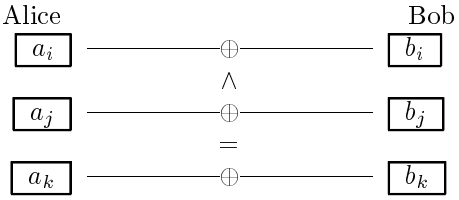
1 Lecture 13

1.1 2-party computation

The $\wedge$ gate

In this situation, Alice owns two bits, a_i and a_j, while Bob owns b_i and b_j. They want to compute the $\wedge$ gate without revealing their own bits to each other. The computing scheme is the following:



In other words, Alice and Bob have to agree on the bits a_k and b_k without knowing the value of the bit that belongs to the other party. Hence, a_k and b_k must verify the following equation:

$$(a_i \oplus b_i) \wedge (a_j \oplus b_j) = (a_k \oplus b_k) \tag{1}$$

which can be rewritten as

$$(a_i \wedge a_j) \oplus a_k \oplus (a_i \wedge b_j) \oplus (a_j \wedge b_i) \oplus (b_i \wedge b_j) = b_k \tag{2}$$

From Alice's point of view, this equation can be seen as a function of two unknown values b_i and b_j:

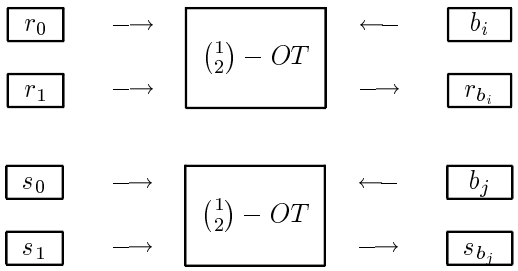
$$b_k = f(b_i, b_j) = \begin{cases} (a_i \wedge a_j) \oplus a_k & \text{if } \ b_i = b_j = 0 \\ (a_i \wedge a_j) \oplus a_k \oplus a_i & \text{if } \ b_i = 0 \ \ b_j = 1 \\ (a_i \wedge a_j) \oplus a_k \oplus a_j & \text{if } \ b_i = 1 \ \ b_j = 0 \\ (a_i \wedge a_j) \oplus a_k \oplus a_i \oplus a_j & \text{if } \ b_i = b_j = 1 \end{cases} \tag{3}$$

She computes these four values and commits to r_0, r_1, s_0 and $s_1 \in_R \{0,1\}$. She then computes the four values

$$w_{ij} = f(b_i, b_j) \oplus r_i \oplus s_j \tag{4}$$

and makes them public. Bob cannot learn anything about the values of a_i, a_j and a_k since the w_{ij}'s are random. Furthermore, Alice can prove to Bob that the w_{ij}'s were computed properly since all of its components are commited to.

It is clear from eq.**??** that revealing the values of r_i and s_j yields the value of $f(b_i, b_j)$, and nothing else. Hence, to get the desired value that must be assigned to b_k, Alice and Bob must perform the following oblivious transfers:



After these oblivious transfers, Bob is commited to the values of r_{b_i} and s_{b_j}. He can then compute

$$b_k = w_{b_i b_j} \oplus r_{b_i} \oplus s_{b_j} \tag{5}$$

and prove to Alice that he did it properly.

Remark 1. To make this proof, Bob would have to be commited to $w_{b_i b_j}$ which has not been mentioned in the above protocole. One way of doing this is to compute

$$\left(w_{00} \wedge \boxed{\overline{b_i}} \wedge \boxed{\overline{b_j}}\right) \oplus \left(w_{01} \wedge \boxed{\overline{b_i}} \wedge \boxed{b_j}\right) \oplus \left(w_{10} \wedge \boxed{b_i} \wedge \boxed{\overline{b_j}}\right) \oplus \left(w_{11} \wedge \boxed{b_i} \wedge \boxed{b_j}\right) = \boxed{\mathrm{w}_{b_i b_j}} \tag{6}$$

and to prove to Alice that the computation was done properly.

2-party computation based on QR assumption

In the beginning of this protocole, Alice and Bob have to agree on some QR assumptions

Alice Bob

$$\xrightarrow{\quad p_A \times q_A = N_A,\ \ y_A \in \mathrm{QNR}_{N_A}[+1] \quad}$$

$$\xrightarrow{\quad \mathrm{ZK}\ \ (y_A \in \mathrm{QNR}_{N_A}) \quad}$$

$$\xleftarrow{\quad p_B \times q_B = N_B,\ \ y_B \in \mathrm{QNR}_{N_B}[+1] \quad}$$

$$\xleftarrow{\quad \mathrm{ZK}\ \ (y_B \in \mathrm{QNR}_{N_B}) \quad}$$

From now on, we will no longer talk of Alice or Bob but of P which can be either Alice or Bob. Obviously, $\overline{P}$ will denote the other party.

To commit to a bit b, P must send $z = r^2 y_p^b \mod N_p$ to $\overline{P}$ for a random $r \in \mathbb{Z}_{N_P}^*$. To unveil the bit, P simply sends r and b to $\overline{P}$ as we have already seen.

The protocole that allows one to obliviously transfer a bit from P to $\overline{P}$ is more complex. Basically, P has to compute

$$z_0 = r_0^2 y_P^{b_0} \mod N_P \tag{7}$$

$$z_1 = r_1^2 y_P^{b_1} \mod N_P \tag{8}$$

and send z_0 and z_1 to $\overline{P}$. To get b_c, $\overline{P}$ first computes $Q = z_c s^2 y_P^a$ for a random $a \in_R \{0,1\}$ and random $s \in_R \mathbb{Z}_{N_P}^*$. He then sends Q to P who answers the question $Q \in QR_{N_P}$? by returning a bit $m = (Q \in QR_{N_P})$. The problem with this protocol is that $\overline{P}$ can construct Q the way he wants which allows him to use P as an oracle to compute quadratic residuosity. Therefore, it must be proven in zero-knowledge that Q a correct question about z_c.

To execute this task, $\overline{P}$ will have to construct two sutch questions Q_0 and Q_1

$$Q_0 = z_c s_c^2 y_{N_P}^{a_c} \qquad \boxed{c} \quad \boxed{a_c} \tag{9}$$

$$Q_1 = z_{\overline{c}} s_{\overline{c}}^2 y_{N_P}^{a_{\overline{c}}} \qquad \boxed{\overline{c}} \quad \boxed{a_{\overline{c}}} \tag{10}$$

and present them in random order to P. Obviously, the boxes mean that the random bits c, $\overline{c}$, a_c and $a_{\overline{c}}$ are commited values. $\overline{P}$ will then present k set of question to P

$$Q_0^i = z_{c_i} s_{c_i i}^2 y_{N_P}^{a_{c_i i}} \qquad \boxed{c_i} \quad \boxed{a_{c_i i}} \tag{11}$$

$$Q_1^i = z_{\overline{c_i}} s_{\overline{c_i} i}^2 y_{N_P}^{a_{\overline{c_i} i}} \qquad \boxed{\overline{c_i}} \quad \boxed{a_{\overline{c_i} i}} \tag{12}$$

It is obvious that if all of the commited bits c_i, $\overline{c_i}$, $a_{c_i i}$ and $a_{\overline{c_i} i}$ are unveiled, P can check that the questions were computed correctly. On the other hand, $\overline{P}$ can prove to P that $\left(Qx^i, Q_{\overline{x}}^i\right)$ maps to (Q_0, Q_1) by showing

the proper association

$$\begin{array}{lcl} c = c_i & & a_c \oplus a_{c_i i} = x \\ \overline{c} = \overline{c_i} & \Longrightarrow & a_{\overline{c}} \oplus a_{\overline{c_i} i} = \overline{x} \\ & \text{or} & \\ c = \overline{c_i} & & a_c \oplus a_{\overline{c_i} i} = x \\ \overline{c} = c_i & \Longrightarrow & a_{\overline{c}} \oplus a_{c_i i} = \overline{x} \end{array} \tag{13}$$

Hence, the zero-knowledge proof can be peformed if P randomly asks $\overline{P}$ to either i) unveil the commited bits to prove that the i^{th} question was formulated properly, or ii) prove that the questions Q_0^i and Q_1^i can be mapped to Q_0 and Q_1.