Massachusetts Institute of Technology
6.857: Network and Computer Security
Professors Ronald L. Rivest and Yael Tauman Kalai

Handout R7
April 3rd, 2020

Christos Nestor Chachamis

Recitation 7: Zero Knowledge

1 Definitions

Interactive Protocol: Given a Prover P, a Verifier V, a language L and an input x, P and V hold a conversation so that P proves that $x \in L$. This is called **proof-of-knowledge**, since P proves that a statement is correct or that she knows the solution to a problem. **Zero-knowledge** is a condition that guarantees that the Verifier V gets no other information besides whether or not a statement is correct ($x \in L$). Some examples including P proving that he knows a 3-coloring of a graph, a Hamiltonian cycle of a graph, or the discrete log of a given number, without revealing the specific 3-coloring, Hamiltonian cycle or discrete log, respectively.

Completeness: An honest Verifier "accepts" the proof when the statement is actually correct (e.g. P knows the discrete log).

Soundness: Verifier "rejects" the proof when the statement is incorrect, with high probability (e.g. a cheating P cannot convince V that she knows the discrete log, except with little probability).

Common Assumptions: We usually assume that P has unlimited computation power, while V is bounded by probabilistic polynomial time.

2 Examples

Below there are three examples of zero-knowledge interactive protocols. We assume that we have an honest Verifier.

2.1 Graph 3-Colorability (Review)

Given a graph $G = (V, E)$, the Prover wants to prove that she knows a way to color the vertices using 3 colors such that no two adjacent vertices have the same color. The Prover does not want the Verifier to find any information about the valid coloring, only one exists and P knows it.

The protocol works as follows:

1. P permutes the colors of the valid 3-coloring. Let $c_1, c_2, \ldots, c_n$ be the final values
2. P sends over to V the commitments $\mathrm{com}(c_1), \mathrm{com}(c_2), \ldots, \mathrm{com}(c_n)$
3. V asks for an edge $(i, j) \in E$
4. P reveals c_i, c_j
5. V accepts if, and only if, c_i and c_j are different colors

Completeness: Completeness follows from the fact that a Prover who knows the coloring can answer all requests of an honest Verifier.

Soundness: Consider a cheating Prover that wants to convince Verifier. Since the Prover does not know a 3-coloring, at least one of the edges is monochromatic. Then, Verifier chooses a monochromatic edge with probability at least $|E|^{-1}$. By repeating N times, the probability of Prover convincing the Verifier becomes at most $(1 - |E|^{-1})^N$, which can be arbitrarily small.

Zero Knowledge: Verifier learns only if the edge he chose is monochromatic or not, but does not get any other information about the original graph.

2.2 Hamiltonian Cycle

Given a graph $G = (V, E)$, a Hamiltonian cycle is a cycle that visits every vertex exactly once. Finding whether a Hamiltonian cycle exists is considered an NP-complete problem. Prover wants to prove that he knows a Hamiltonian cycle of graph G, without revealing the actual path. How to achieve this?

Consider the following protocol:

1. P permutes the vertices of G. Let H be the resulting graph
2. P sends over to V the commitments $\mathrm{com}(\pi)$ and $\mathrm{com}(H)$, where π is the permutation of the vertices
3. V replies with $b \in \{0, 1\}$
4. If $b = 0$, then P reveals H and π and V accepts if π is a valid permutation and H is the valid graph implied by the permutation π
5. If $b = 1$, then P reveals **only** a Hamiltonian cycle of H and V accepts if the revealed path is Hamiltonian.

Completeness: Completeness follows from the fact that a Prover who knows the Hamiltonian cycle can answer all requests of an honest Verifier. Notice that given the Hamiltonian cycle and the permutation π, it is easy to find a Hamiltonian cycle in H.

Soundness: Consider a cheating Prover that wants to convince Verifier. Prover cannot construct H and its Hamiltonian cycle at the same time, so he needs to guess whether $b = 0$ or $b = 1$. If he guesses $b = 0$, then he simply permutes G and sends the resulting graph. If he guesses $b = 1$, then he constructs a different H with a Hamiltonian cycle, so he reveals the cycle, but V does not know that H and G are not isomorphic. Prover can guess correctly with probability $\frac{1}{2}$. By repeating N times, the probability of Prover convincing the Verifier drops to 2^{-N}, which can be arbitrarily small.

Zero Knowledge: If $b = 0$, Verifier only sees a permutation of G. If $b = 1$, Verifier only sees a Hamiltonian cycle, but does not know how the cycle is related to G. Thus, he cannot find how to construct the cycle.

2.3 Discrete Log

Given a prime p, a generator g of $\mathbb{Z}_p^*$ and a number $y \in \{1, 2, \ldots, p-1\}$ Prover wants to prove that he knows the discrete log x of y, that is $g^x = y \mod p$, without revealing x.

Consider the following protocol:

1. P chooses a random number $r \in \{1, 2, \ldots, p-1\}$, computes $c = g^r \mod p$ and sends c over to V
2. V replies with a random $u \in \{1, 2, \ldots, p-1\}$
3. P computes $s = r + ux \mod p-1$ and sends it to V
4. V accepts if $g^s = c \cdot y^u \mod p$

Completeness: Completeness follows from the fact that a Prover who knows the discrete log of y can answer all requests of an honest Verifier.

Soundness: Consider a cheating Prover that wants to convince Verifier. If P chooses r randomly, then in the last step she needs to find the discrete log of $c \cdot y^u$, which we assume is a hard problem. If P chooses s, so that $g^s = c \cdot y^u$, then this means that she needs to have guessed u and then choose c accordingly. Thus, Prover cannot find r and s at the same time, without knowing x, except with little probability (guessing u correctly).

Zero Knowledge: Verifier sees $s = r + ux \mod p-1$ and knows u, so he needs r in order to find x. However, that would mean that he could solve the discrete log for c, which is assumed a hard problem.

3 Simulating the Interactive Protocol

The formal argument for Zero Knowledge requires the use of a *Simulator*. Simulators are programs that take an input x and create a "dialog" between two parties P and V. Let $S(x)$ be the dialog created by a Simulator S and $(P,V)(x)$ be an interactive protocol. Then, (P,V) is zero-knowledge if there exists a simulator S such that $S(x) \approx (P,V)(x)$, that is, the distributions of the two outputs are the same. Intuitively, the Simulator has no knowledge of x, so producing the dialogs with the same probability distributions means that the Verifier learns only as much as the Simulator knows, which is nothing.

In class, we saw how to create a simulator for 3-coloring in the case of an honest Verifier. The proposed Simulator works as follows:

1. S chooses randomly a valid edge $e = (i,j) \in E$
2. S chooses the two distinct colors c_i and c_j. All other vertices are colored with the same color
3. S "sends" the commitments of $c_1, \ldots, c_n$ to V. S cannot actually send anything, since V does not exist. He only simulates doing so.
4. S "receives" $e = (i,j)$ from V
5. S reveals c_i and c_j

The output of the above program is valid and has the same distribution as the interactive protocol between P and V. However, we make the assumption that V is an honest Verifier, e.g. he picks the pre-selected edge. A stronger version of zero-knowledge requires the Simulator to run with *any* Verifier, including a malicious one.

In this case, we modify the Simulator as follows:

1. S chooses randomly a valid edge $e = (i,j) \in E$
2. S chooses the two distinct colors c_i and c_j. All other vertices are colored with the same color
3. S "sends" the commitments of $c_1, \ldots, c_n$ to V^*
4. S "receives" an edge e' from V^*
5. If $e' = e$, S reveals c_i and c_j
6. If $e' \neq e$, S does not reveal anything and repeats until some colors are revealed

On expectation, after $|E|$ repetitions S has produced a valid transcript.

4 Applications of Zero Knowledge

Some applications include:

1. ID schemes
2. Electronic Voting
3. Cryptocurrencies
4. Nuclear Disarmament