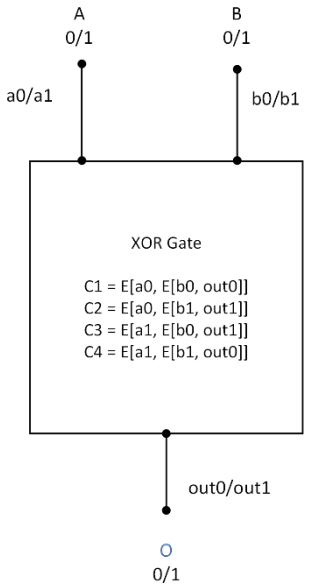
Secure Software Development (CSCE 5565/4565)
Lab 4 - Secure Computation

Background

Garbled Circuit, also known as Yao's protocol, is a cryptographic technique used for secure function evaluation (SFE). In SFE, multiple parties wish to jointly compute a function over their inputs without revealing their respective input to each other. Each gate in the circuit is encoded in such a way that the input and output values are concealed.



The garbling process involves encoding the parties' input, output, and function such that the decoding of the output is the application of the function on the parties' input. For example, the above representation shows the garbling of an XOR Gate circuit, by encrypting the possible gate outputs using the codes representing the input.

Garbled Circuits are particularly useful for secure function evaluation because they enable parties to collaborate on computations without exposing their private inputs. They are essential tool in various applications such as secure data analysis, private information retrieval, and secure multiparty computation in fields like finance, healthcare, and privacy-preserving machine learning.

Goal: The goal of this lab is to have the students get familiar with the use of cryptography. The application shows computing a function without sharing the input.

Pre-Setup:

- Get the VM attached in for Lab 4 from the Canvas Link.
- Customize your command prompt to reflect your EUID by using the following command and substituting <EUID> with your exact UNT EUID.
    - $ hostnamectl set-hostname <EUID>

Secure Software Development (CSCE 5565/4565)
Lab 4 - Secure Computation

- Get a refresher on the different logic gates and their respective truth tables. For instance, below is the XOR gate.

XOR gate truth table

Input		Output
A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

The tasks follow.

Task 1:

Provide a brief description of Yao's Garbled Circuits and How they function in your own words.

Provide your references.

Task 2:

Open the terminal application and clone the repo which has implemented the Garbled Circuit for One-Gate from the link provided below for the assignment.

- Check if your Python setup is working correctly.
  - $ python --version
- Clone the repository.
  - $ git clone https://github.com/kalyancheerla/SSS_Lab4_Garbled_Circuit_Protocol.git
  - $ cd SSS_Lab4_Garbled_Circuit_Protocol/
- Kindly install the dependencies required for the program by using pip.
  - $ pip install -r requirements.txt
- Check if the `onegate_gc.py` is working correctly using the following commands.
  - $ python onegate_gc.py XOR 0 0
  - $ python onegate_gc.py XOR 0 1
  - $ python onegate_gc.py XOR 1 0
  - $ python onegate_gc.py XOR 1 1

Attach the screenshot and provide an explanation of what you observe.

Task 3:

Secure Software Development (CSCE 5565/4565)
Lab 4 - Secure Computation

The main function encodes the inputs A & B were using *generate_labels()* function. Modify the code to print the encoded values using the command:

$ python onegate_gc.py XOR 1 1

Attach the screenshot and provide an explanation of what you observe.

Revert the code change.

Task 4:

Now, modify the code to print the Garbled Table, Output Labels & Encoded Output and run the same above command.

$ python onegate_gc.py XOR 1 1

Attach the screenshot and provide an explanation of what you observe. Also, did you find the encoded Output being present in the Output labels list?

Task 5:

Change the Gate to an AND gate and test if the program is working correctly for all the inputs in the truth table.

- $ python onegate_gc.py AND 0 0
- $ python onegate_gc.py AND 0 1
- $ python onegate_gc.py AND 1 0
- $ python onegate_gc.py AND 1 1

Attach the screenshot and provide an explanation of what you observe.

Task 6:

Now try to update the code to take gate called 'XNOR'.

Attach a screenshot with the code change and evaluate the program with all the different combinations of input Boolean values.

Task 7:

Look at the Garbler Logic in onegate_gc.py and explain briefly in your own words how the whole garbling logic works.

Task 8:

Now look at the Evaluator Logic in onegate_gc.py and explain briefly in your own words how the whole evaluation logic works.

Task 9:

Secure Software Development (CSCE 5565/4565)
Lab 4 - Secure Computation

What is the encryption cipher that is being used to perform the double encryption? Can we use a different symmetric cipher in place of that? Suppose the order double encryption is changed like below then how the order of decryption should be changed.

```
-    return cipher2.encrypt(cipher1.encrypt(data))
+    return cipher1.encrypt(cipher2.encrypt(data))
```

How many parties are involved in the above Secure Function Evaluation? And can we update the logics to support Multiple parties (MPC – Multi Party Computation).

Task 10:

Let us consider the below circuit, consisting of three inputs, A, B, and C, and involving two logic gates OR and AND. Change the program to support this circuit configuration.