CS 203.4860	Secure Multi-Party Computation	Fall 2021

Homework 7: Report

Lecturer: Dr. Adi Akavia *Student(s): ()*

Work space:

We used Python to implement a Yao's protocol for the Boolean circuit evaluating Equation 2, we used OT and ElGamal from previous assignment.

Classes,Functions and implementation:

modular sqrt function:

Solve the congruence for the given parameters a,p of the form:$x^2 = a(modp)$,and returns x or 0 if there is no square root exists for these a and p.

legendre symbol function:

Compute the Legendre symbol a—p using Euler's criterion. p is a prime, a is relatively prime to p (if p divides a, then a—p = 0), the function returns 1 if a has a square root modulo p, and -1 otherwise.

PKE ElGamal class:

we implemented class with the name PKE Elgamal, with init function which choose a random number p and check if it's prime number and if 2*p+1 is prime too using helper functions we've implemented, in addition it initialize q which is equal to 2*p+1. we also implemented Gen,OGen,Enc and dec as functions in this class as we've learned. (same class as previous assignment)

OT class:

OT class is a class we've implemented instead the dealer we've implemented in previous assignments , and it was implemented as we've learned in lectures, init function saves the PKE scheme, load function saves the messages into data variable to use it later,in getkeys function the function gets key as parameter and create a message using enc function we've implemented in PKE ElGamal class. Sendmsgs functions returns the message we've created in getkeys, chose defines sk and pk using Gen and OGen functions we've implemented before(pk and sk same as it was defined in lecture), sendchose returns the pks (Gen and OGen), getData returns output dec as defined in ElGamal.(same as previous assignment)

label class:

label is value (bytes) and the function valid chacks that all the bytes after the first l1 bytes are zero.

labelTuple class:

labelTuple is two labels that represent the True,False values for specific wire.

genTT function:

genTT creates the gate of Garbled Circuit from two given inputs wire inp1,inp2, genTT supports 'or','xor','add','and','mul' and 'not' (all of the above are mod 2)

class GC:

The of the garbled circuit and we've used hashlib.sha256 that work with 32 bytes (512 = 32*8 bits) , and we chose l2=32 as default.

Alice and Bob classes:

In previous assignments we've implemented Alice and Bob as different objects but same class, in Yao's protocol we should define Alice and Bob as different classes because they're dealing with different parts/wires of the circuit, but the functionality is almost the same as before Alice get garbled circuit and input and sends and receive messages from Bob , Bob sends and receives messages too without knowing each others inputs and finding the output. **circuits and equation 2 are defined as previous assignments.**

Test and complexity:

We tested and checked if all the classes are correct and the output is always right , we also calculated average communication and got 5 KB as average.

Execution Time:

We also calculated the executing time by using package time and subtract the start time from the time when we end running the tests and we got different times of execution such as 0.8872568607330322 seconds, 0.7954013347625732 seconds, 0.8208820819854736 seconds for 10 tests the in average every test execution takes between 0.07 to 0.08 seconds.