TFHE

September 28, 2024

1 Introduction

TFHE is a Fully Homomorphic Encryption (FHE) scheme. It is an encryption scheme that allows you to perform computations over encrypted data.

2 TFHE Ciphertexts

TFHE mainly uses three ciphertext types: LWE, RLWE, and RGSW. All of them have different properties which will be useful in the homomorphic operations

3 GLWE

General LWE, or GLWE includes both of LWE, RLWE

3.1 LWE

LWE encryption supports encrypting small-bit-width integers by placing those bits in the most significant bits of a machine word—for simplicity, say it's a 4-bit integer in the top-most bits of a 32-bit integer with all the other bits initialized to zero, and call that whole encoded plaintext. The secret key is a random binary vector of some fixed length chosen to achieve a specific security. Then, sample a random length vector of 32-bit integers to encrypt, take a dot product with the secret key, add the message, and add some random noise. The encrypted value is both the random samples chosen and the noise-masked result. LWE decryption then reverses this process: re-compute the dot product and subtract it from the output of the encryption. But at the end, you must apply a rounding step to remove the noise added to the ciphertext during encryption. Because the message is in the highest-order bits of the message (say, bits 28-31) and because the noise added was not very large, rounding to the nearest multiple removes it.

3.2 RLWE

In RLWE, the scalar multiplications and additions from LWE are upgraded to polynomial multiplications and additions. We pick a polynomial degree as the maximum degree (say 1024), the coefficients are always integers modulo some chosen modulus q, and finally, we pick a polynomial, usually $x^n + 1$, and represent the result of every operation as a remainder when divided by that polynomial. One or more small integer messages are encoded into a polynomial to encrypt. The secret key is a list of random polynomials with binary coefficients, and the samples are random polynomials with uniformly random mod q coefficients. Then, you take a dot product, add the message, and add a similar "noise polynomial" to mask the result. The main advantage of using RLWE over LWE is that you can pack many messages into a single polynomial and the homomorphic operations you apply to all the messages.

3.3 Secret key

To generate any ciphertext, we first need a secret key. With GLWE ciphertexts, the secret key is a list of random polynomials from R:

$$\overrightarrow{S} = (S_0, S_1, \ldots S_{k-1}) \in R^k$$

The coefficients of the elements can be sampled from a uniform binary distribution, a uniform ternary distribution, a Gaussian distribution, or a uniform distribution. Please note that we can find parameters to achieve the desired security level for these secret keys.

3.3.1 Example

Let's choose N (degree) = 4 and k =2. Let's sample the secret key with a uniform binary distribution of a degree N — 1 polynomial. In this example, the secret keys are [0,1,1,0] and [1,0,1,1].

$$\begin{aligned}
\overrightarrow{S} &= ([0,1,1,0],[1,0,1,1]) \in R^2 \\
\overrightarrow{S} &= (0 + x + x^2 + 0x^3, 1 + 0x + x^2 + x^3) \in R^2 \\
\overrightarrow{S} &= (x + x^2, 1 + x^2 + x^3) \in R^2
\end{aligned}$$

3.4 Message

The message is a polynomial of degree smaller than N with coefficients maximum whose maximum value depends on the value p.

$$M \in R^p$$

3.4.1 Example

Let's choose N (degree) = 4 and p (plain modulus) = 4. The coefficient values of the message are stored in 2 bits (p=4 = 2^2). The possible coefficient value in binary format is 11, 10, 00, and 01. The possible coefficient value in a signed integer is -2, -1, 0 and 1. The possible coefficient value in an unsigned integer is 3, 2, 0 and 1. Let Message be [-2 1, 0 -1] in this example.

3.5 Mask

To encrypt the message, we need to sample a uniformly random mask with coefficients whose maximum value depends on q (modulus).

$$\overrightarrow{A} = (A_0, A_1, \ldots A_{k-1}) \in R_q^k$$

3.5.1 Example

Let's choose N (degree) = 4, k =2 and q = 64 (modulus). The coefficient values of the mask are stored in 6 bits (q=64 = 2^6). The possible coefficient value in binary format is 111111, 111110 ..., 100000, 000000, 000001, ... 011111. The possible coefficient value in a signed integer is -31, -30, ...-1, 0, 1, .. 32. The possible coefficient value in an unsigned integer is 64, 63, ... 33, 0, 1, ..32. In this example, let Mask be [17, -2, -24, 9], [-14, 0, -1, 21].

$$\overrightarrow{A} = ([17,-2,-24,9],[-14,0,-1,21]) \in R^2_{64}$$
$$\overrightarrow{A} = (17-2x-24x^2+9x^3,-14-x^2+21x^3) \in R^2_{64}$$

3.6 Error

We must add a discrete Gaussian Error (small coefficients) to encrypt the message. $\chi_{\mu,\sigma}$ is a Gaussian probability distribution with mean μ and standard deviation σ

$$E \in R_q$$

3.6.1 Example

Let's add [-1,1,0,1] error.

$$E = ([-1,1,0,1]) \in R_q$$
$$E = (x+x^2,1+x^3) \in R_q$$

3.7 Body

The body of an encrypted message is:

$$B = \sum_{i=0}^{k-1} A_i.S_i + \triangle M + E \in R_q$$

where $\triangle$ = q/p.

3.7.1 Example

Let's continue with previous examples for N (degree) = 4, p (plain modulus) = 4, k = 2, q = 64 (modulo) and $\triangle$ = q/p = 16.

$$
\begin{aligned}
B &= \sum_{i=0}^{k-1} A_i.S_i + \triangle M + E \in R_q \\
&= \sum_{i=0}^{2} A_i.S_i + 16M + E \in R_q \\
&= A_0.S_0 + A_1.S_1 + 16M + E \in R_q
\end{aligned}
$$

When we compute R_q, we do polynomial operations modulo $x^N + 1$ and modulo q. To reduce modulo $x^N + 1$, you can observe that:

x^N = x^N ≡ -1 mod x^N + 1

So

$$
\begin{aligned}
A_0.S_0 &= (17 - 2x - 24x^2 + 9x^3).(x + x^2) \\
&= 17x + (17-2)x^2 + (-2-24)x^3 + (-24+9)x^4 + 9x^5 \\
&= 17x + 15x^2 - 26x^3 - 15x^4 + 9x^5 \\
&= 17x + 15x^2 - 26x^3 + (-15 + 9x)x^4 \\
&= 17x + 15x^2 - 26x^3 + (-15 + 9x)(-1) \\
&= 17x + 15x^2 - 26x^3 + 15 - 9x \\
&= 15 + 8x + 15x^2 - 26x^3 \in R_q
\end{aligned}
$$

In the same way:

$$
\begin{aligned}
A_1.S_1 &= -13 - 20x + 28x + 7x3 \in R_q \\
\triangle M &= -32 + 16x - 16x^3
\end{aligned}
$$

Then:

$$
\begin{aligned}
B &= A_0.S_0 + A_1.S_1 + \triangle M + E \in R_q \\
&= -31 + 5x - 21x^2 + 30x^3 \in R_q
\end{aligned}
$$

3.8 Encryption

A GLWE ciphertext encrypting the message M under the secret key $\overrightarrow{S}$ is a tuple:

$$GLWE_{\overrightarrow{S},\sigma}(\triangle M) = (A_0, A_1, \ldots A_{k-1}, B) \subseteq R_q^{k+1}$$

3.8.1 Example

Let's continue with previous examples for N (degree) = 4, p (plain modulus) = 4, k = 2, q = 64 (modulo) and $\triangle$ = q/p = 16.

$$\begin{aligned} GLWE_{\overrightarrow{S},\sigma}(\triangle M) &= (A_0, A_1, B) \subseteq R_{64}^3 \\ &= (17 - 2x - 24x^2 + 9x^3, -14 - x^2 + 21x^3, -31 + 5x - 21x^2 + 30x^3) \subseteq R_{64}^3 \end{aligned}$$

3.9 Decryption

We can decrypt the ciphertext using the following equation:

$$B - \sum_{i=0}^{k-1} A_i.S_i = \triangle M + E$$
$$(\triangle M + E)/\triangle = M$$

Observe that the message M is in the MSB part (thanks to the multiplication by $\triangle$) while E is in the LSB part. If $|E| < \triangle/2$ (so if every coefficient of $|e_i| < \triangle/2$), then the second step of the decryption M returns as expected.

3.9.1 Example

Let's continue with previous examples.

$$B = -31 + 5x - 21x^2 + 30x^3$$

$$A_0.S_0 = 15 + 8x + 15x^2 - 26x^3 A_1.S_1 \quad = -13 - 20x + 28x + 7x3 \triangle M + E = B - \sum_{i=0}^{1} A_i.S_i \triangle M + E \quad = B - A_0.S_0 - A_1$$