6.876/18.426: Advanced Cryptography April 9, 2003

Lecture 16: Defining ZK Proofs of Knowledge

Scribed by: Vitaly Feldman

1 Overview

Today's lecture will cover the following topics:

- Rabin's oblivious transfer protocol.
- Fixing Rabin's protocol.
- Formal definition of ZK proof of knowledge.
- Examples of ZK-PoK.

2 Oblivious Transfer

The topic of today's lecture is formal definition of the concept of ZK proof of knowledge (ZK-PoK). We start the discussion by describing a seemingly unrelated notion and a protocol that implements it. Let $n = p_1 p_2$. We assume that p_1 and p_2 are known to A and B knows only n. We are interested in a two-party protocol with the following outcome:

- with probability $1/2$ B "learns" p_1 and p_2; and with probability $1/2$ it "learns nothing"
- A does not "learn" if B has got the factors or not

This notion and the first implementation were given by Rabin [R81]. In his protocol

1. B chooses a random $x \in Z_n^*$ and send $x^2 \bmod n$ to A
2. A finds all the four roots of x^2, chooses one of them randomly (denote it by z), and sends z to B.

The validity of this protocol easily follows from the following observation. x^2 has four roots. Denote them by x, $-x$, y, $-y$. If A sends B either x or $-x$, B has not earned any new information. On the other hand, if z equals to y or $-y$, then by computing the GCD of either $x + z$ or $x - z$ with n B will find the factorization. Clearly this happens with probability $1/2$. If B has chosen his x randomly, A has no information on which of the four roots B has and therefore B will factor n with probability $1/2$ and A cannot know whether B has factored n or not.

A more thorough inspection of the protocol raises the following concerns.

1. What if n has more than 2 prime factors? Then another root will not necessarily give the factorization of n but rather a non-trivial factor of n. The probability that this happens will be higher than $1/2$ (since there are more than four roots). This concern can be handled by modifying the notion accordingly and is relatively minor.

2. We can notice that, in fact, security of this protocol against a malicious B is not equivalent to hardness of factoring. For example, assume the existence of two PPT algorithms A_1 and A_2 such that $A_1(n) = s$, where s is some special "square" (i.e., $(n, s) \in QR$) and $A_2(n, s, t) = (p_1, p_2)$ for any t being a square root of s. This assumption does not contradict the hardness of factoring since it is hard to find any square root of s. But on the other hand, by using A_1 and A_2 while interacting with A, B can factor n.

3 Fixing Rabin's protocol

In order to handle the second concern in the Rabin's protocol Goldwasser, Micali and Racoff [GMR89] gave the following modification of the Rabin's idea to protect against a malicious B. Let k be a security parameter and $n = p_1 p_2$ as before (all the arithmetic is modulo n)

1. B chooses randomly $x, x_1, x_2, \ldots, x_k$ and sends $x^2, x_1^2, x_2^2, \ldots, x_k^2$;

2. A chooses a random binary string $b_1, b_2, \ldots, b_k$ and sends it.

3. For every i, if $b_i = 0$ then B sends x_i; otherwise sends xx_i.

4. A verifies that B's replies are consistent and (if they are) sends a random root of x^2.

This modification forces B to prove that he knows at least one root of x and therefore is playing the game fairly. If B can provide both x_i and xx_i then he knows x. Otherwise, for every i, B will be able to answer at most one question and therefore will be caught with probability at least $1 - 2^{-k}$. In order to prove that A cannot modify the probability with which B factors and also still cannot know whether B factored or not we have to prove that A cannot gain any information about B's root from the proof that B knows it. In other words, we have to prove that B's proof of knowledge is ZK.

For this particular protocol we can easily observe that for every i if $b_i = 0$, then A gets no information related to x at all. If $b_i = 1$, then the pair x_i^2, xx_i does not give any additional information about the root since for every other root there exists (an equally likely) x_i' which is consistent with the given pair.

Thus we can conclude that we achieved improved oblivious transfer.

4 Formal definition of ZK proof of knowledge

In the previous section we first referred to the notion of ZK-PoK which we never formally defined. Informally, the prover wants to prove that he knows some y related to the common input x without revealing y. It is quite clear that we require that if prover indeed knows y he could easily prove this to the verifier. We are also familiar with what ZK requirements on the protocol mean. In this particular case, we want the view of the protocol to be

simulatable without knowing y. The usual soundness definition has to be modified though. We would like to ensure that if the prover manages to convince the verifier that he knows y, he should actually "know" it. An elegant way to formalize this "knowledge" is via the notion of *extractor* algorithm. That is, P "knows" y if an efficient ITM E can output y via an interaction with P. This property of a PoK protocol is called *validity*. As usual in the definition of ZK-ness, validity seems to directly contradict the ZK-ness of a protocol since a malicious verifier would also be able to extract y. But unlike verifier, the extractor can *reset* the prover, namely it can set the prover's random tape and therefore has a complete control over the prover's state. Thus one may think of a knowledge extractor as the "sub-conscious" of the prover in the sense that the sub-conscious has a lot of flexibility in its ability to access the prover.

We will now try to formalize this notion. The basic definition is as follows (given in this form by Bellare and Goldreich [BG92]). Let R be a polynomial-time computable and polynomially balanced relation. Let P and V be PPT ITMs. We denote by $(P,V)(x,y)$ the output of the verifier in the execution of protocol $(P(x,y),V(x))$ with 1 meaning `accept`, and 0 - `reject`. We say that the protocol (P,V) is a ZK proof of knowledge for R with error ϵ if the following properties hold:

1. *Completeness.* For all $(x,y) \in R$, $\mathsf{Pr}[(P,V)(x,y) = 1] = 1$.

2. *Validity.* For every PPT ITM P', if $\mathsf{Pr}[(P',V)(x,y) = 1] = p > \epsilon$, then there exists a PPT resetting ITM E, such that w.p. $p - \epsilon$, $E^{P'}(x) = y$ and $(x,y) \in R$.

The value $p - \epsilon$ is the advantage of the the prover over the inherent error of the protocol ϵ. We can relax the probability of extraction to be $(p-\epsilon)^c/\mathrm{poly}(|x|)$ for some constant c. We can now note that the GMR protocol from the previous chapter has proof of knowledge with error 2^{-k} for relation $((n,x),y)$ where y is a square root of x modulo n. Later we will prove this.

While capturing the idea of ZK-PoK, this definition still allows some situations in which the process of knowledge extraction from the prover does not follow our intuition. To be more specific, in the current definition the knowledge extractor can interact with prover in a way that is completely different from the way in which the verifiers are capable of interacting with P'. This can allow the extractor to extract the knowledge in an "unfair" or "bizarre" way. To avoid this problem the following restrictions were suggested by Halevi and Micali [HM03]:

1. *Tighter Semantics.* Messages in every interaction between P' and E should be distributed in the same way as in the interaction between V and P'.

2. *Prover-Time Preservation.* Interaction between P' and E should take "about" the same time as the interaction between V and P'.

Note that since V and E are polynomial time, the second restriction limits the prover's running time in the interaction with the extractor. Another rather technical limitation is that in Halevi-Micali's definition the extractor has to be strictly polynomial time (and not expected as in [BG92]).

5 Examples

We would now like to show that some of the protocols that we have already seen are restricted ZK-PoK protocols. We start by showing that the GMR protocol conforms to the above requirements. For simplicity we examine the case $k = 1$, that is, $\epsilon = 1/2$. The square x and n are known to both P' and V. Denote the random square sent by P' to V by z. V then randomly picks either 0 or 1. P' replies with either $\sqrt{z}$ or $\sqrt{xz}$ accordingly. Let

$$p_{n,x} = \mathsf{Pr}[(P', V)(x, n) = 1]$$

with probability taken over the random choices of P' and V's single coin flip. E will choose a random setting of P' coins and execute P' twice for both for a random challenge $b \in \{0, 1\}$ and then its negation. We can immediately notice that the views in both of these executions are distributed in the same way as $\mathrm{VIEW}^{P',V}(x, n)$. By the definition of $p_{n,x}$ with probability at least $p_{n,x} - 1/2$ both $\sqrt{z}$ and $\sqrt{xz}$ are accepted (that is, are consistent with x and z) and therefore we can efficiently extract $\sqrt{x}$.

It is easy to see that in this protocol we could also use an extractor which gives independent challenges to P' and achieves extracting probability of $p_{n,x}(p_{n,x} - 1/2)$.

Another ZK-PoK protocol that we have seen is the GMW protocol for 3-COL [GMW91]. While we used it as a ZKP for 3-COL, it is easy to see that, with slight modifications it can be viewed as a ZK-PoK for 3-coloring of a graph, that is for the relation

$$R_{3col} = \{(G, \pi) \ : \ \pi \text{ is a legal 3-coloring of graph } G\} \ .$$

The protocol for graph G and coloring π consists of the following steps:

1. P commits to π and send the commitment s;
2. V picks a random edge $e = (i, j)$ in G and sends it;
3. P sends $\pi(i)$ and $\pi(j)$;
4. V verifies that $\pi(i)$ and $\pi(j)$ match their commitments and are different colors.

We claim that this protocol is ZK-PoK for R_{3col} with error $\frac{m-1}{m}$ where m is the number of edges in G.

Proof: Completeness is obvious and ZK-ness we have already proved. To prove that the protocol is valid we define the extractor algorithm. Sample m conversations as follows: denote the edges in G by $e_1, e_2, \ldots, e_m$. E picks at random $k_0 \in \{1, \ldots, m\}$ and in the r'th experiment it sends in step 2 of the protocol the edge e_{k_r} where $k_r = k_0 + r \bmod m$. It is easy to see that this ITM generates a conversation randomly distributed in the same way as conversations between P' and V since for each experiment e_{k_r} is uniformly distributed among all the edges in G. Notice that since we are using perfectly binding commitments, then any node in G must have the same color in all the accepting conversations in which it is one of the endpoints of the edge sent in Step 2. Hence, if all m conversations are accepting, then it is easy to compute from them a valid 3-coloring of G. As in the previous proof let

$$\lambda_G = \mathsf{Pr}[(P', V)(G) = 1] - \frac{m-1}{m} \ .$$

Notice that in Step 2 V sends one of m equiprobable values. Thus, for any given commitment string s (which may sent by the prover in Step 1), the acceptance probability conditioned on this acceptance string, is either 1 or at most $\frac{m-1}{m}$. Therefore, if $\lambda_G > 0$, then with probability of at least $m \cdot \lambda_G$, the commitment string sent in Step 1 has a conditional acceptance probability of 1. For such a string, we are sure to extract easily a valid 3-coloring. Thus, a valid 3-coloring is extracted with probability at least $m \cdot p_G$.

References

[BG92] M. Bellare, O. Goldreich, "On Defining Proofs of Knowledge," *Advances in Cryptology – CRYPTO '92, Lecture Notes in Computer Science* No. 740, Springer-Verlag, pp. 390–420

[GMR89] S. Goldwasser, S. Micali, C. Rackoff, "The knowledge complexity of interactive proof systems",*SIAM Journal on Computing*, v.18 n.1, p.186–208, Feb. 1989

[GMW91] O. Goldreich, S. Micali, A. Wigderson, "Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems",*Journal of the ACM (JACM)*, Volume 38, Issue 3 (July 1991)

[HM03] S. Halevi, S. Micali, "Conservative Proofs of Knowledge",*manuscript*, Feb 2003

[R81] M. Rabin, "How to exchange secrets by oblivious transfer",*Technical Report TR 81*, Aiken Computation Lab, Harvard University (1981).