Homework 3

Name:
CWID:

Pack all the files into a zip file "*yourname_hw3.zip*" (**Bonus: 2 Points**) and submit it on the Blackboard by **Nov 28, 2021 (11:59 PM CST)**. Notice that:

- Penalty will apply for late submissions (per our syllabus).
- If you would like to let us know how to run your program, please feel free to include it in the report (but this is optional). Thank you.
- The portion of this homework in the overall grade is 15%. 20 bonus points are available (equivalent to 3 extra points on the overall grade 100 points).

1. Secure Multiparty Computation (30 points)

Alice holds a private Boolean vector $\vec{A}$ with 10 Boolean entries ($\{0,1\}^{10}$) while Bob holds another private Boolean vector $\vec{B}$ with another 10 Boolean entries ($\{0,1\}^{10}$). Design and implement a protocol using the *Fairplay* to securely compute the scalar product $\vec{A} \cdot \vec{B}$ without sharing their inputs to each other.

- The scalar product computation should be converted to garbled circuits using SFDL.
- *Fairplay* secure function evaluation: https://www.cs.huji.ac.il/project/Fairplay/.
- Readme file for running *Fairplay* SFE:
  https://www.cs.huji.ac.il/project/Fairplay/Fairplay/Readme.txt

Tasks:

(a) Alice generates random Boolean entries for $\vec{A}$ while Bob generates random Boolean entries for $\vec{B}$. (**3 points**)

(b) Write the SFDL program for Alice and Bob. (**12 points**)

(c) Compile it for Alice and Bob, and run the protocol (communication is integrated in *Fairplay*). (**5 points**)

(d) Report the input Boolean vectors, the SFDL program, SHDL circuit, and output results $\vec{A} \cdot \vec{B}$ (for two parties). (**10 points**)

Submission Part I: (1) a report including the protocol design, implementation details, input matrices and output results, (2) screenshots of the major steps of each task and your answers (you can explain your findings with tables or figures), and (3) source code files – all named with the prefix "hw3-1-" (e.g., *hw3-1-report.pdf*, and *hw3-1-matrix.txt*).

2. Application of Homomorphic Encryption (40 points)

Alice holds a private matrix A (nonnegative integer entries) with size 5×8 while Bob holds a private matrix B (nonnegative integer entries) with size 8×4. Design and implement a two-party protocol to securely compute the product $A \times B$. Hint: Homomorphic Encryption (e.g., Paillier Cryptosystem which is a public key-based scheme) can be used to design the protocol.

- Paillier in Python:

  https://python-paillier.readthedocs.io/en/develop/

  https://github.com/mikeivanov/paillier

- Paillier in Java:

  https://www.csee.umbc.edu/ kunliu1/research/Paillier.html

Tasks:

(a) Alice generates random nonnegative integer entries for A while Bob generates random nonnegative integer entries for B. (**5 points**)

(b) Design the cryptographic protocol between Alice and Bob to perform secure computation. (**10 points**)

(c) Write the programs for Alice and Bob: computation and communication. Note that communication should be established to exchange encrypted messages, e.g., using Socket programming or establishing other distributed computing systems. (**20 points**)

- Socket Programming in Python: https://realpython.com/python-sockets/
- Socket Programming in Java: https://www.tutorialspoint.com/java/java_networking.htm

If you have difficulties on distributed computing systems, you can write a single program to simulate the procedures of Alice and Bob. In that case, you should mark all the steps (computation by which party; message sent from which party to which party) in the source codes.

(d) Report the input matrices, the last ciphertext (right before the decryption) and the decrypted product $A \times B$ using two different key sizes 512-bit and 1024-bit. (**5 points**)

Submission Part II: (1) a report including the protocol design, implementation details, input matrices, last ciphertext, and decrypted result, and (2) source code files – all named with the prefix "hw3-2-" (e.g., *hw3-2-report.pdf*, and *hw3-2-alice.java*).

3. SMC Protocol Design (30 points + 20 bonus points)

Four different parties (Alice, Bob, Chris, and David) locally hold four different vectors, respectively (10 integers in each vector).

- Alice holds: $\vec{V_a} = [a_1, \ldots, a_{10}]$.
- Bob holds: $\vec{V_b} = [b_1, \ldots, b_{10}]$.
- Chris holds: $\vec{V_c} = [c_1, \ldots, c_{10}]$.
- David holds: $\vec{V_d} = [d_1, \ldots, d_{10}]$.

Design a cryptographic protocol to securely sum all the four vectors:

$$\vec{V} = \vec{V_a} + \vec{V_b} + \vec{V_c} + \vec{V_d}$$

and find the maximum value (out of the 10 entries) in $\vec{V}$.

Hints:

- You can use the combination of Homomorphic Encryption (e.g., Paillier's Cryptosystem), Garbled Circuit (e.g., Fairplay) and Permutation to solve this problem.
- Sum $\vec{V}$ should not be disclosed to any party. The maximum value in $\vec{V}$ will be the only output.
- If necessary, two-party Fairplay can also be used to securely compare multiple values by executing multiple comparisons.
- Only using Garbled Circuit (e.g., Fairplay) may not be computationally practical.
- Try to reduce the information leakage in the protocol as much as possible.

Tasks:

(a) Protocol Design (write the pseudocode in the report). (**30 points**: partial credits will be given to different functions and building blocks)

(b) Implementation of the Protocol (similar to the setting). (**20 bonus points**)

Submission Part III: (1) a report including the protocol design, (2) implementation and testing details in the report (bonus), and (3) source code files (bonus) – all named with the prefix "hw3-3-" (e.g., *hw3-3-report.pdf*, and *hw3-3-alice.java*).