CS555 Cryptography Notes - Lecture 16

Zero-Knowledge Proofs

October 21, 2025

1 How to Define Zero-Knowledge

1.1 What Does the Verifier Learn?

After the interaction, V knows:

- The theorem is true
- A view of the interaction (= transcript + coins of V)

P **gives zero knowledge to** V**:** When the theorem is true, the view gives V nothing that he couldn't have obtained on his own without interacting with P.

Definition 1. (P, V) is zero-knowledge if V can "simulate" his VIEW of the interaction all by himself in probabilistic polynomial time.

1.2 The Simulation Paradigm

For the Quadratic Residuosity (QR) protocol:
$\text{view}_V(P, V) : (s, b, z)$ where $s = z^2 \bmod N$

- If $b = 0$, then $z = x$ (check: $s = x^2 = z^2$)
- If $b = 1$, then $z = xy$ (check: $z^2 = sy^1 \bmod N$)

Simulator: Produces (s, b, z) that is indistinguishable from view_V.

1.3 Formal Definitions

1.3.1 Honest-Verifier Zero-Knowledge

Definition 2. An Interactive Protocol (P, V) is **honest-verifier perfect zero-knowledge** for a language L if there exists a PPT simulator S such that for every $x \in L$, the following two distributions are identical:

1. $\text{view}_V(P, V)(x)$
2. $S(x, 1^\lambda)$

1.3.2 Malicious-Verifier Zero-Knowledge

Definition 3. An Interactive Protocol (P, V) is **perfect zero-knowledge** for a language L if for every PPT V^*, there exists an (expected) polynomial-time simulator S such that for every $x \in L$, the following two distributions are identical:

1. $\text{view}_{V^*}(P, V^*)(x)$
2. $S(x, 1^\lambda)$

1.4 Simulator for Malicious Verifier (QR Protocol)

Simulator S works as follows:

1. First set $s = z^2/y^b$ for a random z and feed s to V^*
2. Let $b' = V^*(s)$
3. If $b' = b$, output (s, b, z) and stop
4. Otherwise, go back to step 1 and repeat (called "rewinding")

Lemma 1. *S runs in expected polynomial time and when it outputs a view, it is identically distributed to the view of V^* in a real execution.*

1.5 What Makes Zero-Knowledge Possible?

1. Each statement has multiple proofs of which the **prover** chooses one at random
2. Each such proof is made of two parts: seeing either one on its own gives the verifier no knowledge; seeing both imply 100% correctness
3. Verifier chooses to see either part, at random. The prover's ability to provide either part on demand convinces the verifier

2 Zero-Knowledge Proof Systems

Definition 4. An Interactive Protocol (P, V) is a **perfect/statistical/computational zero-knowledge proof system** for a language L if it is:

(a) **Complete**

(b) **Sound**

(c) **Zero knowledge:** for every PPT V^*, there exists an (expected) poly-time simulator S s.t. for every $x \in L$, the following two distributions are identical/statistically close/computationally close:

   (a) $\text{view}_{V^*}(P, V^*)(x)$
   (b) $S(x, 1^\lambda)$

3 Zero-Knowledge Proof for Graph Isomorphism

3.1 Protocol

Common Input: Graphs G and H where $H = \pi(G)$ for some isomorphism π

Prover:

- Choose random permutation ρ
- Compute $K = \rho(G)$
- Send K to verifier

Verifier: Send random challenge bit $b \in \{0, 1\}$

Prover's response:

- If $b = 0$: send $\pi_0 = \rho$ such that $K = \pi_0(G)$
- If $b = 1$: send $\pi_1 = \pi \circ \rho^{-1}$ such that $H = \pi_1(K)$

3.2 Properties

Completeness: When G and H are isomorphic and the prover knows π, the prover can always answer both challenges correctly.

Theorem 1 (Soundness)**.** *Suppose G and H are non-isomorphic, and a prover could answer both the verifier challenges. Then, $K = \pi_0(G)$ and $H = \pi_1(K)$. In other words, $H = \pi_1 \circ \pi_0(G)$, a contradiction!*

Zero Knowledge: The protocol is zero-knowledge (shown via simulation).

3.3 Efficient Prover Given a Witness

In both the QR and Graph Isomorphism protocols, the (honest) prover is actually polynomial-time given the NP witness:

- The square root of y in the case of QR
- The isomorphism π in the case of Graph Isomorphism

Soundness is nevertheless against any, even computationally unbounded, prover P^*.

4 Zero-Knowledge Proofs for All of NP

4.1 Limitations of Perfect Zero-Knowledge

Theorem 2 (Fortnow'89, Aiello-Hastad'87)**.** *Not all NP languages have perfect ZK proofs, unless bizarre stuff happens in complexity theory (technically: the polynomial hierarchy collapses).*

4.2 Computational Zero-Knowledge for All of NP

Theorem 3 (Goldreich-Micali-Wigderson'87)**.** *Assuming one-way functions exist, all of NP has computational zero-knowledge proofs.*

This theorem is amazing: it tells us that everything that can be proved (in the sense of Euclid) can be proved in zero knowledge!

5 Zero-Knowledge Proof for 3-Coloring

5.1 The 3-Coloring Problem

NP-Complete Problem: Every other problem in NP can be reduced to 3-Coloring.

5.2 Protocol

Common Input: Graph $G = (V, E)$
Prover:

1. Come up with a random permutation of the colors $\rho : V \to \{R, B, G\}$
2. Commit to each vertex color: $\{\text{Com}(\rho(k); r_k)\}_{k=1}^{n}$
3. Send all commitments to verifier

Verifier: Choose random edge $(i, j) \in E$
Prover: Send openings $(\rho(i), r_i)$ and $(\rho(j), r_j)$
Verifier: Check:

1. Check the openings are valid
2. Check: $\rho(i), \rho(j) \in \{R, B, G\}$
3. Check: $\rho(i) \neq \rho(j)$

5.3 Properties

Completeness: Exercise.

Theorem 4 (Soundness)**.** *If the graph is not 3-colorable, in every 3-coloring (that P commits to), there is some edge whose endpoints have the same color. V will catch this edge and reject with probability $\geq 1/|E|$.*

Amplification: Repeat $|E| \cdot \lambda$ times to get the verifier to accept a false statement with probability $\leq (1 - 1/|E|)^{|E| \cdot \lambda} \leq 2^{-\lambda}$.

6 Commitment Schemes

6.1 Introduction

We need a commitment scheme (aka a "promise hiding scheme").

6.2 Definition

Parties: Sender S and Receiver R

Commitment Protocol:

$$(DEC, COM) \leftarrow (S(b, 1^\lambda), R(1^\lambda))$$

Protocol Flow:

1. S sends COM to R
2. S sends (b, DEC) to R
3. R outputs ACCEPT or REJECT

6.3 Properties

6.3.1 1. Completeness

R always accepts in an honest execution.

6.3.2 2. Computational Hiding

For every possibly malicious (PPT) R^*,

$$\text{view}_{R^*}(S(0), R^*) \approx_c \text{view}_{R^*}(S(1), R^*)$$

Intuition: The locked box should completely hide b.

6.3.3 3. Perfect Binding

For every possibly malicious S^*, let COM be the receiver's output in an execution of (S^*, R). There is no pair of decommitments DEC_0, DEC_1 s.t. R accepts both $(COM, 0, DEC_0)$ and $(COM, 1, DEC_1)$.

Intuition: Sender shouldn't be able to open to $1 - b$.

6.4 Construction from One-Way Permutations

Let f be a one-way permutation and HCB be a hardcore bit.

Commitment:

$$COM = (f(r), HCB(r) \oplus b)$$

Decommitment:

$$DEC = r$$

Opening: (b, r)

Verification: Let $COM = (x, y)$. Check that:

1. $f(r) = x$ and
2. $HCB(r) \oplus b = y$

If both conditions hold, output ACCEPT. Otherwise, output REJECT.
Security Properties:

1. **Completeness:** Exercise.
2. **Computational Hiding:** By the hardcore bit property.
3. **Perfect Binding:** Because f is a permutation.

7 Why is 3-Coloring Zero-Knowledge?

7.1 Hybrid Argument

We show zero-knowledge using a sequence of hybrids that transform the simulator into the real protocol.
Key difference between hybrids: How vertices are colored before commitment.

7.1.1 Hybrid 0 (Simulator)

Simulator S works as follows:

1. First pick a random edge (i^*, j^*)
2. **Color vertices i^* and j^* with random, different colors. Color all other vertices red.**
3. Feed the commitments of the colors to V^* and get edge (i, j)
4. If $(i, j) \neq (i^*, j^*)$, go back and repeat (rewinding)
5. If $(i, j) = (i^*, j^*)$, output the commitments and openings r_i and r_j as the simulated transcript

7.1.2 Hybrid 1 (Not-a-Simulator)

1. First pick a random edge (i^*, j^*)
2. **Permute a legal coloring and color all vertices correctly.**
3. Feed the commitments of the colors to V^* and get edge (i, j)
4. If $(i, j) \neq (i^*, j^*)$, go back and repeat
5. If $(i, j) = (i^*, j^*)$, output the commitments and openings r_i and r_j as the simulated transcript

7.1.3 Hybrid 2 (Real View)

Here is the real view of V^*:

1. First pick a random edge (i^*, j^*)
2. **Permute a legal coloring and color all edges correctly.**
3. Feed the commitments of the colors to V^* and get edge (i, j)
4. If $(i, j) \neq (i^*, j^*)$, go back and repeat
5. If $(i, j) = (i^*, j^*)$, output the commitments and openings r_i and r_j as the transcript

7.2 Indistinguishability of Hybrids

Claim 1. *Hybrids 0 and 1 are computationally indistinguishable, assuming the commitment scheme is computationally hiding.*

Proof: By contradiction. Show a reduction that breaks the hiding property of the commitment scheme, assuming there is a distinguisher between hybrids 0 and 1.

Claim 2. *Hybrids 1 and 2 are identical.*

Proof: Hybrid 1 merely samples from the same distribution as Hybrid 2 and, with probability $1 - 1/|E|$, decides to throw it away and resample.

7.3 Main Result

Lemma 2.

1. *Assuming the commitment is hiding, S runs in expected polynomial-time.*
2. *When S outputs a view, it is computationally indistinguishable from the view of V^* in a real execution.*

Theorem 5. *The 3COL protocol is zero knowledge.*

8 Examples of NP Assertions

- **Well-formed public keys:** My public key is well-formed (e.g. in RSA, the public key is N, a product of two primes together with an e that is relatively prime to $\phi(N)$.)
- **Encrypted cryptocurrency:** "I have enough money to pay you." (e.g. I will publish an encryption of my bank account and prove to you that my balance is $\geq \$X$.)
- **Running programs on encrypted inputs:** Given $\text{Enc}(x)$ and y, prove that $y = \text{PROG}(x)$.
- **More generally:** A tool to enforce honest behavior without revealing information.