CS 600.442 – Modern Cryptography 10/26/2015

Lecture 17: Secure Computation - III (GMW)

Instructor: Abhishek Jain *Scribe: Zhenyu Liu*

1 Recap from the previous lecture

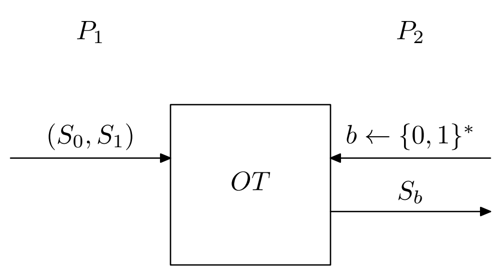
In the last lecture, if we let W denote the set of wires, G denote the set of gates and C denotes the circuit, when $Garble(C, x)$ computes f, for every wire $w \in W$, choose $labels(S_w^0, S_w^1)$; then P_1 will send his label $labels^{x_1}$ directly to P_2 to compute. However, this method is not secure, because for P_2 he can always chooses his input after learning the message from P_1.

So, in order to ensure that two parties, which respectively hold the secret inputs x and y, can jointly compute any function $f(x, y)$ in a secure manner, we decide to construct our protocol based on zero-knowledge proofs. Then, we brought in the definition of garbled circuit for the two-party computation. With this idea, in the last lecture, we have already had the basis of a protocol; that is one of the parties can construct a garbled circuit for C, and the other party can evaluate it. Now, the next is to find a method to transfer the keys corresponding to the evaluator's input to the evaluating player in Yao's circuit method.

2 Oblivious Transfer

In order for the evaluating player to start evaluating the circuit, the player must know the key corresponding to his input. Except for the situation discussed in the last section, P_2 also cannot send his all the keys, then P_1 will know the result of the circuit calculated by input (x, y) and $(x, \overline{y})$, which violates the security definitions at first.

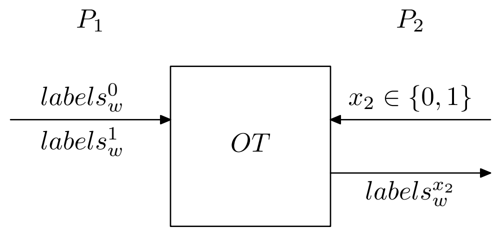
Thus, we bring in *oblivious transfer* protocol to deal with this situation - specifically, here is $1/2 - oblivious\ transfer$. This protocol is a secure computation for party A to learn one of k secret bits held by party B without any knowledge leaking out. The concrete process is displayed in the following example.



In the picture, P_1 input his two message S_0 and S_1 and P_2 randomly choose a String from $\{0, 1\}^*$ to input, then P_2 will get the output S_b. The most important characteristics of this protocol are

- P_1 should not learn b
- P_2 should not learn S_{1-b}

Particularly, if the input of P_2 is just one bit long, according to our notations in the previous lecture, the result is



3 GMW87 Protocol

3.1 Background

There are two parties, P_1 and P_2. Define that x, y are respectively the inputs of these two parties. Each input has the same length n and the two parties will use their input to evaluate a garbled circuit denoted by C. In this protocol, the circuit consists of many gates, such as XOR, NOT and AND. We will construct the gate to evaluate one bit a time.

3.2 Initialization

Let $x = x_1x_2 \ldots x_n$ and $y = y_1y_2 \ldots y_n$. For every wire, P_1 holds all the a_w share and P_2 holds all the b_w share. $\forall i \in [n]$, for two parties, they can initial and split their value based on the following way:

$$Sample\ a_{w_i} \leftarrow \{0, 1\}$$
$$b_{w_i} = a_{w_i} \oplus x_i$$

3.3 Evaluation

Now, we will mainly discuss three example gate evaluations - XOR, NOT and AND. According to the previous notations, P_1 has two shares, $a^1_{w_i}$ and $a^2_{w_i}$, whereas P_2 has two shares $b^1_{w_i}$ and $b^2_{w_i}$; and $x_i = a^1_{w_i} \oplus b^1_{w_i}$ while $y_i = a^2_{w_i} \oplus b^2_{w_i}$. Using the intuitions in secure computation, we need to evaluate the gate without learning extra value. Here starts from XOR gate.

(1) **XOR gate**

For XOR gate, $G(x_i, y_i) = x_i \oplus y_i = (a^1_{w_i} \oplus b^1_{w_i}) \oplus (a^2_{w_i} \oplus b^2_{w_i})$. Define $a_{XOR(V_{w_1}, V_{w_2})} = a_{w_1} \oplus a_{w_2}$ and $b_{XOR(V_{w_1}, V_{w_2})} = b_{w_1} \oplus b_{w_2}$. If we just simply switch the position of the each value,

$$G(x_i, y_i) = XOR(a_{w_1} \oplus a_{w_2}, b_{w_1} \oplus b_{w_2}).$$

Obviously, the whole process is not required any communication.

(2) **NOT gate**

For NOT gate, P_1 only has a_{w_1} and P_2 only has b_{w_1}. Thus, define $a_{NOT(V_{w_1})} = 1 - a_{w_1}$ and $b_{NOT(V_{w_1})} = 1 - b_{w_1}$. Then, we can get

$$G(x_i, y_i) = (1 - a_{w_1}) \oplus (1 - b_{w_1}) = -(a_{w_1} \oplus b_{w_1})(mod\ 2)$$

(3) **AND gate**

To be continued in the next lecture.