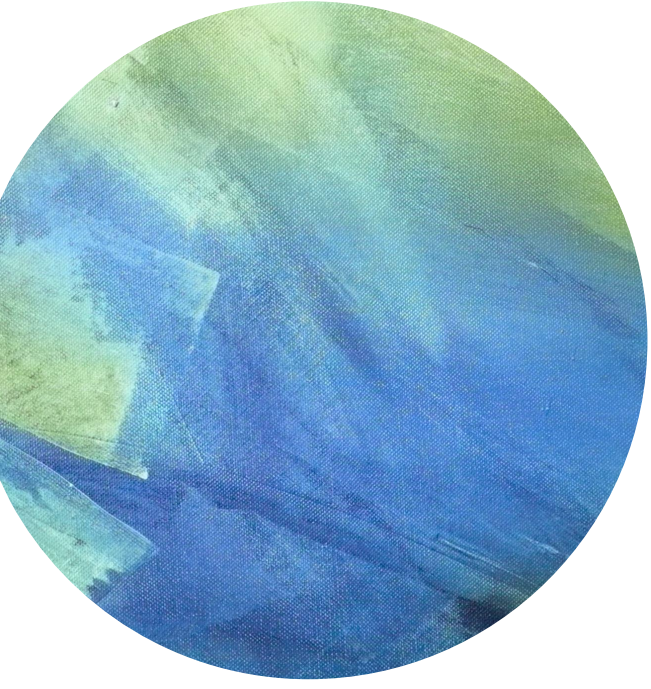
# Lecture ~~22~~ 21



# Outline

Secure Computation

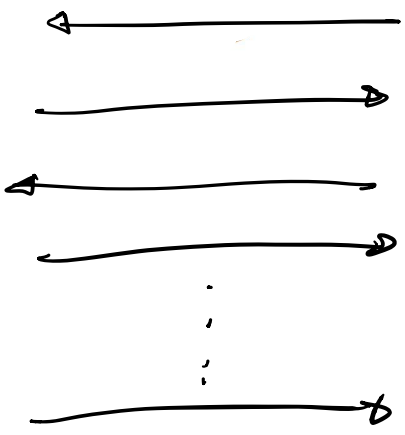
Garbled Circuits

# Oblivious Transfer

# Oblivious Transfer

S

$m_0, m_1$



R

$b$

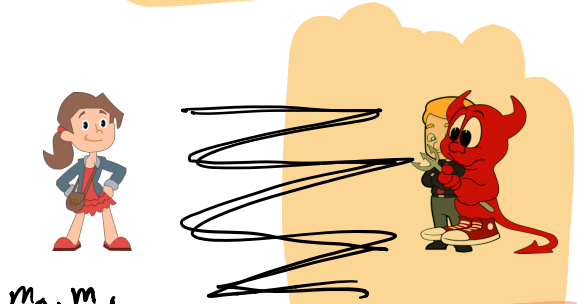
Doesn't learn b.

postprocessing : $m_b$

Doesn't learn $m_{1-b}$

# OT: Security against Malicious Receivers

"Real World/Expmt"



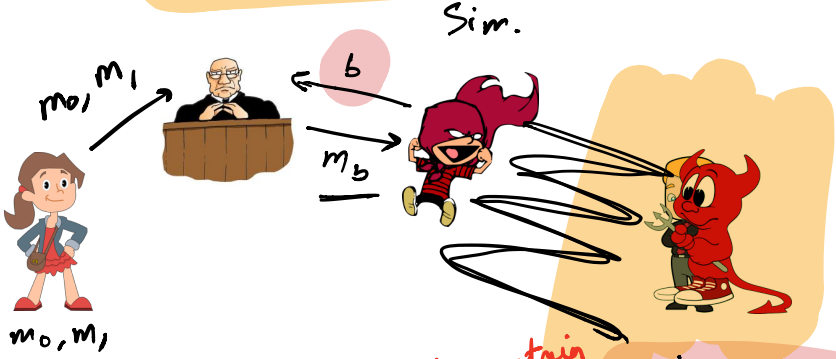

$View_{Bob}$

$= (State_{Bob})$

$+ (transcript)$

does not contain info. about $m_{1-b}$, by virtue of indistinguishability from the ideal view

"Ideal Expmt".




$View_{Bob}$.

$= (State_{Bob})$

$+ (transcript)$

cannot contain info. about $m_{1-b}$!

(1) Sim has to pretend to be Alice

2) Sim has to "find" implicit $b$ that Bob uses.

(3) Sim must generate view w/o knowledge of $m_{1-b}$

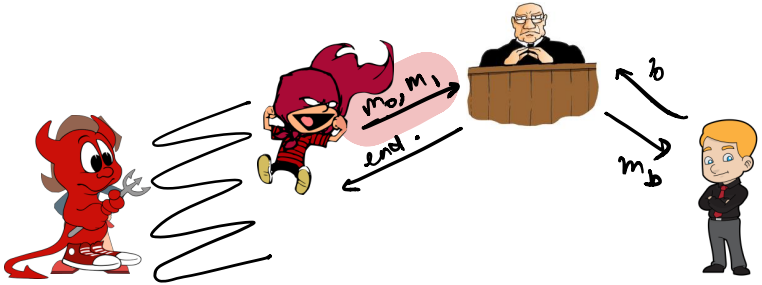
# OT: Security against Malicious Senders

"Real World"


View Alice

≈

"Ideal World"




View Alice

use PKE with pseudorandom public keys.

# Oblivious Transfer Secure Against Malicious Adversaries:

CONSTRUCTION

RANDOM ORACLE

$m_0, m_1$

$s_0, s_1$

$b$

$pk_0 = H(s_0) \oplus s_1$

$ct_0 = Enc_{pk_0}(m_0)$

$pk_1 = H(s_1) \oplus s_0$

$ct_1 = Enc_{pk_1}(m_1)$

$ct_0, ct_1$

1) Sample $(pk, sk)$

2) $r_0, r_1$ s.t. $\underline{H(r_0)} \oplus r_1 = pk$

3) If $b=0$, $s_0 = r_0$, $s_1 = r_1$

If $b=1$, $s_0 = r_1$, $s_1 = r_0$

[2019]

# Oblivious Transfer Secure Against Malicious Adversaries:

CONSTRUCTION

[PKE with pseudorandom public keys]

RANDOM ORACLE

Bob: Has input $b$

$\xleftarrow{s_0, s_1}$

1) Sample $(pk, sk) \leftarrow Gen$

$pk_0 = H(s_0) \oplus s_1$

$pk_1 = H(s_1) \oplus s_0$

$ct_0 = Enc_{pk_0}(m_0)$

$ct_1 = Enc_{pk_1}(m_1)$

$\xrightarrow{ct_0, ct_1}$

$\{m_b \leftarrow Dec_{sk}(ct_b)\}$

3) Sample $r_0 \xleftarrow{\$} \{0,1\}^n$

Compute $H(r_0)$.

4) Set $r_1 = H(r_0) \oplus pk$

Note: $pk = r_1 \oplus H(r_0)$

5) Set $pk' = r_0 \oplus H(r_1)$

Bob doesn't know $sk'$.

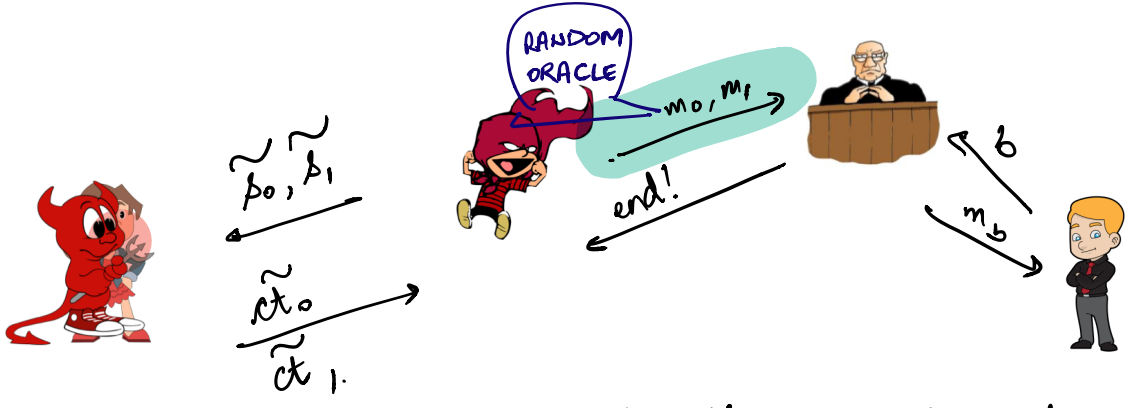
6) If $b = 0$

$s_0 = r_0$, $s_1 = r_1$

If $b = 1$,

$s_1 = r_0$, $s_0 = r_1$

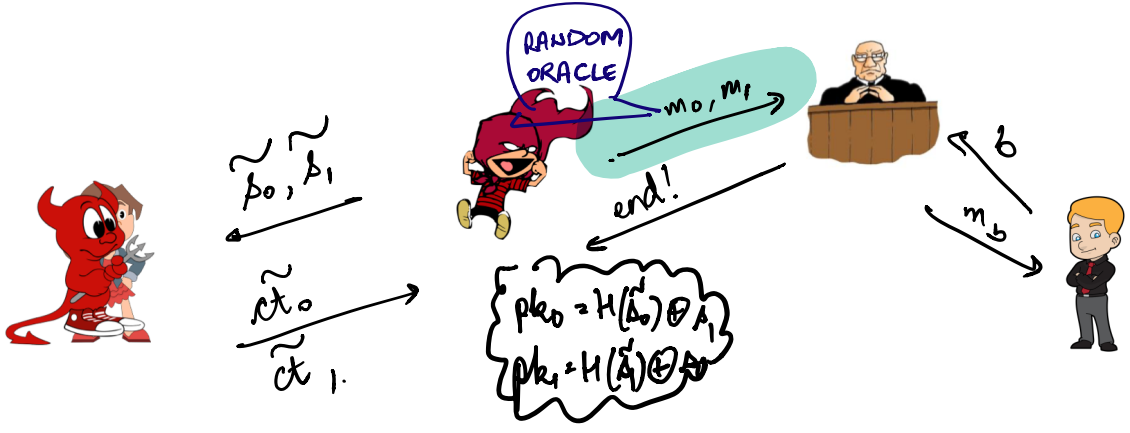
# Oblivious Transfer Secure Against Malicious SENDERS




Simulator's goal: Sample $\tilde{s}_0, \tilde{s}_1$ s.t. they know $(sk_0, sk_1)$ for $(pk_0, pk_1)$ where $pk_0 = H(\tilde{s}_0) \oplus \tilde{s}_1$, $pk_1 = H(\tilde{s}_1) \oplus \tilde{s}_0$.

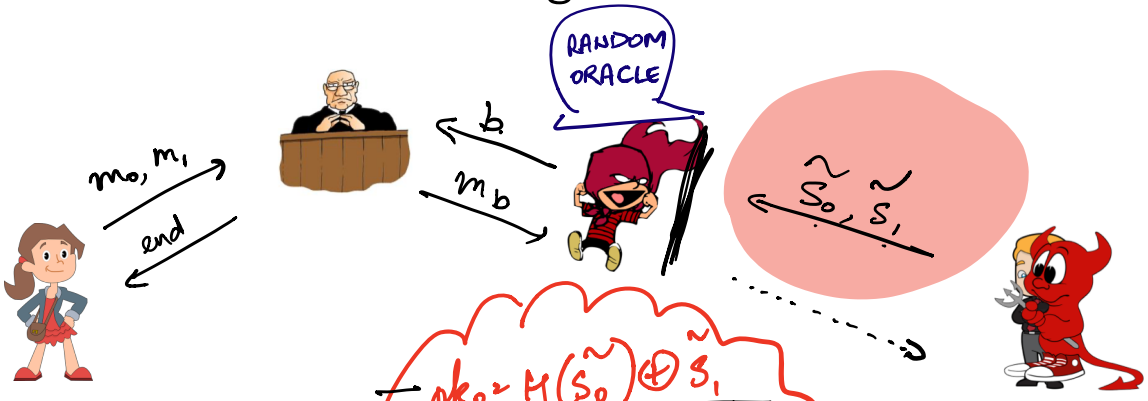
* Sim will first sample $(pk_0, sk_0) \leftarrow Gen$, $(pk_1, sk_1) \leftarrow Gen$. Sample $(\tilde{s}_0, \tilde{s}_1)$. Set $H(\tilde{s}_0) = pk_0 \oplus \tilde{s}_1$, $\tilde{s}_1 = pk_1 \oplus \tilde{s}_0$

# Oblivious Transfer Secure Against Malicious SENDERS



# Oblivious Transfer Secure Against Malicious RECEIVERS

# Oblivious Transfer Secure Against Malicious RECEIVERS



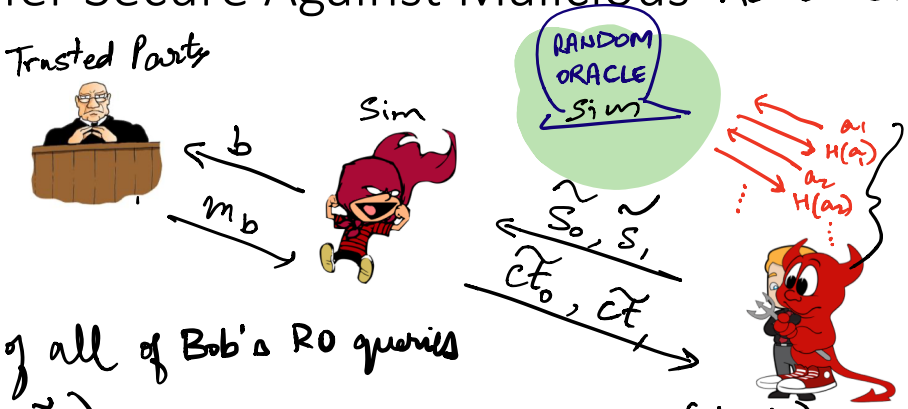

$pk_0 = H(\tilde{s}_0) \oplus \tilde{s}_1$

$pk_1 = H(\tilde{s}_1) \oplus \tilde{s}_0$

Adv knows $sk_b$.

what is $b$?..

# Oblivious Transfer Secure Against Malicious RECEIVERS




Sim's algorithm:

1) Keep an ordered list of all of Bob's RO queries

2) On receiving $(\tilde{s}_0, \tilde{s}_1)$

a) Both $(\tilde{s}_0, \tilde{s}_1)$ were not queried. Bob learns nothing!

b) Only $\tilde{s}_0$ was queried. Bob potentially knows sk for $pk = H(\tilde{s}_0) \oplus \tilde{s}_1$.

c) only $\tilde{s}_1$ was queried. Bob pot. knows sk for $pk = H(\tilde{s}_1) \oplus \tilde{s}_0$.

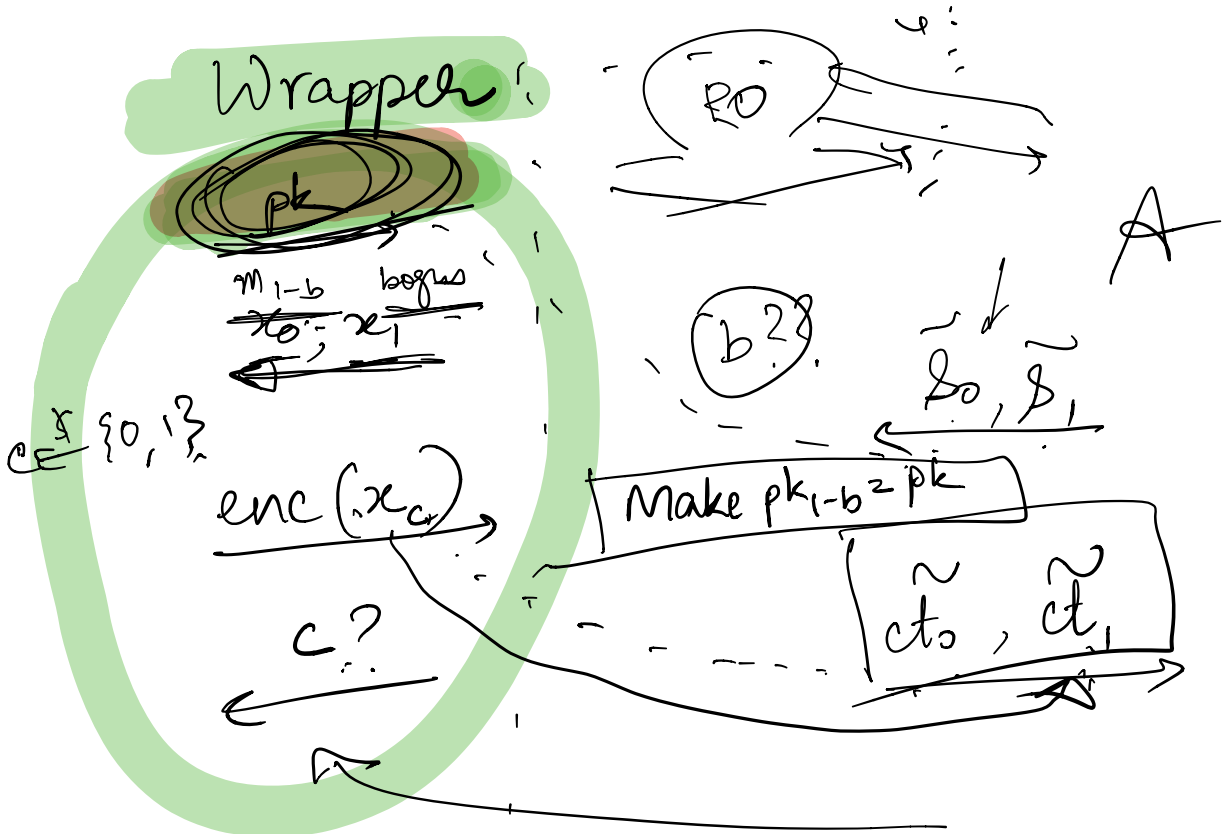
d) Both were queried. Find which was queried first.

If $\tilde{s}_0$ was queried first. If $\tilde{s}_1$ was queried first.

$(pk, sk)$

$r_0, r_1$ s.t. $pk = H(r_0) \oplus r_1$

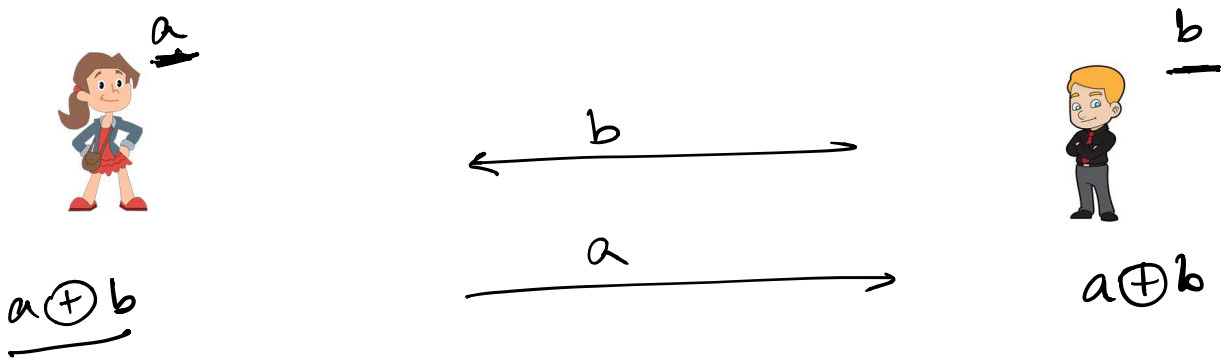
* First, sample $r_0$
* Query RO to obtain $H(r_0)$
* Set $r_1 = pk \oplus H(r_0)$




Given RO queries and $\tilde{s}_0$ and $\tilde{s}_1$, Figure out what $b$ is.

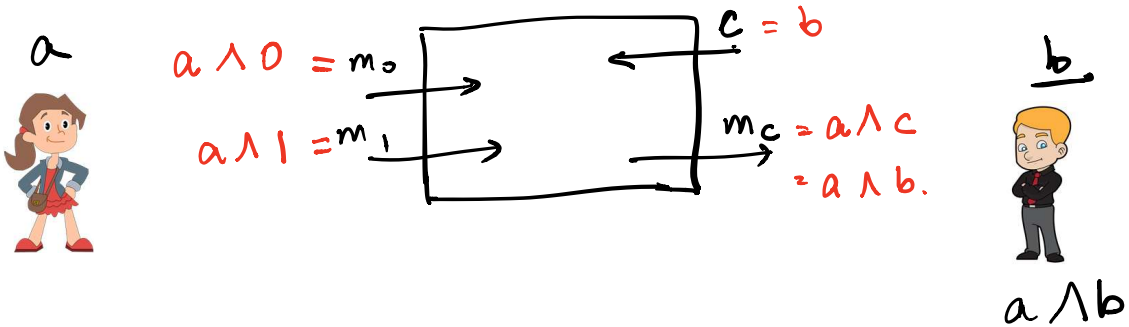
# Secure XOR

$f = \text{XOR}$



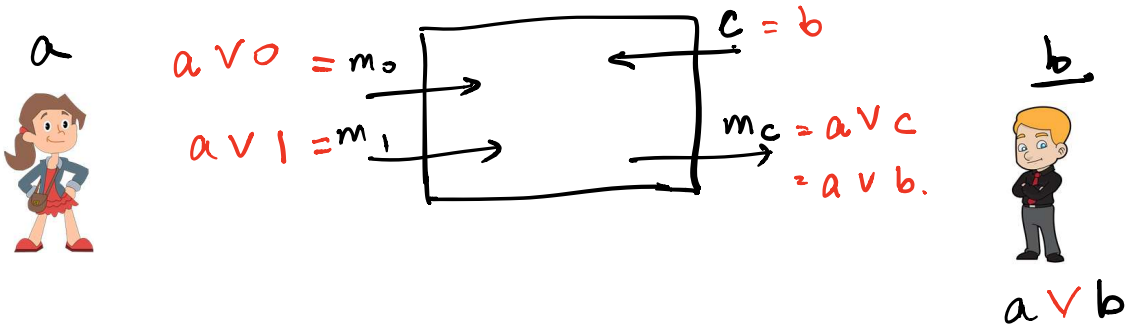
# Secure AND Computation from OT

| $a$ | $b$ | $a \wedge b$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |



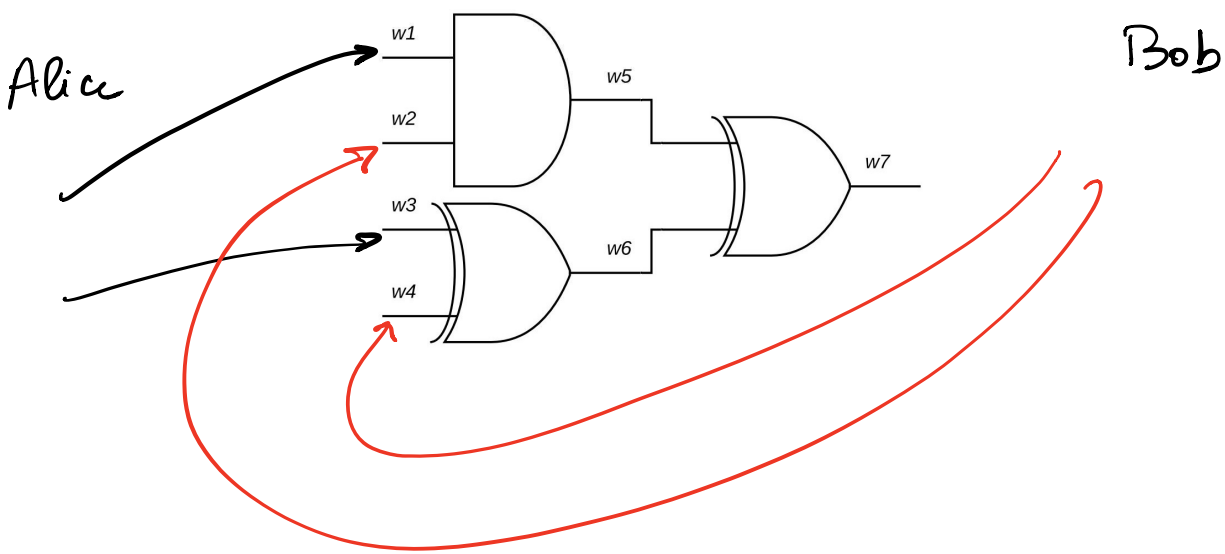
# Secure ~~AND~~ OR Computation from OT

| $a$ | $b$ | $a \vee b$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

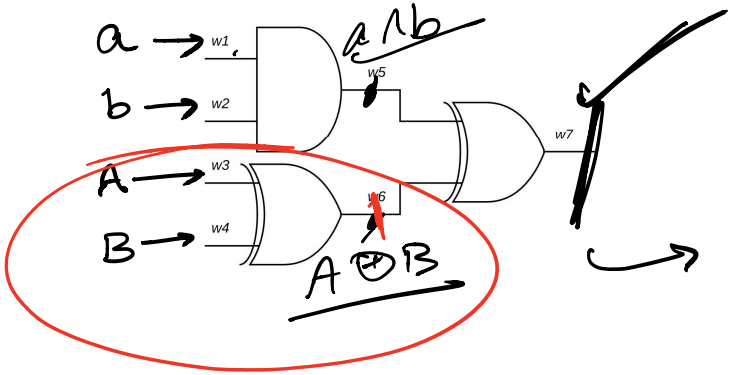


# How about more general circuits?

Boolean Circuit (informal): a collection of logical gates that are connected by wires, where each wire represents a particular boolean value. Inputs to the circuit are placed onto input wires, these wires are fed through a series of logic gates, and eventually one or more outputs are produced.



# General circuits?




# Secret Sharing (2 out of 2)

$a$

$\downarrow$

$(a_1, a_2)$

$SS(\underline{a}) = (\overset{\text{random}}{a_1, a_2})$ s.t. $a_1 \oplus a_2 = a$.

[Shamir's Secret Sharing].

$a \longrightarrow (a_1, \ldots, a_n)$ s.t. <u>any</u> $t$-sized subset determines $a$.

~~XOR on secret shared data~~

Alice & Bob distributed shares of inputs

$a \xrightarrow{ss} (a_1, a_2)$

$$\begin{pmatrix} a_1 \\ b_1 \end{pmatrix}$$

$a_2 \longrightarrow$

$\longleftarrow b_1$

$b \xrightarrow{ss} (b_1, b_2)$

$$\begin{pmatrix} a_2 \\ b_2 \end{pmatrix}$$

# XOR on secret-shared data

$a$

$\begin{pmatrix} a_1 \\ b_1 \end{pmatrix}$

$f = \text{XOR}$

$b$

$\begin{pmatrix} a_2 \\ b_2 \end{pmatrix}$

$d_1 = a_1 \oplus b_1$

END UP WITH

SHARES OF $f(a,b) = a \oplus b$.

$d_1, d_2$ s.t. $d_1 \oplus d_2 = a \oplus b$.

$d_2 = a_2 \oplus b_2$

# AND on secret-shared data

$a_1, a_2$

$(a_1, b_1)$

$(a_1 \wedge b_1)$

$d_1$

$b_1, b_2$

$(a_2, b_2)$

$(a_2 \wedge b_2)$

$d_2$

s.t. $d_1 \oplus d_2 = (a \wedge b) = (a_1 \oplus a_2) \wedge (b_1 \oplus b_2)$

$= \cancel{(a_1 \wedge b_1)} \oplus (a_1 \wedge b_2) \oplus (a_2 \wedge b_1) \oplus \cancel{(a_2 \wedge b_2)}$

# ~~AND~~ NOT on secret-shared data

$c_1$

$d_1$

$c_2$

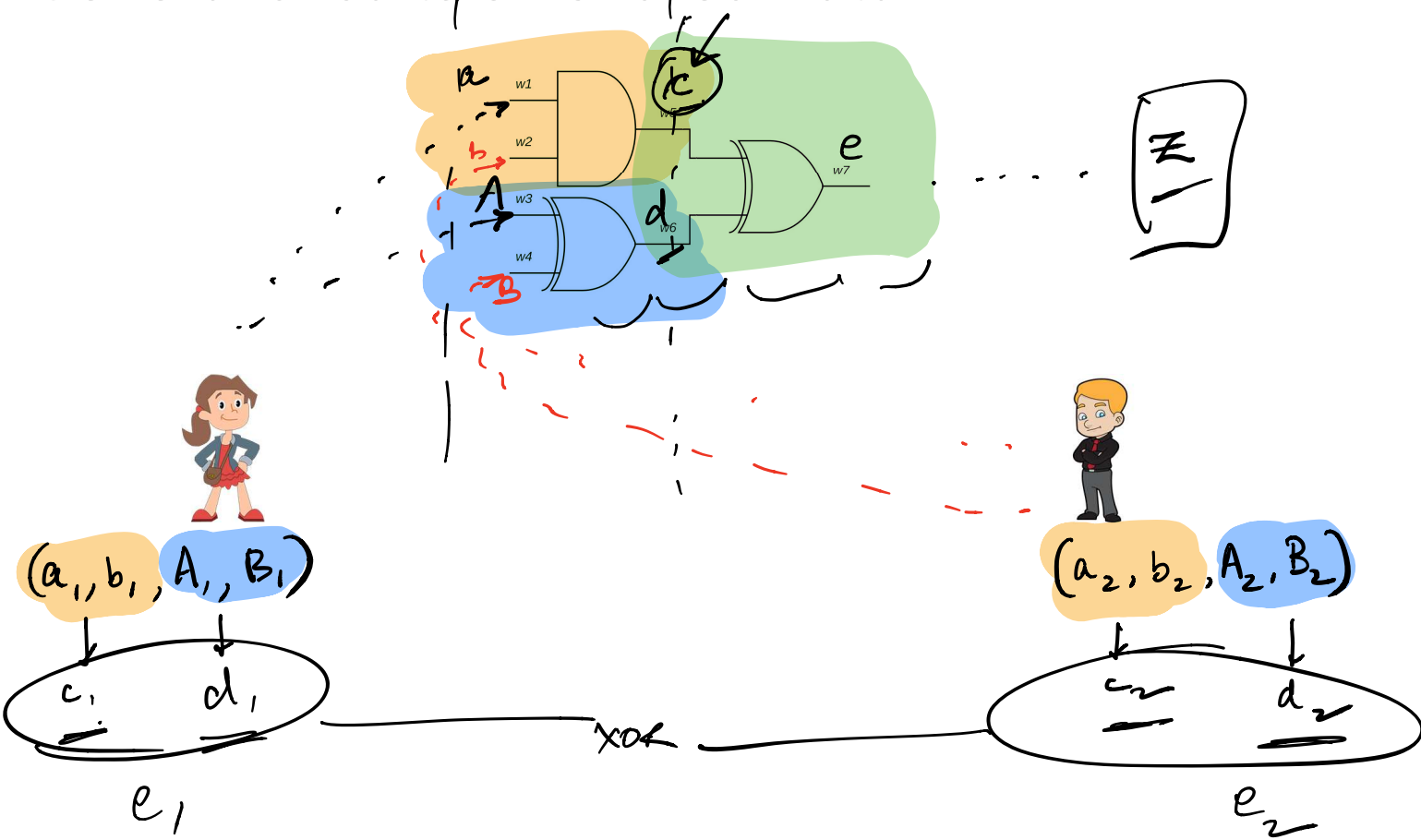
$d_2$

(Alice, Bob) have $(c_1, c_2)$ as S.S. of $c$.

End up with $(d_1, d_2)$ as S.S. of $1-c$.

One of them (Bob) compute $(1-d_2)$.

# General circuits on Shared Data



| | | | | |
|---|---|---|---|---|
| a | 0 | 0 | 1 | 1 |
| b | 0 | 1 | 0 | 1 |
| | 00 | 01 | 01 | 10 |