CSE 5854: Class 04

Benjamin Fuller

January 25, 2018

1 Zero-knowledge Definitional Issues

In the previous reading and class today we introduced the concept of a zero knowledge proof. Roughly it is a proof system where the verifier learns nothing other than the validity of the statement. That is, a proof system P, V is zero-knowledge if for all V^* there exists a simulator S where S can produce views indistinguishable from the view of V^*. We'll spend this reading digging into some important definitional issues.

First we recall that the simulator was allowed to fail with probability $\leq 1/2$. It is always possible to amplify a simulator's success probability so that it rarely fails. Suppose we have a simulator S that fails with probability $\leq 1/2$. We can build a simulator S' that fails with negligible probability. The simulator works as follows:

```
1. Input $S, x$.
2. For $i = 1$ to $p(n)$.
   (a) Execute $S$ on fresh random coins.
   (b) Compute $y \leftarrow S(x)$. If $y \neq \perp$ break.
3. Return $y$.
```

Clearly the simulator S' runs in time $T(S'(n)) \leq p(n) \cdot T(S(n))$ and is polynomial if S runs in polynomial time. Furthermore, the simulator outputs $\perp$ with probability at most $1 - 2^{-p(n)}$.

Ideally one could hope to have the simulator never output $\perp$. Unfortunately, given a simulator that succeeds with some constant probability it isn't possible to do this is pure polynomial time. The composed simulator could get very unlucky and always have V^* output the "wrong" graph. This means that the composed simulator would only run in expected polynomial time, that is $\mathbb{E}_r \, T(S'(n; r)) = p(n)$ for some polynomial p, however there are strings r for which the time exceeds $p(n)$ for any fixed polynomial.

For most of the class we won't care about these issues. One of main concerns is that expected polynomial time does not compose as well as polynomial time. When run a polynomial number of times an expected polynomial time machine does not necessarily yield an expected polynomial time machine.

Second, the zero-knowledge can be a bit unsettling if a simulator can convince any verifier without knowing anything about the witness why should we feel comfortable that the prover has actually convinced us of anything. The first thing that should be slightly reassuring is that S is only expected to produce good transcripts when $x \in L$. We are not saying anything about S's behavior when $x \notin L$. Furthermore, the fact that a simulator and a $< P, V^* >$ interaction look the same demonstrates the power of interactivity in the process. Suppose we have an interactive proof system for some language L, an outside party should not be convinced by a transcript of the interaction. It is crucial to participate in the protocol to be sure that $x \in L$. Furthermore, the way that P and S operate are very different. P is interacting with some party interactively and they have no control over that party, it could be passing messages to other parties, running similar protocols at the same time, etc. We can think of S as knowing the complete code of V^* and being able to set its random tape, read its state, and control messages. Its not surprising that this additional power is helpful (and allows S to produce a good transcript without knowing why $x \in L$).

The first introduction we gave (in Class 03) is called *perfect zero-knowledge* as the simulator produces the exact same distribution of transcript (conditioned on not outputting $\perp$). There are two weaker definitions that we'll use in class. These are statistical zero-knowledge and computational zero-knowledge.

Definition 1 (Statistical Zero Knowledge)**.** *Let (P, V) be an interactive proof system for some language L. We say that (P, V) is* statistical zero knowledge *if there exists a negligible function* $\mathsf{ngl}(n)$ *such that for every PPT A there exists a PPT S such that for every $x \in L$,*

$$\sum_y |\Pr[S(x) = y] - \Pr[< P, V^* > (x) = y]| \leq \mathsf{ngl}(n).$$

In this definition we can safely remove discussion about S outputting $\perp$ as we can include the cases when S outputs $\perp$ in the $\mathsf{ngl}(n)$ probability that the distributions don't line up. However, there is a asymmetry in this definition. Assuming that we are "running" V^* as part of S we are only being expected to fool a polynomial time machine while producing our transcript then we are being subject to an information-theoretic requirement on the closeness of our output distribution. This is remedied the following definition that is known as computational zero-knowledge:

Definition 2 (Computational Zero Knowledge)**.** *Let (P, V) be an interactive proof system for some language L. We say that (P, V) is* statistical zero knowledge *if there exists a negligible function* $\mathsf{ngl}(n)$ *such that for every PPT A, there exists a PPT S such that for every PPT D and every $x \in L$,*

$$|\Pr[D(S(x)) = 1] - \Pr[D(< P, V^* > (x)) = 1]| \leq \mathsf{ngl}(n).$$

Not surprisingly these definitions are ordered. Perfect zero knowledge implies statistical zero knowledge which implies computational zero knowledge. We believe that statistical zero knowledge is stronger than computational zero knowledge. There is little evidence on whether perfect is stronger than statistical.

1.1 View vs. output

All of our definitions of zero knowledge ask for indistinguishability of the simulator output and the verifiers output. Note that the behavior of any V^* is a deterministic function of the messages that it receives from P and any random choices it makes. Thus, we can think of V^*'s behavior as a function of this. This is notion is used frequently enough that we give it a name, V^*'s *view* of the computation. Its not hard to see that we can compute the output of any verifier given its view. Furthermore, there is some verifier that outputs all of the messages it sees as well as its random state. In some sense, this is hardest verifier to simulate. Thus, we can replace the computational zero-knowledge condition with the following:

Definition 3. *Define $(P, V), L, V^*$ as above. Denote by $view^P_{V^*}(x)$ a random variable of the random tape of V^* and the messages that V^* receives from P on input x. (P, V) is zero-knowledge if for all PPT V^* there exists a PPT S such that*

$$|\Pr[D(S(x)) = 1] - \Pr[D(view^P_{V^*}(x)) = 1]| \leq \mathsf{ngl}(n).$$

In the computational setting these two definitions are equivalent and we can think of the goal of a simulator as producing messages that look like P's messages. (Note in our example this is exactly what our simulator did.)

1.2 Auxiliary Input

Recall in the definition of semantic security for encryption there was a function $h(m)$ that was used to represent the adversary's additional information about what message was being sent. We wanted to make sure if the adversary knew the first bit of m we weren't leaking information to that adversary. This was used to codify the notion that we weren't leaking information no matter what information the adversary starts with. For many applications where zero-knowledge is used in other cryptographic protocols we need a similar modification of knowledge. This extra information is known as auxiliary information.

Definition 4. *Let (P, V) be an interactive proof for L. Each machine P and V has an additional input called an auxiliary input that is read only. This input is not considered when determining if a machine is polynomial time (though the amount that a machine can read may be limited by its complexity). We denote by $< P(y), V(z) > (x)$ the output of V when P and V have common input x and auxiliary inputs y, z respectively.*

The machines are an interactive proof with respect to auxiliary input if:

- ***Completeness:*** *For every $x \in L$, there exists a y such that for every $z \in \{0, 1\}^*$,*
$$\Pr[< P(y), V(z) > (x) = 1] \geq \frac{2}{3}$$
- ***Soundness:*** *For every $x \notin L$ for every P^*, and for every $y, z \in \{0, 1\}^*$,*
$$\Pr[< B(y), V(z) > (x) = 1] \leq \frac{1}{3}$$

Denote by $P_L(x)$ the set of strings y where the proof is complete. The machines are zero-knowledge if for every PPT V^* there exists a PPT S such that for all PPT D

$$|\Pr[D(< P(y_x), V^*(z) > (x)) = 1] - \Pr[D(S(x, z)) = 1]| \leq \mathsf{ngl}(n).$$

in this definition y_x is an arbitrary value in $P_L(x)$.