Running the Homomorphic Encryption Utility

Version 1.0 – 5/15/2023

The program takes an input of integers formatted as a CSV (Comma-Separated Values) file, encrypts them via homomorphic encryption, allows the user to perform simple addition, subtraction, or multiplication on the ciphertext, and decrypts the result.

Table of Contents

Loading Integer Values

The CSV file can be selected via the top GUI file chooser button labeled “CSV Input File”. This button will open a directory navigation screen allowing the user to select a CSV file. The utility comes with one sample CSV file named “Sample.csv” with 100 sample integer values. It reads the integer values into mapping whereby the first integer value has a key of 1 and the last integer value has a key equal to the number of integers in the file. Once loaded into the program, the integer values from the CSV file and their associated key values will be shown in the “Original Integer” scroll viewport located on the left side of the application window.

Encrypting Integer Values

To encrypt the integers, click on the “Encrypt ↓” button located under the “Original Integers” viewport. The integers will be encrypted via the SWHE (SomeWhat Homomorphic Encryption) scheme BFVRNS with the default or user supplied “Homomorphic Encryption Options” set at the bottom of the application. Once encrypted, a JSON object representing the ciphertext will be shown in the “Original Ciphertext” viewport located on the left side of the application window.

Setting the Modulus:

The prime modulus number (q) must satisfy the condition:

(q-1) / m = an integer

Whereby, (q) is the prime modulus, and (m) is the cyclotomic number.

Note:
Any mathematical operations performed on the input values in the CSV file must result in a number less than half of the modulus or an overflow condition will occur.

Any mathematical operation performed on an input value resulting in a value greater than half the modulo (q/2) will have the modulus value subtracted from it, repeatedly if necessary, until the resultant value is between the values of -q/2 and q/2.

The default prime modulus value is ‘557057’. Try using the smaller prime modulus value of ‘65537’ for smaller integer inputs and faster calculations.

Setting the Multiplicative Depth:

In homomorphic encryption, the multiplicative depth refers to the number of times one can perform multiplication operations on encrypted data without having to decrypt it. It measures the number of successive multiplications that can be carried out while preserving the encrypted form.

The multiplicative depth is typically limited in homomorphic encryption schemes. Each multiplication operation increases the depth by one, and exceeding a certain depth limit can lead to exponential growth in the complexity and computational requirements of performing operations. This limitation is known as the "depth bootstrapping problem."

The default value is set to a multiplicative depth of ‘2’ and can be increased to a maximum of ‘4’.

Performing Mathematical Operations on Ciphertext

In the middle of the application window, the user can select eight different mathematical operations to perform on the ciphertext. All mathematical operations are applied uniformly across all original integer inputs. The user can select from the following mathematical operations:

- “Add 1” – This adds 1 to all original integer values.
- “Add 3” – This adds 3 to all original integer values.
- “Add 5” – This adds 5 to all original integer values.
- “Subtract 1” – This subtracts 1 from all original integer values.
- “Subtract 3” – This subtracts 3 from all original integer values.
- “Subtract 5” – This subtracts 5 from all original integer values.
- “Multiply by 3” – This multiplies all original integer values by 3.
- “Multiply by 5” – This multiplies all original integer values by 5.

Once a user clicks on a mathematical operation to perform, the application runs the operation on the ciphertext and displays the resultant ciphertext as a JSON object in the “Resultant Ciphertext” viewport on the right side of the application window. Additionally, a history of all performed ciphertext operations are listed in the scroll window viewport located below the mathematical operation buttons. Running multiple mathematical operations on the ciphertext will result in the most recent ciphertext JSON object being displayed first in the “Resultant Ciphertext” viewport. A history of the resultant ciphertexts will be shown if the user scrolls down the viewport. The most recent resultant ciphertext is always displayed first.

Note:
Multiplicative operations on the ciphertext are limited by the ‘Multiplicative Depth’ option set prior to plaintext encryption. The program will prevent the user from additional multiplicative operations if the multiplicative depth is reached.

Decrypting the Ciphertext

When the user is done performing mathematical operations on the ciphertext, they may decrypt the resultant ciphertext via the “Decrypt ↑” button located above the “Resultant Ciphertext” viewport on the right side of the application window. The decrypted resultant integers will be displayed in the “Resultant Integer” viewport above the decrypt button. By comparing the resultant integers to the original integers, you can validate that all performed mathematical operations were calculated successfully.

Note:
Refer to the “Setting the Modulus” section for information on resultant integers which seem incongruent. All resultant integers must fall within the range of -q/2 and q/2 where q is the prime modulus number used for encryption. Apparently incongruent integer results will be found to be correct, but reduced by the modulo to fall within this range.

Resetting the Program

The user may click the “Reset” button to reset the integer input, plaintext, and ciphertext. Once the user clicks the reset button, they will be able to load a new CSV file and perform new homomorphic encryption operations on the new integer data.