COM S 6830 – Cryptography | Sept 15, 2009

Lecture 6: Collections of One-Way Functions and Hard-Core Bits

Instructor: Rafael Pass | *Scribe: Gabriel Bender*

1 Collections of One-Way Functions

Definition 1 *A collection of one-way functions is a family of functions $\mathcal{F} = \{f_i : \mathcal{D}_i \to \mathcal{R}_i\}_{i \in I}$ such that:*

1. *It is easy to sample a function: There is some $Gen \in \mathsf{PPT}$ such that $Gen(1^n)$ outputs an index $i \in I$.*
2. *It is easy to sample the domain: There is a PPT machine which, on input $i \in I$, samples from $\mathcal{D}_i$.*
3. *It is easy to evaluate: There is a PPT machine which, on input $i \in I, x \in \mathcal{D}_i$, can compute $f_i(x)$.*
4. *It is hard to invert:*

$$(\forall A \in \mathsf{nuPPT}\ )(\exists\ neg\ \epsilon)(Pr[i \leftarrow Gen(2^n); x \leftarrow \mathcal{D}_i : A(1^n, i, f_i(x)) \in f_i^{-1}(f_i(x))] \leq \epsilon(n))$$

Here are some candidates:

- Multipling large primes: $I = \mathbb{N}$, $\mathcal{D}_n = \{p, q : p \text{ and } q \text{ are } n\text{-bit primes}\}$, $Gen(1^n) \to n$, and $f_i(x, y) \to xy$.
- Exponentiation: $Gen(1^n) \to (p, g)$ where p is a random n-bit prime and g sis a generator for $\mathbb{Z}_p^*$. In this case, $f_{p,g}(x) \to g^x \mod p$. This function is one-to-one, ie. is a *permutation*. The Discrete Log Assumption states that this gives us a collection of one-way functions.
- RSA Collection: $Gen(1^n) \to (n, e)$ where $n = pq$ for random n-bit primes p and q, and e is a random element in $\mathbb{Z}_{\varphi(n)}$. In this case, $f_{n,e}(x) = x^e \pmod{n}$. This setup gives us a trapdoor permutation (ie. it's invertible with some extra information; more in this to come).

Proposition 1 *There exists a collection of one-way functions iff there exists a one-way function.*

Proof. If we have a one-way function g, define $Gen(1^n) \to n$, and $\mathcal{D}_n = \{0,1\}^n$. To sample from the domain for index set n, we generate a random string in $\{0,1\}^n$. Then we can define $f_i(x) = g(x)$.
Now suppose we have a collection of one-way functions with index generator $Gen : \{0,1\}^n \to I$ and sampling function $\sigma : i \to \mathcal{D}_i$. Then we can define a one-way function $g(r_1, r_2)$ with $|r_1| = |r_2|$ by setting $i \leftarrow Gen(r_1)$ and then using r_2 to sample from $\mathcal{D}_i$.

2 Hard-Core Bits

We know that if one-way functions exist then there exists a one-way function f such that, given $f(x)$ with $x \in \{0,1\}^n$, we can guess any individual bit of x with decent probability.

Definition 2 *A predicate $b : \{0,1\}^* \to \{0,1\}$ is hard-core for a function f if*

- *b is* PPT*-computable*
- $(\forall A \in \mathsf{nuPPT})(\exists \text{ } neg \text{ } \epsilon)(\forall n \in \mathbb{N})(Pr[x \leftarrow \{0,1\}^n : A(1^n, f(x)) = b(x)] \leq \frac{1}{2} + \epsilon(n))$

Every one-way function can be slightly modified to have a hard-core bit.

Theorem 1 *Let f be a OWF (OWP). Then $f'(x,r) = f(x), r$ with $|x| = |r|$ is a OWF (OWP) and $b(x,r) = \langle x, r \rangle = \sum_{i=1}^{n} x_i r_i \pmod 2$ is hard-core for f'.*

We will prove a full version of this theorem for next class. For now, let us look at two simplified versions of this theorem.
Fact: $\langle a, b + c \rangle = \langle a, b \rangle + \langle a, c \rangle$

- First Proof Suppose A computes $b(x)$ from $f(x)$ with probability 1. We will construct a turing machine B that inverts $f(x)$. Then we compute the ith bit of x as $x_i = A(y, e_i)$, where $e_i \in \{0,1\}^n$ has a 1 at position i and 0 everywhere else.
- Second Proof Now suppose that A computes $\langle x, r \rangle$ with probability $\frac{3}{4} + \epsilon$, where ϵ is $\frac{1}{\text{poly}}$. Let $S = \{x : Pr[A(1^n, f(x), r)] = b(x,r) > \frac{3}{4} + \frac{\epsilon}{2}\}$. It follows that $Pr[x \in S] \geq \frac{\epsilon}{2}$.

We show B s.t. B inverts $y = f(x)$ with high probability when $x \in S$:
for $i \leftarrow 1, \ldots, n$

- $r \leftarrow \{0,1\}^n$
- $e' \leftarrow e_i \oplus r$
- Compute our guess $g_i = A(y, r) \oplus A(y, r')$

- Repeat $poly(\frac{1}{\epsilon})$ times, and set x_i to be the result (0 or 1) which was obtained the most times.

Our results is the the concatenation of the bits $x_1 x_2 \ldots x_n$.
Then $Pr[A(y,r) \neq b(x,r)] \leq \frac{1}{4} - \frac{\epsilon}{2}$, and $Pr[A(y,r') \neq b(x,r')] \leq \frac{1}{4} - \frac{epsilon}{2}$. By the union bound, $Pr[A(y,r) = b(x,r) \wedge A(y,r') = b(x,r')] \geq \frac{1}{2} + \epsilon$. By the Chernoff bound, each x_i is correct with probability $1 - 2^{-n}$, and so the entire string is correct with probability $1 - \frac{n}{2^n}$.