CS 600.442 – Modern Cryptography 4 November 2015

Lecture 20: Non-Interactive Zero Knowledge

Instructor: Abhishek Jain *Scribe: Ren Hao*

1 Non-Interactive Zero Knowledge System

1.1 Definition

Define a NIZK proof system for language $\mathbb{L}$ is a set of PPT algorithms (K, P, V) $s.t.$

- Generation: $\sigma \leftarrow K(1^n)$ generates a public random string;
- Prover: $\pi \leftarrow P(\sigma, x, \omega)$, for inputs x and ω;
- Verifier: $\{0,1\} \leftarrow (\sigma, x, \pi)$, where 1 : accept, 0 : reject.

1.2 Properties

NIZK should satisfy completeness, soundness and ZK properties, especially Non-adaptive NIZK and adaptive NIZK should fulfill different soundness property:

- Completeness: $\forall x \in \mathbb{L}, \omega \in R_{\mathbb{L}}(x) : Pr[\sigma \leftarrow K(1^n), \pi \leftarrow P(\sigma, x, \omega) : (\sigma, x, \pi) = 1] = 1$;
- Non-adaptive Soundness: $\forall x \notin \mathbb{L}, \omega \in R_{\mathbb{L}}(x) : Pr[\sigma \leftarrow K(1^n), \exists \pi s.t.(\sigma, x, \pi) = 1] = negl(n)$;
- Adaptive Soundness: $\forall x \notin \mathbb{L}, \omega \in R_{\mathbb{L}}(x) : Pr[\sigma \leftarrow K(1^n), \exists (x, \pi) s.t.(\sigma, x, \pi) = 1] = negl(n)$.
- Non-adaptive ZK: $\exists$ a PPT simulator S $s.t.$ $\forall$ x $\in \mathbb{L}$, $\omega \in R_{\mathbb{L}}(x)$, the following two list are computationally indistinguishable:

$\sigma \leftarrow K(1^n)$	$(\sigma, \pi) \leftarrow S(1^n)$
$\pi \leftarrow P(\sigma, x, \omega)$	Output (σ, π)
output (σ, π)	

- Adaptive ZK: $\exists$ PPT simulators S_1, S_2 $s.t.$ $\forall$ PPT adversary $\mathcal{A}$, the following two list are computationally indistinguishable:

$\sigma \leftarrow K(1^n)$	$(\sigma) \leftarrow S_1(1^n)$
$(x, \sigma) \leftarrow \mathcal{A}$ $s.t. x \in \mathbb{L}$	$(x, \omega) \leftarrow \mathcal{A}(\sigma)$
$\pi \leftarrow P(\sigma, x, \omega)$	$\pi \leftarrow S_2(S_1, x, \mathbb{L})$
output (σ, π)	Output (σ, π)

1.3 Theorem:

Given a NIZK (K, P, V) with non-adaptive soundness, we can construct a NIZK(K',P',V') with adaptive soundness.

2 From NIZK to CRS

Given $\mathbb{L} \bigcap \{0,1\}^{\mathbb{L}^{(n)}}$ for some polynomial $q()$,

- ① Define $NIZK$ in hidden bit model,
- ② Generate transformation form $(NIZK)_{HIDmodel} \rightarrow (NIZK)_{CRSmodel}$,
- ③ Construct $(NIZK)_{HIDmodel}$,
- ② + ③ $\Rightarrow (NIZK)_{CRSmodel}$.

2.1 NIZK proof system in Hidden-Bit model

A $NIZK$ proof system in HID model for $\mathbb{L}$ is a set of PPT algorithm (K_H, P_H, V_H),

- Generator: $r \leftarrow K_H(1^n)$ generates random string CRS,
- Prover: $(I, \phi) \leftarrow P_H(1, x, \omega)$ where $I \in [l], |r| = l$
- Verifier: $\{0,1\} \leftarrow V_H(I, \{r_i\}_{r \in l}, x, \phi)$, where 1 : accept , 0 : reject.

2.2 Theorem:

Assuming the existence of trapdoor permutations and any NIZK proof system (P', V') in the hidden-bits model, we may construct an NIZK proof system (P, V) in the common random string model.
Proof :

- Firstly, we give how to construct(P, V) using (P', V')
  Prover $P(1^k, r, x, \omega)$:
  - Generate (f, f^{-1}). It then computes an n-bit string r' by setting $r'_i = h(f^{-1}9r_i))$.
  - P then runs $P'(1^k.r', x, \omega)$ to obtain π, I.
  - P output $f, \pi, I, \{f^{-1}(r_i)\}_{r \in I}$.

  Verifier $V(1^k, r, x, (f, \pi, I, \{z_i\}_{i \in I}))$:
  - check if f is valid.
  - for each $i \in I$, checks if $f(z_i) = r_i$
  - for each $i \in I$,set $r'_i = h(z_i)$
  - Output $V'(1^k.r'_I, x, \pi, I)$
- The intuition is as follows: assume for a moment that the prover honestly generates (f, f^{-1}) at random, independent of r. Then the string r' constructed by the prover is uniformly distributed.Having the prover send $z + i = f^{-1}(r_i)$ to the verifier has the effect of "revealing" the i^{th} bit of r' to the verifier; also once f^{-1} is fixed the prover cannot "cheat" by changing the value of r'_i.Finally, at least informally, the bits of r' that are not revealed by the prover to the verifier remain "hidden" by the security of f^{-1} and its associated hard-core bit h.

- It is easy to see that (P, V) satisfies the completeness feature and the soundness feature. However, we need to show that (P, V) is zero-knowledge.
- Let Sim' be the simulator for (P', V'), we construct $Sim(1^k, x)$ for (P, V) as follows:
  - $(r'_I, \pi, I) \leftarrow Sim'(1^k, x)$
  - $(f, f^{-1}) \leftarrow Gen(1^k)$
  - foreach $i \in I : z_i \leftarrow \{0,1\}^k s.t. h(z_i) = r'_i, r_i = f(z_i)$
  - foreach $i \notin I : r_i \leftarrow \{0,1\}^k$
  - output $(r, f, \pi, I, \{z_i\}_{i \in I})$

  Intuitively, there are two differences between the real proofs (P) and the simulated proofs (Sim): first, the simulated proofs use the simulator Sim' for the original proof system rather than the actual prover P' for the original proof system. Second, the values $\{r_i\}_{i \notin I}$ now define completely random bits fr0i $\{r'_i\}_{i \notin I}$ in the underlying string r'.
  We will use a hybrid argument to show that the above differences are inconsequential: the first due to the zero-knowledge of (P', V') and the second due to the security of the trapdoor permutation family.
  The formal proof will be given in the next section.

2.3 FORMAL PROOF of computability indistinguishable between P and Sim

2.3.1 the construction of Hybrid

Our goal is to show that: (1) and (2) are computationally indistinguishable, where
(1) $\{(x, \omega) \leftarrow A(1^k); r \leftarrow \{0,1\}^{kn}; (f, \pi, I, \{z_i\}_{i \in I}) \leftarrow P(1^k, r, x, \omega) : (r, x, f, \pi, I, \{z_i\}_{i \in I})\}$
and
(2) $\{(x, \omega) \leftarrow A(1^k); (r, f, \pi, I, \{z_i\}_{i \in I}) \leftarrow Sim(1^k, x) : (r, x, f, \pi, I, \{z_i\}_{i \in I})\}$
We define a intermediate experiment via an algorithm $Hybrid(1^k, x, \omega))$ as follows:

$$\left| \begin{array}{c} r' \leftarrow \{0,1\}^n; \\ (\pi, I) \leftarrow P'(1^k, r', x, \omega) \\ (f, f^{-1}) \leftarrow Gen(1^k); \\ \text{foreach } i \in I z_i \leftarrow \{0,1\}^k s.t. h(z_i) = r'_i; r_i = f(z_i); \text{ foreach } i \notin I : r_i \leftarrow \{0,1\}^k; \\ \text{output}(r, f, \pi, I, \{z_i\}_{i \in I})) \end{array} \right|$$

We need to separately prove that P and Sim are computational indistinguishable with Hybrid. That is to say, we need to separately prove (1) and (2) are computational indistinguishable with the following:
(3) $\{(x, \omega) \leftarrow A(1^k); (r, f, |pi, I, \{z_i\}_{i \in I}) \leftarrow Hybrid(1^k, x, \omega) : (r, x, f, \pi, I, \{z_i\}_{i \in I})\}$.

2.3.2 From Sim to Hybrid

We prove this by contradiction. Assume (2) and (3) are not computational indistinguishable, then there is a distinguisher D can distinguish them with the non-negligible probability. We construct D' that violates Sim' as a zero-knowledge simulator for (P', V') in the hidden-bits model.

$$\left|\begin{array}{c}(f, f^{-1} \leftarrow Gen(1^k) \\ \text{foreach } i \in I : z_i \leftarrow \{0,1\}^k s.t. h(z_i) = r'_i; r_i = f(z_i) \\ \text{foreach } i \notin I : r_i \leftarrow \{0,1\}^k; \\ \text{output} D(r, x, f, \pi, I, \{z_i\}_{i \in I}))\end{array}\right|$$

Let A give (x, ω), D' is the given a tuple (r'_I, x, π, I) and runs as follows :
One can check that if (r'_I, x, π, I) is distributed according to real proofs generated by P, then the input to D is distributed according to (3). On the other hand, if $(r'_I, x, \pi I)$ is distributed as the output of Sim', then the input to D is distributed according to (2). So the distinguishing advantage of D' is equal to the distinguishing advantage of D. But this contradicts the zero-knowledge property of (P', V') with respect to Sim'.

2.3.3 From P to Hybrid

We also prove this by contradiction. Assume that there is a distinguisher between (1) and (3) with non-negligible probability. Then we can construct D' that violates the security of the trapdoor permutation family. We should notice that the security of Gen implies that given a randomly-generated f, no D' output a sequence of bits $r'_1, ..., r'_l$, and receiving in return a sequence of k-bit values $r_1, ...r_l$, can distinguish between the case when each r_i is randomly chosen in $\{0,1\}^k$ and the case when each r_i is randomly chosen in $\{0,1\}^k$ subject to $h(f^{-1}(r_i)) = r'_i$.
Define $D'(1^k, f)$ as:

$$\left|\begin{array}{c}(x, \omega) \leftarrow A(1^k) \\ r' \leftarrow \{0,1\}^n \\ (\pi, I) P'(1^k, r', x, \omega) \\ \text{foreach } i \in I : z_i \leftarrow \{0,1\}^k s.t. h(z_i) = r'_i; r_i = f(z_i) \\ \text{foreach } i \notin I : r_i \leftarrow \{0,1\}^k; \\ \text{output } r'_{\bar{I}} \text{ and get back } r_{\bar{I}} \text{ output} D(r, x, f, \pi, I, \{z_i\}_{i \in I}))\end{array}\right|$$

It is easy to see that in case the values r_I are randomly chosen in $\{0,1\}^k$, then the input to D is distributed according to (3). Though harder to see, it is also the case that when the values $r_{\bar{I}}$ are randomly chosen in $\{0,1\}^k$ subject to $h(f^{-1}(r_i)) = r'_i$ then the input to D is distributed according to (1).
The above shows that the distinguishing advantage of D' is equal to the distinguishing advantage of D. But,as we have noted above, this contradicts the security of the trapdoor permutation family.

3 Citation

Rengarajan Aravamudhan, Nan Wang CMSC 858K Advanced Topics in Cryptography by Jonathan Katz, March 2, 2004 http://www.cs.umd.edu/ jkatz/gradcrypto2/NOTES/lecture11.pdf