Lecture 20

- Secure two party computation
- Yao's encrypted circuits
- Oblivious transfer
- Breaking and fixing
- Defs of security
- Provably secure protocols

2 party computation

Say, x is 2-bit and C only 1 gate.

<table>
<thead>
<tr>
<th>x</th>
<th>C₀₀</th>
<th>Enc(x₀₀, K₀₀)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0</td>
<td>C₀₀</td>
<td>Enc(x₀₀, K₀₀)</td>
</tr>
<tr>
<td>0 1</td>
<td>C₀₁</td>
<td>Enc(x₀₁, K₀₁)</td>
</tr>
<tr>
<td>1 0</td>
<td>C₁₀</td>
<td>Enc(x₁₀, K₁₀)</td>
</tr>
<tr>
<td>1 1</td>
<td>C₁₁</td>
<td>Enc(x₁₁, K₁₁)</td>
</tr>
</tbody>
</table>

So, Alice

encrypted gate #1:

- C₀₀ ← Enc((K₀₀, K₀₀), K₀₀)
- C₀₁ ← Enc((K₀₁, K₀₁), K₀₁)
- C₁₀ ← Enc((K₁₀, K₁₀), K₁₀)
- C₁₁ ← Enc((K₁₁, K₁₁), K₁₁)

encrypted gate #2:

similar

Bob learns K₀₀

encrypted gate #3:

Encrypted gate #3:

Bob doesn’t know what the gates are here

Encrypted gate #3:

so that Bob doesn’t learn anything about the wire inputs to gate #3.
Constructing the underlying cryptosystem:

1. Need 2 keys, not one
2. If decrypt C under wrong key, fail.

We are given \((\text{SymGen, SymEnc, SymDec})\)

- \(\text{keyGen: generate } k_a, k_b\)
- \(\text{Enc: } ((k_a, k_b, m): \text{SymEnc}(\cdots m \cdot 0))\) you need to do it twice, so:

\[\text{SymEnc}(k_a, \text{SymEnc}(k_b, m \cdot 0))\]

TRY 1.2

We are given \(P_k: \text{PRP over } \{0,1\}^{2k}\)

- \(\text{keygen}(sk) \text{ generate } k_a, k_b\)
- \(\text{Enc}(sk, (k_a, k_b, m): \text{output } P_{k_a}(P_{k_b}(m \cdot 0)))\)

Alice

\[\begin{array}{c}
\text{Circuit} \\
\rightarrow \text{Encrypt gate #i} \rightarrow X
\end{array}\]

Bob

Repeat for every input wire

\(k_a, k_b \rightarrow \text{OT} \rightarrow X_j \rightarrow k'_a, k'_b\)

Encrypted gate \(i = (c_i, c_i', c_i, c_i')\) where

1. Compute \(c_{b,b_2} = \text{Enc}((k_i, k_b), k_3, (c_i, c_i'))\) where

   - \(k_i, k_i': \text{keys corresponding to } \text{1st and input gate}\)
   - \(k_0, k_1: \text{2nd } k_2, k_3: \text{3rd } k_4, k_5: \text{next}\)

2. Permute them to get

\((c_i, c_i', c_i, c_i')\)

Oblivious Transfer

Alice

\(s_0, s_1 \rightarrow \text{OT} \rightarrow b \rightarrow \text{Suppose } b = 0. \text{ Then } y_1 = h y_0 = y_0 h^1\)

Bob

\(a, g, h \rightarrow \text{Bob}

\[\begin{array}{c}
\text{verify } y_1 = y_0 h^1 \\
\text{c_0, c_1, ca = (g_0, g_1, k_a, k_b)} \\
=a = 3, 0, 13
\end{array}\]

\(0, g, h \rightarrow \text{Bob}

\[\begin{array}{c}
\text{sk x, y_b = y_b} \\
\text{y_b = y_b h^{-1}}
\end{array}\]

Bob decrypts \(c_b\)