Focus thus far in the course: protecting communication (e.g., message confidentiality and message integrity)

Next few weeks: protecting computations.

Zero-knowledge: a defining idea at the heart of theoretical cryptography
- Idea will seem very counter-intuitive, but surprisingly powerful
- Showcases the importance and power of definitions (e.g., “What does it mean to know something?”)

We begin by introducing the notion of a “proof system”
- Goal: A prover wants to convince a verifier that some statement is true
  - e.g., “This Sudoku puzzle has a unique solution”
  - “The number N is a product of two prime numbers p and q”
  - “I know the discrete log of x to base g”
  {there are all examples of statements}

We model this as follows:

\[ \text{prover}(X) \xrightarrow{\pi} \text{verifier}(X) \]
\[ X: \text{statement that the prover is trying to prove (known to both prover and verifier)} \]
\[ \pi: \text{the proof of } X \]
\[ b \in \{0,1\}^* \quad \text{given statement } X \text{ and proof } \pi, \text{ verifier decides whether to accept or reject} \]

Properties we care about:
- Completeness: Honest prover should be able to convince honest verifier of true statements
  \[ \forall x \in L: \Pr[\pi \leftarrow P(x) : V(x, \pi) = 1] = 1 \]
- Soundness: Dishonest prover cannot convince honest verifier of false statement
  \[ \forall x \notin L: \Pr[\pi \leftarrow P(x) : V(x, \pi) = 1] < \frac{2}{3} \]
  [Important: we are not restricting to efficient prover]

Typically, proofs are “one-shot” (i.e., single message from prover to verifier) and the verifier’s decision algorithm is deterministic
- Languages with these types of proof systems precisely coincide with NP (proof of statement x is to send NP witness w)

Going beyond NP: we augment the model as follows
- Add randomness: the verifier can be a randomized algorithm
- Add interaction: verifier can ask “questions” to the prover

Interactive proof systems [Goldwasser-Micali-Rackoff]:

\[ \text{prover}(X) \xrightarrow{\pi} \xleftarrow{} \text{verifier}(X) \]
\[ \xrightarrow{\pi} \xrightarrow{\pi} \xrightarrow{\pi} \xrightarrow{\pi} \]
\[ \xleftrightarrow{\pi} \xleftrightarrow{\pi} \]
\[ \xrightarrow{\pi} \xrightarrow{\pi} \xrightarrow{\pi} \xrightarrow{\pi} \]
\[ b \in \{0,1\}^* \]

Set of languages that have an interactive proof system is denoted IP.

Theorem (Shamir): IP \subseteq PSPACE

\text{languages that can be decided in polynomial space [very large class of languages!]}
Takeaway: interaction and randomness is very useful

⇒ In fact, enables a new property called zero-knowledge

Consider following example: Suppose prover wants to convince verifier that \( N = p \ q \) where \( p, q \) are prime (and secret).

\[
\begin{align*}
\text{prover } \left( N, p, q \right) & \quad \text{ verifier } \left( N \right) \\
\Rightarrow \quad N = p \ q & \quad \downarrow \text{ accept if } N = p \ q \text{ and reject otherwise.}
\end{align*}
\]

Proof is certainly complete and sound, but now verifier also learned the factorization of \( N \) (may not be desirable if prover was trying to convince verifier that \( N \) is a proper RSA modulus (for a cryptographic scheme) without revealing factorization in the process

⇒ In some sense, this proof conveys information to the verifier [i.e., verifier learns something it did not know before seeing the proof]

Zero-knowledge: ensure that verifier does not learn anything (other than the fact that the statement is true)

How do we define “zero-knowledge”? We will introduce a notion of a “simulator.”

**Definition.** An interactive proof system \( \langle P, V \rangle \) is zero-knowledge if for all efficient (and possibly malicious) verifiers \( V^* \), there exists an efficient simulator \( S \) such that for all \( x \in L \):

\[
\text{View} \langle P, V^* \rangle (x) \approx S(x)
\]

Random variable denoting the set of messages sent and received by \( V^* \) when interacting with the prover \( P \) on input \( x \)

What does this definition mean?

\[
\text{View} \langle P, V^* \rangle (x) : \text{ this is what } V^* \text{ sees in the interactive proof protocol with } P
\]

\[
S(x) : \text{ this is a function that only depends on the statement } x, \text{ which } V^* \text{ already has}
\]

If these two distributions are indistinguishable, then anything that \( V^* \) could have learned by talking to \( P \), it could have learned just by invoking the simulator itself, and the simulator output only depends on \( x \), which \( V^* \) already knows.

⇒ In other words, anything \( V^* \) could have learned (i.e., computed) after interacting with \( P \), it could have learned without ever talking to \( P \)!

Very remarkable definition!

More remarkable: If one-way functions exist, then every language \( L \in \text{IP} \) has a zero-knowledge proof system. 

⇒ Namely, anything that can be proved can be proved in zero-knowledge!

We will state this theorem for \( \text{NP} \) languages. Here, it suffices to construct a single zero-knowledge proof system for an \( \text{NP} \)-complete language. We will consider the language of graph 3-colorability.

3-coloring: given a graph \( G \), can you color the vertices so that no adjacent nodes have the same color?
We will need a commitment scheme (see [H02]). A (non-interactive) commitment scheme consists of two main algorithms (Commit, Verify):

- Commit((r,n) → (c,r)): Takes a message m and outputs the commitment c and an opening r.
- Verify(m,c,r) → b: Checks if c is a valid opening to m (with respect to opening r).

[The commitment scheme might also take public parameters (see [H02]), but for simplicity, we omit them / leave them implicit]

Requirements:
- Correctness: for all messages m:
  \[ \Pr[(c,r) → Commit((r,n)): Verify(m,c,r) = 1] = 1 \]
- Hiding: for all efficient adversaries A, if \((m,m') → A(I)\)
  \[ \Pr[(c,r) → Commit((r,n)): c \neq c'] \leq \Pr[(c,r) → Commit((r,n)): c'] \]
- Binding: for all efficient adversaries A, if
  \[ \Pr[(m,m',c,c',r) → A(I)]: m \neq m' \text{ and } \forall m, \exists c, r \text{ such that } Verify(m,c,r) = 1 \] = neg(\varepsilon)

We will require perfect binding [for every commitment c there is only 1 possible m to which the power can open c]

A 2K protocol for graph 3-coloring:

- Let \(K \in \{0,1,2\}\) be a 3-coloring of \(G\).
- Choose random permutation \(\pi \in \text{Perm}([n])\).
- For \(i \in \{n\}\):
  \((c_i,r_i) → \text{Commit}(\pi(K))\)

Verifier (G)

\(\pi(K)\)

\(c_i, \ldots, c_n\)

\((K_i,r_i), (K_j,r_j)\)

\(\langle i, j \rangle \in E\)

\(\Rightarrow\) accept if \(K_i \neq K_j\) and \(K_i, K_j \in \{0,1,2\}\)

\(\Rightarrow\) reject otherwise

Initiative: Power commits to a coloring of the graph
Verifier challenges power to reveal coloring of a single edge
Power reveals the coloring on the chosen edge and opens the entries in the commitment

Completeness: By inspection, [if coloring is valid, power can always answer the challenges correctly]

Soundness: Suppose \(G\) is not 3-colorable. Let \(K_1, \ldots, K_n\) be the coloring the power committed to. If the commitment scheme is perfectly binding, \(c_i, \ldots, c_n\) uniquely determine \(K_i, \ldots, K_n\). Since \(G\) is not 3-colorable, there is an edge \((i,j) \in E\) where \(K_i = K_j\) or \(i \notin \{0,1,2\}\) or \(j \notin \{0,1,2\}\). Otherwise, \(G\) is 3-colorable with coloring \(K_1, \ldots, K_n\). Since the verifier chooses an edge to check at random, the verifier will choose \((i,j)\) with probability \(1/|E|\). Thus, if \(G\) is not 3-colorable,

\[ \Pr[\text{ verifier rejects }] \geq \frac{1}{|E|} \]

Thus, this protocol provides soundness \(1 - \frac{1}{|E|}\). We can repeat this protocol \(O(|E|^3)\) times sequentially to reduce soundness error to

\[ \Pr[\text{ verifier accepts proof of false statement }] \leq \left(1 - \frac{1}{|E|}\right)^{1/2 \cdot |E|} \leq \frac{1}{|E|} = \epsilon \quad \text{[since } (1 - \frac{1}{2})^2 \leq \frac{1}{e}] \]
We need to construct a simulator that outputs a valid transcript given only the graph $G$ as input.

Let $V^*$ be a (possibly malicious) verifier. Construct simulator $S$ as follows:

1. Choose $K_i \leftarrow \{0,1,2,3\}$ for all $i \in [n]$. 
   
   Let $(c_i, r_i) \leftarrow \text{Commit}(1^3, K_i)$.
   
   Give $(c_i, ..., c_n)$ to $V^*$.

2. $V^*$ outputs an edge $(i,j) \in E$

3. If $K_i \neq K_j$, then $S$ outputs $(K_i, K_j, r_i, r_j)$.
   
   Otherwise, restart and try again (it fails $\lambda$ times, then abort)

Simulator succeeds with probability $2/3$ (over choice of $K_i, ..., K_n$). Thus, simulator produces a valid transcript with prob. $1 - \frac{1}{3^2} = 1 - \frac{1}{9}$ after $\lambda$ attempts. It suffices to show that simulated transcript is indistinguishable from a real transcript.

- True scheme: prover opens $K_i, K_j$ where $K_i, K_j \leftarrow \{0,1,2,3\}$ [since prover randomly permutes the colors]
- Simulation: $K_i$ and $K_j$ sampled uniformly from $\{0,1,2,3\}$ and conditioned on $K_i \neq K_j$, distributions are identical.

In addition, $(i,j)$ output by $V^*$ in the simulation is distributed correctly since commitment scheme is computationally-hiding (e.g. $V^*$ behaves essentially the same given commitments to a random coloring as it does given commitment to a valid coloring).

If we repeat this protocol (for soundness amplification), simulator simulate one transcript at a time.