6.876/18.426: Advanced Cryptography February 19, 2003

Lecture 4: ZK Proofs and Proofs of Knowledge

Scribed by: Susan Hohenberger

1 Recap

Recall our working definition of a zero knowledge proof system. We say that (P, V) is a $ZKPS$ for language L if the following conditions hold:

1. **Completeness:** $\forall x \in L \; Pr[(\mathsf{P}, \mathsf{V})[x] = \text{“Yes”}] \geq 1 - negl(k)$.
2. **Soundness:** $\forall x \in L \; \forall \mathsf{P}' \; Pr[(\mathsf{P}', \mathsf{V})[x] = \text{“Yes”}] \leq \frac{1}{2}$.
3. **Zero Knowledgeness:** (most general definition)
   $\forall \mathsf{V}'_{\mathsf{PPT}} \; \exists \mathsf{S}_{\mathsf{PPT}} \; \forall x \in L \; \forall a \in \{0,1\}^* \;\; VIEW_{\mathsf{V}'}^{\mathsf{PV}'}(x; x, a) = \mathsf{S}(x, a)$.

In this definition, we attempted to capture our intuition about what a zero knowledge proof system should be. We wanted that for *all* knowledge that V' gets from talking to P, there is some “easy” strategy to obtain this knowledge, and therefore V' learns nothing important from talking to P (except the truth of the statement that they are discussing). We learned in Lecture 3, that V' must still not learn anything important, even if she talks to P using knowledge that she learned in an earlier conversation. We were able to satisfy these intuitions, at least for the graph isomorphism protocol (ISO), by adding an advice string. So, are we done? Have we correctly defined a zero knowledge proof system? Let's test it on another example.

2 Another Example: Graph Non-Isomorphism

The language $NISO$ consists of all pairs of graphs (G, H) such that $G \ncong H$. Recall our previous (IPS) proof system for proving membership in $NISO$.

1. V randomly selects either G or H (wlog, assume G is chosen). Next, V randomly selects a permutation π, and sets $\pi(G) \rightarrow C$. V sends C to P.
2. Since $G \ncong H$ and P has infinite time, P figures out which graph C is isomorphic to, and returns one bit to V indicating G or H.
3. If P was incorrect, V rejects, otherwise, V accepts.

Now, we ask the question: is this a zero knowledge proof or not? It is not immediately clear. We reason that our definition captures our intuition; we have blind belief; we hope that any bugs that arise from this protocol might become features in our definition. But, alas, there is cause to worry. In the above protocol, P is deterministic, which seems very dangerous. Can a malicious V' extract something from P? Unfortunately, the answer is yes.

2.1 Break in the Naive Protocol for NISO

To understand why the above protocol is not zero knowledge, let us first consider an example:

> Suppose a company is selling three new aspirin pills. Your friend claims that all three new pills are really made out of the same protein. You are a master scientist, so you are able to test these pills. You are willing to compare two of them and tell your friend, but you want to be paid extra for telling him anything about the third pill. You worry, however, that in telling him about the first two, you might reveal something to him about the last pill for free!

The same problem might arise in our previous protocol if V' behaves maliciously. Suppose she was given three graphs G, H, C and she does not know anything about them. Now, P might agree to prove that $G \ncong H$, but no more. Here V' can take advantage of P by sending C instead of a random permutation of G or H according to the protocol. If $G \cong C$ or $H \cong C$, then P would not be able to tell that V' had cheated him, and V' would learn something nontrivial about C. This is certainly not zero knowledge (assuming that testing graph isomorphism is hard)!

An observation was made at this point that in this $NISO$ protocol V' is allowed to send many bits of information to P (i.e. C), but in the ISO protocol V' only sends one bit. Another way to look at this is to say that in the ISO protocol, S only has to handle two possible messages from V', while in this protocol, S must handle exponentially many messages. Indeed, there is a real problem with this protocol, which we will fix shortly, but we must be careful – even one bit of information (i.e. "She loves me, she loves me not.") can be a lot!

2.2 A New Protocol for NISO

We give a new protocol for proving membership in $NISO$. Our two biggest concerns from the old protocol were (1) P was deterministic, and (2) a malicious V' could ask about a graph C that was isomorphic to neither G nor H. We handle the second concern by putting into the protocol pressure on V' to make the statement:

> I know which of G or H the graph C is isomorphic to, but I'm checking that you can tell, P.

Our new protocol that captures this idea is:

1. V' flips a coin to choose either G or H (wlog, assume G is chosen). Then V' randomly chooses a permutation π, and computes $\pi(G) \to X$. Now, V' flips another coin to choose either G or H (say, H), randomly chooses a permutation π_1, and computes $\pi_1(H) \to X_1$. This process is repeated until V' has computed $X, X_1, X_2, \ldots X_{2k}$, and sends all these graphs to P.
2. P then flips $2k$ coins, one for each X_i, and sends them to V'.

3. Now, V' must send one last message back to P. For each X_i with a corresponding 0 from P, V' reveals π_i. But for each corresponding 1, V' either reveals ϕ_i, such that $\phi_i(X) \to X_i$, or outputs $\perp$. Note that $\phi_i = \pi_i \cdot \pi^{-1}$ when X and X_i share the same root graph. (We will later see that it is important that V' not output *too many* $\perp$s. In fact, if V' does return ϕ_i for at least $\frac{k}{3}$ of the X_is with 1s, then we allow P to abort the conversation.[1])

4. P sends back one bit, indicating whether $X \cong G$ or $X \cong H$. If P is incorrect, V' rejects, otherwise, V accepts.

Is this protocol correct? Let's take it one step at a time. First, we are glad to see that P now acts randomly. But can a malicious V' still fool us with a graph C isomorphic to neither G nor H? We will address this issue shortly, but intuitively we are satisfied that this protocol puts pressure on V' to make the statement:

> I know if $G \cong H$ or not, but if $G \cong H$, then you, P, would not be able to tell what my mind thinks the answer is (i.e. you can not tell for certain my first coin flip).

P can use the X_is, π_is, and ϕ_is, as a way to check that V' is not trying to trick him with an X isomorphic to neither H nor G. The π_is check that the X_is are not "junk", while the ϕ_is give strong evidence that X is not "junk". But, you say, wouldn't a sneaky V' get away with cheating (i.e. asking about an X not isomorphic to either G or H), if P selects random coins such that it puts 1s next to all the $\frac{k}{3}$ X_is placed strategically for cheating? Yes; but the chance of this event occurring is exponentially small.

Thus, we have a good feeling that if V' goes through the protocol with P, then we can conclude that V' knew the a mapping from X to G or H, and therefore is not trying to gain information about an arbitrary third graph C. Now, let's prove the correctness of this new protocol under our working definition.

2.3 Proof of New Protocol

We will prove the completeness, soundness, and zero knowledgeness of our new *NISO* protocol.

1. **Completeness** $\forall x \in L \; Pr[(\mathsf{P}, \mathsf{V})[x] = \text{"Yes"}] \geq 1 - negl(k)$.

When $G \ncong H$, P can always decide if $X \cong G$ or $X \cong H$ (by running his exponential time algorithm) and send back a correct answer, as long as he is confident that V is not trying to cheat him. When V is honest, the chance that she looks like she is trying to cheat is exactly the case where she is unable to answer at least $\frac{k}{3}$ of her 1-queries, which we already saw to be negligible. Thus, when both P and V play according to the rules, P convinces V of a

[1] Why is P allowed to abort? Because it is very unlikely that V' is behaving honestly. Consider that when V' follows the protocol for choosing X_is honestly and P sends a random string, then we expect half the graphs with 1s to be isomorphic to X, for an expected total of $\frac{k}{2}$. Using Chernoff bounds, we can show that $Pr[\text{number of graphs isomorphic to } X < \frac{k}{3}] < O(e^{-k})$ (i.e. exponentially small). Therefore, if V' does not produce approximately $\frac{k}{3}$ correct ϕ_is, then with high probability she is withholding information or trying to cheat and P should walk away.

true statement with probability $\geq 1 - Pr[\mathsf{V} \text{ looks like she is cheating}] = 1 - negl(k)$. Note that we are really using the property that completeness hold *with high probability*, and not absolutely, in our $NISO$ example today.

2. **Soundness** $\forall x \in L \; \forall \mathsf{P}' \; Pr[(\mathsf{P}', \mathsf{V})[x] = \text{"Yes"}] \leq \frac{1}{2}$.

Assume that $G \cong H$ and V is honest, now we wish to show that P''s chances of successfully fooling V are less than $\frac{1}{2}$. To fool V, P' must successfully guess her first coin flip, that created X from G or H. He can obviously guess with probability $\frac{1}{2}$, but does he get any information from an unwitting V to do better?

In the first pass, V sends $X, X_1, X_2, \ldots, X_{2k}$. All of these graphs are isomorphic to one another, chosen independently at random, and therefore, they reveal nothing about the "real" root graph of X, since they have the same probability of being a descendant of either. Now, P' has a chance to send back an arbitrary string, and V will reveal π_i for all 0-queries, and either ϕ_i or $\perp$ for all 1-queries. A π_i shows a relation between X_i and one of the input graphs; it says nothing about X and therefore, it can reveal nothing about the first coin flip. A ϕ_i shows a mapping to X_i from X, but since both of those graphs are isomorphic to G and H, this also reveals nothing. Lastly, $\perp_i$ tells P that V did not know a mapping from X to X_i, and thus the "root" graph of X_i is not the root graph of X in V's mind. But P' can see that $X_i \cong G \cong H$, so what was the root graph of X_i? There is no way for P to tell! Again, P can do no better than guessing.

3. **Zero Knowledgeness** (most general definition)

$\forall \mathsf{V}'_{\mathsf{PPT}} \; \exists \mathsf{S}_{\mathsf{PPT}} \; \forall x \in L \; \forall a \in \{0,1\}^* \;\; VIEW_{\mathsf{V}'}^{\mathsf{PV}'}(x; x, a) = \mathsf{S}(x, a)$.

We will now make a habit out of the seemingly contrary practice of making this proof easier on ourselves by actually proving something *stronger*. To be exact, we will instead show:

$\exists \mathsf{S}_{\mathsf{PPT}} \; \forall \mathsf{V}'_{\mathsf{PPT}} \; \forall x \in L \; \forall a \in \{0,1\}^* \;\; VIEW_{\mathsf{V}'}^{\mathsf{PV}'}(x; x, a) = \mathsf{S}(x, a)$

We now construct such an S, that when given $[(G, H), a]$ as input, can create a view, talking to any verifier, that is identical to the view between that verifier and the prover.

1. S interacts with V' in a black box manner. First, S flips a fair coin repeatedly and places the resulting string onto V''s random tape.

2. Now, V' awakens, creates $X, X_1, X_2, \ldots, X_{2k}$ in any manner she likes, and sends them to S.

3. S flips $2k$ coins and sends the resulting bit string to V'.

4. Now V' will send a final message. It is the job of S to test that for each 0-query, V' correctly returned a π_i such that $\pi_i(G) \to X_i$ or $\pi_i(H) \to X_i$. If this is not true, S aborts. Next, S must count the number of 1-queries for which ϕ_is were returned. If there are less than $\frac{k}{3}$ of them, S also aborts. Lastly, S checks that for each 1-query with a ϕ_i, it holds that $\phi_i(X) \to X_i$; if not, abort.

5. At this point, S should output a bit, indicating if $X \cong G$ or $X \cong H$. *But*, assuming $P \neq NP$, he does not know the answer, and guessing is not good enough. So, S puts his conversation with V' on hold. He now starts a new conversation with V'' on the same graphs (G, H), giving V'' the same random tape as V'.

6. Since all the input is the same for both V' and V'', they choose the same root graph for X, and send the same $X, X_1, X_2, \ldots, X_{2k}$ to S.

7. S uses its cleverness here, creates and sends a new random string to V''. Then it eagerly awaits the reply.

8. Once V'' replies, S runs all the checks in step 4. If they all pass, S scans the two response strings that it has from the two verifiers. If it finds a position j in which one verifier gave π_j (for a 0-query) and the other verifier gave ϕ_j (for a 1-query), then it celebrates. S can now solve for the root graph of X, by computing $\pi_j^{-1}(\phi_j^{-1}(X))$, which will equal either G or H. It returns to V' and sends back the correct answer. However, if no such position j occurs between these two message strings, then S takes two steps. First, S does a constant amount of work on the exponential algorithm that decides membership in $NISO$ on (G, H). Secondly, he aborts his conversation with V'' and starts up a new clone of the verifier on the same input and random string.

9. S will always respond correctly to V'; either by finding a match in two response strings, or by solving the exponential time $NISO$ algorithm, a few steps at a time.

Okay! To convince ourselves that this achieves zero knowledgeness, we should focus on two main concerns (all other parts of the definition are fairly easy to confirm): (1) will S finish in probabilistic polynomial time, and (2) is the conversation between S and V' indistinguishable from what we would expect from P and V'?

Concern 1: Does the simulator run in probabilistic polynomial time?

First, let's define what we mean by PPT. Generally speaking, there are two types:

1. Strict polynomial time, with coin flips.

2. Expected polynomial time, with coin flips.

Unfortunately for us as cryptographers, we must allow S to be the latter for our $NISO$ example to succeed.[2] If S is so unlucky as to never find a match , then his worst-case running time will be exponential. But what about his expected running time? Each conversation between S and a verifier is polynomial in time; the question is how many conversations will be necessary?.

Let p be the probability that the verifier knows the answer to an entire bit string from S. First, it is obvious that we need two good answers from V' (either by opening a new session or rewinding). In this case, we have a formula for the E[runtime of S]:

$$Pr[\mathsf{V} \text{ aborts}]1 + Pr[\mathsf{V} \text{ does not abort}]E[\text{time to second right answer}] = (1-p) + p\frac{1}{p} \leq 2$$

But we actually need something more than just that V' responds correctly twice. We need to be sure that the responses are to two different random strings. Let n be the number

[2] See the paper by Barak and Lindell[BL02] on the uncomposability of expected polynomial time, the heartache it causes in cryptography, and our current inability to do without it.

of strings of length k for which V' knows a valid response. Then the E[runtime of S]:

$$Pr[\mathsf{V} \text{ knows response}]E[\text{time to ask diff string}] = \left(\frac{n}{2^k}\right)\left(\frac{2^k}{n-1}\right) = \frac{n}{n-1}$$

If V' only knows the valid response for 1 string, then the expected runtime of S is ∞! So this is obviously a problem. Also, what if V' responds correctly to two different strings, but in the only position in which they differ, she returns $\perp$ for the 1-query?

We need for her to give us π_j and ϕ_j information about the same X_j; we want to prove that we will get this information after a polynomial number of expected queries. Suppose that the probability that when the verifier V' is given a random string challenge it makes a valid response is p, so it should only take us an expected $1/p$ trials for each valid response to a random string we need to get. All we need is that the verifier V' correctly answers a new random challenge which is a 0 in a place where V' gave a ϕ_i before. Since, on expectation, $1/6$ of the bits are ones where this is possible, the probability that a second, correctly answered, random challenge doesn't give the information we need is $1/2^{k/3}$, we can see that it is very likely we'll recover the answer of V' in polynomial time.

One interesting question that arose in class was: why not just save time by running the protocol with V'', guessing the correct answer, seeing whether or not V'' accepted, and then giving an answer to V' accordingly. Do you know what a poker face is? Maybe V'' has a good poker face. What does it mean for V'' to accept? V'' may not give an indication if she liked the response from S or not, and S must simulate the view regardless of V'''s behavior. Afterall, V'' could be a robot, and whether or not she would even have emotions to show, is well outside the scope of this class.

Concern 2: $VIEW_{\mathsf{V}'}^{\mathsf{P}\mathsf{V}'}(x;x,a) = \mathsf{S}(x,a)$??

We look at the views of S and the true P; are they the same? Yes, they are. Observe that S and P both generate a random string, then they both abort if V' responds in a suspicious manner. Otherwise, they will both always return the correct answer. P can do this, because he has infinite time; and S can do this, because he can either check that $\pi_j^{-1}(\phi_j^{-1}(X)) \to G$ or $\pi_j^{-1}(\phi_j^{-1}(X)) \to H$, or run the brute force algorithm, a few steps at a time, whenever he gets into trouble. Thus, S has the same odds as P in producing the $\{(conversations, coins)\}$ that this protocol creates.

We already argued that S "gets into trouble" an exponentially small amount of the time. With that idea in mind, we ask the question: why not just let S abort when he gets into trouble instead of running exponentially long? One objection might be that S would no longer be always acting like P, since P never aborts unless V' behaves suspiciously. But, we know this difference in behavior would only occur an exponentially small amount of the time, so it is unlikely that S will ever get caught acting differently than P. In fact, this is good enough. Previously, we have been working with an idea called **Perfect Zero Knowledge**, in which the views of S and P must be identical. An example of Perfect Zero Knowledge was our ISO proof. Let X and Y be two distributions over a finite set D.

Definition: X and Y are *perfectly indistinguishable* if $\forall a \in D Pr[X = a] = Pr[Y = a]$

Theorem: Let Z_0 and Z_1 be perfectly indistinguishable. Then $\forall A Pr[b \leftarrow \{0,1\}; a \leftarrow Z_b; b' \leftarrow A(a) : b' = b] = \frac{1}{2}$.

Now, we introduce a concept called **Statistical Zero Knowledge**, in which the views of S and P need only be *statistically indistinguishable*. The intuition here is that even an unbounded adversary, given only polynomially many samples from either distribution $VIEW_{\mathsf{V}'}^{\mathsf{PV}'}(x;x,a)$ or $\mathsf{S}(x,a)$, can not tell which it is with an advantage better than negligible.

Definition: X and Y are *statistically indistinguishable* probability distributions over set D if $\sum_{a\in D} |Pr[X=a] - Pr[Y=a]| \leq negl(k)$.

Theorem: Let X and Y be statistically indistinguishable. Then, for any polynomial p, given $p(k)$ independently chosen samples, no adversary can tell X from Y with advantage better than negligible.

Thus, for all practical purposes, two statistically indistinguishable distributions, notated $\cong$, are the same, since any polynomially bounded verifier can not tell them apart. We add this notation into our definition for a $ZKPS$. Notice that three different forms of fuzziness in our working definition were all used by this $NISO$ example: (1) completeness with high probability ($\geq 1 - negl(k)$), (2) expected polynomial time (PPT), and (3) statistical indistinguishability. The good news is that we are making our notion more precise!

Zero Knowledgeness: (most general definition)
$\forall \mathsf{V}'_{\mathsf{PPT}}\ \exists \mathsf{S}_{\mathsf{PPT}}\ \forall x \in L\ \forall a \in \{0,1\}^*\ \ VIEW_{\mathsf{V}'}^{\mathsf{PV}'}(x;x,a) \cong \mathsf{S}(x,a)$.

3 Zero Knowledge Proofs of Knowledge

What else can we learn from the $NISO$ example? Recall the situation. The verifier got the prover to convince her that $G \ncong H$, by in turn convincing the prover that the graph X that she was using in her test was indeed isomorphic to G or H. But the beauty here, is that the verifier was able to convince the prover of this fact *without* revealing the root graph of X. One might say that the verifier gave a "zero knowledge proof of knowledge" about X.

Let's consider a second example: discrete logs. Let G be a group of prime order p. Let $g \in G$ be a generator for this cyclic group. We choose a random y such that $1 \leq y \leq p-1$. Now, we know that there *exists* an x such that $g^x \ mod\ p = y$, but finding that x is a different story, generally considered to be very hard. If you know an x such that $g^x \ mod\ p = y$, how could you convince me of that without telling me which x you have in mind?

Our take-home assignment from this lecture is to try to formalize in a definition the intuition behind a Zero Knowledge Proof of Knowledge (ZKPOK). The idea is "I know a secret π, but I want to prove to you that I know it without revealing which π I know." Good luck!

References

[BL02] Boaz Barak and Yehuda Lindell Strict Polynomial-time in Simulation and Extraction. Proceedings, 34th Annual ACM Symposium on Theory of Computation (STOC), pp. 484-493, 2002.