CSC 2426: Fundamentals of Cryptography **Fall 2023**

Lecture 2: One-Way Functions

Instructor: Akshayaram Srinivasan *Scribe: Brian Fu*

Date: 2023-09-18

Recap.

- **Negligible Functions**: A function $\mu : \mathbb{N} \to [0,1]$ is said to be negligible if $\forall$ polynomials p, $\exists\ n_0 \in \mathbb{N}\ s.t.\ \forall\ n \geq n_0$, where $\mu(n) < 1/p(n)$. In other words, negligible functions are smaller than any inverse polynomial for large enough n.
- **Non-uniform PPT machines**: It is given by an infinite family of TMs $\{M_0, M_1, \ldots, M_n, \ldots\}$ where M_i is used on inputs of length i. Specifically, there is no bound on time needed to generate the description of M_n for each $n \in \mathbb{N}$. This distinguishes it from uniform PPT machines. We will use nuPPT to denote non-uniform PPT machines.

2.1 One-Way Functions

Recall the definition of One-Way Functions (OWFs) from last class.

Definition. Let $f = \{f_n : \{0,1\}^{k(n)} \to \{0,1\}^{m(n)}\}_{n\in\mathbb{N}}$ be a family of functions. We say f is a *one-way function* if:

- **Easy to evaluate:** There exists a polynomial algorithm that for every $n \in \mathbb{N}$ takes $x \in \{0,1\}^{k(n)}$ and outputs $f_n(x)$.
- **Hard to Invert:** For all non-uniform PPT $\mathcal{A}$,

$$\Pr_{x\leftarrow\{0,1\}^{k(n)}}[\mathcal{A}(1^{k(n)}, f_n(x)) = y \text{ s.t. } f(y) = f(x)] \text{ is negligible}$$

Note on $k(n)$. Suppose $2^{k(n)} = p(n)$ for some polynomial $p(\cdot)$ for infinitely many n. In that case, we will argue that the function cannot be one-way. Specifically, for every such n, we can build a polynomial-size table containing the inverse of each possible image of the OWF. This table can be built in polynomial time and can be used to trivially invert the one-way function. Hence, it is necessary that $2^{k(n)} > p(n)$ for every polynomial $p(\cdot)$ for large enough n. In other words, we require $2^{-k(n)}$ to be negligible.

No string has many pre-images. Suppose there exists a string $z \in \{0,1\}^{m(n)}$ such that $|\{x \in \{0,1\}^{k(n)} : f_n(x) = z\}| \geq \frac{2^{k(n)}}{p(n)}$ for some polynomial $p(\cdot)$ for infinitely many n. Then, we will now argue that the function cannot be one-way. Specifically, for every such n, we can non-uniformly find a pre-image of z and hardwire that pre-image in the description of the adversary. If we sample $x \leftarrow \{0,1\}^{k(n)}$ and evaluate f_n on the

sampled input, we get z with $1/p(n)$ probability. In that case, we can trivially output the hardwired inverse and invert the one-way function with non-negligible probability. Hence, for any $z \in \{0,1\}^{m(n)}$, we have $\frac{|\{x \in \{0,1\}^{k(n)} : f_n(x) = z|}{2^{k(n)}}$ is negligible.

Existance of OWFs. If OWFs exist then $\mathsf{P} \neq \mathsf{NP}$. Note that checking if a candidate inverse is valid or not can be done in polynomial time. Hence, if $\mathsf{P} = \mathsf{NP}$, then any function can be inverted in polynomial time and hence, OWFs cannot exist. Therefore, proving the existence of OWFs is at least as hard as separating P from NP.

Candidates for OWFs. Given that we are very far from proving $\mathsf{P} \neq \mathsf{NP}$ (which is a necessary condition for the existence of OWF), we do not know if OWFs exist. However, we have several candidates for OWFs. The security of these candidates is based on conjectured hardness of certain mathematical problems. These mathematical problems have been analysed by many mathematicians and computer scientists over several decades and we do not yet have polynomial-time solutions for these problems. Some examples of hard problems that give rise of one-way functions are: (1) Factoring, (2) Discrete Logarithm (DLog) (Factoring and DLog are in BQP via Shor's algorithm and hence, do not give rise to candidate one-way functions that are secure against quantum computers), (3) Decoding random linear codes, and (4) Finding short vectors in lattices (the last two candidates are conjectured to be secure against quantum computers).

2.2 Composition of OWFs

Let $f = \{f_n : \{0,1\}^{k(n)} \mapsto \{0,1\}^{k(n)}\}$ be a one-way function. Here, we assume that $m(n) = k(n)$ and such one-way functions are said to be length-preserving. What can we say about $g = \{g_n = f_n(f_n(\cdot))\}_{n \in \mathbb{N}}$? Is it one-way or is there a one-way function f for which g is not one-way?

Theorem 2.1 *If length-preserving one-way functions exist, then there exists a one-way function f such that g is not one-way.*

Proof: Let $h = \{h_n : \{0,1\}^{k(n)} \to \{0,1\}^{k(n)}\}$ be a length-preserving OWF. We will define another length-preserving one-way function f and prove that f is one-way assuming h is one-way. However, for this particular f, we will prove that composition of f with itself is not one-way.

Define $f = \{f_n : \{0,1\}^{2k(n)} \to \{0,1\}^{2k(n)}\}_{n \in \mathbb{N}}$ where

$$f_n(x_1, x_2) = \begin{cases} 0^{2k(n)} & \text{if } x_1 = 0^{k(n)} \\ 0^{k(n)} \| h_n(x_2) & \text{otherwise} \end{cases}$$

Here, $x_1 \in \{0,1\}^{k(n)}$ denotes the first $k(n)$ bits and $x_2 \in \{0,1\}^{k(n)}$ denotes the last $k(n)$ bits of the input. $\|$ denotes concatenation.

We must prove:

1. f is one-way.
2. g is NOT one-way

The second part is trivial to prove. Note that $f_n(x_1, x_2)$ for any $x_1, x_2 \in \{0,1\}^{k(n)}$ is either the bitstring $0^{2k(n)}$ or bitstring $0^{k(n)} \| h_n(x_2)$. As the first half of both the strings is $0^{k(n)}$, $f_n(f_n(x_1, x_2)) = 0^{2k(n)}$. A constant function cannot be one-way as any domain point is a valid inverse.

Let us now prove the first part. Suppose f is not one-way. Then, there must exist a nuPPT $\mathcal{A}$ such that

$$\mu(n) = \Pr_{(x_1,x_2)\leftarrow\{0,1\}^{2k(n)}}[\mathcal{A}(1^{2k(n)}, f_n(x_1, x_2)) = (x_1', x_2'), \text{ s.t. } f_n(x_1, x_2) = f_n(x_1', x_2')]$$

(i.e., $Pr[\mathcal{A} \text{ inverts } f]$) is non-negligible. We will use $\mathcal{A}$ to design an inverter $\mathcal{B}$ for h that succeeds with non-negligible probability. This would contradict the one-wayness of h.

Define $\mathcal{B}(1^{k(n)}, h_n(x))$ as follows:

1. $\mathcal{B}$ runs $\mathcal{A}(1^{2k(n)}, 0^{k(n)} \| h_n(x))$ to obtain $x_1' \| x_2'$.
2. $\mathcal{B}$ outputs x_2'.

If $\mathcal{A}$ runs in polynomial time, then so does $\mathcal{B}$. We now prove that $\mathcal{B}$ produces a valid inverse for h with non-negligible probability. To see why this is the case,

$$\begin{aligned}
\mu(n) &= \Pr[x_1 = 0^{k(n)}] \Pr[\mathcal{A} \text{ inverts } f | x_1 = 0^{k(n)}] + \Pr[x_1 \neq 0^{k(n)}] \Pr[\mathcal{A} \text{ inverts } f | x_1 \neq 0^{k(n)}] \\
&\leq \frac{1}{2^{k(n)}} + \Pr[\mathcal{A} \text{ inverts } f | x_1 \neq 0^{k(n)}]
\end{aligned}$$

Note that:

$$\begin{aligned}
\Pr[\mathcal{A} \text{ inverts } f | x_1 \neq 0^{k(n)}] &= \Pr[h_n(x_2) = 0^{k(n)}] \Pr[\mathcal{A} \text{ inverts } f | x_1 \neq 0^{k(n)}, h_n(x_2) = 0^{k(n)}] \\
&\quad + \Pr[h_n(x_2) \neq 0^{k(n)}] \Pr[\mathcal{A} \text{ inverts } f | x_1 \neq 0^{k(n)}, h_n(x_2) \neq 0^{k(n)}] \\
&\leq \Pr[h_n(x_2) = 0^{k(n)}] + \Pr[\mathcal{A} \text{ inverts } f | x_1 \neq 0^{k(n)}, h_n(x_2) \neq 0^{k(n)}] \\
&\leq \nu(n) + \Pr[\mathcal{B} \text{ inverts } h]
\end{aligned}$$

Note that $\nu(n)$ is negligible as any particular image cannot have a non-negligible probability of occurring (see earlier discussion about this in Section 2.1). Also, whenever A inverts f conditioned on $x_1 \neq 0^{k(n)}$ and $h_n(x_2) \neq 0^{k(n)}$, B produces a valid inverse for h.

Hence, $\Pr[B \text{ inverts } h] \geq \mu(n) - \frac{1}{2^{k(n)}} - \nu(n)$. Note that $\frac{1}{2^{k(n)}}$ and $\nu(n)$ are negligible while $\mu(n)$ is non-negligible. Hence, $\Pr[B \text{ inverts } h]$ is non-negligible and this concludes the proof. ■

2.3 Hard-core Predicate

Definition 2.2 *Let $h = \{h_n : \{0,1\}^{k(n)} \to \{0,1\}\}_{n\in\mathbb{N}}$ be a family of predicates. We say that h is a hard-core predicate for $f = \{f_n : \{0,1\}^{k(n)} \to \{0,1\}^{m(n)}\}_{n\in\mathbb{N}}$ if:*

- ***Easy to Evaluate:*** *There exists a polynomial-time that for every $n \in \mathbb{N}$ takes $x \in \{0,1\}^{k(n)}$ and outputs $h_n(x)$.*

- ***Hard to predict given the output of*** f*: For all nuPPT* A*:*

$$\Pr_{x \in \{0,1\}^{k(n)}}[A(1^{k(n)}, f_n(x)) = h_n(x)] \leq 1/2 + \mathsf{negl}(n)$$

In the rest of the discussion, we will restrict ourselves to the case where f_n is injective for each $n \in \mathbb{N}$. If this was the case, then $f_n(x)$ completely determines x and hence, $h_n(x)$. However, the definition of hard-core predicate guarantees that $h_n(x)$ is almost like a random bit to all computationally-bounded machines. In other words, even if $h_n(x)$ is determined information-theoretically, it is computationally random (a.k.a. pseudorandom).

Our next observation is that if h is a hard-core predicate for f then f is necessarily one-way. Suppose f is not one-way, then there exists an inverter for f that succeeds with non-negligible probability $\epsilon(n)$. We can use this inverter to predict $h_n(x)$ with probability non-negligibly more than $1/2$. Specifically, whenever the inverter succeeds, we can compute $h_n(x)$ correctly. Whenever the inverter fails, we output a random bit. This predictor manages to correctly predict $h_n(x)$ with probability $\epsilon(n) + 1/2(1 - \epsilon(n)) = 1/2 + \epsilon(n)/2$. As $\epsilon(n)/2$ is non-negligible, this contradicts the definition of hard-core predicate.