CS 7880 Graduate Cryptography November 13th 2017

Lecture : Zero Knowledge Proofs

Lecturer: Daniel Wichs *Scribe: Lucianna Kiffer*

1 Topics Covered

- Zero knowledge proofs
- Interactive proofs
- Commitment schemes

2 Informal Example of Zero Knowledge Proofs

Imagine you want to prove to someone you know where Waldo is on a page without revealing his location. You do this because you don't want the proof to be transferable: if you prove to someone you found Waldo, you don't want them using your proof to claim that they were who found Waldo first. You can do this by taking a piece of cardboard 4x the size of the page from the Where's Waldo book, then cutting out just the outline of Waldo's face in the middle of the cardboard. This way you can place the book behind the cardboard exactly where Waldo's face meets the cutout and show to anyone Waldo's face without revealing where the book is placed behind it. Thus anyone can see Waldo, but not where he is, and only someone who knows where Waldo is can place the book correctly behind the cardboard.

Informally, zero-knowledge proofs are proofs that carry no information except for the validity of what you are trying to prove. In this lecture we will see that all NP-languages have zero-knowledge proofs, meaning for each element in the language, we can provide a proof that a certificate exists for that element without revealing the certificate.

3 Zero-knowledge proof model

Definition 1 *Zero-knowledge proofs* are interactive proofs with a PPT prover and verifier (P, V).
Syntax: NP relation $R = R(x, w)$ or NP language L where x is an element of L and w a witness for $x \in L$.
P has inputs x, w and V has input x. P is given the witness w otherwise it might not efficient for P to find w. P and V interact and then V outputs either *accept* or *reject*. The properties we want the proof to satisfy are as follows:

(1) *Completeness*: honest provers can convince honest verifiers of honest claims

$$\forall x \in L \text{ w/ witness } w, \quad \Pr[< P(x, w), V(x) >= acc] = 1$$

where $< P(x,w), V(x) >$ is V's output.

(2)*Soundness*: If x is not in the language, no prover can convience the verifier that x is in the language.

$$\forall x \notin L, \forall(\text{possibly malicious, possibly unbounded } P^*, \quad \Pr[< P^*(x), V(x). = acc] \leq \frac{1}{3}$$

$\diamondsuit$

(1)(2) so far have defined an interactive proof. The idea of an interactive proof is to save on time (both for the prover who needs to find a proof and the verifier who needs to read and verify the proof). Interactive proofs capture the class of all languages that you can prove and use polynomial space (I=PSPACE). The main point being that if you have interaction, you can save on computation. We can amplify soundness by re-running the proof and get the probability bound down to negligible. You can always do this as long as you start with a non-negligible bound).

(3)*Zero-knowledge(zk)*: Imagine an ideal world where we start by knowing x is in L and anything the verifier sees during the proof you could generate just by knowing $x \in L$.
$View_{V^*}(x,w)$ = view of $V^*(x)$ when interacting with a prover $P(x,w)$. $\forall$ possibly malicious PPT verifiers $V^*, \exists$ PPT Sim $s.t. \forall x \in L$,

$$Sim(x) \approx View_{V^*}(x,w)$$

In this lecture we consider an honest verifier zero-knowledge (HVZK) proof model, meaning we want to guarantee that honest verifiers learn nothing. A stronger statement would be to show that there is a single simulator that $\forall V^*$ satisfies the proof. Note that we write $View_{V^*}(x,w)$ as the view of the verifier V^* though the verifier does not see w, the view is dependent on both x and w.

4 Commitment Schemes

Before we proceed with the main theorem of this lecture, we first cover an import tool called *commitment schemes*. We want these schemes to have 3 propers: *perfect binding, computational hiding,* and be *for bits*. We are only interested in committing to single bits since once we can do that, we can extend it for arbitrary length messages. We are not trying to optimize for number of rounds or communication as long as its all PPT.

In the commitment scheme we have a PPT $Com.$ protocol and a polynomial verifier $Ver.$ $Com.$ has inputs $b, 1^\lambda$, and outputs a commitment and decommitment (c,d). $Ver.$ has inputs $b, 1^\lambda, c, d$ and outputs $acc./rej.$. The scheme should satisfy the following properties:

(1)*Correctness* $\forall b \in \{0,1\}$

$$\Pr[Ver(b, 1^\lambda, c, d) = acc \ : \ (c,d) \leftarrow Com(b, 1^\lambda)] = 1$$

Note that the randomness comes from the commitment scheme since the verifier is deterministic.

(2) *Computational hiding*:

$$\mathrm{EXP}_b(\lambda) = \text{run } (c,d) \leftarrow Com(b, 1^\lambda), \text{ output } c$$

$$\{\mathrm{EXP}_0(\lambda)\} \approx \{\mathrm{EXP}_1(\lambda)\}$$

(3) *Perfect binding*: $\nexists c^*, d_0^*, d_1^*$

$$Ver(0, 1^\lambda, c^*, d_0^*) = Ver(1, 1^\lambda, c^*, d_1^*) = acc$$

i.e. the same commitment cannot have two decommitments for two different values.

5 zk-proof for NP

Again, the main goal of this lecture is to prove the following theorem:

Theorem 1 (GMM) *all NP languages have zk-proofs*

We will prove this by showing a proof for a specific NP language (since any other language can be reduced to it). We prove the theorem for the graph 3-coloring problem (G3C).

$$L_{G3C} = \{\text{graph } G : \exists \text{coloring } \varphi \text{ of } G \text{ in 3 colors}\}$$

Proof Idea: If there is a coloring, no matter what edge you wish to see of the coloring, it must be valid. If the prover commits to all colors of each edge and the verifier asks for the color of a random edge, it should be valid. This alone is an interactive proof. To make it zk we add in that the prover permutes the colors.

The proof proceeds with the following steps,

zk-proof: $G = (Vert, E)$, $\varphi = 3$ coloring

(1) P picks a random permutation

$$\Pi : \{1,2,3\} \to \{1,2,3\} \text{ and computes } \varphi' = \Pi \circ \varphi$$

(2) $\forall v \in Vert$, P computes

$$(c_v, d_v) \leftarrow Com(\varphi(v), 1^\lambda)$$

(3) P sends $\{c_v : v \in Vert\}$ to V

(4) V picks a random edge $e = (u,v) \in E$ and sends it to P

(5) P sends $\varphi'(u), \varphi'(v), d_u, d_v$ to V

(6) V checks: $\varphi'(u), \varphi'(v), \in \{1,2,3\}$ and $\varphi'(u) \neq \varphi'(v)$

$$Ver(\varphi'(u), 1^\lambda, c_u, d_u) = Ver(\varphi'(v), 1^\lambda, c_v, d_v) = acc$$

if it all works out then output acc, otherwise rej

Proof
(1)*Completeness*: if φ is a 3-color, then φ' is also a 3 color and $(c_v, d_v) \leftarrow Com(\varphi(v), 1^\lambda)$ and we can show all V checks acc with probability 1.

(2)*Soundness*: Let G be a graph with **no** 3-coloring, we have some prover P^* which behaves arbitrarily but must send some c_v^* such that $\forall v \in Vert$ in step (3). Because of perfect binding, each commitment opens at most 1 value, thus $\exists$ a single $\varphi^*(v)$ that c_v^* can be decommited to. This gives a coloring φ^* and it is not a valid 3-color of G by definition.

Thus $\exists$ some edge e that violates the coloring

$$e = (u, v) \in E \ s.t. \ \varphi^*(u) = \varphi^*(v) \text{ or } \varphi^*(u) \notin \{1, 2, 3\} \text{ or } \varphi^*(v) \notin \{1, 2, 3\}$$

if V chooses e in step (4), then V rejects

$$\Pr[V \text{ accepts }] \leq \Pr[V \text{ doesn't choose } e] \leq 1 - \frac{1}{|E|} \neq negl$$

The probability of V accepting on a graph $G \notin$ 3COLOR has a noticeable gap to the probability V accepts a $G \in$ 3COLOR. To get the probability that V accepts wrongfully to $negl(\lambda)$, repeat entire protocol $\lambda|E|$ times.

We now prove for the special case an honest verifier (but the proof stands for all malicious verifiers). We construct a Sim for the honest verifier, but because we claim it'll work for all verifiers, the Sim needs to interact with the verifier so it can simulate the probability the verifier requests each edge.

Sim :
(1) pick $e = (u, v) \in_R E$ and random $\varphi'(u) = \varphi'(v)$ (2) $w \in \{u, v\}$
$(c_w, d_w) \leftarrow Com(\varphi'(w), 1^\lambda)$
if $w \notin \{u.v\}$:
$(c_w, d_w) \leftarrow Com(1, 1^\lambda)$
Sim is betting it's money that the verifier will ask for e, otherwise Sim looses.
(3) Sim sends $\{c_w : w \in Vert.\}$ to V and gets $e' = (u', v')$.
(4) if $e = e'$, then output $\{c_w : w \in vert\}, e, d_u, d_v, \varphi'(u,\varphi(v)$, otherwise Sim reruns (2-4).
Sim can rewind the verifier polynomial times.
(5) if $e \neq e'$,go back to (1) and try again
(6) after $\varphi|E|$ iterations, if there is still no success, give up.

Analysis:
If $\exists$ a successful iteration, $(e, \varphi'(u), \varphi'(v))$ is identical to the real world distribution because e is random and the two end points have 2 different colors that are random due to the permutation. Because of hiding of the commitment scheme $\{c_w : w \in Vert\}, d_u, d_v$ is computationally indistinguishable from real world.

If not, Sim has no Com and we reach (6) with negligible probability,

$$\Pr[\text{reaching (6)}] \leq (1 - \frac{1}{|E|})^{\lambda|E|} = negl(\lambda)$$