CS 6160 Cryptology Lecture 4: Hardcore Bits

Maria Francis

September 9, 2021

Hardcore Bit of an OWF

- Note: we skip concepts like weak and strong one-way functions and family of OWF.
- Hard core bit $h(x)$ is easy to compute from x, but impossible to guess from $f(x)$.
- We will:
  - see examples of hardcore bits for modular exponentiation and RSA,
  - state the seminal result of Goldreich-Levin that shows a general hardcore bit for any OWF (without proof),
  - describe a Private Key Encryption which encrypts 1 bit!

Motivation for Hardcore Bits

- What if $f(x)$ reveals some information about x? We would like to hide all the information, but then we are back to perfect secrecy.
- We focus on hiding specific and carefully chosen partial information about x given $f(x)$.
- If you want to encrypt something, you need to know not just that your whole message is hard to decipher, but that every (or which) bit in your message is hard to guess
- For e.g: the bit which encodes whether you are sending a buy/sell a stock!
- First step: how to hide one bit of information about x.

Definition of a Hardcore Bit

A function $h : \{0,1\}^* \to \{0,1\}$ is called a hardcore bit for a function f if

- $h(x)$ is poly-time computable from x
- No PPT algo that can predict $h(x)$ given $f(x)$ better than flipping a coin:

$$(\forall PPT\ A)\ Pr[A(f(x)) = h(x) : x \xleftarrow{R} \{0,1\}^k] \leq \frac{1}{2} + \mathrm{negl}(k).$$

Ex: Definition of hard-core predicates do not require f to be one-way, but if f is a permutation then it cannot have a hardcore predicate unless it is one-way.

Hardcore Bits

$$(\forall PPT\ A)\ Pr[A(f(x)) = h(x) : x \xleftarrow{R} \{0,1\}^k] \leq \frac{1}{2} + \mathrm{negl}(k).$$

- Why add $\frac{1}{2}$ rather than $\mathrm{negl}(k)$?
- Output of h is only one bit, so A can always guess with probability $\frac{1}{2}$ by flipping a coin.
- The aim is to say h is not only hard to compute, it is even hard to predict.
- The knowledge of $f(x)$ does not allow us to predict $h(x)$ any better than a guess, so $h(x)$ looks random given $f(x)$.
- $h(x)$ need not be from the string x but should depend on x in complex but efficiently computable ways.

Construction of Hardcore Bits

Construction can happen in two ways:

- Exhibit a hardcore bit for a concrete OWF
- Exhibit a hardcore bit for an arbitrary function – strongest construction!

Modular Exponentiation Function

- $f(x) = y = g^x \mod p$, g is the generator of $\mathbb{Z}_p^*$.

$$MSB(x) = \begin{cases} 0, & \text{if } x < \frac{p-1}{2} \\ 1, & \text{if } x \geq \frac{p-1}{2} \end{cases}$$

Theorem

If $f(x) = g^x \mod p$ is a OWP, then $MSB(x)$ is a hardcore bit for f.

LSB is NOT a hardcore bit

Given $y \in \mathbb{Z}_p^*$ and p,g such that $y = g^x \bmod p$, there exists an efficient algorithm to compute $LSB(x)$.

- What we will show is x is even iff $y^{\frac{p-1}{2}} \equiv 1 \bmod p$.
- If $x = 2w$ is even, then $y^{(p-1)/2} = g^{(p-1)w} = 1 \bmod p$ (by Fermat's theorem)
- If $g^{x(p-1)/2} = 1$ and g is a generator, then $x \cdot \frac{(p-1)}{2} = 0 \bmod (p-1)$
- Why? The order of an element is the least value raised to which gives 1, the order divides any other value for which the element is raised to gives 1
- Also order of a generator is the number of elements in the group.
- $x \cdot \frac{(p-1)}{2} = k(p-1)$ for some integer k and $x = 2k$ is even!

MSB as a hardcore bit

The idea is to show this lemma.

Lemma

If there exists a PPT that can always compute $MSB(x)$ from $f(x)$ then there is a PPT that can always invert $f(x)$, i.e. compute the discrete log.

PPT algorithm for inversion of OWF

- Given $y = g^x \mod p$, $LSB(x) = x_k$ is easy to compute, so we can determine x_k.
- If x is even, then y can be viewed like this: $g^{[x_1 \dots x_{k-1} 0]} = (g^{[x_1 \dots x_{k-1}]})^2 \mod p$.
- If x is odd, $x_k = 1$ then first divide y by g and then do the same.

PPT algorithm for inversion of OWF

even $g^{110} \stackrel{?}{=} (g^{11})^2 \bmod p$

$= g^{11+11}$

$= g^{110}$

if i take square root of this value I will get g power the value with LSB removed!

odd $g^{111}/g = g^{110}$

back to even case!

PPT algorithm for inversion of OWF

- We can extract square roots modulo p efficiently.
- We have $g^{[x_1 \dots x_{k-1}]}$ and keep doing the same thing – take LSB and take square root and we will have all bits of x.
- But $g^{[x_1 \dots x_{k-1} 0]}$ has two square roots : $y_0 = g^{[x_1 \dots x_{k-1}]}$ (we want this one) and $y_1 = (-g^{[x_1 \dots x_{k-1}]}) = g^{\frac{p-1}{2} + [x_1 \dots x_{k-1}]}$. ($-1 = g^{(p-1)/2}$)

Hardcore Bits of OWF

- So which square root is which? Use *MSB*!
- $MSB(Dlog(y_0)) = 0$ and $MSB(Dlog(y_1)) = 1$.
- *MSB* plays a critical role in deciding which square root of y corresponds to $x/2$.
- All the bits of x are hardcore bits of RSA.

Construction of Hardcore bit for Arbitrary OWF

- The aim is to see if *any OWF f has some easy and natural hardcore bits*.
- Can any OWF have some particular bit x_i in x which is hardcore for f.
- Ans: No! From any arbitrary OWF f it is possible to construct another OWF g such that none of the bits of x are hardcore for g.
- A concrete boolean function h (not necessarily an input bit) is a hardcore bit for ALL OWFs. No universal h exists.
  - Since for such a h and OWF f, let $g(x) = f(x) \circ h(x)$.
  - g is an OWF (an inverter for g implies an inverter for f (check!)) h is clearly not a hardcore for g.

Examples for why simple ideas won't work

- Consider $h(x) = \oplus_{i=1}^{n} x_i$, x_is are bits of x.
- Our idea is:if f is a one-way function, then $f(x)$ must hide at least one of the bits $x_i \Rightarrow$ XOR of the bits of x is hard to compute.
- But it doesn't work!Let $g(x) = (f(x), \oplus_{i=1}^{n} x_i)$. g is one-way if f is one-way.
- But $h(x)$ is not a hard-core predicate for g since it is part of the output.
- Thus, for any fixed predicate you can always find another one-way function for which it is not a hard-core predicate.

Trivial hard-core predicates

- Let $f(x) = x_1 \ldots x_{n-1}$ you drop x_n.
- Seeing the output no way to find x_n so it is a hard-core predicate for $f(x)$.
- But f is not one-way!Why? Since for a given output $x_1 \ldots x_{n-1}$ append 0 or 1 and you have the required preimage!
- For a one-way function we need to find only any preimage not the original preimage!

One-Way Functions to Pseudorandomness

- Aim: Construct Pseudorandom Generators using one-way functions/permutations.
- First step: T.S.T a hard-core predicate exists for any one-way function. Actually we do not know if that is true, we do something weaker but that is good enough!
- What the Goldreich-Levin theorem shows is that given a one-way function/permutation f we can construct a different one way function g with a hard-core predicate of g.

Hardcore bit for Arbitrary OWF - What Do We Have?

Theorem (Goldreich-Levin Theorem)

Assume one-way functions/permutations exist. Then there exists a one-way function/permutation g and a hard-core predicate h of g.

- We wont go into the technical details of the proof but lets try and be more specific.

Parity of $x = x_1 x_2 \cdots x_k \in \{0,1\}^k$ w.r.t. $r = r_1 r_2 \cdots r_k \in \{0,1\}^k$ is defined as
$h(x, r) = r_1 x_1 \oplus r_2 x_2 \ldots \oplus r_k x_k = r_1 x_1 + r_2 x_2 + \ldots + r_k x_k \mod 2$.
r is like a selector of bits of x have to be included in parity, based on r_i being 1 or 0.

Goldreich Levin Construction/Theorem

- Given a OWF f we construct an auxiliary function $g_f(x, r) = f(x) \circ r$, where $|x| = |r|$.
- A random input for g samples both r and x at random from $\{0, 1\}^k$
- Since f is a OWF, g is a OWF too. In fact if f is a permutation then g_f is also one.

Theorem (Goldreich Levin)

f is a OWF, then $h(x, r)$ (random parity for randomly selected x) is a hardcore bit for g_f.

Goldreich Levin Construction/Theorem

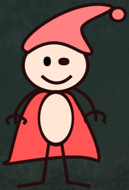
- A hardcore bit for g_f is almost as same as that for f, since $f \equiv g_f$ (r is part of the output, inverting g_f exactly equiv. to inverting f).
- We abuse terminology when we say "since f is a OWF, let us take its hardcore bit $h(x)$" – we can take hardcore bit for f or Goldreich-Levin bit for g_f.
- Same way when we say "every OWF has a hardcore bit" or "most parities of f are hardcore".
- For the proof of the theorem, O. Goldreich's textbook is a good resource.

A detour! – PKC for just one bit

Alice

PK(Bob) : *TDP*, *f*
SK(Bob): *t*, trapdoor info
Hardcore bit: *h* of *f*
(or use Goldreich-Levin)

Bob



Eve

Protocol

$$c = (f(x), h(x) \oplus b) \longrightarrow$$

Alice

For encryption of b:
Choose a random $x \in \{0,1\}^k$

Bob

x from $f(x)$ using t
$h(x)$ from x
$b = (h(x) \oplus b) \oplus h(x)$

Eve

Can Eve attack?

Alice

$$c = (f(x), h(x) \oplus b) \longrightarrow$$

Bob

To know b, Eve should learn $h(x)$
But Eve only knows f
$h(x)$ is hardcore bit, so Eve can't guess better than $1/2$