CMSC 858K — Introduction to Secure Computation	Sept 16, 2013
Lecture 5	

Lecturer: Jonathan Katz

Scribe(s): Andrew Miller

1 Oblivious Transfer Cont.

1.1 Pre-processing Oblivious Transfer

For the pre-processing OT protocol, one round of OT is used to establish keys. Thereafter, an arbitrary number of OT's over different messages can be performed for using these keys.

$\frac{\text{Sender}}{k_0, k_1 \leftarrow \{0, 1\}^n}$	Pre-processing p OT	$\frac{\text{hase}}{c \leftarrow \{0, 1\}}$
	k_0, k_1	$\begin{array}{c} c \\ \hline k_c \end{array}$
	Second phase	9
$\underline{\operatorname{Sender}(m_0,m_1)}$	_	$\underline{\operatorname{Receiver}(b)}$
if z=0	$z = b \oplus c$	
$y_0 = m_0 \oplus k_0$		
$y_1 = m_1 \oplus k_1$		
else		
$y_0 = m_0 \oplus k_1$		
$y_1 = m_1 \oplus k_0$	y_0, y_1	R learns $y_b \oplus k_c = m_b$

Figure 1: Protocol for Pre-processing OT

1.2 OT Extension

An OT extension protocol turns k OTs on m-bit strings into m OTs on n-bit strings, where k is the security parameter.



Figure 2: Protocol for OT extension

1.2.1 Assumption on Randomness of the Hash Function

For arbitrary $T^1, ..., T^m$ and s, the hash function outputs, $\mathcal{H}(s \oplus T^1), \mathcal{H}(s \oplus T^2), ..., \mathcal{H}(s \oplus T^m)$, should be indistinguishable from uniform random, even given $T^1, ..., T^m$.

1.3 GMW (Goldreich-Micali,Wigderson) Approach to semi-honest twoparty computation

Secure computation of arbitrary circuits from OT. Assume we have a Boolean circuit with 2ℓ inputs, the first half are from P₁, the second half are from P₂. The gates may have arbitrary fan-in and fan-out. At the bottom we have some number of output gates, and both parties learn all the outputs.



The approach is to have 2-out-of-2 secret sharing for every wire value. The protocol proceeds layer by layer, beginning with the input layer.

1.3.1 Input Layer

First layer (input wires)				
$\underline{\mathrm{P}_1(x_1,,x_\ell)}$		$\mathrm{P}_2(y_1,,y_\ell)$		
choose $s_1,, s_{\ell} \in \{0, 1\}$		choose $r_1,, r_\ell \in \{0, 1\}$		
	$s_1,, s_\ell$			
	$r_1,, r_\ell$			
P_1 has	•	P_2 has		
P_1 's input wires,		P_2 's input wires,		
and $x_1 \oplus s_1,, x_\ell \oplus s_\ell$ (shares of P ₂ 's inputs)		and $y_1 \oplus r_1,, y_\ell \oplus r_\ell$ (shares of P ₁ 's inputs)		

1.3.2 XOR Gate

No communications are required for an XOR gate - each party can construct the shares of the output using their existing shares of the inputs.

$$\begin{array}{ccc} \gamma_3 = \gamma_1 \oplus \gamma_2 \\ \mathbf{r}_1 \oplus \mathbf{s}_1 = \gamma_1 \\ \mathbf{r}_2 \oplus \mathbf{s}_2 = \gamma_2 \end{array}$$

$$\begin{array}{ccc} \mathbf{P}_1 \text{ has } r_1, r_2 \\ \text{define } r_3 = r_1 \oplus r_2 \end{array}$$

$$\begin{array}{ccc} \mathbf{P}_2 \text{ has } s_1, s_2 \\ \text{define } s_3 = s_1 \oplus s_2 \end{array}$$

1.3.3 NOT Gate

NOT gates are easy - just agree that one player (e.g., $\mathbf{P}_1)$ flips the bit.

1.3.4 AND Gate

Each AND gate requires an invocation of OT.

		$\gamma_{ m r}$	$egin{array}{lll} {}_3 = \gamma_1 \wedge \gamma_2 \ {}_1 \oplus {}_{\mathrm{S}_1} = \gamma_1 \end{array}$	
		$\mathbf{r}_{\mathbf{c}}$	$\mathbf{s}_2 \oplus \mathbf{s}_2 = \gamma_2$	
\mathbf{P}_1 has r_1, r_2				\mathbf{P}_2 has s_1, s_2
choose $r_3 \stackrel{\$}{\leftarrow} \{0,1\}$				use 1-out-of-4 OT to select
this is P_1 's share of γ_3				appropriate row from table
	s ₁	s ₂	S3	
	0	0	$(\mathbf{r}_1 \wedge \mathbf{r}_2) \oplus \mathbf{r}_3$	
	0	1	$(r_1 \wedge \neg r_2) \oplus r_3$	
	1	0	$(\neg r_1 \wedge r_2) \oplus r_3$	
	1	1	$(\neg r_1 \land \neg r_2) \oplus r_3$	