Proving Computation Integrity

CS 276: Introduction to Cryptography

Sanjam Garg

April 8, 2026

Overview

1. Commitment Schemes
2. Zero-Knowledge for NP via 3COL
3. NIZK Proof Systems
4. NIZKs from Trapdoor Permutations
5. NIZK for NP: Graph Hamiltonicity

Commitment Schemes

Definition 1 (Commitment Scheme)

A *commitment scheme* is a PPT machine C taking input (b, r) satisfying:

- **Perfect binding**: $\forall\, r, s,\ C(0, r) \neq C(1, s)$.
- **Computational hiding**: $\{C(0, U_n)\} \approx_c \{C(1, U_n)\}$.

Commitment Schemes

Definition 1 (Commitment Scheme)

A *commitment scheme* is a PPT machine C taking input (b, r) satisfying:

- **Perfect binding**: $\forall\, r, s,\ C(0, r) \neq C(1, s)$.
- **Computational hiding**: $\{C(0, U_n)\} \approx_c \{C(1, U_n)\}$.

Opening

To "open" a commitment $C(b, r)$, the sender reveals r (the *decommitment*). The receiver recomputes and checks.

Commitment Schemes

Definition 1 (Commitment Scheme)

A *commitment scheme* is a PPT machine C taking input (b, r) satisfying:

- **Perfect binding**: $\forall\, r, s,\ C(0, r) \neq C(1, s)$.
- **Computational hiding**: $\{C(0, U_n)\} \approx_c \{C(1, U_n)\}$.

Opening

To "open" a commitment $C(b, r)$, the sender reveals r (the *decommitment*). The receiver recomputes and checks.

Why perfect binding?

Even a single pair r, s with $C(0, r) = C(1, s)$ lets a cheating sender open the same commitment as either 0 or 1.

Commitment Schemes from Injective OWFs

Theorem 2

If injective one-way functions exist, then commitment schemes exist.

Commitment Schemes from Injective OWFs

Theorem 2

If injective one-way functions exist, then commitment schemes exist.

Construction

Let f be an injective OWF. Define $f'(x, r) := (f(x), r)$, also an injective OWF with hard-core bit $B(x, r) := \langle x, r \rangle$.

$$C(b, x, r) := \big(f'(x, r),\ b \oplus B(x, r)\big)$$

Commitment Schemes from Injective OWFs

Theorem 2

If injective one-way functions exist, then commitment schemes exist.

Construction

Let f be an injective OWF. Define $f'(x, r) := (f(x), r)$, also an injective OWF with hard-core bit $B(x, r) := \langle x, r \rangle$.

$$C(b, x, r) := \big(f'(x, r),\ b \oplus B(x, r)\big)$$

Proof

- **Perfect binding**: since f' is injective, $(x, r) \neq (y, s) \Rightarrow C(0, x, r) \neq C(0, y, s)$. Also $C(0, x, r) = (f'(x, r), B(x, r)) \neq (f'(x, r), \overline{B(x, r)}) = C(1, x, r)$.
- **Computational hiding**: distinguishing $C(0, U_n)$ from $C(1, U_n)$ means distinguishing $B(x, r)$ from $\overline{B(x, r)}$ given $f'(x, r)$, contradicting hard-core bit security.

□

Multi-bit Commitments

Extension to longer messages

Concatenate bit commitments: for messages $m = m_1 m_2 \cdots m_\ell$, define

$$C(m, r_1, \ldots, r_\ell) := C(m_1, r_1) \| C(m_2, r_2) \| \cdots \| C(m_\ell, r_\ell).$$

Multi-bit Commitments

Extension to longer messages

Concatenate bit commitments: for messages $m = m_1 m_2 \cdots m_\ell$, define

$$C(m, r_1, \ldots, r_\ell) := C(m_1, r_1) \| C(m_2, r_2) \| \cdots \| C(m_\ell, r_\ell).$$

Security

For any two messages m_0, m_1 of the same length:

$$\{C(m_0, U_n)\} \approx_c \{C(m_1, U_n)\}.$$

Follows by a standard hybrid argument over the ℓ positions.

3-Coloring is NP-Complete

Definition 3 (n-Coloring)

An n-coloring of graph $G = (V, E)$ is a function $c : V \to \{1, \ldots, n\}$ such that $(i, j) \in E \Rightarrow c(i) \neq c(j)$.

3-Coloring is NP-Complete

Definition 3 (n-Coloring)

An n-coloring of graph $G = (V, E)$ is a function $c : V \to \{1, \ldots, n\}$ such that $(i, j) \in E \Rightarrow c(i) \neq c(j)$.

3COL

Given a graph G, decide if a 3-coloring exists. This problem is **NP-complete**.

3-Coloring is NP-Complete

Definition 3 (n-Coloring)

An n-coloring of graph $G = (V, E)$ is a function $c : V \to \{1, \ldots, n\}$ such that $(i, j) \in E \Rightarrow c(i) \neq c(j)$.

3COL

Given a graph G, decide if a 3-coloring exists. This problem is **NP-complete**.

Strategy

A computational zero-knowledge proof for 3COL $\Rightarrow$ ZK proofs for **all of NP** (via polynomial-time reductions).

Intuition: Cups and Colors

Physical protocol

1. P asks V to leave the room. P samples a random color permutation ϕ and colors the vertices according to $\phi \circ c$, then covers all vertices with **cups**.
2. V returns and picks one edge; P uncovers its two endpoints.
3. If endpoints share a color, V **rejects**. Otherwise, repeat with a fresh random ϕ.

Intuition: Cups and Colors

Physical protocol

1. P asks V to leave the room. P samples a random color permutation ϕ and colors the vertices according to $\phi \circ c$, then covers all vertices with **cups**.
2. V returns and picks one edge; P uncovers its two endpoints.
3. If endpoints share a color, V **rejects**. Otherwise, repeat with a fresh random ϕ.

Key observations

- **Soundness**: if G is not 3-colorable, some edge is monochromatic; V catches it with probability $\geq 1/|E|$.
- **Zero knowledge**: the random permutation ϕ masks the true coloring; V learns nothing.
- **Problem**: this requires P not to recolor after uncovering $\Rightarrow$ we need **commitments**.

3COL Protocol

Protocol $P(x, z) \leftrightarrow V(x)$

Input: graph $G = (\{1, \ldots, n\}, E)$, witness z = 3-coloring c.

1. **Commit**: P picks random permutation $\pi : \{1, 2, 3\} \to \{1, 2, 3\}$ and sets $\beta := \pi \circ c$. For each $i \in V$, computes $\alpha_i = C(\beta(i), U_n)$. Sends $\alpha_1, \ldots, \alpha_n$ to V.
2. **Challenge**: V samples a random edge $e = (i, j) \in E$ and sends it to P.
3. **Open**: P opens commitments α_i and α_j (reveals decommitments).
4. **Verify**: V accepts iff decommitments are valid and $\beta(i) \neq \beta(j)$ are distinct valid colors.

3COL: Completeness and Soundness

Completeness

If c is a valid 3-coloring and π is any permutation, then $\beta = \pi \circ c$ is also a valid 3-coloring: every edge has distinct colors. V always accepts.

3COL: Completeness and Soundness

Completeness

If c is a valid 3-coloring and π is any permutation, then $\beta = \pi \circ c$ is also a valid 3-coloring: every edge has distinct colors. V always accepts.

Soundness (one round)

If G is **not** 3-colorable, then for any coloring β, at least one edge (i, j) has $\beta(i) = \beta(j)$.

$$\Pr[V \text{ rejects in one round}] \geq \frac{1}{|E|}.$$

Since $|E| \leq n^2$, repeating $\text{poly}(n)$ times amplifies to soundness error $\leq \text{negl}(n)$.

3COL Simulator (for ZK)

Simulator S for verifier V^*

1. Sample a random edge $e = (i, j) \in E$.
2. Assign c_i, c_j to distinct random values from $\{1, 2, 3\}$. Set $c_k := 1$ for all $k \neq i, j$.
3. Compute random keys $r_1, \ldots, r_n$ and set $\alpha_k = C(c_k, r_k)$ for all k.
4. Let $e' \in E$ be V^*'s response upon receiving $\alpha_1, \ldots, \alpha_n$.
5. If $e' \neq e$: go back to step 1. If restarted more than $2n|E|$ times, output fail.
6. If $e' = e$: output $\alpha_1, \ldots, \alpha_n$, e, open r_i, r_j, and print V^*'s final output.

3COL Simulator (for ZK)

Simulator S for verifier V^*

1. Sample a random edge $e = (i, j) \in E$.
2. Assign c_i, c_j to distinct random values from $\{1, 2, 3\}$. Set $c_k := 1$ for all $k \neq i, j$.
3. Compute random keys $r_1, \ldots, r_n$ and set $\alpha_k = C(c_k, r_k)$ for all k.
4. Let $e' \in E$ be V^*'s response upon receiving $\alpha_1, \ldots, \alpha_n$.
5. If $e' \neq e$: go back to step 1. If restarted more than $2n|E|$ times, output fail.
6. If $e' = e$: output $\alpha_1, \ldots, \alpha_n$, e, open r_i, r_j, and print V^*'s final output.

Key property

S runs in **polynomial time** (at most $2n|E|$ restarts), but may output fail. We must show fail occurs with negligible probability.

Simulator: Bounding Fail Probability

Claim

For all but finitely many graphs G, V^* outputs $e' = e$ with probability $\geq 1/(2|E|)$.

Simulator: Bounding Fail Probability

Claim

For all but finitely many graphs G, V^* outputs $e' = e$ with probability $\geq 1/(2|E|)$.

Proof idea

Suppose $\Pr[e' = e] < 1/(2|E|)$ for infinitely many G.

- Consider modified simulator $\tilde{S}$: set $c_k = 1$ for **all** k (including i, j).
- Then V^* cannot distinguish edges $\Rightarrow \Pr[e' = e] = 1/|E|$ exactly.
- But S and $\tilde{S}$ differ only in two committed values (c_i, c_j vs. $1, 1$).
- Distinguishing S from $\tilde{S} \Rightarrow$ breaking hiding of commitment scheme C. Contradiction.

Simulator: Bounding Fail Probability

Claim

For all but finitely many graphs G, V^* outputs $e' = e$ with probability $\geq 1/(2|E|)$.

Proof idea

Suppose $\Pr[e' = e] < 1/(2|E|)$ for infinitely many G.

- Consider modified simulator $\tilde{S}$: set $c_k = 1$ for **all** k (including i, j).
- Then V^* cannot distinguish edges $\Rightarrow \Pr[e' = e] = 1/|E|$ exactly.
- But S and $\tilde{S}$ differ only in two committed values (c_i, c_j vs. $1, 1$).
- Distinguishing S from $\tilde{S} \Rightarrow$ breaking hiding of commitment scheme C. Contradiction.

Fail probability

$$\Pr[S \text{ outputs fail}] < \left(1 - \frac{1}{2|E|}\right)^{2n|E|} < \frac{1}{e^n} = \text{negl}(n). \square$$

Simulator: Indistinguishability of Transcripts

Modified simulator S'

Like S, but given a valid 3-coloring as auxiliary input: in step 2, use a random permutation of the true coloring (instead of setting $c_k = 1$ for $k \neq i, j$).

Simulator: Indistinguishability of Transcripts

Modified simulator S'

Like S, but given a valid 3-coloring as auxiliary input: in step 2, use a random permutation of the true coloring (instead of setting $c_k = 1$ for $k \neq i, j$).

Argument (two steps)

1. $P \leftrightarrow V^*$ and S' are computationally indistinguishable: they behave identically except S' may output fail, which happens with $\mathrm{negl}(n)$ probability.
2. S and S' are computationally indistinguishable: they differ only in committed values. A distinguisher would break computational hiding of C (distinguishing commitments to a valid 3-coloring vs. commitments with $c_k = 1$ for $k \neq i, j$).

Simulator: Indistinguishability of Transcripts

Modified simulator S'

Like S, but given a valid 3-coloring as auxiliary input: in step 2, use a random permutation of the true coloring (instead of setting $c_k = 1$ for $k \neq i, j$).

Argument (two steps)

1. $P \leftrightarrow V^*$ and S' are computationally indistinguishable: they behave identically except S' may output fail, which happens with $\mathrm{negl}(n)$ probability.
2. S and S' are computationally indistinguishable: they differ only in committed values. A distinguisher would break computational hiding of C (distinguishing commitments to a valid 3-coloring vs. commitments with $c_k = 1$ for $k \neq i, j$).

Therefore $P \leftrightarrow V^* \approx_c S' \approx_c S$, and the protocol is **zero-knowledge**. □

ZK for All of NP

Theorem 4

Assuming injective one-way functions exist, every language in NP has a computational zero-knowledge interactive proof.

ZK for All of NP

Theorem 4

Assuming injective one-way functions exist, every language in NP has a computational zero-knowledge interactive proof.

Proof

1. Injective OWFs $\Rightarrow$ commitment schemes exist (shown earlier).
2. The 3COL protocol using commitments is a computational ZK proof for 3COL.
3. 3COL is NP-complete: for any $L \in \text{NP}$, there is a poly-time reduction $x \mapsto G_x$ such that $x \in L \iff G_x \in \text{3COL}$.
4. To prove $x \in L$: reduce to G_x, run the ZK proof for G_x. $\square$

From Interactive to Non-Interactive ZK

Motivation

Can the prover send a **single message** to the verifier (no interaction)?

From Interactive to Non-Interactive ZK

Motivation

Can the prover send a **single message** to the verifier (no interaction)?

Common Reference String (CRS) model

Both prover and verifier have access to a common random public string σ (trusted to be random by both).

From Interactive to Non-Interactive ZK

Motivation

Can the prover send a **single message** to the verifier (no interaction)?

Common Reference String (CRS) model

Both prover and verifier have access to a common random public string σ (trusted to be random by both).

NIZK = non-interactive zero-knowledge proof in the CRS model.

NIZK: Definition

Definition 5 (NIZK Proof System)

A NIZK for language L is efficient algorithms (K, P, V):

1. **Key generation**: $\sigma \leftarrow K(1^k)$ generates the CRS.
2. **Prover**: $\pi \leftarrow P(\sigma, x, \omega)$ produces a proof (given witness ω).
3. **Verifier**: $V(\sigma, x, \pi) \in \{0, 1\}$ accepts or rejects.

NIZK: Properties

Completeness

$\forall x \in L,\ \forall \omega \in R_L(x)$:

$$\Pr[\sigma \leftarrow K(1^k),\ \pi \leftarrow P(\sigma, x, \omega) : V(\sigma, x, \pi) = 1] = 1.$$

NIZK: Properties

Completeness

$\forall\, x \in L,\ \forall\, \omega \in R_L(x)$:

$$\Pr[\sigma \leftarrow K(1^k),\ \pi \leftarrow P(\sigma, x, \omega) : V(\sigma, x, \pi) = 1] = 1.$$

Non-adaptive soundness

$\forall\, x \notin L$:

$$\Pr[\sigma \leftarrow K(1^k) : \exists\, \pi \text{ s.t. } V(\sigma, x, \pi) = 1] = \mathsf{negl}(k).$$

The cheating prover must fix x **before** seeing σ.

NIZK: Properties

Completeness

$\forall\, x \in L,\ \forall\, \omega \in R_L(x)$:

$$\Pr[\sigma \leftarrow K(1^k),\ \pi \leftarrow P(\sigma, x, \omega) : V(\sigma, x, \pi) = 1] = 1.$$

Non-adaptive soundness

$\forall\, x \notin L$:

$$\Pr[\sigma \leftarrow K(1^k) : \exists\, \pi \text{ s.t. } V(\sigma, x, \pi) = 1] = \mathsf{negl}(k).$$

The cheating prover must fix x **before** seeing σ.

Adaptive soundness (stronger)

$$\Pr[\sigma \leftarrow K(1^k) : \exists\, (x, \pi) \text{ s.t. } x \notin L \text{ and } V(\sigma, x, \pi) = 1] = \mathsf{negl}(k).$$

The cheating prover may choose x **after** seeing σ.

NIZK: Zero-Knowledge Property

Non-adaptive zero-knowledge

$\exists$ PPT simulator S such that $\forall\, x \in L$, $\omega \in R_L(x)$, the following are computationally indistinguishable:

Real:

1. $\sigma \leftarrow K(1^k)$
2. $\pi \leftarrow P(\sigma, x, \omega)$
3. Output (σ, π)

Simulated:

1. $(\sigma, \pi) \leftarrow S(1^k, x)$
2. Output (σ, π)

NIZK: Zero-Knowledge Property

Non-adaptive zero-knowledge

$\exists$ PPT simulator S such that $\forall\, x \in L$, $\omega \in R_L(x)$, the following are computationally indistinguishable:

Real:

1. $\sigma \leftarrow K(1^k)$
2. $\pi \leftarrow P(\sigma, x, \omega)$
3. Output (σ, π)

Simulated:

1. $(\sigma, \pi) \leftarrow S(1^k, x)$
2. Output (σ, π)

Key point

The simulator produces **both** σ and π together. If S could not generate σ, the definition would be trivial (verifier could run S itself instead of interacting with P).

Adaptive ZK

Adaptive zero-knowledge

$\exists$ PPT simulator (S_1, S_2) such that $\forall$ PPT $\mathcal{A}$, the following are computationally indistinguishable:

Real:

1. $\sigma \leftarrow K(1^k)$
2. $(x, \omega) \leftarrow \mathcal{A}(\sigma)$ with $(x, \omega) \in R_L$
3. $\pi \leftarrow P(\sigma, x, \omega)$
4. Output (σ, x, π)

Simulated:

1. $(\sigma, \tau) \leftarrow S_1(1^k)$
2. $(x, \omega) \leftarrow \mathcal{A}(\sigma)$
3. $\pi \leftarrow S_2(\sigma, x, \tau)$
4. Output (σ, x, π)

Adaptive ZK

Adaptive zero-knowledge

$\exists$ PPT simulator (S_1, S_2) such that $\forall$ PPT $\mathcal{A}$, the following are computationally indistinguishable:

Real:

1. $\sigma \leftarrow K(1^k)$
2. $(x, \omega) \leftarrow \mathcal{A}(\sigma)$ with $(x, \omega) \in R_L$
3. $\pi \leftarrow P(\sigma, x, \omega)$
4. Output (σ, x, π)

Simulated:

1. $(\sigma, \tau) \leftarrow S_1(1^k)$
2. $(x, \omega) \leftarrow \mathcal{A}(\sigma)$
3. $\pi \leftarrow S_2(\sigma, x, \tau)$
4. Output (σ, x, π)

Here τ is local state ("trapdoor") passed from S_1 to S_2: the simulator sets up σ in a way that lets it later produce fake proofs for any x chosen by $\mathcal{A}$.

Non-Adaptive $\Rightarrow$ Adaptive Soundness

Theorem 6

Given a NIZK (K, P, V) that is non-adaptively sound, we can construct a NIZK (K', P', V') that is adaptively sound.

Non-Adaptive $\Rightarrow$ Adaptive Soundness

Theorem 6

Given a NIZK (K, P, V) that is non-adaptively sound, we can construct a NIZK (K', P', V') that is adaptively sound.

Construction

Repeat (K, P, V) polynomially many times with fresh randomness. V' accepts iff V accepts in **every** repetition. Ensure soundness error $\leq 2^{-2P(k)}$ (where $P(k)$ bounds $|x|$).

Non-Adaptive $\Rightarrow$ Adaptive Soundness

Theorem 6

Given a NIZK (K, P, V) that is non-adaptively sound, we can construct a NIZK (K', P', V') that is adaptively sound.

Construction

Repeat (K, P, V) polynomially many times with fresh randomness. V' accepts iff V accepts in **every** repetition. Ensure soundness error $\leq 2^{-2P(k)}$ (where $P(k)$ bounds $|x|$).

Proof

Call σ "bad for x_0" if $\exists\, \pi$ s.t. $V(\sigma, x_0, \pi) = 1$ (and $x_0 \notin L$). By non-adaptive soundness: $\Pr_\sigma[\text{bad for } x_0] \leq 2^{-2P(k)}$. By union bound over all $x \in \{0,1\}^{P(k)}$:

$$\Pr[\sigma \leftarrow K'(1^k) : \exists\, (x, \pi) \text{ s.t. } V'(\sigma, x, \pi) = 1] \leq 2^{P(k)} \cdot 2^{-2P(k)} = 2^{-P(k)}. \square$$