Lecture 9

Gidon Rosalki

2026-01-14

Notice: If you find any mistakes, please open an issue at `https://github.com/robomarvin1501/notes_intro_to_crypto`

1 Introduction

Zero knowledge proofs are proofs that reveal nothing beyond the validity of the assertion being proved. This seems to be a contradicting definition, but we will see that it is not. They were introduced by Goldwasser, Micali, and Rackoff in 1985, and are a central tool in cryptographic protocols. We are not going to pass over the entire powerpoint presentation, this is simply going to be an introductory lecture.
Consider for example proving that I have 1000NIS in my pocket. I could prove by taking it out and counting it, but I could potentially want to prove it without showing you what types of notes I have.
A less contrived example would be finding a mathematical proof, which also has monetary value. I do not want to publish this proof, since then I no longer have the monetary value of the secret (suppose it is an algorithm, which you do not want everyone else to be able to implement). So, zero knowledge proofs can be helpful here, to show that I have proven the theorem, but not distribute the solution to everybody.

This is often thought of as an interactive process (it does not have to be, but it can be easier to think of it as such, and so we will begin by doing so).
A classic example is proving that you have solved a sudoku, without demonstrating the result. This is classic, and there are papers that demonstrate it, that are supposedly very readable, and easy to understand.

Sudoku is a slightly contrived example, since it is very clearly in P, consider instead the 3 colour problem. Can Alice prove that the graph is 3 colourable without revealing the 3 colouring? Well, hopefully we will discuss that later. Furthermore, given $n = pq$, we can show that Alice knows how to factor n, without revealing p or q.

2 Interactive proofs

We will begin by defining classic NP problems. NP is the class of

Definition 2.1 (NP)**.** *All languages L, equipped with an efficiently computable relation R_L such that*

$$x \in L \Leftrightarrow \exists w : (x, w) \in R_L \wedge |w| = poly(|x|)$$

So, given a statement $x \in L$, and proof w, NP proofs are inherently non-ZK, since Bob gains the ability to prove $x \in L$ to others.

Let us extend this with two new ideas:

- Interaction: Replace a static proof with an interactive protocol
- Randomisation: Allow the verifier to toss coins, and err with a small probability

We will create the prover $\mathcal{P}$, which has the random tape $r_{\mathcal{P}}$, and the verifier $\mathcal{V}$, with the random tape $r_{\mathcal{V}}$. We will define the notation:

- $\langle \mathcal{P}, \mathcal{V} \rangle (x)$ is the distribution of the transcript of the protocol
- $\text{out}_{\mathcal{V}} [\langle \mathcal{P}, \mathcal{V} \rangle (x)]$ is the distribution of $\mathcal{V}$'s output

Definition 2.2 (Interactive proof system)**.** *An **interactive proof system** for a language L is a protocol $\langle \mathcal{P}, \mathcal{V} \rangle$ where $\mathcal{V}$ is computable in probabilistic polynomial time, and the following holds:*

- *Completeness: For every $x \in L$:*
$$\Pr_{r_{\mathcal{P}}, r_{\mathcal{V}}} [out_{\mathcal{V}} [\langle \mathcal{P}, \mathcal{V} \rangle (x)] = Accept] = 1$$
- *Soundness: For every $x \notin L$, and for every computationally unbounded $\mathcal{P}^*$:*
$$\Pr_{r_{\mathcal{P}}, r_{\mathcal{V}}} [out_{\mathcal{V}} [\langle \mathcal{P}^*, \mathcal{V} \rangle (x)] = Accept] \leq \frac{1}{2}$$

We will state that **IP** is the class of all languages with an interactive proof system. IP contains NP, and in fact, $IP = PSPACE$. We can reduce the soundness error from $\frac{1}{2}$ to ε with $\log\left(\frac{1}{\varepsilon}\right)$ independent repetitions.

Definition 2.3 (Isomorphic)**.** *Two graphs $G_0 = (V_0, E_0)$, and $G_1 = (V_1, E_1)$ are **isomorphic** if there exists a one to one mapping $\pi : V_0 \to V_1$ such that $(u, v) \in E_0 \Leftrightarrow (\pi(u), \pi(v)) \in E_1$ for every $e, v \in V_0$*

That is to say, two graphs are isomorphic if we can rename the nodes in order to transform one into the other. We will note that this problem is in NP, since we can trivially build a verifier, but beyond this, we know nothing. We know not if it is NP-hard, NP-complete, or in P.
We can define the set of isomorphic graphs $GI = \{(G_0, G_1) : G_0 \text{ is isomorphic to } G_1\} \in NP$. Similarly, we can define the other class of graphs that are **not** isomorphic: $GNI = \{(G_0, G_1) : G_0 \text{ is } \textbf{not} \text{ isomorphic to } G_1\} \in NP$. This class is not known to be in NP.
We can now ask the question as to how to prove to an efficient verifier that G_0, G_1 are not isomorphic. we will posit that GNI $\in IP$: Given the common input (G_0, G_1), the prover, and the verifier, the verifier will try and find the points where (if) the prover is guessing.
Intuitive solution: The verifier will create two new graphs, which are permutations on the originals, $\pi(G_0), \pi(G_1)$. The verifier will also create $\pi(G_b) : b \leftarrow \{0, 1\}$. If the graphs are isomorphic, then all the graphs, including the new ones, are also isomorphic. If they are not, then $\pi(G_0)$ is not isomorphic to $\pi(G_1)$, and $\pi(G_b)$ will be isomorphic to **one** of them.
Formal: The verifier will create $H = \pi(G_b)$ for a random permutation π, and $b \leftarrow \{0, 1\}$. H is then sent to the prover. The prover will then find $z \in \{0, 1\}$, such that H is isomorphic to G_z, and will then respond with z. The verifier will accept **if and only if** $z = b$.

Theorem 1. *This protocol is an interactive proof for GNI*

Proof . Firstly, we will claim completeness. If $(G_0, G_1) \in$ GNI, then the verifier always accepts, since no graph can be isomorphic to both G_0, and G_1.
Next, soundness, if $(G_0, G_1) \notin$ GNI, then for every $\mathcal{P}^*$ the verifier accepts with probability $\frac{1}{2}$. This is true since $\mathcal{P}^*$'s view is independent of b, so therefore for any $z \in \{0, 1\}$ that $\mathcal{P}^*$ will output, we have $\Pr_{b \leftarrow \{0,1\}}[z = b] = \frac{1}{2}$. □

3 Zero knowledge proofs

An interactive proof system is zero-knowledge if whatever can be efficiently computed after interacting with $\mathcal{P}$ on input $x \in L$ can also be computed given only x. This should be true even when $\mathcal{P}$ is interacting with a malicious verifier.

Let us return to the isomorphic classes. Can we prove that G_0 and G_1 are isomorphic without revealing the isomorphism? A solution needs to enable completeness, soundness, and zero knowledge (ZK). Hear me and rejoice, for we are able so to do. Gird thy loins, and behold the following solution:
Intuition: Two graphs are isomorphic if there exists a permutation that transforms from one to the other (π). There therefore also exists π^{-1}, which transforms in the opposite direction. Let us consider another permutation σ, which transforms G_0 to H. Should G_0 and G_1 be isomorphic, then there also exists σ' which transforms from G_1 to H. We can send the transformed graph $H = \sigma(G_0)$ to the prover, and it will respond with the permutation that transforms G_1 to H. This is ZK, since it teaches us nothing on the permutations π, π^{-1}, but demonstrates that the prover can find these permutations.
Formal: The prover will sample a random permutation σ, and send $H = \sigma(G_0)$ to the verifier. The verifier then responds with a request that the prover show that H is isomorphic to G_b, for $b \leftarrow \{0, 1\}$. The prover will then respond with

$$\gamma = \begin{cases} \sigma, & \text{if } b = 0 \\ \sigma \circ \pi^{-1}, & \text{if } b = 1 \end{cases}$$

Which provides the required transformation to H for G_b. The verifier then accepts **if and only if** $\gamma(G_b) = H$.
Correctness: If the two graphs are isomorphic, then the verifier will trivially receive what it requested.
ZK: We need to show that the prover did not leak information. Since the only leak that can happen is $\sigma \circ \pi^{-1}$, but since we have further transformed π^{-1} with σ, it acts sort of like a one time pad, and so it does not leak π.
Soundness: The two graphs are not isomorphic, so the prover sends some graph, it can be a transformation of G_0, or G_1, or perhaps some other graph G_5, and in every case, the prover will make a mistake with probability of $\frac{1}{2}$, as required.

4 Zero knowledge proofs for NP

Theorem 2. *Assuming that OWFs exist, then any $L \in NP$ has a zero knowledge interactive proof. Furthermore, the prover's strategy can be implemented in probabilistic polynomial time, provided an NP witness for membership of the ocmmon input.*

Steps. 1. Construct a ZK proof for some NP complete language. Use G3C (graph 3 colouring), and the tool commitments schemes (which are based on OWFs)

2. Given any NP language L, a common input x, and a witness w, we reduce them to G3C, and then use the above ZK proof.

Since we showed reductions in computability, and complexity, we will focus on step one, and commitments schemes $\square$

4.1 Tool: Commitment schemes

These are the basic ingredient in many cryptographic protocols. They are a digital analogue of locked boxes. The sender S has a value v, and a random tape r_S. The sender sends the commit phase to the receiver, and receives a protocol $\langle S, R\rangle$ in response. It sends (v, r_S) in the reveal phase, and the receiver R accepts v **if and only if** (v, r_S) are consistent with the commit phase. Commitment schemes have the following security requirements:

- **Hiding**: At the end of the commit stage, the receiver has no knowledge of v
- **Binding**: The sender cannot find two valid openings (v, r_S) and (v', r'_S) for $v \neq v'$

Definition 4.1 (Hiding)**.** *A commitment scheme $\langle S, R\rangle$ is* ***computationally hiding*** *if for every PPT receiver R^* and for every two values $v \neq v'$ it holds that*

$$\mathit{view}_{R^*}\left[\langle S(v), R^*\rangle(1^n)\right] \approx^c \mathit{view}_{R^*}\left[\langle S(v'), R^*\rangle(1^n)\right]$$

Perfect (statistical) hiding is when $\mathit{view}_{R^*}\left[\langle S(v), R^*\rangle(1^n)\right]$ and $\mathit{view}_{R^*}\left[\langle S(v'), R^*\rangle(1^n)\right]$ are identical (statistically indistinguishable) for any ***unbounded*** *R^**

Definition 4.2 (Binding)**.** *A commitment scheme $\langle S, R\rangle$ is* ***computationally binding*** *if for every PPT sender S^* there exists a negligible function $v(n)$ such that*

$$\Pr\left[((v, r, v', r'), \mathit{com}) \leftarrow \langle S^*, R\rangle(1^n) : v \neq v' \wedge (v, r) \text{ is consistent with com } \wedge (v', r') \text{ is consistent with com}\right]$$

where

$$\mathit{com} \overset{def}{=} \mathit{view}_R\left[\langle S^*, R\rangle(1^n)\right]$$

4.1.1 Some applications of commitments

Consider if you are playing some board game over the phone, which involves throwing dice. Your opponent tells you that they threw 6. Commitments can be used to to ensure that this is the truth. Let us simplify this somewhat into coin flipping, neither party wants to speak first. Alice may send a commitment of what she tossed, Bob then responds with his toss. Alice then sends the reveal for her toss, which Bob can then verify, so this way, Alice could not change her result, and Bob can believe it.

Let us return to ZK proofs for NP.

Definition 4.3. *A graph $G = (V, E)$ is 3 colourable if there exists a mapping $\varphi : V \to \{1, 2, 3\}$ such that $\varphi(u) \neq \varphi(v)$ for every $(u, v) \in E$.*

We want to prove that G is 3 colourable, without revealing a 3 colouring. The high level idea is to break that G is 3-colourable into polynomially many pieces. Each piece does not reveal any information, but combining all the pieces yields a proof that G is 3-colourable. We can implement this using commitments.

Solution: Given the common input $G = (V, E)$, and an auxiliary input to the prover, which is a 3 colouring $\psi : V \to \{1, 2, 3\}$. The protocol is as follows:

- $\mathcal{P}$ uniformly chooses a permutation π over $\{1, 2, 3\}$, and lets $\varphi \overset{def}{=} \pi \circ \psi$
- $\mathcal{P}$ commits to the value $\varphi(w)$ for every $w \in V$ using a statistically binding commitment
- $\mathcal{V}$ uniformly chooses an edge $(u, v) \in E$, and sends it to $\mathcal{P}$
- $\mathcal{P}$ revelas the openings of $\varphi(u)$, and $\varphi(v)$
- $\mathcal{V}$ accepts **if and only if** the openings are valid, i.e. $\varphi(u), \varphi(v) \in \{1, 2, 3\} \wedge \varphi(u) \neq \varphi(v)$

This protocol is repeated $t \cdot |E|$ times for soundness e^{-t}.

Since we have shown ZK proofs for G3C, and there exist reductions for every language in NP, we can thus show a ZK proof for every language in NP.