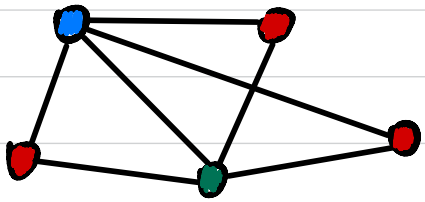
CSCI 1515 Applied Cryptography

This Lecture:

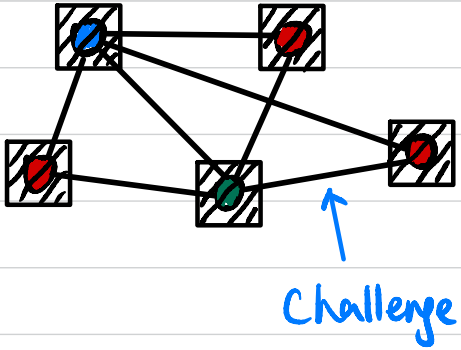
- Zero-Knowledge Proofs for All NP
- Succinct Non-Interactive Arguments (SNARGs)

Zero-Knowledge Proof for Graph 3-Coloring (All NP)



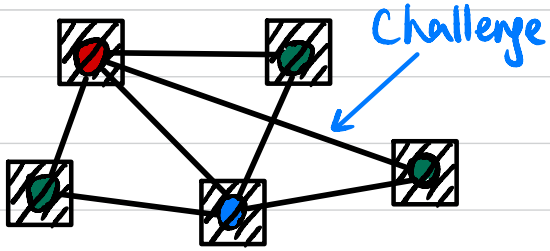
NP language $L = \{ G : G \text{ has 3-coloring} \}$

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



$G=(V,E)$

Shuffle Color



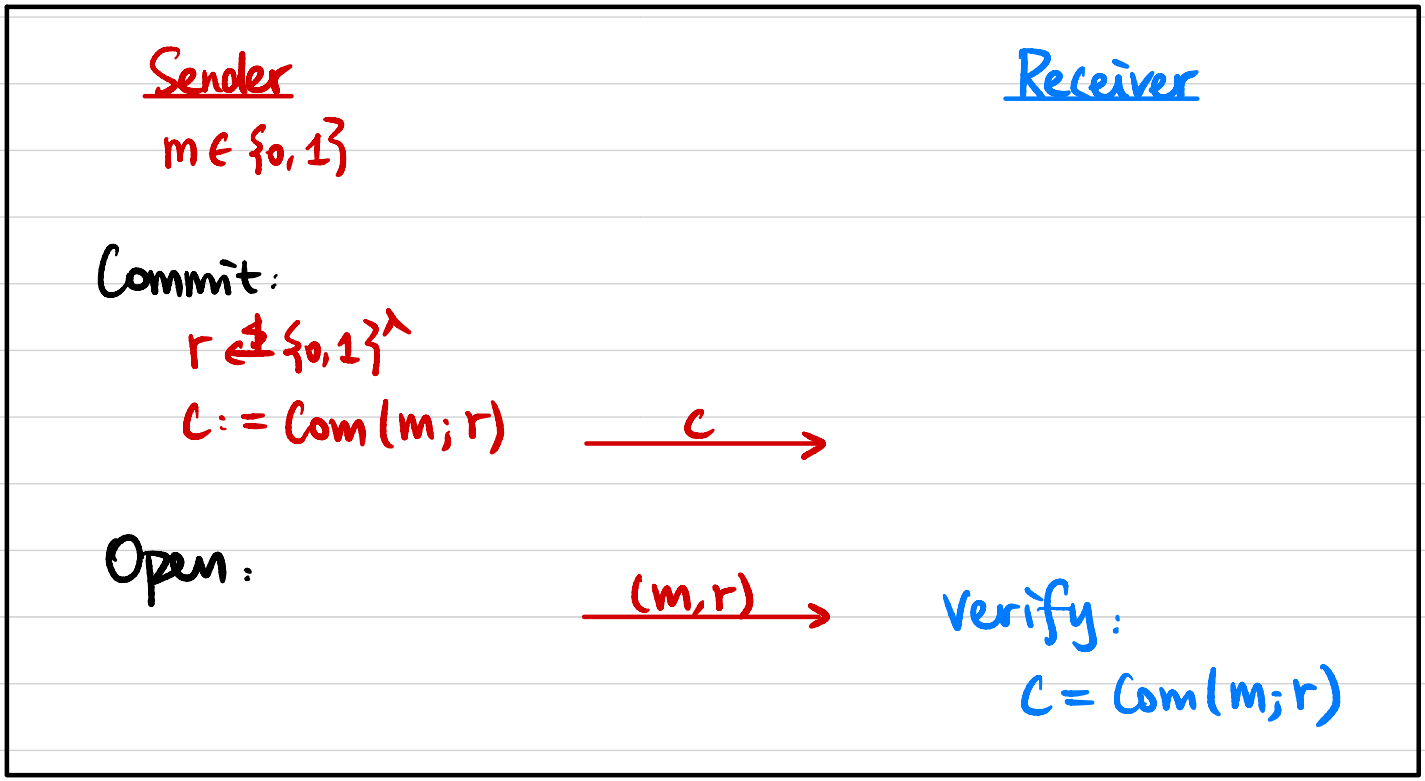
If $G \notin L$, $\Pr[P^* \text{ is caught}] \geq \frac{1}{|E|}$

$\Pr[P^* \text{ convinces } V] \leq \left(1 - \frac{1}{|E|}\right)^{\lambda \cdot |E|}$

$\left(1 - \frac{1}{N}\right)^N \simeq \frac{1}{e}$ $\leq \left(\frac{1}{e}\right)^{\lambda}$

How to amplify soundness?

Commitment Scheme



Example: Pedersen Commitment

Cyclic group G of order q with generator g. $h \xleftarrow{\$} G$ ← can be generated by Receiver

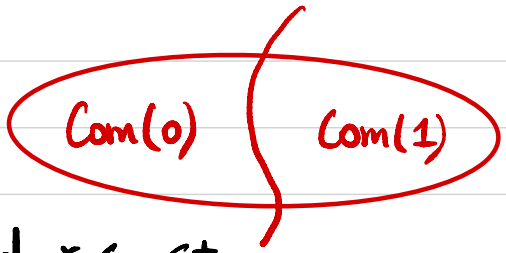
$r \xleftarrow{\$} \mathbb{Z}_q$

$Com(m;r) = g^m \cdot h^r = c$

Commitment Scheme

- **Hiding:** $\text{Com}(0; r) \simeq \text{Com}(1; s)$
  - **Perfectly hiding:** $\text{Com}(0; r) \equiv \text{Com}(1; s)$
  - **Computationally hiding:** $\text{Com}(0; r) \overset{c}{\simeq} \text{Com}(1; s)$

- **Binding:** Hard to find r, s s.t. $\text{Com}(0; r) = \text{Com}(1; s)$
  - **Perfectly binding:** $\forall r, s,\ \text{Com}(0; r) \neq \text{Com}(1; s)$
  - **Computationally binding:** Any PPT sender cannot find r, s s.t. $\text{Com}(0; r) = \text{Com}(1; s)$



find r, s s.t. for some c,
$c = \text{Com}(0; r) = \text{Com}(1; s)$
$c = h^r = g \cdot h^s$
$\Leftrightarrow h^{r-s} = g$
$\Leftrightarrow h = g^{(r-s)^{-1}}$
$(r-s)^{-1} = \log_g h$

What does Pedersen commitment scheme satisfy?

$r \xleftarrow{\$} \mathbb{Z}_q$ (random in G (OTP))

$\text{Com}(m; r) = g^m \cdot h^r = c$

perfectly hiding
computationally binding

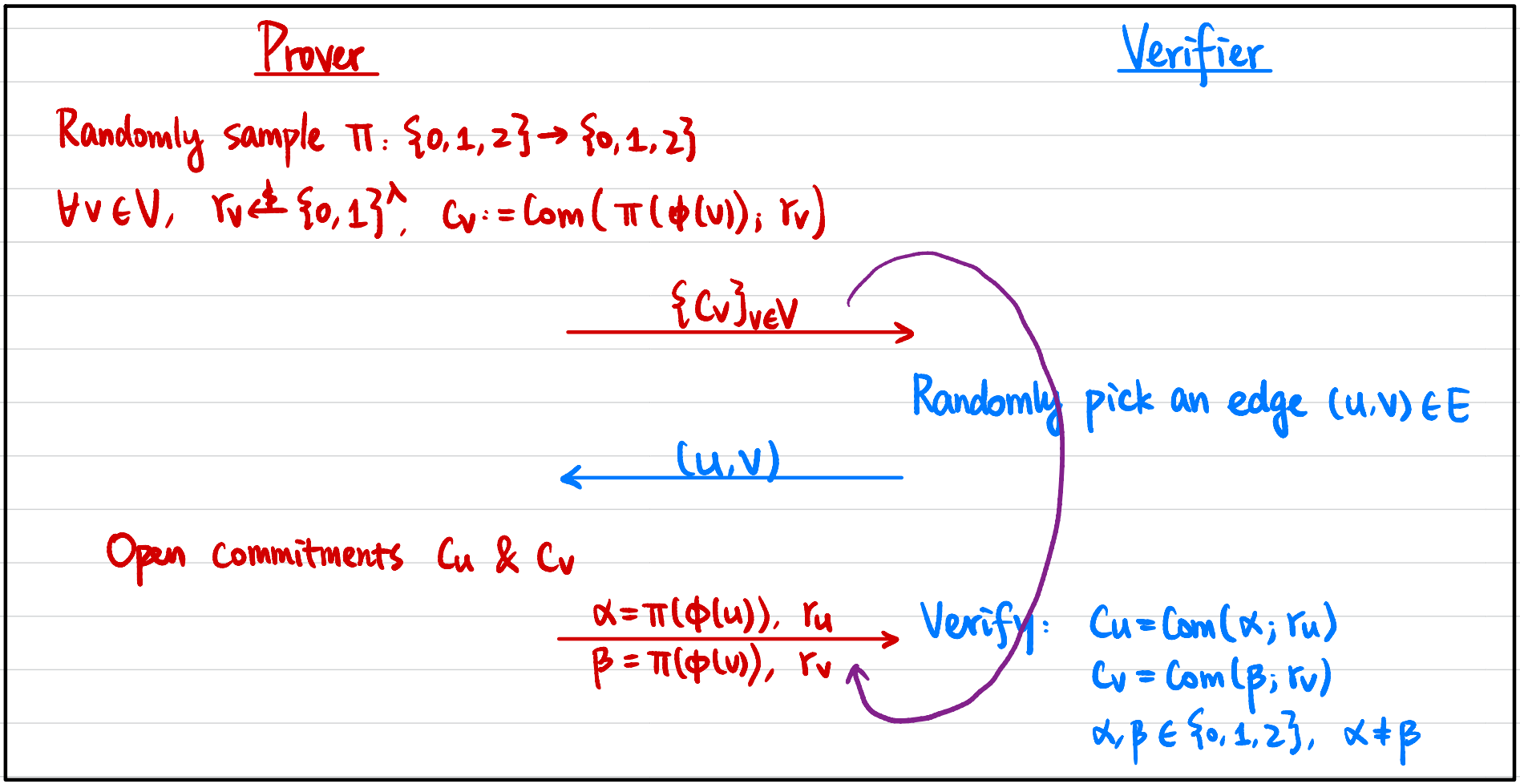
Can a commitment scheme be both perfectly hiding & perfectly binding? No!

Zero-Knowledge Proof for Graph 3-Coloring

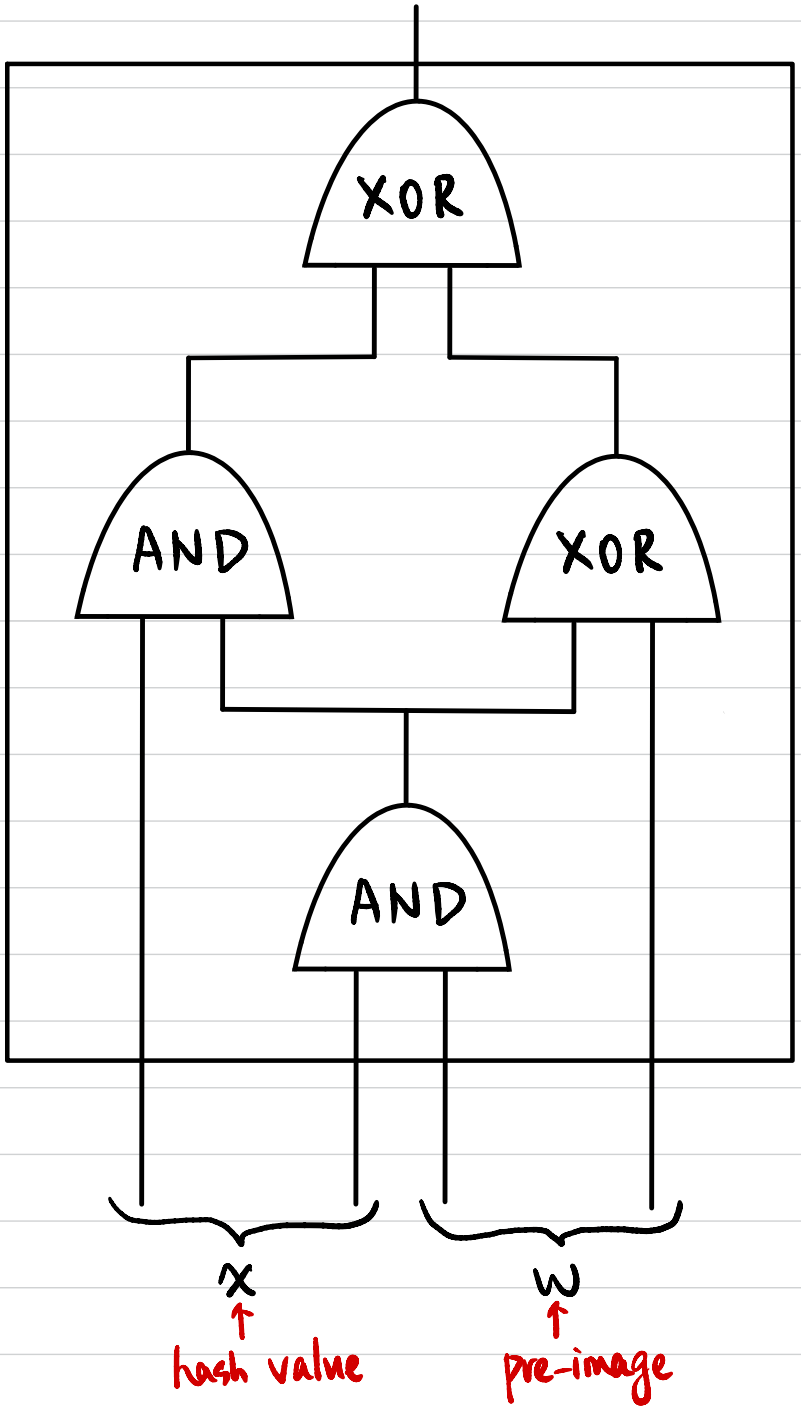
Input: $G = (V, E)$

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

Given a computationally hiding, perfectly binding commitment scheme.



Circuit Satisfiability (NP Complete)



NP language $L_C = \{x \in \{0,1\}^n : \exists w \in \{0,1\}^m \text{ s.t. } C(x,w) = 1\}$

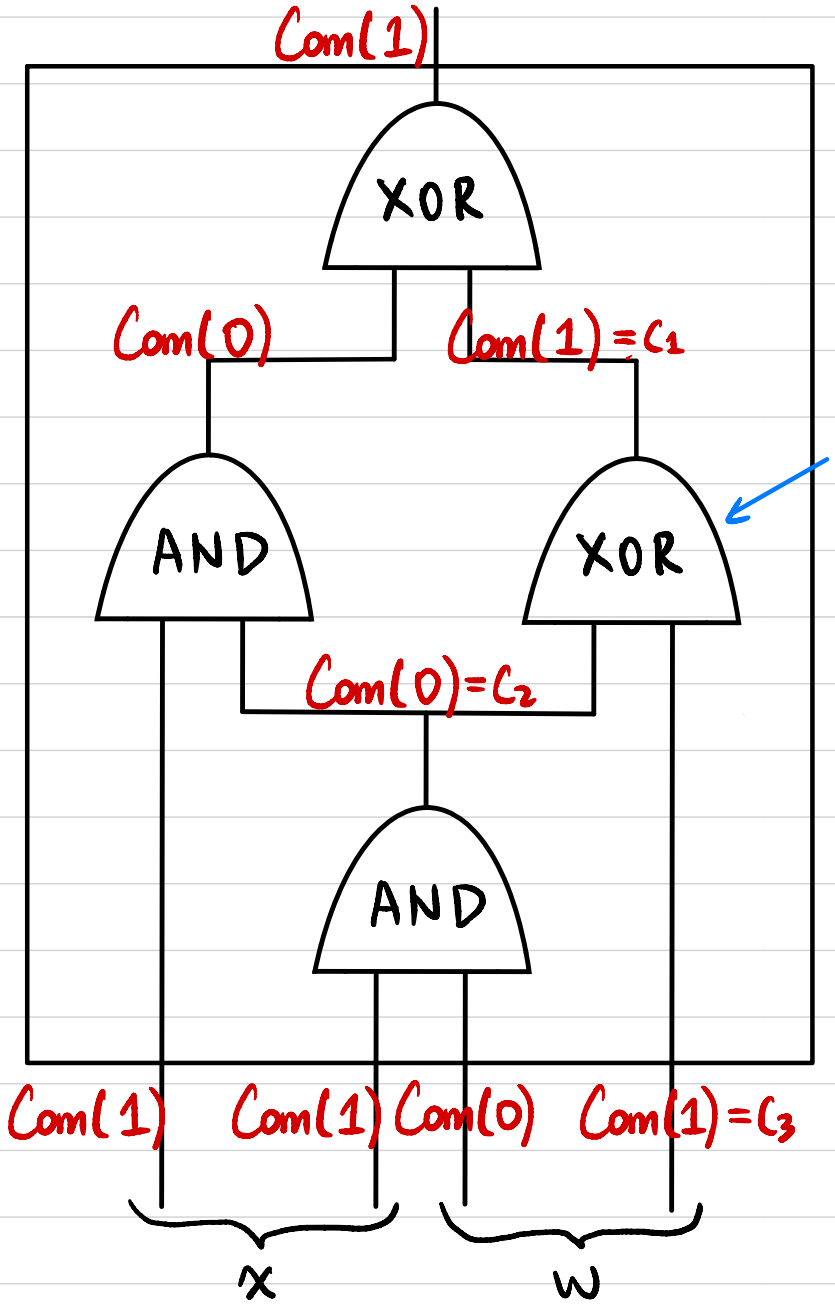
NP relation $R_{L_C} = \{(x, w) : C(x, w) = 1\}$

Example: pre-image of hash function

$C(x, w) = H(w) - x + 1$

$H(w) = x$

ZKP for Circuit Satisfiability



$$\begin{pmatrix} C_1 = Com(0) \\ C_2 = Com(0) \\ C_3 = Com(0) \end{pmatrix}$$

OR

$$\begin{pmatrix} C_1 = Com(1) \\ C_2 = Com(0) \\ C_3 = Com(1) \end{pmatrix}$$

OR

$$\begin{pmatrix} C_1 = Com(1) \\ C_2 = Com(1) \\ C_3 = Com(0) \end{pmatrix}$$

OR

$$\begin{pmatrix} C_1 = Com(0) \\ C_2 = Com(1) \\ C_3 = Com(1) \end{pmatrix}$$

Proof Systems for Circuit Satisfiability

NP relation $R_{\mathcal{C}} = \{(x, w) : C(x, w) = 1\}$

	NP	Σ-Protocol	(Fiat-Shamir) NIZK
	$P(x, w) \xrightarrow{w} V(x)$ $C(x,w)=1$	$P(x, w) \rightleftarrows V(x)$	$P(x, w) \xrightarrow{\pi} V(x)$
Zero-Knowledge	NO	YES	YES
Non-Interactive	YES	NO	YES
Communication	$O(\|w\|)$	$O(\|C\| \cdot \lambda)$	$O(\|C\| \cdot \lambda)$
V's computation	$O(\|C\|)$	$O(\|C\| \cdot \lambda)$	$O(\|C\| \cdot \lambda)$

Can we have communication complexity & verifier's computational complexity sublinear in $|C|$ & $|w|$?

Succinct Non-Interactive Argument (SNARG)

$\forall P^*$ $\forall PPT\ P^*$ (in soundness)

Def A non-interactive proof/argument system is succint if

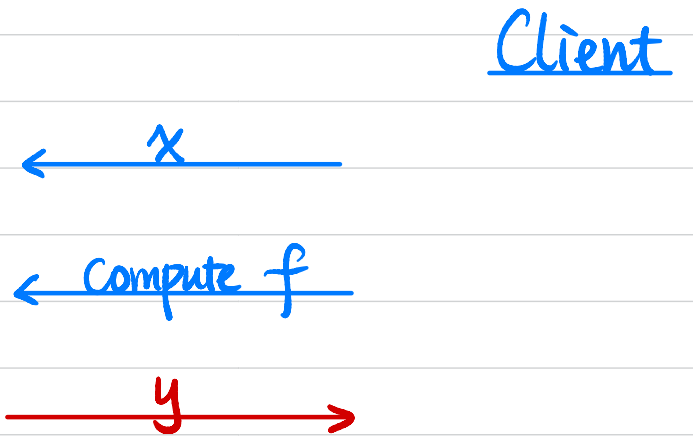
- The proof π is of length $|\pi| = poly(\lambda, \log|C|)$
- The verifier runs in time $poly(\lambda, |x|, \log|C|)$

- SNARK: Succinct Non-Interactive Argument of Knowledge
- zk-SNARG/zk-SNARK: SNARG/SNARK + zero-knowledge

Why Succinct Proofs?

Is it possible?

Verifiable Computation



Anonymous Transactions on Blockchain

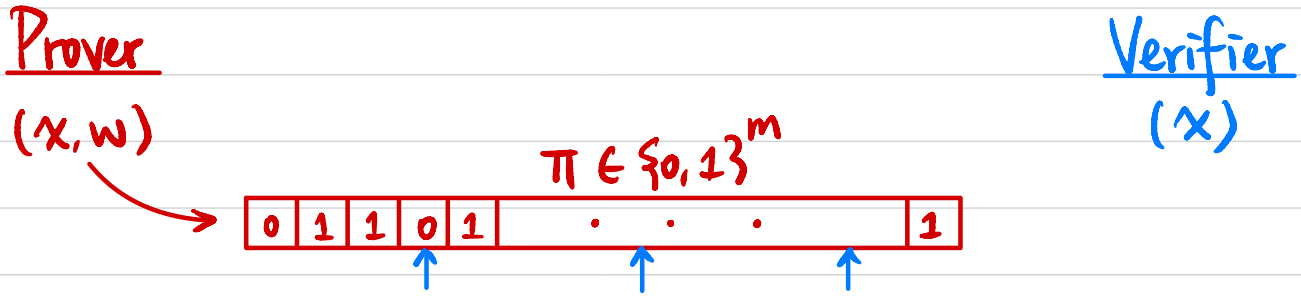
Transaction: $\boxed{vk_A, vk_B, 2BTC}$, $\sigma = Sign_{sk_A}(\cdot)$

Anonymous Transaction:

$$Com\left(\boxed{vk_A, vk_B, 2BTC}, \sigma = Sign_{sk_A}(\cdot)\right)$$

NIZk: valid transaction

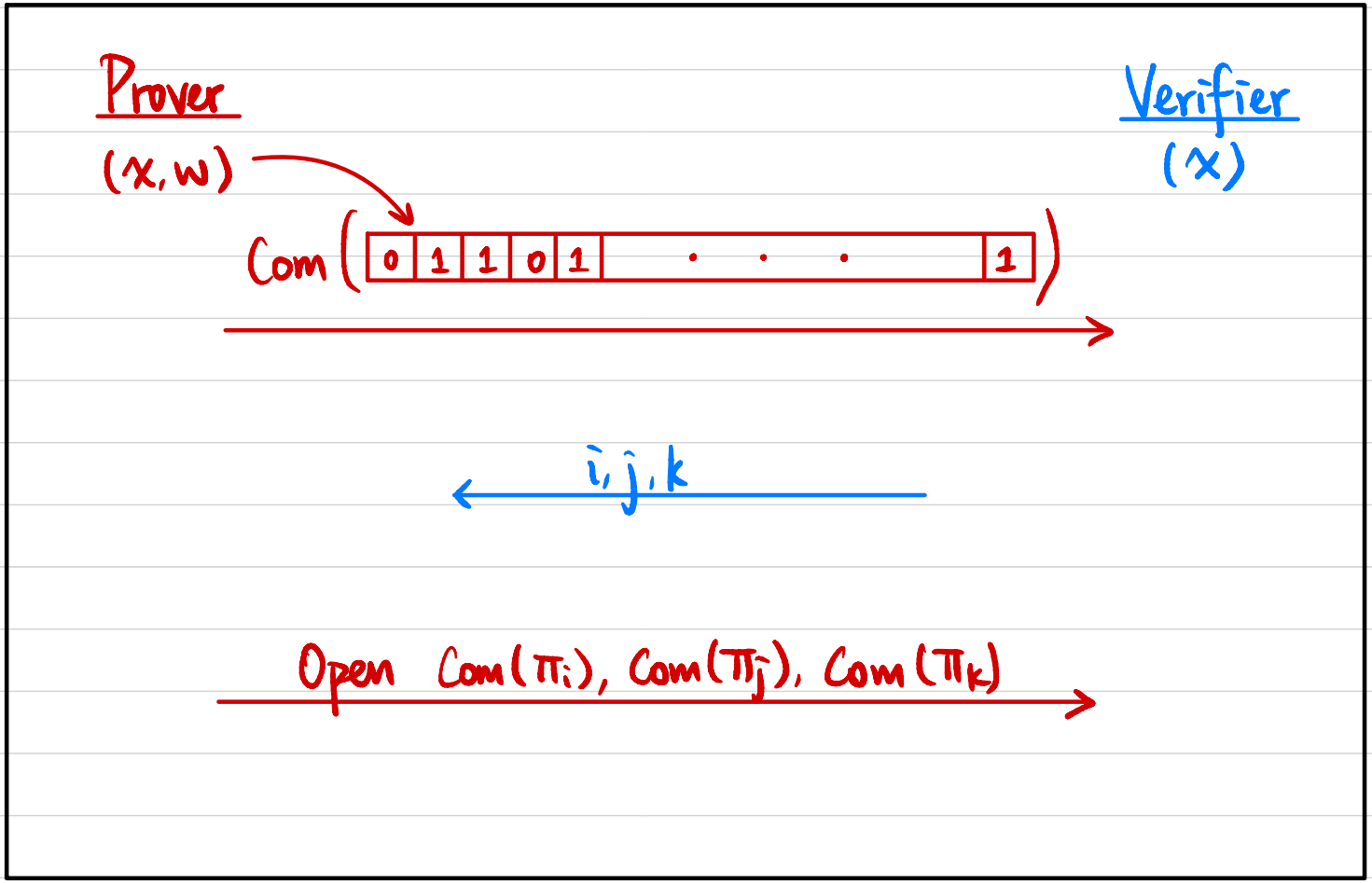
Probabilistically Checkable Proof (PCP)



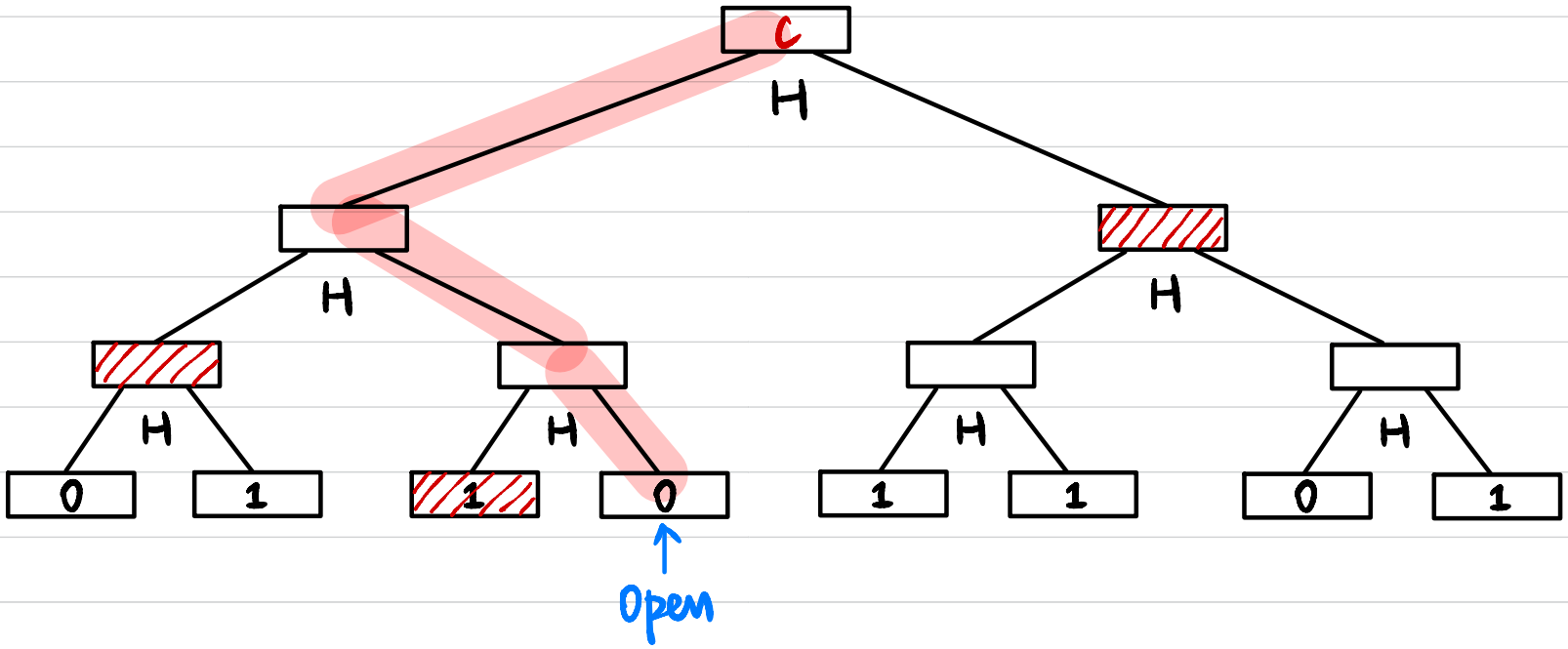
PCP Theorem (Informal):

Every NP language has a PCP where the Verifier reads only a constant number of bits of the proof.

First Attempt



Merkle Tree



Why (computationally) binding ?

Can we make it hiding ?