CS 203.4860	Secure Multi-Party Computation	Spring 2023

Homework 2: Report

Lecturer: Dr. Adi Akavia *Student(s): Nir Segal, Hallel Weinberg, Michael Rodel*

Abstract

In this homework we implemented a secure two-party protocol using the passively secure One-Time Truth-Table protocol. We wrote code in Python that uses this protocol to compute the following function:

$$f_{a,4}(x_1, x_2) = \begin{cases} 1 & ax \geq 4 \\ 0 & otherwise \end{cases}$$

Our notebook colab notebook is here:
`https://colab.research.google.com/drive/1cx1NWtXvYy8IbYRvGaIai27CievS24Me?usp=sharing`.

1 Introduction

Motivation. When 2 parties want to exchange information, they have to use a secure protocol to ensure that the information remains confidential and protected from any malicious actors who may try to intercept it. The One-Time Truth-Table protocol is a passively secure protocol that provides a way for two parties to securely exchange information.

Secure Computation Technique. The One-Time Truth-Table protocol achieves passive security by using a one-time truth table, which is a table that describes the output of a function for all possible inputs. The two parties generate their own random inputs and use the truth table to compute the output of the function without revealing their inputs to each other. This ensures that even if an attacker is listening in, they cannot determine the inputs of the parties and therefore cannot compute the output[Orl].
By using the passively secure OTTT protocol, the two parties can exchange information securely from passive attackers. This is especially useful in scenarios where there is a risk of passive attacks.

Application. In our application we defined 3 classes: the dealer, Alice and Bob, with the first responsible for the offline phase and the last two for the online phase. We first generate a, x randomly to be used as input to the protocol and then we communicate between the classes using a few lines of code:

1. We initialize the dealer and perform the offline phase.
2. We perform the online phase: we initialize Alice and Bob with their inputs a, x and the outputs of the offline phase $(r, M_A), (c, M_B)$.

3. Alice computes u and sends to Bob.

4. Bob computes (z_B, v) and sends to Alice.

5. Alice outputs z.

Note that actually Given a, x, z is equal to the result of the function $f_{a,4}(x)$ for a, x above.

Empirical Evaluation. We conducted experiments as follows: for each possible input a, x we computed using OTTT an output z.
In short, the results we received correspond to the expected output of the function $f_{a,4}(x)$ for a, x the aforementioned.

2 Preliminaries

Secure multi-party computation (SMPC) is a technique that enables multiple parties to compute a function collaboratively, without revealing their inputs to each other. In other words, it allows multiple parties to perform computations on their private data without revealing that data to any of the other parties.[Wik]

Passive attacker is an attacker who only observes the communication between the two parties and does not actively interfere with it. It allows the parties to exchange information even in an adversarial environment where an attacker may be listening in on the communication channel.[Con]

3 Protocols

3.1 Secure Computation Technique: One-Time Truth-Table (OTTT) Protocol

Parties: Alice A, Bob B, Trusted dealer D.
Functionality: $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\} \times \perp$, $(x, y) \to (f(x, y), \perp)$.
Truth Table: Truth table T with 2^n rows and 2^n columns, where $T[i, j] = f(i, j)$.

3.1.1 The Algorithm

Parties: Dealer D with truth table T.

Algorithm 1 The Offline Phase

1. Generate $c, r \in \{1, \ldots, 2^2\}$, the shifts of the rows and columns.
2. Generate $2^2 \times 2^2$ matrix M_B.
3. Compute $M_A[i, j] = M_B[i, j] \oplus T[i - r \ mod \ 2^2, j - c \ mod \ 2^2]$.
4. Output (r, M_A) to Alice and (c, M_B) to Bob.

Parties: Alice A with (r, M_A) and input x, Bob B with (c, M_B) and input y.

Algorithm 2 The Online Phase

1. Alice computes $u = x + r \ mod \ 2^2$ and sends u to Bob.
2. Bob computes $v = a + c \ mod \ 2^2$ and $z_B = M_B[u, v]$ and sends (z_B, v) to Alice.
3. Alice outputs $z = M_A[u, v] \oplus z_B$.

3.2 Application: One-Time Truth-Table (OTTT) Protocol

Algorithm 3 One-Time Truth-Table Protocol

1. Generate $a, x \in \{0, 1, 2, 3\}$, randomly.
2. The offline phase (the dealer):
   (a) Generate a truth table T that represent formula 2:

$$f_{a,4}(x) = \begin{cases} 1 & ax \geq 4 \\ 0 & otherwise \end{cases}$$

	0	1	2	3
0	0	0	0	0
1	0	0	0	0
2	0	0	1	1
3	0	0	1	1

   (b) Generate $c, r \in \{1, \ldots, 2^2\}$, the shifts of the rows and columns.
   (c) Generate $2^2 \times 2^2$ matrix M_B.
   (d) Compute $M_A[i, j] = M_B[i, j] \oplus T[i - r \ mod \ 2^2, j - c \ mod \ 2^2]$.
   (e) Output (r, M_A) to Alice and (c, M_B) to Bob.
3. The online phase (Alice and Bob):
   (a) Alice computes and sends u to Bob.
   $u = x + r \ mod \ 2^2$.
   (b) Bob computes and sends (z_B, v) to Alice.
   $v = a + c \ mod \ 2^2$.
   $z_B = M_B[u, v]$.
   (c) Alice outputs $z = M_A[u, v] \oplus z_B$.

4 Implementation

The code is written in Python. Numpy library is the only one required to run the code. Our code generates 2 random inputs a and x and returns an output z, the result of $f_{a,4}(x)$.

5 Empirical Evaluation

Each row in the table is an experiment: we conducted 16 experiments on all possible inputs of a and x for which we got output z:

a	x	z
0	0	0
0	1	0
0	2	0
0	3	0
1	0	0
1	1	0
1	2	0
1	3	0
2	0	0
2	1	0
2	2	1
2	3	1
3	0	0
3	1	0
3	2	1
3	3	1

Note that for each a, x we got z which corresponds to the result of the function $f_{a,4}(x)$ for these a, x:

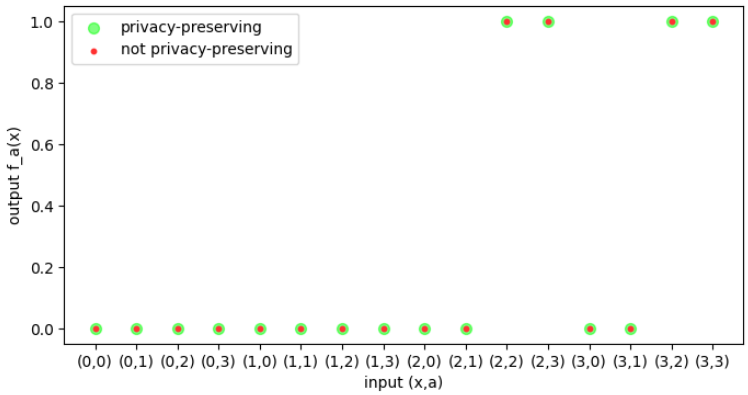

Figure 1: Comparison between privacy-preserving computation and not privacy-preserving computation

Note: You can see our tests file here: `https://colab.research.google.com/drive/1ZkhDAkdqNfAIFrUsUc3jaNJfyLSQlFP9?usp=sharing` and the Benchmark tests file here: `https://colab.research.google.com/drive/1YwCvRyzoSbodK_8Le6fBuw2aUjnKzyXa?usp=sharing`.

6 Conclusions

As shown in the results in the previous seciton, We see that the proposed approach yields correct results for $f_{a,4}(x)$. Therefore, the output correctness of the proposed approach is not compromised by considering privacy.
By using the passively secure OTTT protocol, we were able to maintain participants' (Alice and Bob) privacy because the private data didn't need to be disclosed for computations.

References

[Con] TechTarget Contributor. Passive attack.

[Orl] Claudio Orlandi. "tiny ot" – part 1.

[Wik] Wikipedia. Secure multi-party computation.