Security and Privacy
2025/2026

Project3: Privacy Through Homomorphic Encryption

1. Introduction

Objective: Explore the concepts of Homomorphic Encryption as well as the tools and libraries that implement those concepts.

Groups: groups of 2 or 3 students

Final deadline: **December 14, 2025**

Delivery: Write a final report that should include the reasoning behind the choices made in each of the steps defined below. Include all the relevant results and discussion in your report.

Project defenses: slots will be made available.

2. Description

Step #1 – Problem definition and selection of dataset

In this step, you will define a scenario and a problem that can be effectively addressed using Homomorphic Encryption. In this scenario, two key roles are involved: a **Data Holder** and a **Data Analyzer**.

- The **Data Holder** owns the data, encrypts it, and sends it securely to the **Data Analyzer**.
- The **Data Analyzer** processes the encrypted data, performs operations on it, and returns the encrypted results to the **Data Holder**.
- Finally, the **Data Holder** decrypts the results to access the desired output.

To proceed, identify and prepare a dataset relevant to the problem you want to solve (e.g., a list of values such as salaries). This dataset will serve as the foundation for demonstrating how Homomorphic Encryption can enable secure computations without exposing sensitive data.

Step #2 – Applying Homomorphic Encryption

In this step, you will implement the **Data Holder** functionality, which consists of two main functions: **encrypt()** and **decrypt()**.

- The **encrypt()** function takes a list of values and a file name as input parameters. It encrypts the data using a homomorphic encryption algorithm and saves the encrypted values to the specified file.
- The **decrypt()** function receives a file name containing the results of statistical operations. It decrypts the data from the file and prints the results.

Note: Groups of **two students** should implement the *Data Holder* and *Data Analyzer* using **two Fully Homomorphic Encryption (FHE)** schemes and compare their execution times.
Groups of **three students** should implement **three different FHE** schemes and compare them against each other in terms of efficiency.

You can use ***phe*** or ***tenseal*** or any other libraries in **Python** for this[1].

Step #3 – Analysis of data over data encrypted by Homomorphic Encryption

In this step, you will implement the functionality of the **Data Analyzer**. The core task is to read encrypted data from a file, perform statistical analysis on the data (e.g., calculating the sum and mean), preferably including both addition and multiplication operations, and then write the computed results back to a file. This ensures that the Data Analyzer processes the information without accessing its plaintext form, maintaining the integrity of the homomorphic encryption workflow.

Step #4 – Writing the report

Your report should provide a detailed description of the scenario and the problem you defined, along with an explanation of the dataset used. It should also include the full implementation of the code for both the **Data Holder** and the **Data Analyzer**, accompanied by clear and concise descriptions of how the code works. The report should also include the results regarding the comparison between different schemes.
During the defense, I expect you to demonstrate the execution of the code for both the **Data Holder** and the **Data Analyzer**. This will involve showing how the Data Holder encrypts the dataset, how the Data Analyzer performs computations on the encrypted data, and finally, how the Data Holder decrypts and interprets the results.

Good luck ☺

[1] See this tutorial carefully before start working: https://sefiks.com/2023/04/10/a-step-by-step-fully-homomorphic-encryption-example-with-tenseal-in-python/