Non-Interactive Zero Knowledge (I)

CS 601.442/642 Modern Cryptography

Fall 2017

The Setting

- Alice wants to prove an **NP** statement to Bob without revealing her private witness
- However, Alice only has the resource to send a *single* message to Bob. Therefore, they cannot run an interactive zero-knowledge proof
- To make matters worse, 1-message zero-knowledge is only possible for languages in **BPP**! (Think: Why?)
- Fortunately, they both have access to a *common random string* that was (honestly) generated by someone they both trust
- Can Alice prove statements *non-interactively* to Bob using the common random string?

Non-Interactive Proofs

Syntax. A non-interactive proof system for a language L with witness relation R is a tuple of algorithms $(\mathsf{K}, \mathsf{P}, \mathsf{V})$ such that:

- **Setup:** $\sigma \leftarrow \mathsf{K}(1^n)$ outputs a common random string
- **Prove:** $\pi \leftarrow \mathsf{P}(\sigma, x, w)$ takes as input a common random string σ, a statement $x \in L$ and a witness w and outputs a proof π
- **Verify:** $V(\sigma, x, \pi)$ outputs 1 if it accepts the proof and 0 otherwise

A non-interactive proof system must satisfy completeness and soundness properties

Non-Interactive Proofs (contd.)

Completeness: $\forall x \in L, \forall w \in R(x)$:

$$\Pr\left[\sigma \leftarrow \mathsf{K}(1^n); \pi \leftarrow \mathsf{P}(\sigma, x, w) : \mathsf{V}(\sigma, x, \pi) = 1\right] = 1$$

Non-Adaptive Soundness: There exists a negligible function $\nu(\cdot)$ s.t. $\forall x \notin L$:

$$\Pr\left[\sigma \leftarrow \mathsf{K}(1^n); \exists\ \pi \text{ s.t. } \mathsf{V}(\sigma, x, \pi) = 1\right] \leqslant \nu(n)$$

Adaptive Soundness: There exists a negligible function $\nu(\cdot)$ s.t.:

$$\Pr\left[\sigma \leftarrow \mathsf{K}(1^n); \exists\ (x, \pi) \text{ s.t. } x \notin L \wedge \mathsf{V}(\sigma, x, \pi) = 1\right] \leqslant \nu(n)$$

Note: In non-adaptive soundness, the adversary chooses x before seeing the common random string whereas in adaptive soundness, it can choose x depending upon the common random string

Non-Interactive Zero Knowledge (NIZK)

Definition (Non-Adaptive NIZK)

A non-interactive proof system $(\mathsf{K}, \mathsf{P}, \mathsf{V})$ for a language L with witness relation R is *non-adaptive zero-knowledge* if there exists a PPT simulator $\mathcal{S}$ s.t. for every $x \in L$, $w \in R(x)$, the output distributions of the following two experiments are computationally indistinguishable:

$\mathsf{REAL}(1^n, x, w)$	$\mathsf{IDEAL}(1^n, x)$
$\sigma \leftarrow \mathsf{K}(1^n)$	$(\sigma, \pi) \leftarrow \mathcal{S}(1^n, x)$
$\pi \leftarrow \mathsf{P}(\sigma, x, w)$	
Output (σ, π)	Output (σ, π)

Note: The simulator generates both the common random string and the simulated proof given the statement x is input. In particular, the simulated common random string can depend on x and can therefore only be used for a single proof

Non-Interactive Zero Knowledge (contd.)

Definition (Adaptive NIZK)

A non-interactive proof system $(\mathsf{K}, \mathsf{P}, \mathsf{V})$ for a language L with witness relation R is *adaptive zero-knowledge* if there exists a PPT simulator $\mathcal{S} = (\mathcal{S}_0, \mathcal{S}_1)$ s.t. for every $x \in L$, $w \in R(x)$, the output distributions of the following two experiments are computationally indistinguishable:

$\mathsf{REAL}(1^n, x, w)$	$\mathsf{IDEAL}(1^n, x)$
$\sigma \leftarrow \mathsf{K}(1^n)$	$(\sigma, \tau) \leftarrow \mathcal{S}_0(1^n)$
$\pi \leftarrow \mathsf{P}(\sigma, x, w)$	$\pi \leftarrow \mathcal{S}_1(\sigma, \tau, x)$
Output (σ, π)	Output (σ, π)

Note 1: Here, τ is a "trapdoor" for the simulated common random string σ that is used by $\mathcal{S}_1$ to generate an accepting proof for x without knowing the witness.

Note 2: This definition captures *reusable* common random strings

Remarks on NIZK Definition

- In NIZK, the simulator is given "extra power" to choose the common random string, along with possibly a trapdoor to enable simulation without a witness
- In interactive ZK, the extra power to the simulator was the ability to "reset" the verifier
- Indeed, a simulator must always have some extra power over the normal prover, otherwise, the definition would be impossible to realize for languages outside **BPP**
- **Meaning of the Definition:** Whatever the adversarial verifier could have learnt from the proof, it could have computed on its own using the CRS trapdoor. (Since the CRS trapdoor is generated independent of the witness, this roughly guarantees that no knowledge regarding the witness is revealed.)

From Non-Adaptive to Adaptive Soundness

Lemma

There exists an efficient transformation from any non-interactive proof system $(\mathsf{K}, \mathsf{P}, \mathsf{V})$ *with non-adaptive soundness into a non-interactive proof system* $(\mathsf{K}', \mathsf{P}', \mathsf{V}')$ *with adaptive soundness*

Proof Strategy: Let $\ell(n)$ be the length of the statements

- Repeat $(\mathsf{K}, \mathsf{P}, \mathsf{V})$ polynomially many times (with fresh randomness) so that soundness error decreases to $2^{-2\ell(n)}$
- Non-adaptive soundness means that a randomly sampled σ is "bad" for a statement x with probability $2^{-2\ell(n)}$
- By Union Bound, σ is "bad" for all statements with probability $2^{-\ell(n)}$. Therefore, we have adaptive soundness

NIZKs for **NP**

I. Non-adaptive Zero Knowledge: We first construct NIZKs for **NP** with non-adaptive zero-knowledge property using the following two steps:

Step 1. Construct a NIZK proof system for **NP** in the **hidden-bit model**. This step is unconditional

Step 2. Using trapdoor permutations, transform any NIZK proof system for language in the hidden-bit model to a non-adaptive NIZK proof system in the common random string model

II. Adaptive Zero Knowledge: Next, we transform non-adaptive NIZKs for **NP** into adaptive NIZKs for **NP**. This step only requires one-way functions, which are implied by trapdoor permutations.

Putting all the steps together, we obtain adaptive NIZKs for **NP** based on trapdoor permutations

Roadmap

- **Today:** Defining NIZKs in hidden-bit model, and transformation from NIZKs in hidden-bit model to NIZKs in common random string model
- **Next time:** NIZKs for **NP** in the hidden-bit model
- **Homework:** Non-adaptive NIZKs to Adaptive NIZKs

NIZK in Hidden-Bit Model

Syntax. A non-interactive proof system for a language L with witness relation R in the hidden-bit model is a tuple of algorithms $(\mathsf{K}_{\mathsf{HB}}, \mathsf{P}_{\mathsf{HB}}, \mathsf{V}_{\mathsf{HB}})$ such that:

- **Setup:** $r \leftarrow \mathsf{K}_{\mathsf{HB}}(1^n)$ outputs the hidden random string
- **Prove:** $(I, \pi) \leftarrow \mathsf{P}_{\mathsf{HB}}(r, x, w)$ generates the indices $I \subseteq [|r|]$ of r to reveal, along with a proof π
- **Verify:** $V_{\mathsf{HB}}(I, \{r_i\}_{i \in I}, \pi)$ outputs 1 if it accepts the proof and 0 otherwise

Such a proof system must satisfy completeness and soundness (similar to as defined earlier)

NIZK in Hidden-Bit Model (contd.)

Definition (NIZK in Hidden Bit Model)

A non-interactive proof system $(\mathsf{K_{HB}}, \mathsf{P_{HB}}, \mathsf{V_{HB}})$ for a language L with witness relation R in the hidden-bit model is *(non-adaptive) zero-knowledge* if there exists a PPT simulator $\mathcal{S}_{\mathsf{HB}}$ s.t. for every $x \in L$, $w \in R(x)$, the output distributions of the following two experiments are computationally indistinguishable:

$\mathsf{REAL}(1^n, x, w)$	$\mathsf{IDEAL}(1^n, x)$
$r \leftarrow \mathsf{K_{HB}}(1^n)$	$(I, \{r_i\}_{i \in I}, \pi) \leftarrow \mathcal{S}_{\mathsf{HB}}(1^n, x)$
$(I, \pi) \leftarrow \mathsf{P_{HB}}(r, x, w)$	
Output $\left(I, \{r_i\}_{i \in I}, \pi\right)$	Output $\left(I, \{r_i\}_{i \in I}, \pi\right)$

From NIZK in HB Model to NIZK in CRS Model

Intuition: How to transform a "public" random string into a "hidden" random string

- Suppose the prover samples a trapdoor permutation (f, f^{-1}) with hardcore predicate h
- Given a common random string $\sigma = \sigma_1, \ldots, \sigma_n$, the prover can compute $r = r_1, \ldots, r_n$ where:

$$r_i = h(f^{-1}(\sigma_i))$$

- If f is a permutation and h is a hard-core predicate, then r is guaranteed to be random
- Now r can be treated as the hidden random string: V can only see the parts of it that the prover wishes to reveal

Construction

Let $\mathcal{F} = \{f, f^{-1}\}$ be a family of 2^n trapdoor permutations with hardcore predicate h. Let $(\mathsf{K_{HB}}, \mathsf{P_{HB}}, \mathsf{V_{HB}})$ be a NIZK proof system for L in the hidden-bit model with soundness error 2^{-2n}

Construction of $(\mathsf{K}, \mathsf{P}, \mathsf{V})$:

$\mathsf{K}(1^n)$: Output a random string $\sigma = \sigma_1, \ldots, \sigma_n$ s.t. $\forall i, |\sigma_i| = n$

$\mathsf{P}(\sigma, x, w)$: Execute the following steps:

- Sample $(f, f^{-1}) \leftarrow \mathcal{F}(1^n)$
- Compute $\alpha_i = f^{-1}(\sigma_i)$ for $i \in [n]$
- Compute $r_i = h(\alpha_i)$ for $i \in [n]$
- Compute $(I, \varphi) \leftarrow \mathsf{P_{HB}}(r, x, w)$
- Output $\pi = (f, I, \{\alpha_i\}_{i \in I}, \Phi)$

$\mathsf{V}(\sigma, x, \pi)$: Parse $\pi = (f, I, \{\alpha_i\}_{i \in I}, \Phi)$ and:

- Check $f \in \mathcal{F}$ and $f(\alpha_i) = \sigma_i$ for every $i \in I$
- Compute $r_i = h(\alpha_i)$ for $i \in I$
- Output $\mathsf{V_{HB}}(I, \{r_i\}_{i \in I}, x, \Phi)$

(K, P, V) is a Non-Interactive Proof

- **Completeness:** α is uniformly distributed since f^{-1} is a permutation and σ is random. Further, since h is a hard-core predicate, r is also uniformly distributed. Completeness follows from the completeness of $(\mathsf{K_{HB}}, \mathsf{P_{HB}}, \mathsf{V_{HB}})$

- **Soundness:** For any $f = f_0$, r is uniformly random, so from (non-adaptive) soundness of $(\mathsf{K_{HB}}, \mathsf{P_{HB}}, \mathsf{V_{HB}})$, we have:

$$\Pr_{\sigma}[P^* \text{ can cheat using } f_0] \leqslant 2^{-2n}$$

Since there are only 2^n possible choices of f (verifier checks that $f \in \mathcal{F}$), by union bound, it follows:

$$\Pr_{\sigma}[P^* \text{ can cheat }] \leqslant 2^{-n}$$

Proof of Zero Knowledge: Simulator

Let $\mathcal{S}_{\mathsf{HB}}$ be the simulator for $(\mathsf{K}_{\mathsf{HB}}, \mathsf{P}_{\mathsf{HB}}, \mathsf{V}_{\mathsf{HB}})$

Simulator $\mathcal{S}(1^n, x)$:

1. $(I, \{r_i\}_{i \in I}, \Phi) \leftarrow \mathcal{S}_{\mathsf{HB}}(1^n, x)$
2. $(f, f^{-1}) \leftarrow \mathcal{F}$
3. $\alpha_i \leftarrow h^{-1}(r_i)$ for every $i \in I$
4. $\sigma_i = f(\alpha_i)$ for every $i \in I$
5. $\sigma_i \stackrel{\$}{\leftarrow} \{0,1\}^n$ for every $i \notin I$
6. Output $(\sigma, f, I, \{\alpha_i\}_{i \in I}, \Phi)$

Note: $h^{-1}(r_i)$ denotes sampling from the pre-image of r_i, which can be done efficiently by simply trying random α_i's until $h(\alpha_i) = r_i$

Proof of Zero Knowledge: Hybrids

Hybrid $H_0(1^n, x, w) := \mathsf{REAL}(1^n, x, w)$**:**

1. $\sigma \leftarrow \mathsf{K}(1^n)$ where $\sigma = \sigma_1, \ldots, \sigma_n$
2. $(f, f^{-1}) \leftarrow \mathcal{F}$
3. $\alpha_i \leftarrow f^{-1}(\sigma_i)$ for every $i \in [n]$
4. $r_i = h(\alpha_i)$ for every $i \in [n]$
5. $(I, \Phi) \leftarrow \mathsf{P}_{\mathsf{HB}}(r, x, w)$
6. Output $(\sigma, f, I, \{\alpha_i\}_{i \in I}, \Phi)$

Proof of Zero Knowledge: Hybrids (contd.)

Hybrid $H_1(1^n, x, w)$**:**

1. $\alpha_i \xleftarrow{\$} \{0,1\}^n$ for every $i \in [n]$
2. $(f, f^{-1}) \leftarrow \mathcal{F}$
3. $\sigma_i \leftarrow f(\alpha_i)$ for every $i \in [n]$
4. $r_i = h(\alpha_i)$ for every $i \in [n]$
5. $(I, \Phi) \leftarrow \mathsf{P}_{\mathsf{HB}}(r, x, w)$
6. Output $(\sigma, f, I, \{\alpha_i\}_{i \in I}, \Phi)$

$H_0 \approx H_1$: In H_1, we sample α_i at random and then compute σ_i (instead of sampling σ_i and then computing α_i as in H_0). This induces an identical distribution since f is a permutation

Proof of Zero Knowledge: Hybrids (contd.)

Hybrid $H_2(1^n, x, w)$**:**

1. $r_i \xleftarrow{\$} \{0,1\}$ for every $i \in [n]$
2. $(f, f^{-1}) \leftarrow \mathcal{F}$
3. $\alpha_i \leftarrow h^{-1}(r_i)$ for every $i \in [n]$
4. $\sigma_i = f(\alpha_i)$ for every $i \in [n]$
5. $(I, \Phi) \leftarrow \mathsf{P}_{\mathsf{HB}}(r, x, w)$
6. Output $(\sigma, f, I, \{\alpha_i\}_{i \in I}, \Phi)$

$H_1 \approx H_2$: In H_2, we again change the sampling order: first sample $r = r_1, \ldots, r_n$ at random and then sample α_i from the pre-image of r_i (as described earlier). This distribution is identical to H_1

Proof of Zero Knowledge: Hybrids (contd.)

Hybrid $H_3(1^n, x, w)$**:**

1. $r_i \xleftarrow{\$} \{0,1\}$ for every $i \in [n]$
2. $(f, f^{-1}) \leftarrow \mathcal{F}$
3. $\alpha_i \leftarrow h^{-1}(r_i)$ for every $i \in [n]$
4. $(I, \Phi) \leftarrow \mathsf{P}_{\mathsf{HB}}(r, x, w)$
5. $\sigma_i = f(\alpha_i)$ for every $i \in I$
6. $\sigma_i \xleftarrow{\$} \{0,1\}^n$ for every $i \notin I$
7. Output $(\sigma, f, I, \{\alpha_i\}_{i \in I}, \Phi)$

$H_2 \approx_c H_3$: In H_3, we output random σ_i for $i \in I$. From security of hard-core predicate h, it follows that:

$$\{f(h^{-1}(r_i)\} \approx_c U_n$$

Indistinguishability of H_2 and H_3 follows using the above equation

Proof of Zero Knowledge: Hybrids (contd.)

Hybrid $H_4(1^n, x) := \mathsf{IDEAL}(1^n, x)$**:**

1. $(I, \{r_i\}_{i \in I}, \Phi) \leftarrow \mathcal{S}_{\mathsf{HB}}(1^n, x)$
2. $(f, f^{-1}) \leftarrow \mathcal{F}$
3. $\alpha_i \leftarrow h^{-1}(r_i)$ for every $i \in I$
4. $\sigma_i = f(\alpha_i)$ for every $i \in I$
5. $\sigma_i \xleftarrow{\$} \{0,1\}^n$ for every $i \notin I$
6. Output $(\sigma, f, I, \{\alpha_i\}_{i \in I}, \Phi)$

$H_3 \approx_c H_4$: In H_4, we swap P_{HB} with $\mathcal{S}_{\mathsf{HB}}$. Indistinguishability follows from the zero-knowledge property of $(\mathsf{K}_{\mathsf{HB}}, \mathsf{P}_{\mathsf{HB}}, \mathsf{V}_{\mathsf{HB}})$