CS 290G (Fall 2014) – Introduction to Cryptography Oct 2nd, 2014

Lecture 4: Strong OWF

Instructor: Rachel Lin *Scribe: Nilo Redini*

1 Recap

Last class, we gave some formal definitions about one-way functions and their properties. Particularly we formally defined when a one-way function is strong (i.e., hard to invert). That is to say when:

$$\begin{aligned} &\forall\ non-uniform\ PPT\ adversaries,\ \exists\ a\ negligible\ \epsilon(n) \\ &Pr_x[A\ inverts\ f(x)] \leq \epsilon(n) \end{aligned} \tag{1}$$

We also formally defined the opposite type of one-way functions, that is to say the weak one-way functions.
Formally a one-way function is considered weak when:

$$\begin{aligned} &\forall\ non-uniform\ PPT\ adversaries: \\ &Pr_x[A\ inverts\ f(x)] < (1 - \frac{1}{p(n)}),\ where\ p\ is\ a\ polynomial \end{aligned} \tag{2}$$

Furthermore, we proved the hardness of factoring[1]:

$$\begin{aligned} &\forall\ adversary\ A\ \exists\epsilon(n): \\ &Pr_{p,q\in\Pi_n}[A\ factors\ N = p * q] < \epsilon(n) \end{aligned} \tag{3}$$

Finally, we proved that the hardness of factoring implies that the inverse function (that is to say the multiplication operation or $f_{mult}(x, y)$) is a weak one way function.
This class will combine these definitions together. Particularly, today's lesson will focus on proving an important theorem which states that if we are able to build a weak one-way function, then we are able to build a strong one-way function. Note that throughout this document when we say "adversary" (or "attacker") we will always implicitly refer to a non-uniform PPT TM adversary.

2 Hardness of factoring and strong one-way functions

Before tackling the main theorem introduced in Section 1, let us start with a simpler theorem which proves an important relationship between the hardness of factoring and strong one-way functions.

[1] π_n represents the set of prime numbers as defined in the previous lesson.

Theorem 1 *Hardness of factoring* $\Longrightarrow$ *Strong one-way function*

Let us recall the weak one-way function we gave last time, which exploits the factoring hardness:

$$f_{mult}(x,y) = \begin{cases} 1 & x \; or \; y \; = \; 1 \\ x * y & oth. \end{cases} \tag{4}$$

Let us now declare another function $f(\overline{x},\overline{y})$ defined as follows:

$$f(\overline{x},\overline{y}) = f_{mult}(x_1,y_1) * f_{mult}(x_2,y_2) * ... * f_{mult}(x_m,y_m) \tag{5}$$

Note that $\overline{x}$ and $\overline{y}$ are two vectors, defined as $\overline{x} = \{x_i, 1 \leq i \leq m\}$ and $\overline{y} = \{y_i, 1 \leq i \leq m\}$.

Like in the last lecture, we are trying to combine different instances of a weak one-way function (i.e., $f_{mult}(x,y)$), hoping that the probability of inverting the resulting function (i.e., $f(\overline{x},\overline{y})$) is much lower. As we already said last time, such a resulting probability is not merely given by the product of the probabilities for breaking each single $f_{mult}(x_i,y_i)$, because the probabilities of inverting different $f_{mult}(x_i,y_i)$ are not independent. To be convinced of this just think about an attacker who decides to try to find the inputs all at once. Such an attacker can decide that if at least one input is not correct – that is to say computing f over such a set of inputs does not equal to $f(\overline{x},\overline{y})$ he is trying to break – he discards all the input and tries again from scratch.

Before proving the theorem, let us define some quantities and claims which we will use hereafter. Particularly, let us define a set of inputs which we know (from the last lecture) are hard to invert based on the hardness of factoring:

$$S_{hard} = \{(x,\ y) \mid x,\ y \; are \; prime\} \tag{6}$$

we know that:

1. by density of primes: $|S_{hard}| \geq \frac{1}{4n^2} \ * \ 2^{2n}$
2. by hardness of factory: $P_{x,y}[A \; inverts \; f_{mult}(x,y) \mid x,\ y \in S_{hard}] \leq \varepsilon(n)$

Even though an adversary can try to find the inputs all together, we know that all the x_i and all the y_i are chosen at random independently.

Let us define $Event_i$ as the event that the the i'th pair (x_i,y_i) in $(\overline{x},\overline{y})$ chosen at random belongs to S_{hard}. Event $Event_i$ satisfies:

1. $Pr[Event_i] \geq \frac{1}{4n^2}$,
2. $Event_i$ for different i's are independent.

We are going to prove now that the probability that there is such a pair (x_i,y_i) in the randomly chosen input vectors $(\overline{x},\overline{y})$, is very high. That is:

$$Pr_{(\overline{x},\overline{y})}[\exists (x_i,\ y_i) \in S_{hard}] \geq 1 - \frac{1}{2^n} \tag{7}$$

Let us see why. The above probability corresponds exactly to:

$$Pr_{(\overline{x},\overline{y})}[\nexists(x_i,\ y_i) \in S_{hard}] <\ \frac{1}{2^n} \tag{8}$$

It suffices to prove the correctness of the second inequality instead of the first one:

$$\begin{aligned} &Pr_{(\overline{x},\overline{y})}[\nexists(x_i,\ y_i) \in S_{hard}] = Pr_{(\overline{x},\overline{y})}[\forall i,\ Event_i\ is\ false] = \\ &= \prod_{1\leq i\leq m} Pr_{(\overline{x},\overline{y})}[Event_i\ is\ false] \leq \prod_{1\leq i\leq m} (1-\frac{1}{4n^2}) \\ &= (1-\frac{1}{4n^2})^m < [if\ m = 4n^3] < \frac{1}{2^n} \end{aligned} \tag{9}$$

which proves the claim.
Note that the very last inequality is conditioned by a particular value of m. Such a variable is set on that particular value because of the following mathematical law:

$$(1-\frac{1}{x})^y \xrightarrow{x\to\infty} \frac{1}{e^n};\ if\ y = x*n \tag{10}$$

Proof. We want to prove that:

$$if\ f(\overline{x},\overline{y})\ is\ not\ strong\ one{-}way\ function\ \implies\ factoring\ is\ not\ hard.\ (proof\ of\ reduction) \tag{11}$$

if $f(\overline{x},\overline{y})$ is not a strong one-way function then we know by definition that it has to exist an attacker B s.t. $Pr_{\overline{x},\overline{y}}[B\ inverts\ f(\overline{x},\overline{y})] > \frac{1}{p(n)}$.
Let us now build another adversary A which is able to factorize efficiently. That is to say an adversary s.t. $Pr_{p,q\in\Pi_n}[A\ factors\ z\ =\ p*q] > \frac{1}{p(n)}$. If the expression defined on B implies the expression defined on A I would prove the theorem.
Let us build now an algorithm where A and B are bounded together. Particularly we want to build an algorithm where A can factorize numbers by exploiting the capability of B to invert strong one-way functions. Algorithm 1 shows an example of this.
Take a look at line 3. Note that the z value is planted only if there is a couple $(x_j, y_j) \in S_{hard}$. This is done in order to not modify the distribution of the random values generated at step 1. If I modify such a distribution, I could not rely on the properties of a set randomly generated. Now, we want to calculate the probability for A to factorize using the Algorithm 1 just defined:

$$\begin{aligned} &Pr_{p,q\in\Pi_n}[A\ factors\ z\ =\ p*q] \geq Pr_{(\overline{x},\overline{y})}[B\ inverts\ z_1z_2...z_m\ in\ A\ \wedge\ z\ planted\ in\ z_j] \geq \\ &\geq Pr_{(\overline{x},\overline{y})}[B\ inverts\ z_1z_2...z_m\ in\ A] - Pr_{(\overline{x},\overline{y})}[z\ is\ not\ planted] \end{aligned} \tag{12}$$

Since we have assumed that there is an attacker B able to invert strong one-way functions, follows that $Pr_{(\overline{x},\overline{y})}[B\ inverts\ z_1z_2...z_m\ in\ A] \geq \frac{1}{p(n)}$.

input: A receives z; z = p * q, where $p, q \in \Pi_n$
1 Randomly select $(x'_1, y'_1), (x'_2, y'_2), ..., (x'_m, y'_m)$
2 Compute $z'_i = f_{mult}(x'_i, y'_i)$, $\forall i$
3 **if** $\exists (x'_j, y'_j) \in S_{hard}$ **then**
4 $\quad z_j = z$
5 $\quad z_i = z'_i, \forall i \neq j$
6 **else**
7 $\quad z_i = z'_i, \forall i$
8 **end**
9 Send $\prod_{1 \leq i \leq m} z_i$ to B which yields the input vector:
10 $\quad B(\prod_{1 \leq i \leq m} z_i) \rightarrow (x_1, y_1, x_2, y_2, ..., x_m, y_m)$
11 **if** $x_j * y_j == z$ **then**
12 $\quad$ output $x_j,\ y_j$
13 **else**
14 $\quad$ output failure
15 **end**

Algorithm 1: A factorizes exploiting B

Furthermore, because of the distribution of prime numbers in a random set, we have that the probability not to have a couple of prime numbers in set of random couple (formed by components randomly chosen), is really tiny (less than $\frac{1}{2^n}$). Therefore $Pr_{(\overline{x},\overline{y})}[z \ is \ not \ planted] < \frac{1}{2^n}$.
Using these two values follows that:

$$Pr_{p,q \in \Pi_n}[A \ factors \ z \ = \ p * q] \geq \frac{1}{2p(n)} \tag{13}$$

■

3 Relationship between weak and strong one-way functions

Let us now prove the main theorem of this lecture. We state that starting from a weak one-way function is always possible to build a strong one-way function. Let us try to prove this.

Theorem 2 $\exists$ *Weak one-way function* $\implies$ $\exists$ *Strong one-way function*

Before to start to prove this theorem let us provide some definition we will use throughout this section.

By definition of weak one-way function: f(x) is a weak one-way function if:

$$\begin{aligned} &\forall\ adversary\ A,\ \exists\ polynomial\ q: \\ &Pr_x[A\ inverts f(x)] \leq (1 - \frac{1}{q(x)}) \end{aligned} \tag{14}$$

Let us finally define another function g built combining different instances of the weak one-way function just described:

$$g(\overline{x}) = f(x_1)f(x_2)...f(x_m) \tag{15}$$

Proof. We know that $g(\overline{x})$ **is not** a strong one-way function if and only if $\exists$ and adversary B such that $Pr_{\overline{x}}[B\ inverts\ g(\overline{x})] \geq \frac{1}{p(n)}$ where p(n) is a polynomial. We are going to demonstrate that if this is true (i.e., $g(\overline{x})$ is not a strong one-way function) then f(x) is not a weak one-way function.
Again, let us build an attacker A such that $Pr_x[A\ inverts\ f(x)] \geq (1 - \frac{1}{q(n)})$ where q is a polynomial.
As we did in Section 2, we have to bound the inversion carried out by A to the one carried out by B. In other words we have to write an algorithm that allows A to invert a weak one-way functions only relying on the B capability to invert strong one-way functions.
Let us have a look at Algorithm 2.

input: A receives z; z = $f(p)$
1 Randomly select $x'_i \leftarrow \{0,1\}^n,\ \forall i\ 1 \leq i \leq m$
2 Compute $z'_i = f(x'_i), \forall i$
3 Pick up a random index j ($1 \leq j \leq m$) and set $z_j = z$
4 $z_i = z'_i,\ \forall i \neq j$
5 Send $\prod_{1 \leq i \leq m} z_i$ to B which yields the input vector:
6 $B(\prod_{1 \leq i \leq m} z_i) \rightarrow (x_1, y_1, x_2, y_2, ..., x_m, y_m)$
7 **if** $f(x_j) == z$ **then**
8 output x_j
9 **else**
10 output failure
11 **end**

Algorithm 2: A inverts weak one-way functions exploiting B

Note that in step 3 we plant the quantity z in a random place within the random generated vector (from now on called coin vector or merely coins). This is due to the fact that this time there are not any particular properties to care about (differently from the case of Algorithm 1 in Section 2, where we were looking for a value with particular properties). In this case as being z a possible random number in the vector of coins, is placed in a random position. We keep in this way the distribution of the vector of coins.

Looking at the algorithm we can infer that the probability that A inverts the weak one-way function f(x) is exactly the same that B has to invert the strong one-way function $g(\overline{x})$, which is greater than $\frac{1}{p(n)}$, as we supposed at the beginning.

$$Pr_x[A\ inverts\ f(x)] = Pr_{(\overline{x},\overline{y})}[B\ inverts\ Z_1 Z_2 ... Z_m] \geq \frac{1}{p(n)} \tag{16}$$

But this does not involve anything useful. We want to prove that A is able to invert f(x) with much higher probability: only in this case we could prove our thesis.
Looking at the last equality we can say that in order to raise such a probability, we have to raise the probability that B has to invert a strong one-way function. How can we achieve this? We can try to merely repeat the algorithm many more times.
Particularly I can build an algorithm C such as Algorithm 3.

```
1 for i = 1 .... t do
2     u_i ← A(z)
3     if f(u_i) == z then
4         output u_i
5     else
6         output failure
7     end
8 end
```

Algorithm 3: Algorithm A in a loop

It is really important to note that each u_i retrieved in Algorithm 3 at every round, are calculated in Algorithm 2 using fresh and independent random coins. This is really important because implies that each round of the loop in Algorithm 3 is completely independent. It is worthy to note also that the value of z (the one we are trying to invert) is retrieved at the beginning (before to start the Algorithm 3) and of course never changed. For the moment we do not talk about the value that the variable t has to assume, it will be find out later.
The natural question we pose is: does this algorithm really improve the chances an attacker A (or B) has to invert a weak (or strong) one-way function? The response is yes, thanks to the next claim.

Proposition 1 *Repeating helps*

Particularly we claim that:

$$\begin{aligned} &\forall p \leftarrow \{0,1\}^n\ and\ \forall\ polynomial\ \alpha(n) \\ &if\ Pr_{p\in\Pi_n}[A\ inverts\ f(p)] \geq \frac{1}{\alpha(n)} \implies Pr_{p\in\Pi_n}[C\ inverts\ f(p)] > (1 - \frac{1}{2^n}) \end{aligned} \tag{17}$$

Proof. Let us try now to prove this claim. Proving that $Pr_{p\in\Pi_n}[C\ inverts\ f(p)] > (1-\frac{1}{2^n})$ is equivalent to prove that $Pr_{p\in\Pi_n}[C\ fails\ to\ invert\ f(p)] \leq \frac{1}{2^n}$.
Let us study the latter probability:

$$\begin{aligned} &Pr_{p\in\Pi_n}[C\ fails\ to\ invert\ f(p)] = \\ &= Pr_{p\in\Pi_n}[A\ fails\ to\ invert\ f(p)\ at\ the\ i-th\ call,\ \forall 1\leq i\leq t] = \\ &= \prod_{1\leq i\leq t} Pr_{p\in\Pi_n}[A\ fails\ to\ invert\ f(p)\ at\ the\ i-th\ call] = \\ &= (1-\frac{1}{\alpha(n)})^t \leq_{(if t=\alpha(n)n)} \frac{1}{2^n} \end{aligned} \tag{18}$$

Note that the second expression is equal to the third one only because each run of the loop in Algorithm 3 is independent. ■
From the latest claim two things can be inferred:

1. the value t in the Algorithm 3 has to be fixed to a precise value.
2. Looking at the formulation of the last claim, we note that we need particular "good" inputs in order to raise so much the probability that C inverts f(x).

The point two is not trivial. In fact there is no certainty that those inputs actually exist. But, if we were able to prove that they actually exist and that they are in a sufficient quantity, we would prove that C can invert f(x) with really high probability, proving thus (proof of reduction) that the main theorem is true.
Let us then focus on proving what has just been said. For convenience let us call the set of this particular good inputs S_{easy}. There is a lemma which says that:

$$\begin{aligned} &\exists\alpha(n)\ polynomial\ s.t. \\ &S^{\alpha}_{easy} = \{x|Pr_x[A(f(x))] \geq \frac{1}{\alpha(n)}\} \\ &|S^{\alpha}_{easy}| > (1-\frac{1}{2q(n)})2^n \end{aligned} \tag{19}$$

To be convinced that $|S^{\alpha}_{easy}| > (1-\frac{1}{2q(n)})2^n$ would prove our theorem, just think that:

$$\begin{aligned} &Pr_x[C\ inverts\ f(x)] \geq Pr_x[C\ inverts\ f(x)|x\in S^{\alpha}_{easy}]Pr_x[x\in S^{\alpha}_{easy}] \geq \\ &\geq (1-\frac{1}{2^n})(1-\frac{1}{2q(n)}) > (1-\frac{1}{q(n)}) \end{aligned} \tag{20}$$

Proof. Let us fix $\alpha(n) = 2q(n)$.
Supposing that the statement $|S^{\alpha}_{easy}| > (1-\frac{1}{2q(n)})2^n$ is false (that is to say that $|S^{\alpha}_{easy}| \leq (1-\frac{1}{2q(n)})2^n$ is true).
Follows that $|S^{\alpha}_{hard}| > (\frac{1}{2q(n)})2^n$, where $|S^{\alpha}_{hard}| = \{x|Pr_x[A\ inverts\ f(x)] < \frac{1}{\alpha(n)}\}$.

We are going to show that if this is true, then $Pr_{\overline{x}}[B\ inverts\ g(\overline{x})] < \frac{1}{q(n)}$, which would lead to a contradiction given that we started from the initial assumption that $Pr_{\overline{x}}[B\ inverts\ g(\overline{x})] \geq \frac{1}{q(n)}$
Let us analyze the probability B has to invert $g(\overline{x})$:

$$\begin{aligned} Pr^* &= Pr_{\overline{x}}[B\ inverts\ g(\overline{x})] = \\ &= Pr_{\overline{x}}[B\ inverts\ g(\overline{x})|\exists x_i \in S_{hard}^{\alpha(n)}]Pr[x_i \in S_{hard}^{\alpha(n)}]+ \\ &+ Pr_{\overline{x}}[B\ inverts\ g(\overline{x})|\nexists\ x_i \in S_{hard}^{\alpha(n)}]Pr[\nexists x_i \in S_{hard}^{\alpha(n)}] \leq Pr_1 + Pr_2 \\ where: & \\ Pr_1 &= Pr_{\overline{x}}[B\ inverts\ g(\overline{x})|\exists x_i \in S_{hard}^{\alpha(n)}] \\ Pr_2 &= Pr[\nexists x_i \in S_{hard}^{\alpha(n)}] \end{aligned} \tag{21}$$

Let us study Pr_1 and Pr_2:

$$\begin{aligned} P_1 &= Pr_{\overline{x}}[A\ inverts\ f(x)|x \in S_{hard}^{\alpha(n)}] \leq \frac{1}{2q(n)} \\ P_2 &< \prod_{1 \leq i \leq m} Pr[x_i \notin S_{hard}^{\alpha(n)}] \leq (1 - \frac{1}{2q(n)})^m \leq_{m=2q(n)n} \frac{1}{2^n} \end{aligned} \tag{22}$$

Note that the equality about Pr_1 is valid because of the probability built on A (i.e., $Pr_{\overline{x}}[A\ inverts\ f(x)|x \in S_{hard}^{\alpha(n)}]$) has the same distribution of the probability built on B (i.e., $Pr_{\overline{x}}[B\ inverts\ g(\overline{x})|\exists x_i \in S_{hard}^{\alpha(n)}$).
In fact as we can see, such a probability is conditioned by the presence of at least one $x_i \in S_{hard}^{\alpha(n)}$. In order to say that this last probability is equal to the one expressed by Pr_1 in Equation 22 they have to have the same distribution. So, if I say that the initial input x belongs to $S_{hard}^{\alpha(n)}$ I do not change such a distribution.
Moreover, the value $\frac{1}{2q(n)}$ in the first equality in Equation 22 is due to the way we fixed $\alpha(n)$ and the definition of $S_{hard}^{\alpha(n)}$.
Finally:

$$Pr^* \leq Pr_1 + Pr_2 \leq \frac{1}{q(n)} \tag{23}$$

which proves that $|S_{easy}^{\alpha}| > (1 - \frac{1}{2q(n)})2^n$ and therefore, as has been said earlier in this section, the main theorem. ■

4 A little summary

In this lecture we stated and proved an important theorem about one-way functions. It is then worthy to try to summarize briefly the approach and the rationale behind the whole demonstration.

4.1 Summary

Thesis: $\exists$ Weak one-way function $\implies$ $\exists$ Strong one-way function
Approach: proof of reduction
Goal: prove that $Pr[A\ inverts\ weak\ one-way\ functions] > (1 - \frac{1}{p(n)})$
Steps:

- B is able to invert strong one way functions.
- A wants to exploit B's ability, in order to invert weak one-way functions.

  But $Pr[B\ inverts\ strong\ one-way\ function\ in\ A] > \frac{1}{p(n)} \implies$

  $\implies Pr[A\ inverts\ weak\ one-way\ functions] > \frac{1}{p(n)}$. Not enough to prove the theorem, we need a higher probability.
- We exploit the repetition approach and we build a machine C which repeats the algorithm carried out by A.
- We find out that: if the initial input belongs to a particular set S_{easy}, then

  $Pr[C\ inverts\ weak\ one-way\ functions] > (1 - \frac{1}{2^n})$, which would prove our theorem! Therefore, now we need to prove that such a set exists and that it is wide enough.
- We prove that S_{easy} exists and that it is big at least $(1 - \frac{1}{2q(n)})2^n$ which prove that $P[C\ inverts\ weak\ one-way-functions] > (1 - \frac{1}{q(n)})$.

  Proving thus the theorem.