COS 533: Advanced Cryptography — Princeton University
Lecture 8 (October 9, 2017)
Lecturer: Mark Zhandry — Scribe: Allen Welkie

Notes for Lecture 8

Problem: We've assumed "Honest but Curious" participants in our MPC protocols. The way to remove this assumption is to have each party prove they're following the protocol without revealing anything beyond what the protocol reveals.

1 Zero-Knowledge Proofs

A **Zero-Knowledge Proof** is an interactive protocol between a proover P and a verifier V satisfying the *soundness* and *zero-knowledge* properties. The proover P has access to some data x and a proof π of some property on x, and the verifier has access to x and outputs {success, failure}.

Soundness

The *soundness* property is that if the property on x is false, no cheating proover P^* can convince V. The soundess property can be held perfectly, statistically, or computationally.

Perfect soundness P^* is computationally unbounded, 0 probability of success

Statistical soundness P^* is computationally unbounded, negligible probability of success

Computational soundness[1] P^* is PPT, negligible probability of success

Zero-Knowledge

The *zero-knowledge* property is that for all cheating verifiers V^*, there exists a PPT simulator S^* such that for all true x with valid proof π:

$$\text{View}_{V^*}(P(x,\pi) \leftrightarrow V^*(x)) \approx_C S^*(x).$$

In other words, the simulator can simulate the messages sent between P and V^* using just the information in x. This is similar to the "simulation security" definition we used for multiparty computation.

[1] Some people call these "arguments" instead of "proofs"

2 A Zero-Knowledge Proof Construction

It suffices to construct a zero-knowledge proof protocol for any NP-complete language to get zero-knowledge proofs for all languages in NP because we can then use NP-style reductions from any language in NP. We will construct a zero-knowledge proof for the 3-coloring language.

A Zero-Knowledge Proof for 3-coloring

Let $G = (V, E)$ be a graph. x_G is the property that there exists a valid three coloring π.

$$x_G = \exists \pi : V \to \{1, 2, 3\} \text{ such that } \forall (u, v) \in E, \pi(u) \neq \pi(v)$$

The proover P chooses a random permutation σ on $\{1, 2, 3\}$

$$\sigma \leftarrow S_3$$

Note that $\pi' = \sigma \circ \pi$ — the permutation composed with the valid 3-coloring — is a valid 3-coloring. P then constructs "locked boxes" $C_1, \ldots, C_n$ containing $\sigma \circ \pi(1), \ldots, \sigma \circ \pi(n)$ and sends these to the verifier V.

V chooses a random edge $(u, v) \in E$ and sends this to P.

P sends the keys for the boxes for u and v

V uses the keys to get the colors for u and v and checks that the colors d_u and d_v are different.

Soundness

Suppose G is *not* 3-colorable. This means that there exists an edge (u, v) such that the colors $d_u = d_v$ match. With probability greater than $\frac{1}{|E|}$, the verifier will catch a cheating proover.

Zero-knowledge

All the verifier sees is n "locked boxes," the keys for 2 of the boxes, and permuted colors for 2 vertices. A simulator S chooses a random coloring $\pi^* : V \to \{1, 2, 3\}$ which may not be a valid 3-coloring. S constructs the locked boxes $C_1, \ldots, C_n$ containing $\pi^*(1), \ldots, \pi^*(n)$. Then S runs V^* on $C_1, \ldots, C_n$ to get the verifier's chosen edge (u, v). S checks that $\pi^*(u) \neq \pi^*(v)$, and if they do match, S starts over at the beginning (by choosing another random coloring). If the colors don't match, then S finishes the simulation by giving the keys to V^* and then outputs the view of V^*.

Problem: Soundness probability

There are two problems with this protocol as specified. The first problem is that soundness is only guaranteed with probability $\frac{1}{|E|}$. To solve this problem, we can just run the whole protocol multiple times. Specifically, we can run it $|E|\lambda$ times so that the probability that we don't catch a cheating proover is less than $(1 - \frac{1}{|E|})^{\lambda|E|} \approx \frac{1}{e^{\lambda}}$. This satisfies statistical soundness.

It matters whether the protocol repeats in parallel or sequentially. In either case the soundness property still holds, but if the repetitions are performed in parallel it's unknown whether the zero-knowledge property still holds. This is because the the simulator S would need to restart the protocol if *any* of the random colorings for each repetition were revealed by the verifier, and so the simulator may not return a valid view in polynomial time. Sequential repetition doesn't have this problem because the simulator can restart rounds individually without needing to restart the entire simulation.

Problem: Locked boxes

The second problem is that we haven't given a construction for these "locked boxes." We can't just use typical encryption because then the proover could potentially send different keys that would decrypt to different values of the proover's choosing. We need a scheme where the proover commits to a value in addition to hiding the value.

3 Commitment scheme

A commitment scheme is an interactive protocol between a sender S and a receiver R. The protocol has two phases; the "commit" phase where the sender commits to a value x and the "reveal" phase where the receiver learns x. A commitment scheme has two properties.

Hiding property: Informally, the *hiding* property is that after the commit phase, R "learns nothing" about x. There are perfect, statistical, and computational definitions.

Binding property: Informally, the *binding* property is that if S commits to x in the commit phase, it can't cause R to output $x' \neq x$ in the reveal phase. There are perfect, statistical, and computational definitions.

Note that either hiding or binding *must* be computational. If there is perfect binding, an unbounded adversary can simulate S and R over all possible x. For the purposes of our zero-knowledge proof construction, we'll have statistical binding and computational hiding.

The commitment protocol

Assume a PRG $G : \{0,1\}^\lambda \to \{0,1\}3\lambda$. This construction has the sender commit to a bit b.

Commit phase: The receiver R chooses $r \leftarrow \{0,1\}^{3\lambda}$ and sends r to the sender S.

The sender chooses $s \leftarrow \{0,1\}^\lambda$. If $b = 0$, S sends $y = G(s)$, else S sends $y = G(s) \oplus r$.

Reveal phase: To reveal, S sends b and s.

Binding

Suppose the sender commits to 0, so $y = G(S)$. Then suppose sender can convince receiver to output 1. So $y = G(s') \oplus r$, which means $G(s) \oplus G(s') = r$.

Claim: With probability $\geq 1 - \frac{1}{2^\lambda}$, no such s, s' exist.

Proof:

$$|\{G(s) \oplus G(s')\}| \leq |\{s, s'\}| \leq 2^{2\lambda}$$

$$\text{Bad} = \{G(s) \oplus G(s')\}$$

$$\Pr[r \in \text{Bad}] = \frac{|\text{Bad}|}{2^{3\lambda}} \leq \frac{1}{2^\lambda}$$

4 Malicious MPC

So how do we use zero-knowledge proofs to get around the honest-but-curious assumption? Have all users commit to their inputs x and broadcast that commitment C. Then engage in the honest-but-curious protocol and whenever a user sends a message m, they also send a zero-knowledge proof that there exists x, r such that C is a commitment to x and m can be gotten from the appropriate algorithm $A(x, r)$.