CSCI 1510

This Lecture:

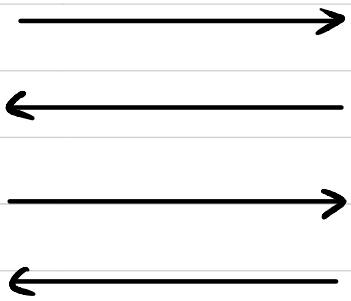
- Oblivious Transfer (continued)
- Semi-Honest MPC for Any Function (GMW)
- Malicious MPC (GMW Compiler)
- Definition of Fully Homomorphic Encryption

Oblivious Transfer (OT)

Sender

Receiver

Input: $m_0, m_1 \in \{0,1\}^\ell$



Input: $c \in \{0,1\}$

Output: $\perp$

Output: m_c

Oblivious Transfer (OT)

Cyclic group G of order q with generator g

$H: G \to \{0,1\}^l$

Sender		Receiver
Input: $m_0, m_1 \in \{0,1\}^l$		Input: $c \in \{0,1\}$
$a \xleftarrow{\$} \mathbb{Z}_q$	$\xrightarrow{A = g^a}$	$b \xleftarrow{\$} \mathbb{Z}_q$
	$\xleftarrow{B = g^b \cdot A^c}$	
$k_0 := H(B^a)$ $k_1 := H((\frac{B}{A})^a)$		
	$\xrightarrow{\begin{array}{l} ct_0 := k_0 \oplus m_0 \\ ct_1 := k_1 \oplus m_1 \end{array}}$	
		Output: $m_c := ct_c \oplus H(A^b)$

Thm If CDH is hard in G and H is modeled as a random oracle, then this protocol is semi-honest secure.

$S_B(1^n, c, m_c)$

Receiver

Input: $c \in \{0,1\}$

$a \xleftarrow{\$} \mathbb{Z}_q$

$A = g^a$ →

$b \xleftarrow{\$} \mathbb{Z}_q$

← $B = g^b \cdot A^c$

$k_c := H(g^{ab})$

$ct_c := k_c \oplus m_c$

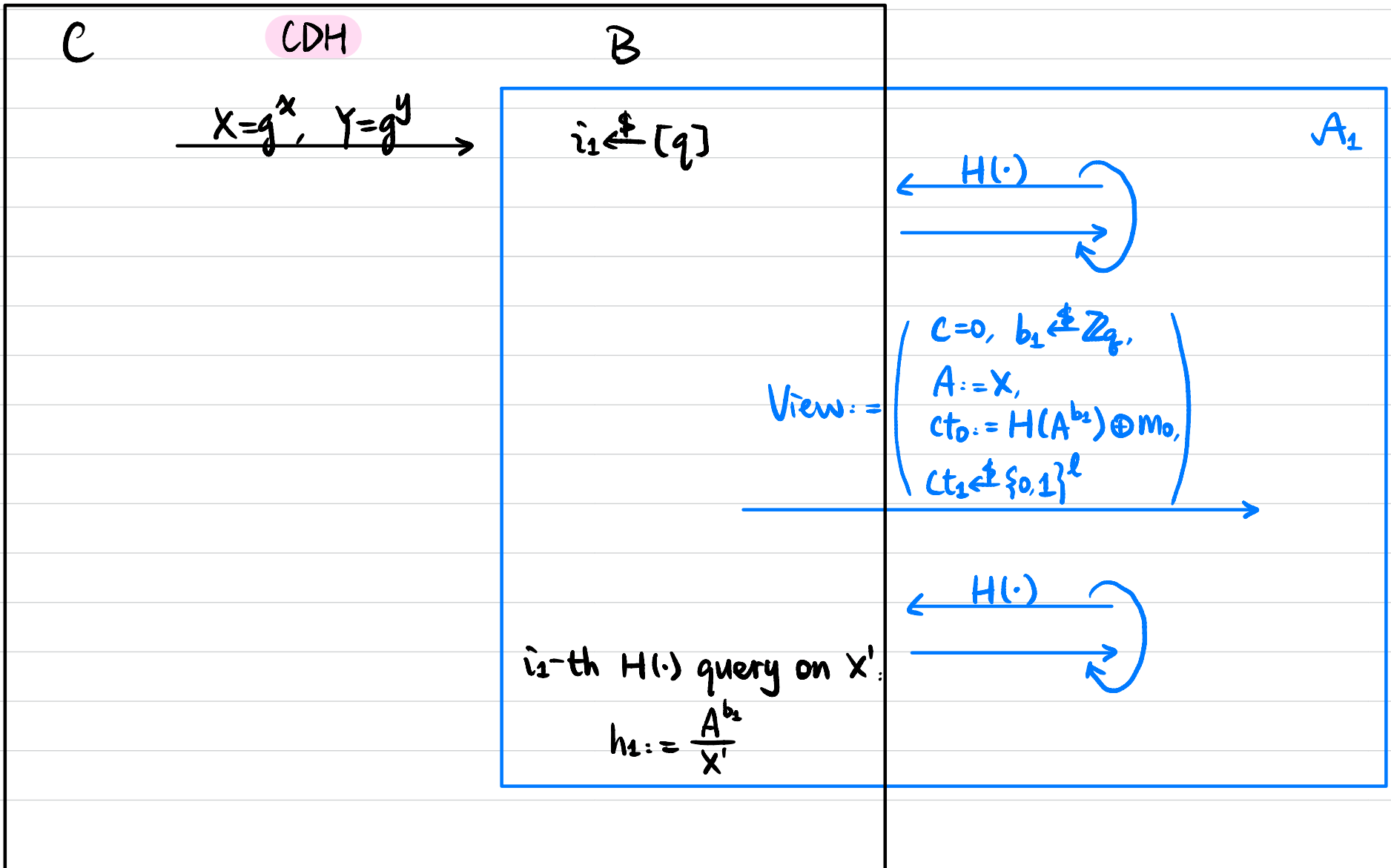
$ct_{1-c} \xleftarrow{\$} \{0,1\}^\ell$ →

Output: $m_c := ct_c \oplus H(A^b)$

$S_B(1^n, c, m_c) \simeq \text{View}_R^{\Pi}((m_0, m_1), c, n)$

Assume $\exists$ PPT A that can distinguish. A must be querying $H(g^{ab-a^2})$ when $c=0$ or $H(g^{ab+a^2})$ when $c=1$ with non-negligible probability. WLOG assume $c=0$.

We construct PPT B to break CDH in the random oracle model.



B (continued)

A_2

$i_2 \xleftarrow{\$} [q]$

$H(\cdot)$

$\text{View} := \left(\begin{array}{l} c=0,\ b_2 \xleftarrow{\$} \mathbb{Z}_q, \\ A := Y, \\ ct_0 := H(A^{b_2}) \oplus m_0, \\ ct_1 \xleftarrow{\$} \{0,1\}^l \end{array} \right)$

$H(\cdot)$

i_2-th $H(\cdot)$ query on Y':

$h_2 := \frac{A^{b_2}}{Y'}$

A_3

$i_3 \xleftarrow{\$} [q]$

$r \xleftarrow{\$} \mathbb{Z}_q$

$H(\cdot)$

$\text{View} := \left(\begin{array}{l} c=0,\ b_3 \xleftarrow{\$} \mathbb{Z}_q, \\ A := (X \cdot Y)^r, \\ ct_0 := H(A^{b_3}) \oplus m_0, \\ ct_1 \xleftarrow{\$} \{0,1\}^l \end{array} \right)$

$H(\cdot)$

i_3-th $H(\cdot)$ query on Z':

$h_3 := \left(\frac{A^{b_3}}{Z'} \right)^{(r^2)^{-1}}$

Output $\left(\frac{h_3}{h_1 \cdot h_2} \right)^{2^{-1}}$

Feasibility Results

Computational Security:

semi-honest Oblivious Transfer (OT)

semi-honest MPC for any function with $t < n$ (# corrupted parties)

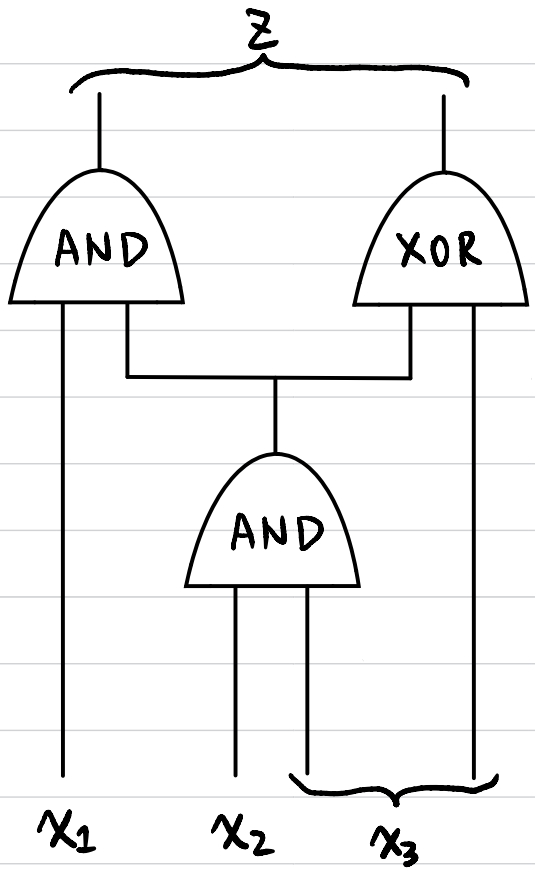
malicious MPC for any function with $t < n$

Information-Theoretic (IT) Security:

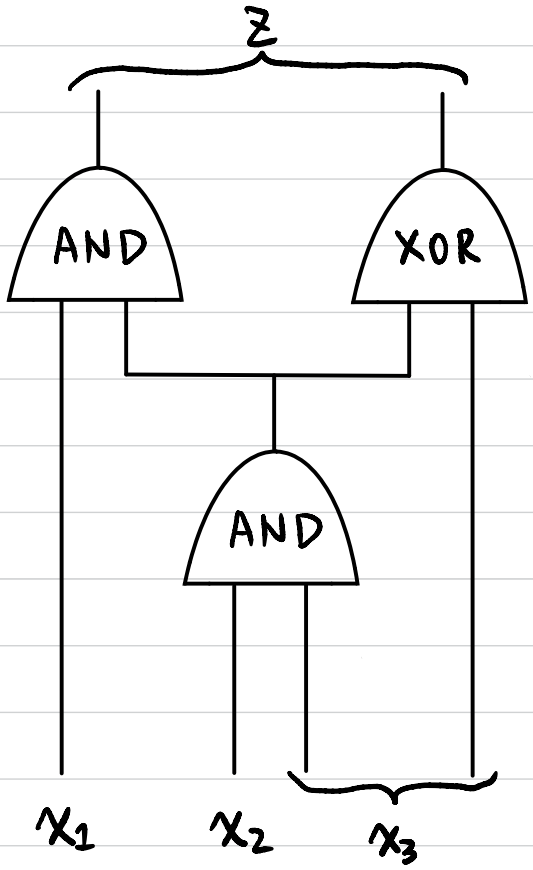
semi-honest/malicious MPC for any function with $t < n/2$ (honest majority)

necessary

Arbitrary Function → Represent it as a Boolean circuit



MPC for any function with $t \leq n-1$ (GMW)



Throughout the protocol, we keep the invariant:

For each wire w:

If the value of the wire is $v^w \in \{0,1\}$,

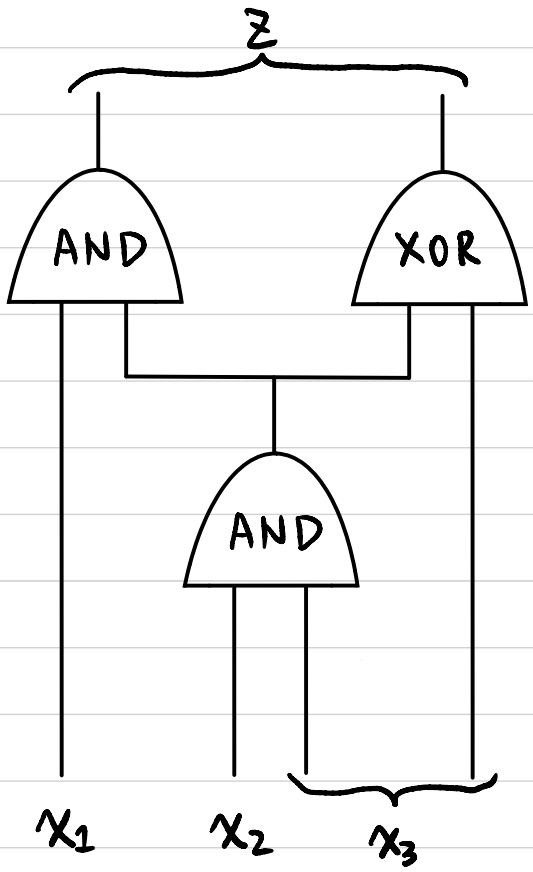
then the n parties hold an additive secret share of v^w

Each party P_i holds a random share $v_i^w \in \{0,1\}$ s.t.

$$\bigoplus_{i=1}^{n} v_i^w = v^w$$

Any $(n-1)$ shares information theoretically hide v^w.

MPC for any function with $t \le n-1$ (GMW)



Each party P_i holds a random share $v_i^w \in \{0,1\}$ s.t. $\bigoplus_{i=1}^{n} v_i^w = v^w$

Inputs:

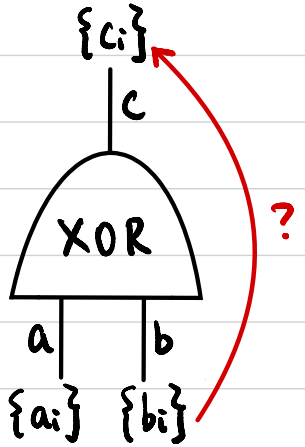
For each input wire w:

If it's from party P_k with input value $v^w \in \{0,1\}$.

P_k randomly samples $v_i^w \xleftarrow{\$} \{0,1\}$ s.t. $\bigoplus_{i=1}^{n} v_i^w = v^w$

↳ Sends v_i^w to party P_i.

XOR gates:

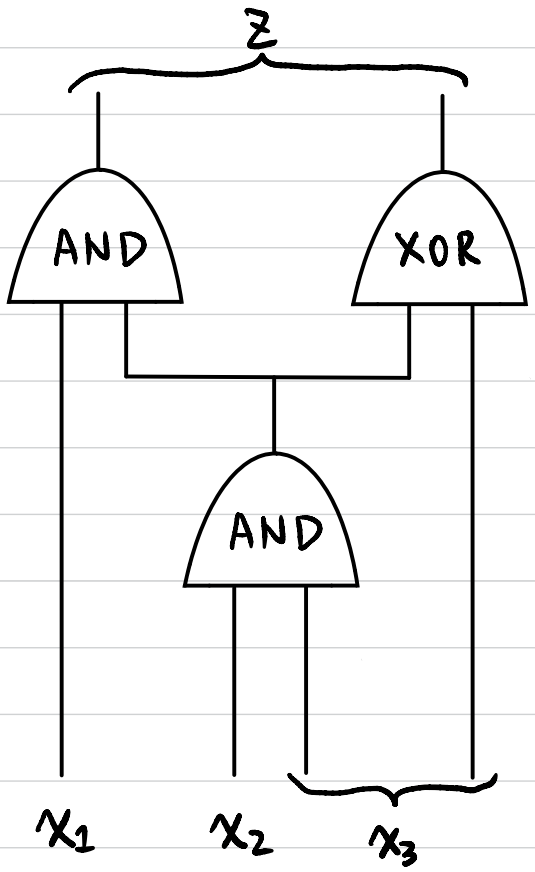


GIVEN: $\bigoplus_{i=1}^{n} a_i = a$ $\quad \bigoplus_{i=1}^{n} b_i = b$

WANT: $\{c_i\}$ s.t. $\bigoplus_{i=1}^{n} c_i = c = a \oplus b$

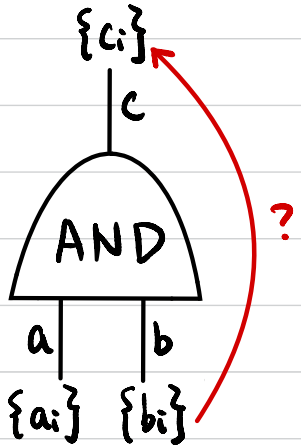
$c_i = ?$

MPC for any function with $t \leq n-1$ (GMW)



Each party P_i holds a random share $v_i^w \in \{0,1\}$ s.t. $\bigoplus_{i=1}^{n} v_i^w = v^w$

AND gates:



GIVEN: $\bigoplus_{i=1}^{n} a_i = a$ $\bigoplus_{i=1}^{n} b_i = b$

WANT: $\{c_i\}$ s.t. $\bigoplus_{i=1}^{n} c_i = c = a \cdot b$

$c_i = ?$

Outputs:

For each output wire w:

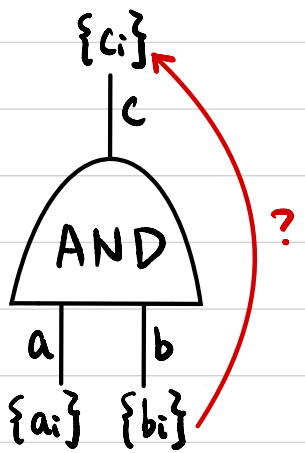
Each party P_i holds a random share $v_i^w \in \{0,1\}$

→ Sends v_i^w to all parties

Each party computes the value $v^w = \bigoplus_{i=1}^{n} v_i^w$

MPC for any function with $t \leq n-1$ (GMW)

AND gates:



GIVEN: $\bigoplus_{i=1}^{n} a_i = a$ $\quad \bigoplus_{i=1}^{n} b_i = b$

WANT: $\{c_i\}$ s.t. $\bigoplus_{i=1}^{n} c_i = c = a \cdot b$

$c_i = ?$

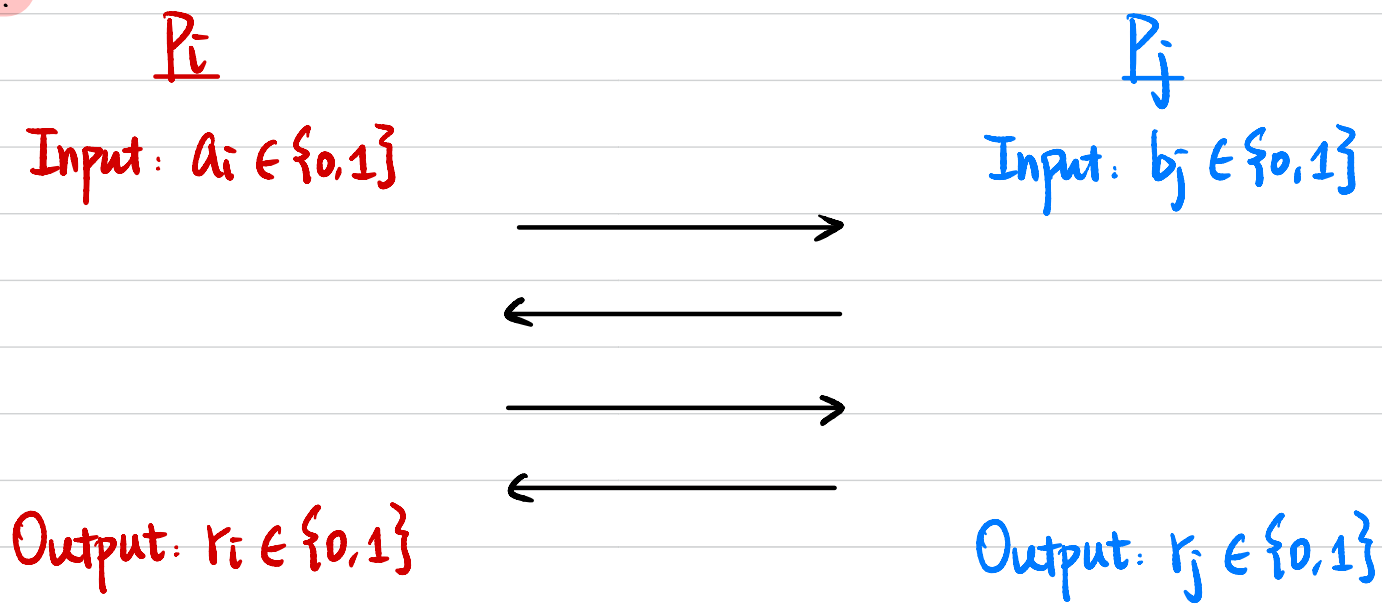
$$a \cdot b = \left(\sum_{i=1}^{n} a_i\right) \cdot \left(\sum_{i=1}^{n} b_i\right) \mod 2$$

$$= \left(\sum_{i=1}^{n} a_i \cdot b_i\right) + \left(\sum_{i \neq j} a_i \cdot b_j\right) \mod 2$$

P_i locally $\quad$?

MPC for any function with $t \leq n-1$ (GMW)

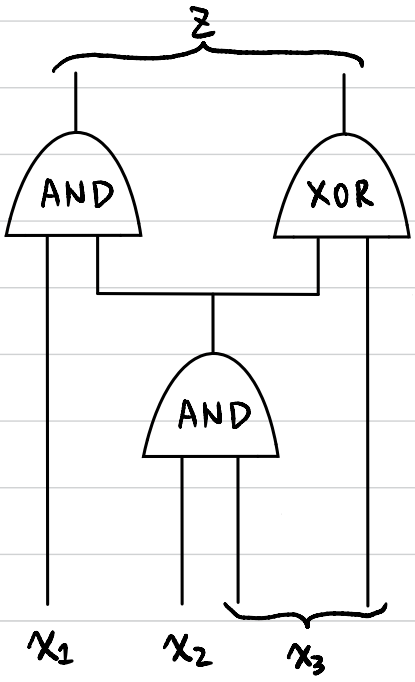
Reshare:



WANT: Random $r_i, r_j \in \{0,1\}$ s.t. $r_i \oplus r_j = a_i \cdot b_j$

1) P_i randomly samples $r_i \xleftarrow{\$} \{0,1\}$
2) How to let P_j learn r_j s.t. $r_i \oplus r_j = a_i \cdot b_j$?

MPC for any function with $t \leq n-1$ (GMW)



Each party P_i holds a random share $v_i^w \in \{0,1\}$ s.t. $\bigoplus_{i=1}^{n} v_i^w = v^w$

Inputs:

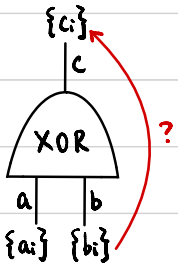
For each input wire w:

If it's from party P_k with input value $v^w \in \{0,1\}$,

P_k randomly samples $v_i^w \xleftarrow{\$} \{0,1\}$ s.t. $\bigoplus_{i=1}^{n} v_i^w = v^w$

↳ Sends v_i^w to party P_i.

XOR gates:

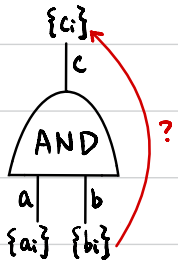


GIVEN: $\bigoplus_{i=1}^{n} a_i = a$ $\bigoplus_{i=1}^{n} b_i = b$

WANT: $\{c_i\}$ s.t. $\bigoplus_{i=1}^{n} c_i = c = a \oplus b$

$c_i = a_i \oplus b_i$

AND gates:



GIVEN: $\bigoplus_{i=1}^{n} a_i = a$ $\bigoplus_{i=1}^{n} b_i = b$

WANT: $\{c_i\}$ s.t. $\bigoplus_{i=1}^{n} c_i = c = a \cdot b$

$c_i = ?$

$$a \cdot b = \left(\sum_{i=1}^{n} a_i\right) \cdot \left(\sum_{i=1}^{n} b_i\right) \mod 2$$

$$= \left(\sum_{i=1}^{n} a_i \cdot b_i\right) + \left(\sum_{i \neq j} a_i \cdot b_j\right) \mod 2$$

P_i locally — Reshare

Outputs:

For each output wire w:

Each party P_i holds a random share $v_i^w \in \{0,1\}$

↳ Sends v_i^w to all parties

Each party computes the value $v^w = \bigoplus_{i=1}^{n} v_i^w$

GMW Compiler

Given a semi-honest protocol:

Once inputs & randomness are fixed, protocol is deterministic.

Step 1: Each party P_i commits to its input x_i & randomness r_i to be used in the semi-honest protocol.

Step 2: Run semi-honest protocol.

Along with every message, prove in ZK that the message is computed correctly (based on its input, randomness, transcript so far)

Homomorphic Properties of Encryption Schemes

Multiplicatively Homomorphic

$$Enc(m_1) \rightarrow Enc(m_1 \cdot m_2) \leftarrow Enc(m_2)$$

Additively Homomorphic

$$Enc(m_1) \rightarrow Enc(m_1 + m_2) \leftarrow Enc(m_2)$$

ElGamal:

$$C_1 = (g^{r_1},\ h^{r_1} \cdot m_1)$$
$$C_2 = (g^{r_2},\ h^{r_2} \cdot m_2)$$
$$\rightarrow \left(g^{r_1+r_2},\ h^{r_1+r_2} \cdot (m_1 \cdot m_2)\right)$$

Exponential ElGamal:

$$Enc(m) = (g^r,\ h^r \cdot g^m)$$
$$C_1 = (g^{r_1},\ h^{r_1} \cdot g^{m_1})$$
$$C_2 = (g^{r_2},\ h^{r_2} \cdot g^{m_2})$$
$$\rightarrow \left(g^{r_1+r_2},\ h^{r_1+r_2} \cdot g^{m_1+m_2}\right)$$

Regev:

$$C_1 = \left(r_1^T \cdot A,\ r_1^T \cdot b + \mu_1 \cdot \lfloor \tfrac{q}{2} \rfloor\right)$$
$$C_2 = \left(r_2^T \cdot A,\ r_2^T \cdot b + \mu_2 \cdot \lfloor \tfrac{q}{2} \rfloor\right)$$
$$\Downarrow$$
$$\left((r_1+r_2)^T \cdot A,\ (r_1+r_2)^T \cdot b + (\mu_1+\mu_2) \cdot \lfloor \tfrac{q}{2} \rfloor\right)$$

Fully Homomorphic: Additively & Multiplicatively Homomorphic

Application: Privacy-Preserving Query

Server

Client

Input x

Key sk

$ct \leftarrow Enc(x)$

Search / ML / GPT / ⋯

$ct' \leftarrow Eval(f, ct)$

$f(x) \leftarrow Dec_{sk}(ct')$

Is it possible?

- Question was asked back in 1978
- Big breakthough in 2009 (Gentry)
  - Complicated construction
  - Non-standard assumptions
- By now: much simpler constructions from standard assumptions.

Fully Homomorphic Encryption (FHE)

- **Syntax:** A (public-key) homomorphic encryption scheme $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ w.r.t. function family $\mathcal{F}$:
  - $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$
  - $ct \leftarrow \mathsf{Enc}_{pk}(m) \quad m \in \{0,1\}$
  - $m \leftarrow \mathsf{Dec}_{sk}(ct)$
  - $ct_f \leftarrow \mathsf{Eval}(f, ct_1, \cdots, ct_k) \quad f: \{0,1\}^k \to \{0,1\}$

- **Correctness:** $\forall f \in \mathcal{F}, \quad \forall m_1, m_2, \cdots, m_k \in \{0,1\}$

$$\Pr[\mathsf{Dec}_{sk}(ct_f) = f(m_1, \cdots, m_k)] \geq 1 - \mathsf{negl}(n)$$

where $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$, $ct_i \leftarrow \mathsf{Enc}_{pk}(m_i) \quad \forall i \in [k]$,

$ct_f \leftarrow \mathsf{Eval}(f, ct_1, \cdots, ct_k)$.

- **Succinctness:** $|ct_f| \leq$ fixed $\mathrm{poly}(n)$

  Independent of circuit size of f.

- **CPA/CCA Security?**

Fully Homomorphic Encryption (FHE)

- **Syntax:** A (public-key) homomorphic encryption scheme $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ w.r.t. function family $\mathcal{F}$:
  - $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$
  - $ct \leftarrow \mathsf{Enc}_{pk}(m) \quad m \in \{0,1\}$
  - $m \leftarrow \mathsf{Dec}_{sk}(ct)$
  - $ct_f \leftarrow \mathsf{Eval}(f, ct_1, \cdots, ct_k) \quad f: \{0,1\}^k \to \{0,1\}$

- If $\mathcal{F}$ is the set of all poly-sized Boolean circuits, then Π is fully homomorphic.