CR13 - Fully Homomorphic Encryption — Project
Alain Passelègue & Damien Stehlé - Nov. 27, 2023 — Due date: Jan. 12, 2024

The goal is to implement the GSW FHE scheme seen during the class of Nov. 27, 2023. You should use Python and provide a clean, commented code and a README file. You should also provide a written report explaining how you proceeded.

The homework shall be sent to the lecturers, by email (first.last@ens-lyon.fr), in a tgz/zip archive called by your name.

Implementation of the GSW FHE scheme

The LWE variant

Implement the following functions:

- KeyGen: on input public parameters (a modulus q, a dimension n, and an error distribution χ), returns a pair of public/secret keys $(\mathsf{pk}, \mathsf{sk})$;
- Encrypt: on input the public key pk and a bit b, returns a ciphertext $\mathbf{C}$;
- Decrypt: on input the secret key sk and a ciphertext $\mathbf{C}$, returns a bit;
- Eval: on input a binary circuit with ℓ inputs and depth d, composed of fan-in-2 gates `AND`, `XOR`, `OR`, `NAND`, or fan-in-1 gate `NOT`, a list of ℓ ciphertexts, returns a ciphertext.

In the process, you should implement additional functions for homomorphically evaluating a single fan-in 2 gate `AND`, `NAND`, `XOR`, `OR` and a single fan-in 1 `NOT` gate.

The Ring-LWE variant

As you can realize, the LWE version will hardly run on your machine. To gain efficiency, describe a Ring-LWE variant in your report, and implement this version as well. The guidelines are the same as for the previous part.

Parameter selection for leveled GSW

For this part, you should implement a function Setup which on input a security parameter λ and a maximal depth d, returns parameters q and n such that:

- correctness holds;
- the scheme is at least λ-bit secure.

For bit-security, you will use the lattice estimator 'rops'. See:

https://github.com/malb/lattice-estimator