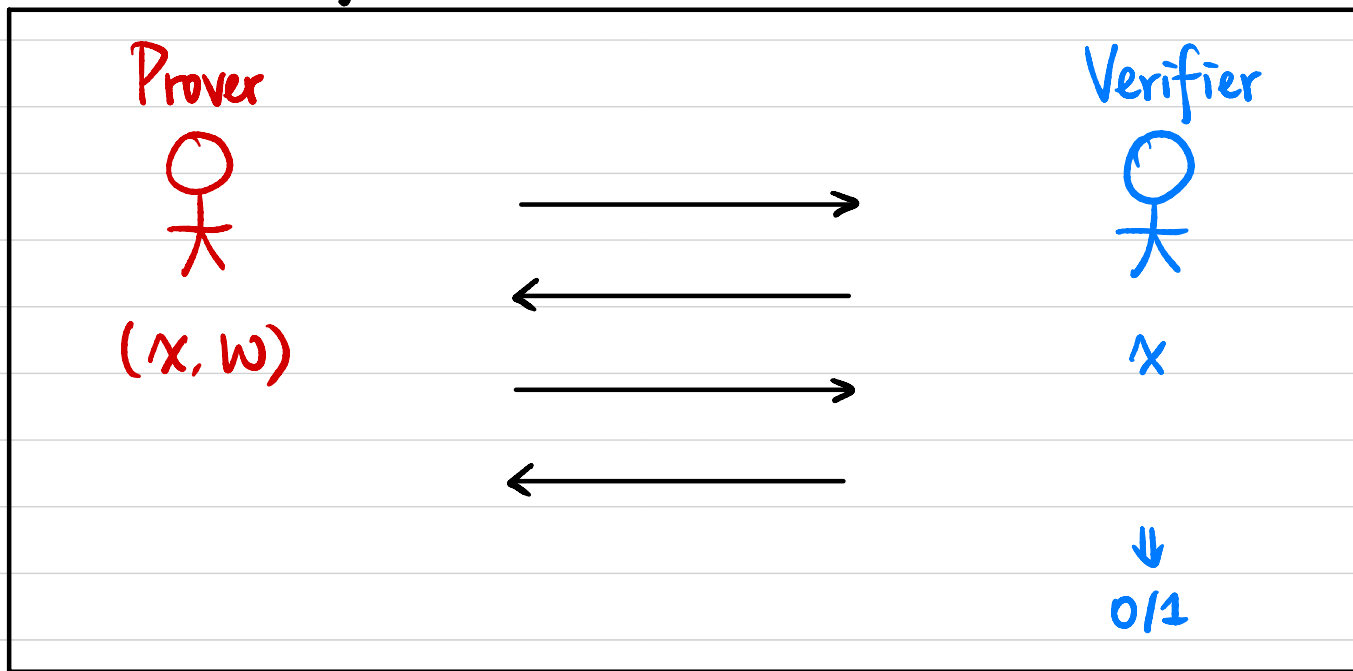


CSCI 1510

This Lecture:

- ZKP for All NP (continued)
- Non-Interactive Zero-Knowledge Proofs
- Definitions of Secure Multi-Party Computation

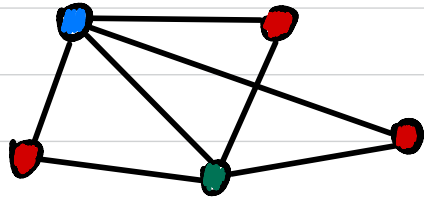
Zero-Knowledge Proof (ZKP)



Let (P, V) be a pair of PPT interactive machines. (P, V) is a **zero-knowledge proof system** for a language L with associated relation R_L if

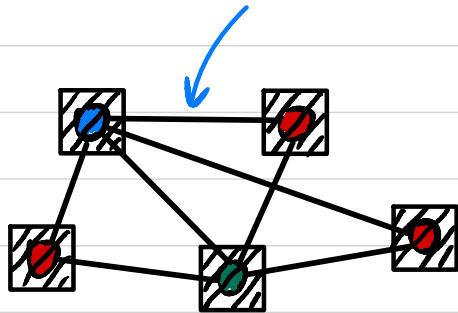
- **Completeness:** $\forall (x, w) \in R_L, \Pr [P(x, w) \longleftrightarrow V(x) \text{ outputs } 1] = 1.$
- **Soundness:** $\forall x \notin L, \forall p^*, \Pr [p^*(x) \longleftrightarrow V(x) \text{ outputs } 1] \leq \text{negl}(n).$
- **Zero-Knowledge:** $\forall \text{PPT } V^*, \exists \text{PPT } S \text{ s.t. } \forall (x, w) \in R_L, \text{Output}_{V^*}[P(x, w) \longleftrightarrow V^*(x)] \simeq S(x)$

ZKP for Graph 3-Coloring (All NP)



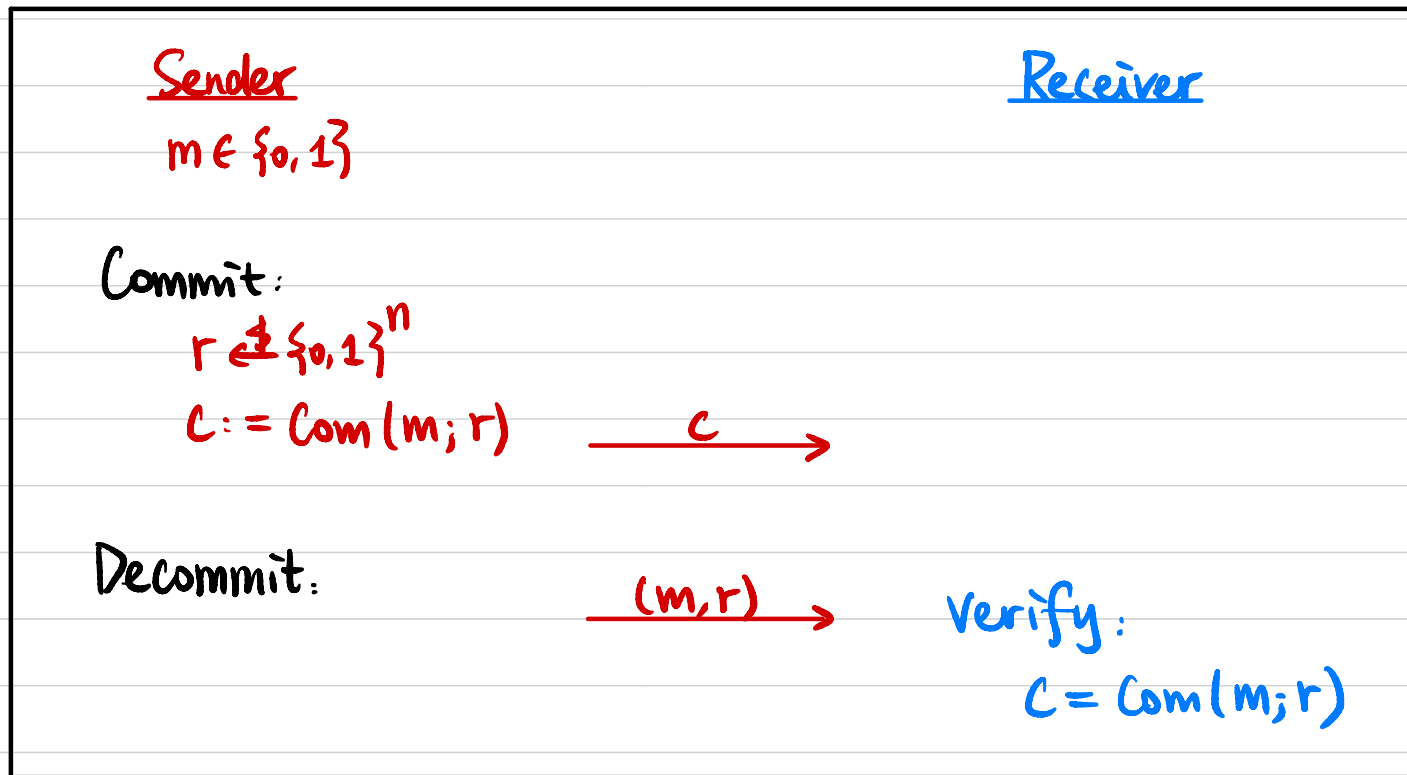
NP language $L = \{ G : G \text{ has 3-coloring} \}$

NP relation $R_L = \{ (G, \text{3COL}) \}$

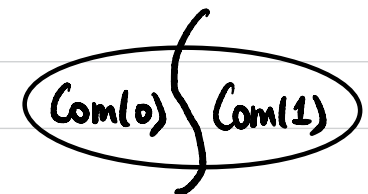


$$\pi : \{ \bullet \bullet \bullet \} \rightarrow \{ \bullet \bullet \bullet \}$$

Commitment Scheme



- **Perfectly Binding:** $\forall r, s \in \{0, 1\}^n, \text{Com}(0; r) \neq \text{Com}(1; s)$



- **Computationally Hiding:** $\text{Com}(0; U_n) \stackrel{c}{\approx} \text{Com}(1; U_n)$

ZKP for Graph 3-Coloring

Input: $G = (V, E)$

Witness: $\phi: V \rightarrow \{0, 1, 2\}$

Given a perfectly binding commitment scheme Com .

Soundness?

$G \notin L$, by perfect binding of Com ,

$$\Pr[P^* \text{ be caught}] \geq \frac{1}{|E|}$$

$$\Pr[V \text{ outputs 1}] \leq \left(1 - \frac{1}{|E|}\right)^{n \cdot |E|} \approx \left(\frac{1}{e}\right)^n$$

Prover

Randomly sample $\pi: \{0, 1, 2\} \rightarrow \{0, 1, 2\}$

$\forall v \in V, r_v \leftarrow \{0, 1\}^n, c_v := \text{Com}(\pi(\phi(v)), r_v)$

$\{c_v\}_{v \in V}$

(u, v)

Reveal decommitments of c_u & c_v

$\alpha = \pi(\phi(u)), r_u$
 $\beta = \pi(\phi(v)), r_v$

$k = n \cdot |E|$

Randomly pick an edge $(u, v) \in E$

Verify: $c_u = \text{Com}(\alpha; r_u)$
 $c_v = \text{Com}(\beta; r_v)$
 $\alpha, \beta \in \{0, 1, 2\}, \alpha \neq \beta$

Completeness?

Zero-Knowledge?

$\forall$ PPT V^* , $\exists$ PPT S s.t. $\forall (x, w) \in R_L$,
 $\text{Output}_{V^*}[P(x, w) \longleftrightarrow V^*(x)] \stackrel{c}{\approx} S(x)$

Simulator

Verifier*

$(u', v') \xleftash E$

$\alpha, \beta \xleftash \{0, 1, 2\}$ s.t. $\alpha \neq \beta$

$r_u \xleftash \{0, 1\}^n$ $C_u := \text{Com}(\alpha; r_u)$

$r_v \xleftash \{0, 1\}^n$ $C_v := \text{Com}(\beta; r_v)$

$\forall v \in V \setminus \{u', v'\}$:

$r_v \xleftash \{0, 1\}^n$, $C_v := \text{Com}(0; r_v)$

$\{C_v\}_{v \in V}$

(u, v)

If $(u, v) = (u', v')$:

Reveal decommitments of C_u & C_v

Otherwise rewind

α, r_u
 β, r_v

$$\Pr[(u, v) = (u', v')] = \frac{1}{|E|}$$

Rewind $n \cdot |E|$ times $\Rightarrow$ failure prob. e^{-n} .

Prover

$\mathcal{H}_1$

Verifier*

$(u, v) \notin E$

$\alpha, \beta \in \{0, 1, 2\}$ st. $\alpha \neq \beta$

Construct $\pi: \{0, 1, 2\} \rightarrow \{0, 1, 2\}$ st.

$\pi(\phi(u)) = \alpha \wedge \pi(\phi(v)) = \beta$

$\forall v \in V, r_v \in \{0, 1\}^n, c_v := \text{Com}(\pi(\phi(v)), r_v)$

$\xrightarrow{\{c_v\}_{v \in V}}$
 $\xleftarrow{(u, v)}$

Reveal decommitments of c_u & c_v

$\xrightarrow{\alpha = \pi(\phi(u)), r_u}$
 $\beta = \pi(\phi(v)), r_v$

$\mathcal{H}_0$

Prover $\longleftrightarrow V^*$

$\equiv$

(identical distribution)

$\mathcal{H}_1$

Prover

$\mathcal{H}_2$

Verifier*

$n \cdot |E|$

$(u', v') \notin E$

$\alpha, \beta \in \{0, 1, 2\}$ st. $\alpha \neq \beta$

Construct $\pi: \{0, 1, 2\} \rightarrow \{0, 1, 2\}$ st.

$\pi(\phi(u')) = \alpha \wedge \pi(\phi(v')) = \beta$

$\forall v \in V, r_v \in \{0, 1\}^n, c_v := \text{Com}(\pi(\phi(v)), r_v)$

$\{c_v\}_{v \in V}$

(u, v)

If $(u, v) = (u', v')$:

Reveal decommitments of c_u & c_v

Otherwise rewind

α, r_u
 β, r_v

$\mathcal{H}_1$

$\stackrel{s}{\approx}$

(negligible failure probability)

$\mathcal{H}_2$

(computational $\stackrel{c}{\approx}$ hiding
of Com)

$\mathcal{H}_3$

Simulator $\longleftrightarrow V^*$

Non-Interactive Zero-Knowledge (NIZK) Proof



- **Completeness:** $\forall (x, w) \in R_L, \Pr [P(x, w) \rightarrow V(x) \text{ outputs } 1] = 1.$
- **Soundness:** $\forall x \notin L, \forall p^*, \Pr [P^*(x) \rightarrow V(x) \text{ outputs } 1] \leq \text{negl}(n)$
- **Zero-Knowledge:** $\forall \text{PPT } V^*, \exists \text{PPT } S \text{ s.t. } \forall (x, w) \in R_L, \text{Output}_{V^*}[P(x, w) \rightarrow V^*(x)] \simeq S(x)$

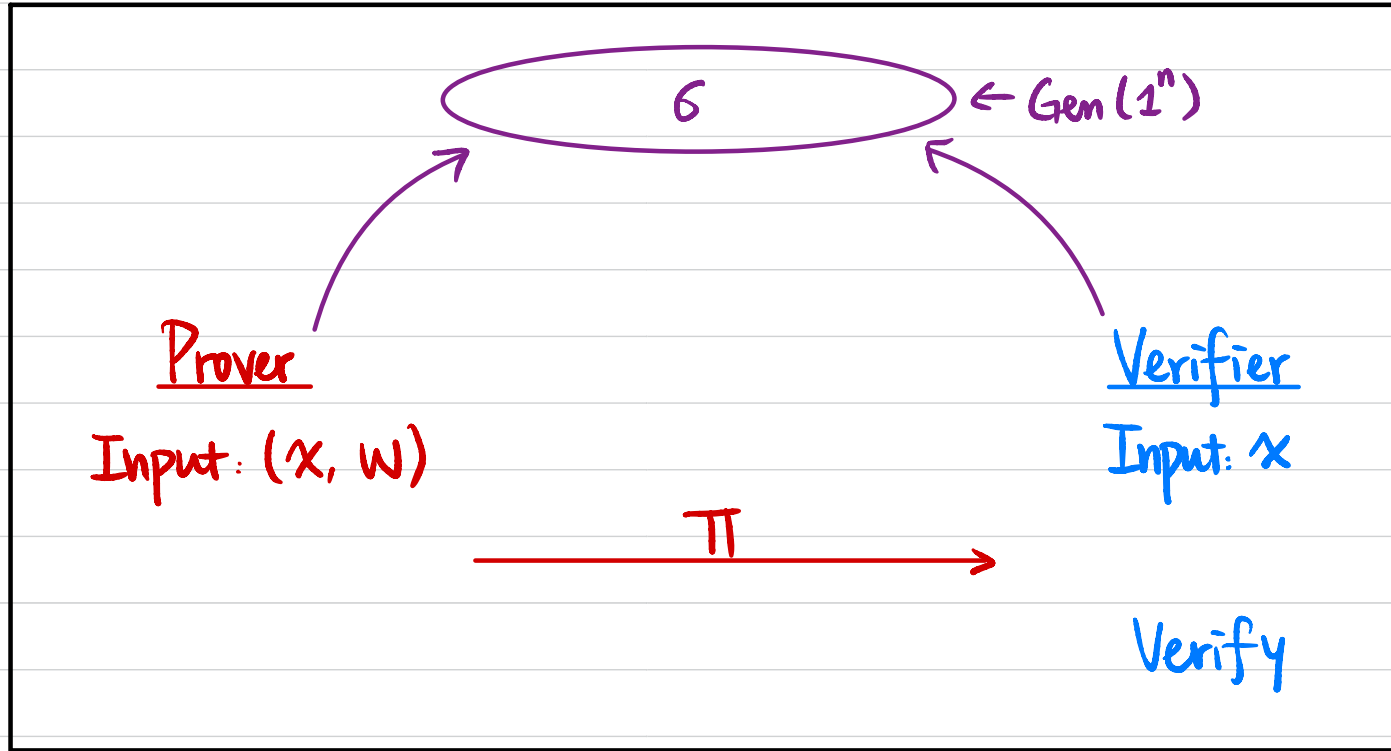
Is it possible?

$G \in L \Rightarrow \pi \simeq S(G) \rightarrow V \text{ outputs } 1$

$G \notin L \Rightarrow S(G) \rightarrow V \text{ outputs } 0$

NOT possible in the "plain" model

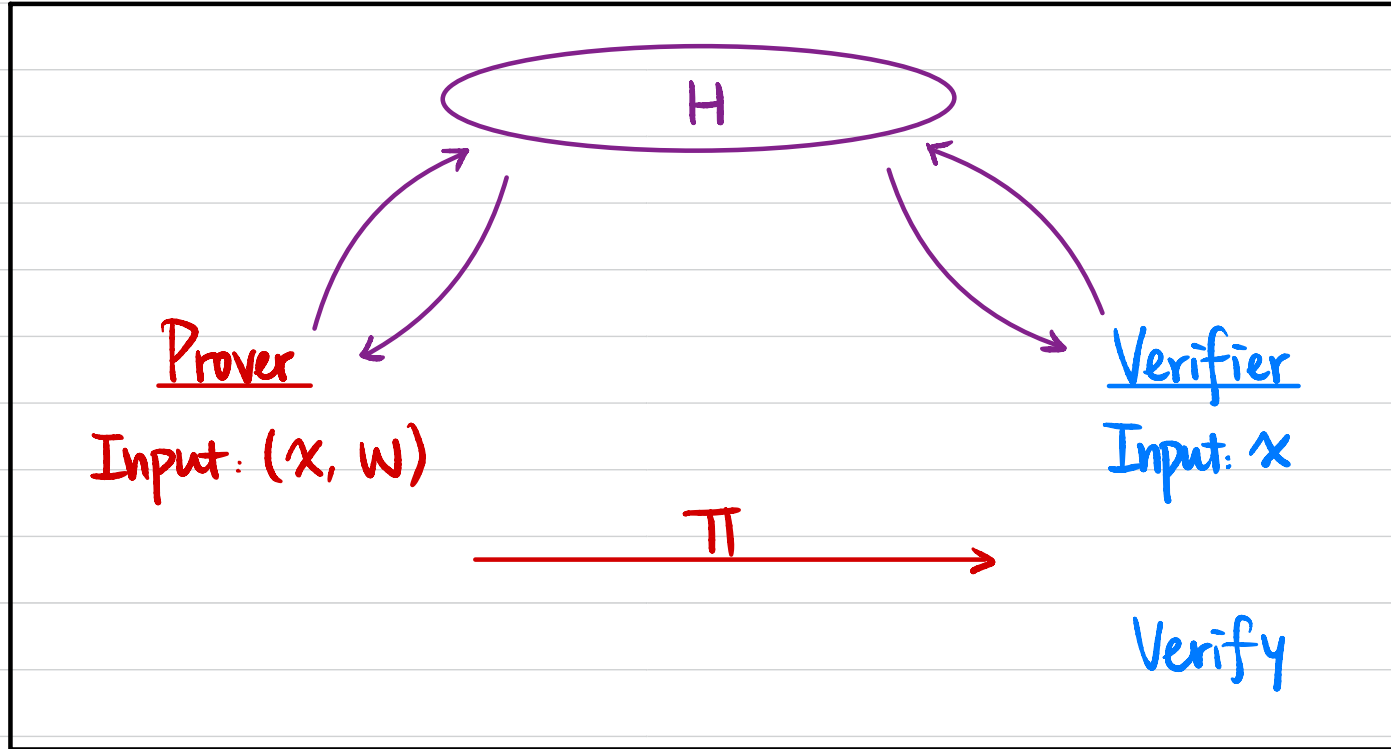
Model 1: Common Random String / Common Reference String (CRS)



$S(x)$ generates both $(\mathcal{G}, \pi)$

- **Zero-Knowledge:** $\forall \text{PPT } V^*, \exists \text{PPT } S \text{ s.t. } \forall (x, w) \in R_L,$
 $\text{Output}_{V^*}[\mathcal{G} \leftarrow \text{Gen}(1^n), P(x, w, \mathcal{G}) \rightarrow V^*(x, \mathcal{G})] \approx S(x)$
Alternatively: $(\mathcal{G} \leftarrow \text{Gen}(1^n), P(x, w, \mathcal{G})) \approx S(x)$

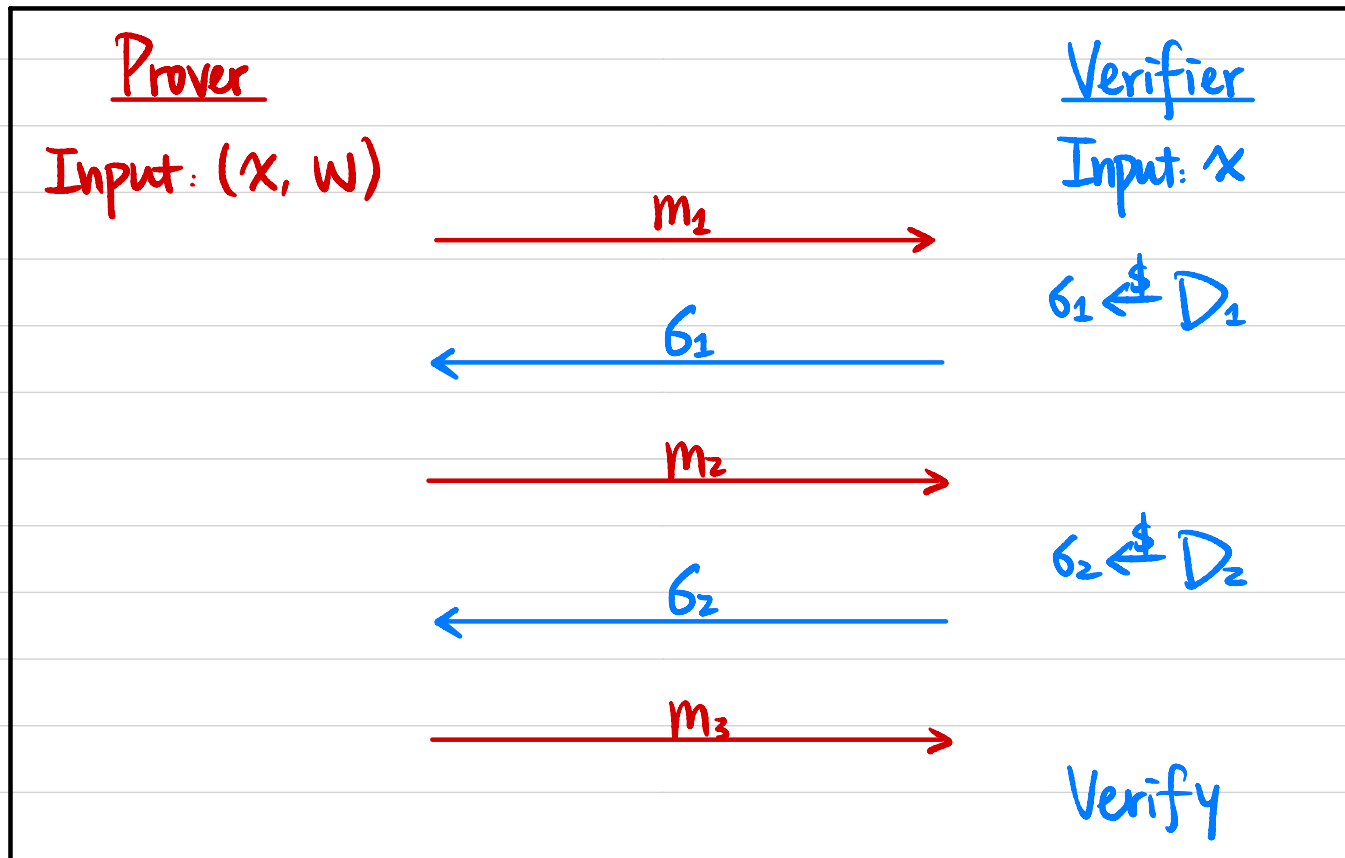
Model 2: Random Oracle Model



S controls input/output behavior of RO

Fiat-Shamir Heuristic

Public-Coin Honest-Verifier ZK (HVZK) $\Rightarrow$ NIZK in the RO model



$$c_1 := H(x \parallel m_1)$$

$$c_2 := H(x \parallel m_1 \parallel m_2)$$

Secure Multi-Party Computation (MPC)

Alice



$x \in \{0, 1\}$

Second date?

$$f(x, y) = x \wedge y$$

$y \in \{0, 1\}$

Bob



Who is richer?

x

$$f(x, y) = \begin{cases} 0 & \text{if } x > y \\ 1 & \text{otherwise} \end{cases}$$

y

Mutual friends?

x

$$f(x, y) = x \cap y$$

y

Secure Two-Party Computation (2PC)

Alice



x

Bob



y

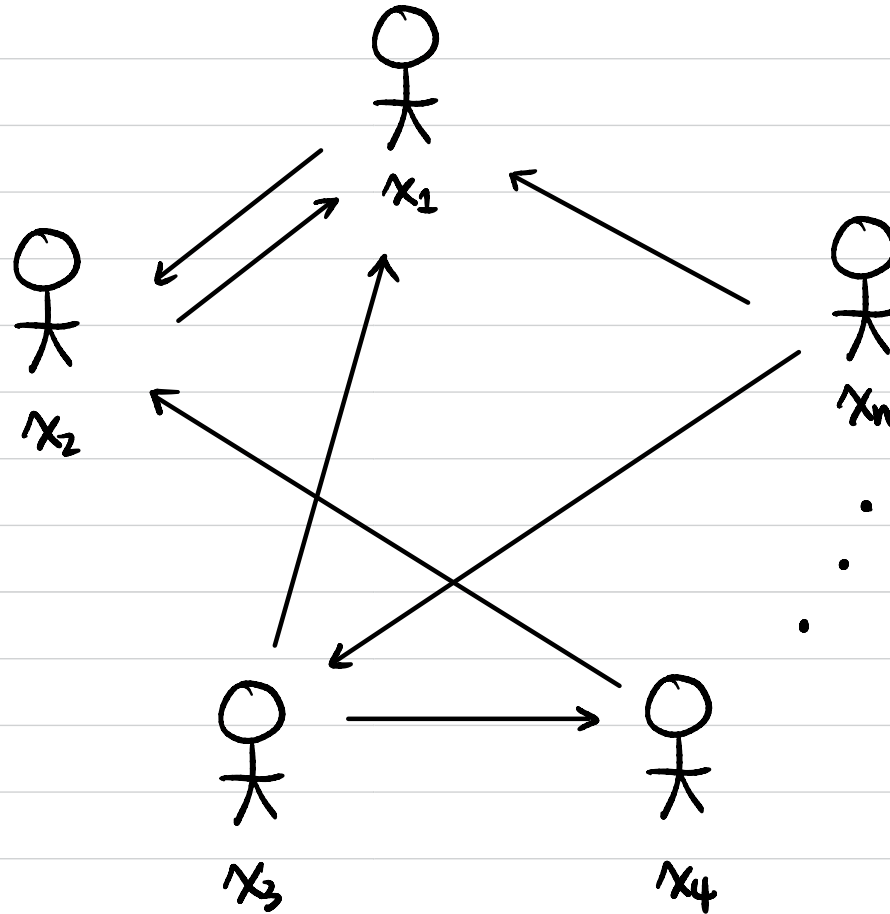


$$z = f(x, y)$$

Applications:

- Password Breach Alert (Chrome / Firefox / Azure / iOS Keychain)
- Privacy-Preserving Contact Tracing for COVID-19 (Apple & Google)
- Ads Conversion Measurements / Personalized Advertising (Google / Meta)

Secure Multi-Party Computation (MPC)



$$z = f(x_1, \dots, x_n)$$

Secure Multi-Party Computation (MPC)

Applications:

- Privacy-Preserving Inventory Matching (J.P. Morgan)
- Setup Ceremony to securely generate CRS (Zcash)
- Distributed Key Management (Unbound / Coinbase)
- Federated Learning (Google Keyboard Search Suggestion)
- Auctions (Danish sugar beet auction)
- Boston gender wage gap (Boston Women's Workforce Council)
- Study / Analysis on Medical Data
- Fraud Detection (banks)

Setting

- n parties $P_1, P_2, \dots, P_n$
with private inputs $x_1, x_2, \dots, x_n$
- Jointly compute $f(x_1, x_2, \dots, x_n)$
← public
- Communication:
Authenticated secure point-to-point channels between each pair (P_i, P_j)
(Sometimes also assume broadcast channel)
- The adversary can "corrupt" a subset of the parties
(e.g. at most t parties)

What properties do we want?

General Security Properties

- **Correctness:** The function is computed correctly.
- **Privacy:** Only the output is revealed.
- **Independence of Inputs:** Parties cannot choose inputs depending on others' inputs.
- **Security with Abort:** Adversary may "abort" the protocol.
(preventing honest parties from receiving the output)
- **Fairness:** If one party receives output, then all receive output.
- **Guaranteed Output Delivery (GOD):** Honest parties always receive output.

Adversary's Power

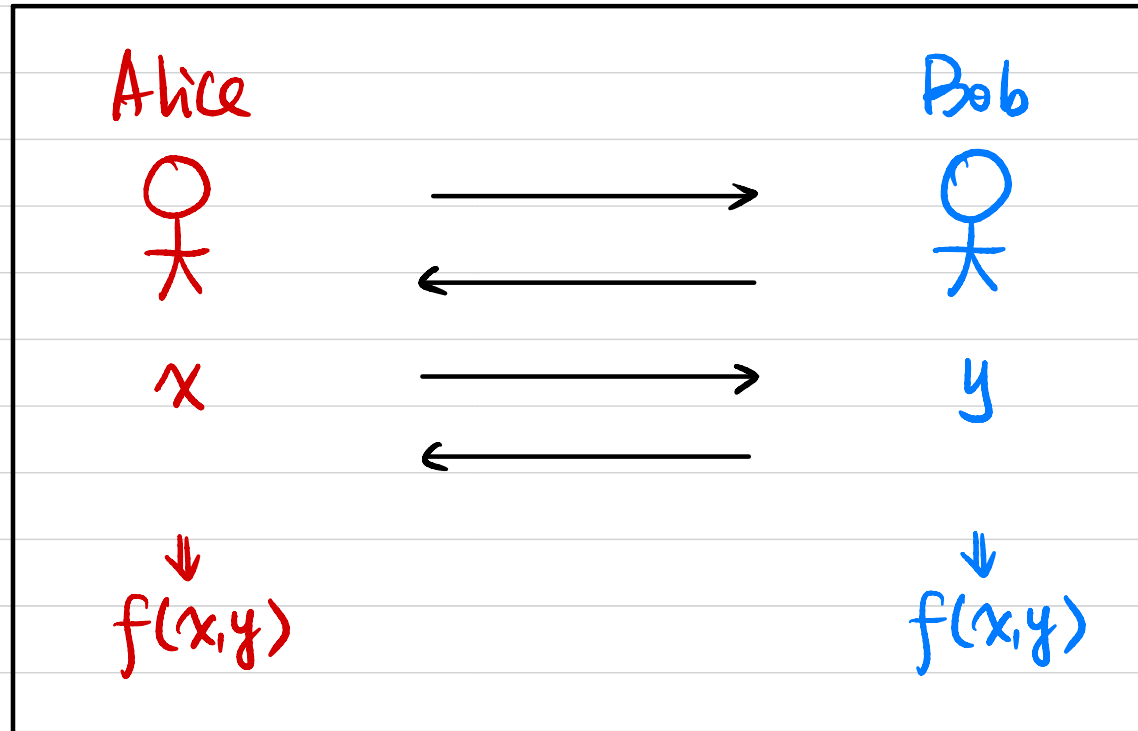
Allowed adversarial behavior:

- Semi-honest / passive / honest-but-curious:
Follow the protocol description honestly,
but try to extract more information by inspecting transcript.
- Malicious / active:
Can deviate arbitrarily from the protocol description.

Adversary's Computing Power:

- Unbounded computing power $\Rightarrow$ Information-Theoretic (IT) Security
- PPT bounded $\Rightarrow$ Computational Security

Security Against Semi-Honest Adversaries



Alice's view:

$\text{View}_A^\pi(x, y, n) := (x, \text{internal random tape } r, \text{ messages from Bob})$

Given $x, f(x, y)$, Alice's view can be "simulated".

Security Against Semi-Honest Adversaries

Def (Semi-honest security for ZPC)

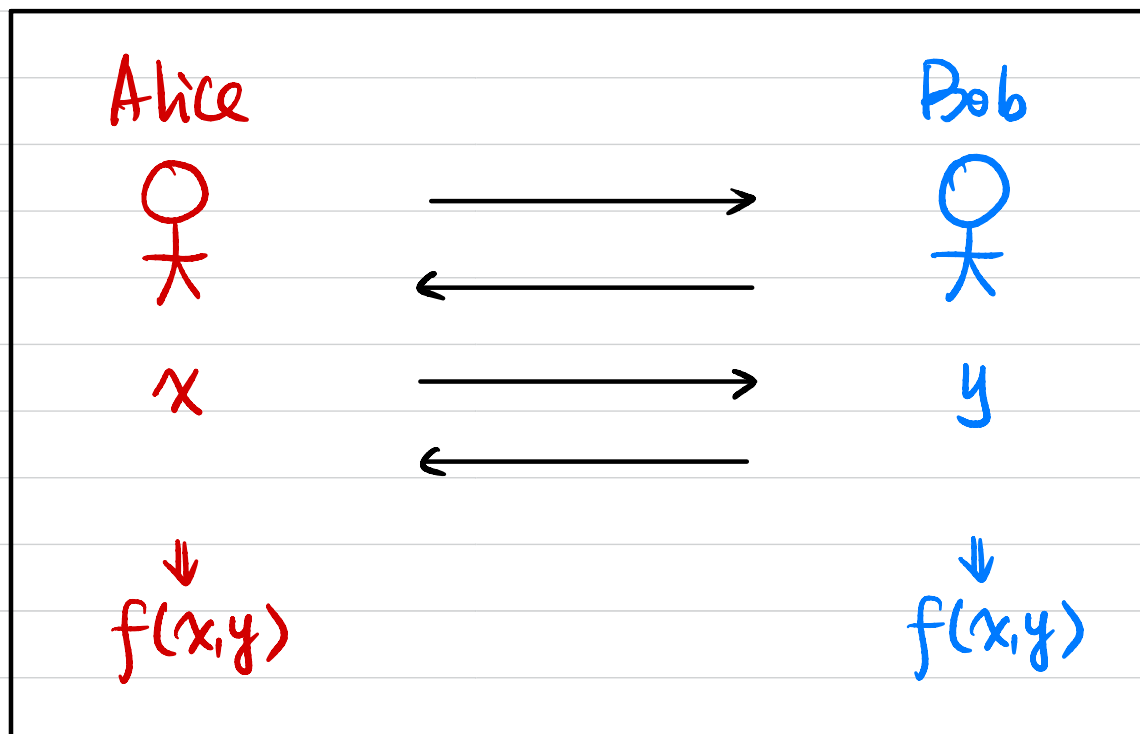
Let f be a functionality. We say a protocol Π securely computes f against semi-honest adversaries if $\exists$ PPT algorithms S_A, S_B s.t. $\forall x, y$,

$$\left\{ \begin{pmatrix} S_A(1^n, x, f(x, y)) \\ f(x, y) \end{pmatrix} \right\}_{n \in \mathbb{N}} \approx \left\{ \begin{pmatrix} \text{View}_A^\Pi(x, y, n) \\ \text{Output}^\Pi(x, y, n) \end{pmatrix} \right\}_{n \in \mathbb{N}}$$

$$\left\{ \begin{pmatrix} S_B(1^n, y, f(x, y)) \\ f(x, y) \end{pmatrix} \right\}_{n \in \mathbb{N}} \approx \left\{ \begin{pmatrix} \text{View}_B^\Pi(x, y, n) \\ \text{Output}^\Pi(x, y, n) \end{pmatrix} \right\}_{n \in \mathbb{N}}$$

perfect / statistical / computational
 $\equiv \quad \stackrel{s}{\approx} \quad \stackrel{c}{\approx}$

Security Against Malicious Adversaries



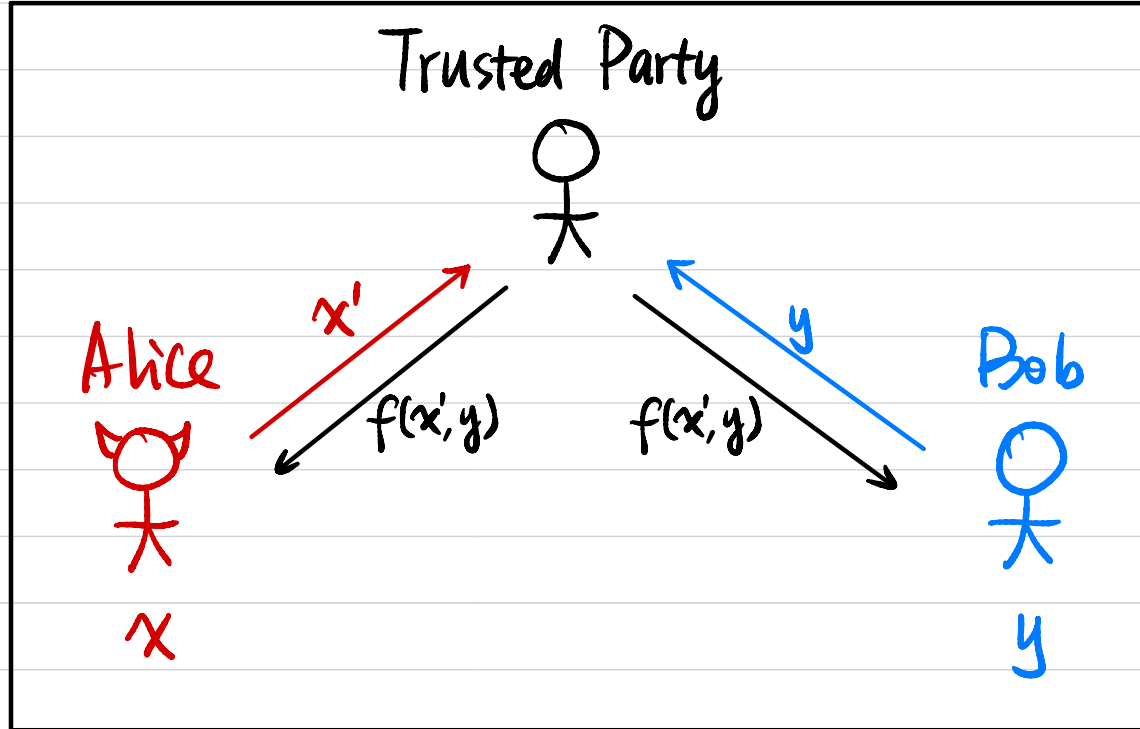
Alice's view:

$\text{View}_A^\pi(x, y, n) := (x, \text{internal random tape } r, \text{ messages from Bob})$

Given $x, f(x, y)$, Alice's view can be "simulated".

↑
What output?

What's the best we can hope for? (Ideal World)



Security Against Malicious Adversaries (Real / Ideal Paradigm)

Execution in the Real World:

(PPT) adversary A corrupting party $i \in \{ \text{Alice}, \text{Bob} \}$

$$\text{REAL}_{A,i}^{\pi} := \begin{pmatrix} A's \text{ output} \\ \text{Honest party's output in Real World} \end{pmatrix}$$

Execution in the Ideal World:

PPT adversary S corrupting party $i \in \{ \text{Alice}, \text{Bob} \}$

$$\text{IDEAL}_{S,i}^f := \begin{pmatrix} S's \text{ output} \\ \text{Honest party's output in Ideal World} \end{pmatrix}$$

Def (malicious security for ZPC)

Let f be a functionality. We say a protocol π securely computes f against malicious adversaries if $\forall$ (PPT) A in the real world, $\exists$ PPT S in the ideal world s.t. $\forall i \in \{ \text{Alice}, \text{Bob} \}, \forall x, y,$

$$\{ \text{REAL}_{A,i}^{\pi}(x, y, n) \}_{n \in \mathbb{N}} \approx \{ \text{IDEAL}_{S,i}^f(x, y, n) \}_{n \in \mathbb{N}}$$