CMSC 858K — Advanced Topics in Cryptography March 4, 2004

Lecture 12

Lecturer: Jonathan Katz *Scribe(s): Omer Horvitz, Zhongchao Yu, John Trafton, Akhil Gupta*

1 Introduction

Our goal is to construct an adaptively-secure non-interactive zero-knowledge (NIZK) proof system for any language in NP; we will do so in several steps. We first define the hidden-bits model, and show how to transform any NIZK proof system for a language L in the hidden-bits model into an NIZK proof system for L in the common random string model, using trapdoor permutations. We will then construct an NIZK proof system for any language in NP in the hidden-bits model.[1] Our exposition draws from the work of Feige, Lapidot, and Shamir [6, 2, 1, 3] and also the presentation of [4, Section 4.10].

1.1 From the Hidden-Bits Model to the CRS model

We begin with a quick review of the definitions at hand.

Definition 1 A pair of PPT algorithms $(\mathcal{P}, \mathcal{V})$ is a *non-adaptive NIZK proof system* for a language $L \in \text{NP}$ in the *common random string* (*CRS*) model if:

1. Completeness: For all $x \in L$ where $|x| = k$ and all witnesses w for x,

$$\Pr[r \leftarrow \{0,1\}^{\text{poly}(k)}; \Pi \leftarrow \mathcal{P}(r, x, w) : \mathcal{V}(r, x, \Pi) = 1] = 1.$$

2. (Adaptive) Soundness: For any (unbounded) algorithm $\mathcal{P}^*$, the following is negligible:

$$\Pr[r \leftarrow \{0,1\}^{\text{poly}(k)}; (x, \Pi) \leftarrow \mathcal{P}^*(r) : \mathcal{V}(r, x, \Pi) = 1 \wedge x \notin L].$$

3. Zero-knowledge: There exists a PPT algorithm Sim such the following ensembles are computationally indistinguishable for all PPT A:

$$\begin{array}{ll} (1) & \{(x, w) \leftarrow A(1^k); r \leftarrow \{0,1\}^{\text{poly}(k)}; \Pi \leftarrow \mathcal{P}(r, x, w) : (r, x, \Pi)\} \\ (2) & \{(x, w) \leftarrow A(1^k); (r, \Pi) \leftarrow \mathsf{Sim}(x) : (r, x, \Pi)\}, \end{array}$$

where $x \in L$, $|x| = k$, and w is any witness for x.

$\diamondsuit$

In the above, r is called the *common random string*.

[1] We focus on the case of *non-adaptive* NIZK. However, careful examination of the constructions show that we actually end up with *adaptively-secure* NIZK without any additional modifications.

Definition 2 A pair of PPT algorithms $(\mathcal{P}, \mathcal{V})$ is a *non-adaptive NIZK proof system* for a language $L \in \mathrm{NP}$ in the *"hidden-bits"* model if:

1. Completeness: For all $x \in L$ where $|x| = k$ and all witnesses w for x,

$$\Pr[b \leftarrow \{0,1\}^{\mathrm{poly}(k)}\,; (\Pi, I) \leftarrow \mathcal{P}(b, x, w) : \mathcal{V}(\{b_i\}_{i \in I}\,, I, x, \Pi) = 1] = 1.$$

2. (Adaptive) Soundness: For any (unbounded) algorithm $\mathcal{P}^*$, the following is negligible:

$$\Pr[b \leftarrow \{0,1\}^{\mathrm{poly}(k)}\,; (x, \Pi, I) \leftarrow \mathcal{P}^*(b) : \mathcal{V}(\{b_i\}_{i \in I}\,, I, x, \Pi) = 1 \wedge x \notin L].$$

3. Zero-knowledge: There exists a PPT algorithm Sim such the following ensembles are computationally indistinguishable for any PPT A:

$$\begin{array}{ll} (1) & \{(x, w) \leftarrow A(1^k); b \leftarrow \{0,1\}^{\mathrm{poly}(k)}\,; (\Pi, I) \leftarrow \mathcal{P}(b, x, w) : (\{b_i\}_{i \in I}\,, I, x, \Pi)\} \\ (2) & \{(x, w) \leftarrow A(1^k); (\{b_i\}_{i \in I}\,, I, \Pi) \leftarrow \mathsf{Sim}(x) : (\{b_i\}_{i \in I}\,, I, x, \Pi)\}, \end{array}$$

where $x \in L$, $|x| = k$, and w is any witness for x.

$\diamondsuit$

In the above, b is called the *hidden-bits string* and the $\{b_i\}_{i \in I}$ are the *revealed bits*. We denote the latter by b_I for brevity.

Let $(\mathcal{P}'', \mathcal{V}'')$ be a non-adaptive NIZK proof system for $L \in NP$ in the hidden-bits model. First, we convert the system into one with a precise bound on the soundness error; this will be useful in the analysis of our main transformation. The idea is to run the given system enough times in parallel. Assume that on input x of length k, $(\mathcal{P}'', \mathcal{V}'')$ uses a hidden-bits string of length $p(k)$, for some polynomial p. Define $(\mathcal{P}', \mathcal{V}')$ as follows[2]:

$\mathcal{P}'(b = b_1 \cdots b_{2k}, x, w)$ $\quad$ // $b_j \in \{0,1\}^{p(k)}$
  For $j = 1$ to $2k$, do
    $(\Pi_j, I_j) \leftarrow \mathcal{P}''(b_j, x, w)$;
  Let $\Pi = \Pi_1 | \cdots | \Pi_{2k}$ and $I = \cup_{j=1}^{2k} I_j$
  Output Π, I.

$\mathcal{V}'(b_I, I, x, \Pi)$
  parse Π as $\Pi_1 | \cdots | \Pi_{2k}$ and I as $\cup_{j=1}^{2k} I_j$ (for simplicity, we assume this can be done easily, in some uniquely-specified way)
  If $\mathcal{V}''(b_{I_j}, I_j, x, \Pi_j) = 1$ for *all* $1 \leq j \leq 2k$ then
    output 1;
  else, output 0.

Claim 1 *If $(\mathcal{P}'', \mathcal{V}'')$ is a non-adaptive NIZK proof system for L in the hidden-bits model, then $(\mathcal{P}', \mathcal{V}')$ is a non-adaptive NIZK proof system for L in the hidden-bits model with soundness error at most 2^{-2k}.*

[2] We will slightly abuse the notation here, formatting the inputs and outputs of the prover and verifier in a manner that strays from the one specified in the definition, for clarity; this is purely syntactic.

In the previous lecture, we proved a substantially similar result; we therefore omit proof here.

We would now like to convert $(\mathcal{P}', \mathcal{V}')$ into a non-adaptive NIZK proof system for L in the CRS model. The idea is to use the CRS to "simulate" the hidden-bits string. This is done by treating the CRS as a sequence of images of a one-way trapdoor permutation, and setting the hidden-bits string to be the hard-core bits of the respective pre-images. By letting the prover have access to the trapdoor, he is able to "see" the hidden-bits and also to reveal bits in positions of his choosing.

As before, assume that $(\mathcal{P}', \mathcal{V}')$ uses a hidden-bits string of length $p(k)$ on security parameter k. Let algorithm Gen be a key-generation algorithm for a trapdoor permutation family which, on input 1^k, outputs permutations over $\{0,1\}^k$. Define $(\mathcal{P}, \mathcal{V})$ as follows:

$\mathcal{P}(r = r_0 | \cdots | r_{p(k)}, x, w)$ // $r_i \in \{0,1\}^k$
  $(f, f^{-1}) \leftarrow \mathsf{Gen}(1^k)$;
  For $i = 1$ to $p(k)$ do
    $b_i = r_0 \cdot f^{-1}(r_i)$; // "$\cdot$" denotes the dot product.
  $(\Pi, I) \leftarrow \mathcal{P}'(b_1 \ldots b_{p(k)}, x, w)$;
  Output $(\Pi, I, \{f^{-1}(r_i)\}_{i \in I}, f)$.

$\mathcal{V}(r, x, (\Pi, I, \{z_i\}_{i \in I}, f))$
  For all $i \in I$
    If $f(z_i) = r_i$ then
      let $b_i = r_0 \cdot z_i$;
    else stop and output 0;
  Output $\mathcal{V}'(\{b_i\}_{i \in I}, I, x, \Pi)$.

Note that b_i is computed as in the Goldreich-Levin construction [5], and is a hard-core bit for f. This particular hardcore-bit construction is used, as it guarantees that the "simulated" hidden bits are uniform with all but negligible probability (as opposed to just negligibly close to uniform when we use a general hardcore bit construction). This follows from that fact that $r_0 \cdot y = 0$ for precisely half of the strings $y \in \{0,1\}^k$, and from the fact that $f^{-1}(r_i)$ is uniform in that set, as r_i is uniform and f is a permutation. (Of course, this assumes $r_0 \neq \{0,1\}^k$, which occurs with all but negligible probability.)

Claim 2 $(\mathcal{P}, \mathcal{V})$ *is a non-adaptive NIZK proof system for L in the CRS model.*

Sketch of Proof (Informal) A full proof appears in the previous lecture, so we just remind the reader of the highlights here. Completeness of the transformed proof system is easy to see, as the prescribed $\mathcal{P}$ runs $\mathcal{P}'$ as a subroutine. For soundness, consider first a *fixed* trapdoor permutation (f, f^{-1}). As argued above, this (with all but negligible probability) results in a uniformly-random string b as seen by a cheating prover. So, soundness of the original proof system implies that a prover can only cheat, using this (f, f^{-1}), with probability at most 2^{-2k}. But a cheating prover can choose whatever (f, f^{-1}) he likes! However, summing over all 2^k possible choices of (f, f^{-1}) (we assume here **(a)** that legitimate output of Gen are easily decidable and **(b)** that Gen uses at most k random bits on security parameter k; see last lecture for further discussion) shows that the probability of cheating (e.g., finding a "bad" (f, f^{-1}) that allows cheating) is at most 2^{-k} over the choice of r.

For zero-knowledge, let Sim' be the simulator for $(\mathcal{P}', \mathcal{V}')$. Define Sim as follows:

$\mathsf{Sim}(x)$
  $(\{b_i\}_{i \in I}, I, \Pi) \leftarrow \mathsf{Sim}'(x)$;
  $(f, f^{-1}) \leftarrow \mathsf{Gen}(1^k)$;
  $r_0 \leftarrow \{0,1\}^k$;   // assume $r_0 \neq 0$
  For $i \in I$ do
    Pick $z_i \leftarrow \{0,1\}^k$ subject to $r_0 \cdot z_i = b_i$;
    Set $r_i = f(z_i)$;
  For $i \notin I$, $i \leq p(k)$ do
    Pick $r_i \leftarrow \{0,1\}^k$;
  Output $(r = r_0 | \cdots | r_{p(k)}, (\Pi, I, \{z_i\}_{i \in I}, f))$.

Intuitively, Sim runs Sim', chooses f, then comes up with a CRS that is consistent with the b_i's that Sim' produced. Note that Sim does not know the actual distribution of values for the "hidden bits" at positions $i \notin I$; yet, informally, the security of the trapdoor permutation (and its hard-core bit) ensure that just choosing random r_i at those positions hides the underlying values at those positions anyway.

A complete proof was given in the previous lecture notes. □

2 NIZK for any $L \in NP$ in the Hidden-Bits Model

We now construct a non-adaptive NIZK proof system for a particular NP-Complete language L_0 in the hidden-bits model. Note that this implies a similar result for *any* $L \in NP$: to obtain a system for any $L \in NP$, simply reduce L to L_0 and proceed with the proof system shown below. Soundness, completeness, and zero-knowledge are all clearly preserved.

Specifically, the language L_0 we consider is Graph Hamiltonicity:

$$L_0 = \{G \mid G \text{ is a directed graph with a Hamiltonian cycle}\}$$

(recall that a Hamiltonian cycle in a graph is a sequence of edges that forms a cycle and passes through every vertex exactly once). In our construction, a graph with n vertices will be represented an an n by n boolean matrix, such that entry (i, j) in the matrix is 1 iff there is an edge from vertex i to vertext j (this is the standard *adjacency matrix* representation). In such representation, an n-vertex graph can be identified with a string of length n^2.

For now, we will make the assumption that the hidden-bits string is drawn from a *non-uniform* distribution: instead of being drawn uniformly over strings of length n^2, we assume it is drawn uniformly from strings of length n^2 representing "cycle graphs" (i.e., directed graphs consisting only of a single Hamiltonian cycle). We will show later how to remove this assumption. Given this assumption, define $(\mathcal{P}, \mathcal{V})$ as follows:

$\mathcal{P}(b, G, w)$ // b represents a (random) cycle graph; w is a Hamiltonian cycle in G
  Choose a permutation π on the vertices of G at random from those π that
    map w onto the directed edges of b;

(Imagine "overlaying" G onto b such that the cycle w in G lies on top of the cycle in b)

Let I be the set of positions in b corresponding (under π) to *non-edges* in G

Output π and I.

$\mathcal{V}(\{b_i\}_{i \in I}, I, G, \pi)$

Verify that π is a permutation, and that I contains all positions in b corresponding (under π) to non-edges in G

If all the revealed bits at those positions are 0, accept; otherwise, reject.

Claim 3 $(\mathcal{P}, \mathcal{V})$ *is a non-adaptive NIZK proof system for L_0 in the "hidden-bits" model.*

Sketch of Proof (Informal) Completeness clearly holds. We show that soundness holds with probability 1 (i.e., it is impossible for the prover to cheat). Let G be a graph and assume the verifier accepts. We know that the hidden-bits string b is guaranteed to be a cycle graph, by assumption on the distribution of b. If the verifier accepts, there must be a permutation π under which every non-edge of G corresponds to a non-edge (i.e., "0") in b. But this means, by contrapositive, that every edge ("1") in b corresponds to an edge in G. But since the edges in b form a cycle, this means there must be a cycle in G as well, and hence $G \in L_0$.

To prove zero-knowledge, define Sim as follows:

$\mathsf{Sim}(G)$

Pick a random permutation π on the vertices of G;

Let I be the set of positions corresponding (under π) to *non-edges* in G

Set the values of all "revealed bits" b_I to 0

Output π, b_I, and I

In fact, this gives a *perfect* simulation of $\mathcal{P}$ (although seeing this takes some thought). To see why, let $G \in L_0$ (recall that simulation only needs to work for statements in the language) and consider the distribution over (π, I, b_I) in the real-world. Since b is a random cycle graph, and π is a random permutation mapping the cycle in G to the cycle in b, this means that π is in fact a random permutation. I is a set of positions to which the non-edges of G are mapped under π. Finally, the b_I are all 0. But this is exactly the distribution produced by the simulator. $\square$

References

[1] U. Feige. *Alternative Models for Zero-Knowledge Interactive Proofs*. PhD Thesis, Dept. of Computer Science and Applied Mathematics, Weizmann Institute of Science, 1990. Available from http://www.wisdom.weizmann.ac.il/~feige.

[2] U. Feige, D. Lapidot, and A. Shamir. Multiple Non-Interactive Zero-Knowledge Proofs Based on a Single Random String. In *FOCS*, pp. 308–317, 1990.

[3] U. Feige, D. Lapidot, and A. Shamir. Multiple Non-Interactive Zero-Knowledge Proofs Under General Assumptions. *SIAM Journal on Computing* 29(1): 1–28, 1999.

[4] O. Goldreich. *Foundations of Cryptography, vol. 1: Basic Tools*, Cambridge University Press, 2001.

[5] O. Goldreich and L. Levin. A hard-Core Predicate for all One-Way Functions. In *Symposium on the Theory of Computation*, 1989.

[6] D. Lapidot and A. Shamir. Publicly Verifiable Non-Interactive Zero-Knowledge Proofs. In *Advances in Cryptology - CRYPTO '90*, pp. 353-365, 1990.