CS276 Lecture 17 Scribe Notes

Scribe: Gautam Altekar

1 One-way Functions

Definition. We say that $f : S \to T$ is a (t, ϵ)-one way function (OWF) if:

1. f is deterministic
2. f is efficiently computable
3. f^{-1} is hard to compute. In other words, for some "negligible" ϵ and "reasonable" t, $\Pr[x \xleftarrow{\$} S; f(A(f(x))) = f(x)] \leq \epsilon$ for $\forall A$ running in time $\leq t$.

Understanding the Definition. Intuitively, a one-way function is a function that is easy to compute but hard to invert. Since our use of one-way functions is in the cryptography setting, we place additional restrictions on f that aren't part of the formal definition. In particular, f shouldn't be a constant function. More generally, there should not be too many x such that $f(x) = y$ for some y.

2 Hardcore Bits for Any One-way Function

Theorem. Let $f : S \to T$ be a (t, ϵ)-OWF. Now define $F(S \times S) \to (T \times S)$ by $F(s, r) = (f(s), r)$ and $B(s, r) = s \cdot r$. Then B is a hardcore bit (hcb) for F.

Understanding the Theorem. First, note that F is also a (t, ϵ)-OWF. To see this, suppose that F is not a (t, ϵ)-OWF. Then there exists an adversary A that can invert F with Adv $A > \epsilon$. Now we can construct an adversary B that uses A to invert f as follows. On input y, B runs A on input $(f(s), r \xleftarrow{\$} S)$ and outputs whatever A outputs. So, we can conclude that Adv $B > \epsilon$, and therefore we have proved the contrapositive. Thus, this theorem is saying that, given a OWF f, we can construct another OWF F that has hardcore bit B.

Proof Idea. To prove this theorem, we prove its contrapositive: if there is an efficient adversary that can predict the hardcore bit B, then there is an efficient adversary that can invert the function f. The key idea is to invoke the procedure from the Goldreich-Levin (GL) theorem (see previous lecture notes) to invert f.

Proof. Suppose B is not a hcb for F. Then by the definition of a hardcore bit, this implies that there is an adversary A that can compute the hcb with probability greater than 1/2. In particular,

$$\Pr[A(f(s), r) = s \cdot r] > \frac{1}{2} + \frac{\epsilon}{2}$$

Using notation from the GL theorem, we let $\alpha(x) = A(f(s), x)$ and equality predicate $EQ(t) = \text{if} f(t) = f(s)$ then 1 else 0. Then, using the procedure from the GL theorem, we can find s given $f(s)$ in $O(n^3/\epsilon^4)$ time.

Theorem. Let F : PK × S $\rightarrow S$ be a TOWP. Now let $F'_{pk}(s, r) = (F_{pk}(s), r)$ also be a TOWP (using a similar argument as in the previous theorem). Then $B(s, r) = s \cdot r$ is a hcb for F'.

Proof Idea. Employ the GL procedure as in the previous proof. But this time, since we are dealing with many functions (one for each public key pk), we must verify that the GL procedure will actually succeed in inverting F for a significant fraction of the public keys.

Proof. Suppose there is an adversary A that predicts hcb B. Then the probability p that he does so is

$$p = \Pr[pk \xleftarrow{\$} \text{PK}; s \xleftarrow{\$} \text{S}; r \xleftarrow{\$} \text{S}; A(pk, F_{pk}(s), r) = B(s, r)] > \frac{1}{2} + \frac{\epsilon}{2}$$

Unlike the OWF in the previous theorem where we are dealing with just one function, here we are dealing with many functions indexed by the public key pk. Therefore, we can think of p as A's average success probability taken over all public-keys.

Now pk and $F_{pk}(s)$, and define $\alpha(x) = A(pk, F_{pk}(s), x)$ and equality predicate $EQ(t) =$ if $F_{pk}(t) = F_{pk}(s)$ then 1 else 0. Then, using the procedure from the GL theorem, we can find s.

Now a problem is that A might do better for some public keys and worse for others. We would like to know the fraction of keys on which the GL procedure will compute s. In particular, we would like to verify that that fraction is non-negligible, which in turn would imply that an adversary has a good chance of finding s. This is key to making our proof by contrapositive work.

To determine this fraction, let's look at p another way. In particular, define

$$q : PK \rightarrow R \text{ by } q(pk) = \Pr[s \xleftarrow{\$} S; r \xleftarrow{\$} S; A(pk, F_{pk}(s), r) = B(s, r)]$$

and

$$p = \frac{1}{|PK|} \sum_{pk \in PK} q(pk).$$

If p is indeed greater than $1/2$ as we are supposing, then at least some of the terms in p's summation are greater than $1/2$. We need to quantify this more precisely, so let's divide the sum into pieces where q is not too big, and pieces where q is big. More precisely, if we define "big" as being greater than $1/2 + \epsilon/4$, then we're mainly interested in the fraction of terms for which $q > 1/2 + \epsilon/4$. This is because the GL procedure will succeed only on public keys that give the adversary a high advantage.

$$\begin{aligned} p &= \frac{1}{|PK|} \sum_{pk \in PK} q(pk) \\ &\leq (1/2 + \epsilon/4) + 1 * \Pr[pk \stackrel{\$}{\leftarrow} PK; q(pk) > 1/2 + \epsilon/4] \end{aligned}$$

We know that p is lower-bounded by $1/2 + \epsilon/2$. Therefore, the right-hand term is at least $\epsilon/4$. Thus, for at least an $\epsilon/4$ fraction of the keys, the GL procedure will compute s. In conclusion, if there is an adversary that predicts the hcb with probability at least $1/2 + \epsilon/2$, then there is some adversary that can invert F.

3 Hardcore Bits and PRGs

We now show how OWPs and hardcore bits can be used to create a PRG that stretches its input by an arbitrary amount. To do so, we need only create a PRG that stretches its input by one bit. From there, we can use a standard construction to generate an arbitrary stretch.

Theorem. If $f : S \to S$ is a OWP with a (t, ϵ)-hcb B, then $G : S \to (S \times \{0, 1\})$ given by $G(s) = (f(s), B(s))$ is a (t, ϵ)-PRG.

Proof. First, note that if s is chosen from a uniform random distribution, then $f(s)$ is uniformly random, since a bijective function (i.e., permutation) on a uniform random distribution maps to a uniform random distribution.

We begin with the definition of a (t, ϵ)-hard core bit, which states that for all A running in time $\leq t$,

$$\Pr[A(f(s)) = B(s)] \leq 1/2 + \epsilon/2.$$

This is equivalent to saying that,

$$\left| \Pr[A'(f(s)), B(s) = 1] - \Pr[b \stackrel{\$}{\leftarrow} \{0, 1\}; A'(f(s), b) = 1] \right| \leq \epsilon.$$

Now observe that $\Pr[A'(f(s), B(s)) = 1] = \Pr[A'(G(s)) = 1]$ and that $\Pr[b \stackrel{\$}{\leftarrow} \{0, 1\}; A'(f(s), b) = 1] = \Pr[A'(y) = 1]$. Thus, we have arrived at the definition of PRG. Thus, G is a PRG.

We can now design a PRG that stretches the input to $l + 1$ bits using the following construction: $(f^l(s), B(f^{l-1}(s)), \ldots, B(f(s)), B(s)))$. This construction is computationally indistinguishable from the uniform distribution, which can be proved using material from past lectures.

4 Hybrid Cryptosystem

Our goal in this section is to derive an encryption scheme that uses both asymmetric and symmetric key cryptography. We will provide motivation for such a scheme first.

Motivation. Public-key encryption and decryption operations are not cheap. Consider, for example, the following encryption scheme using the Rabin trapdoor function (define Rabin trapdoor function as $f(x) = x^2 \mod n$ with hcb $B(x) = lsb(x)$):

$$E_{pk}(m[1 \ldots l]) = (s^{2^l}, lsb(s^{2^{l-1}}) \oplus m[1], \ldots, lsb(s^2) \oplus m[l-1], lsb(s) \oplus m[l])$$

Encryption using this scheme is fast, because modular exponentiation is fairly inexpensive. Decryption, on the other hand, is expensive, since it requires $O(l)$ square-root operations. Computing the square-root takes a lot of time.

Alternatively, consider a similar encryption scheme under RSA (define RSA trapdoor function as $f(x) = x^3 \mod n$, decryption exponent $d = (3^l)^{-1} \mod \varphi(n)$, and hcb $B(x) = lsb(x)$). Observe that the decryption exponent d performs a root operation. However, it need be computed only once, since d can be reused for all subsequent decryptions: $(s^{3^l})^d = s$. Thus, modular exponentiation is performed for both encryption and decryption, making RSA fairly efficient.

Despite the efficiency of RSA, symmetric cryptosystems are still more efficient since they don't require modular exponentiation. Then the question is, can we get the best of both worlds? That is, can we design a new cryptosystem that has asymmetric keys, yet can encrypt and decrypt quickly?

Hybrid Scheme. Define the hybrid encryption scheme in terms of asymmetric and symmetric schemes AE, SE: E = HyE[AE$_{pk}$, SE] and D = HyD[AD$_{tk}$, SD]. The basic idea is to use an asymmetric cryptosystem to distribute a key k and then use a symmetric cryptosystem to encrypt and decrypt data with the shared key k.

More precisely, define the encryption procedure HyE[AE$_{pk}$, SE](m) as

1. $k \xleftarrow{\$}$ SK-keys
2. $y = \mathrm{AE}_{pk}(k)$; $z = \mathrm{SE}_k(m)$
3. return (y, z)

and decryption procedure HyD[AD$_{tk}$, SD](m) as

1. $k = \mathrm{AD}_{tk}(y)$
2. if $k = \perp$ return $\perp$
3. return $\mathrm{SD}_k(z)$

Theorem. If AE is (t, q, ϵ)-IND-CPA and SE is $(t, 1, \epsilon')$-IND-CPA, then $\mathrm{HyE}[\mathrm{AE}_{pk}, \mathrm{SE}]$ is $(t - O(q), q, 2\epsilon + q\epsilon')$-IND-CPA.

Understanding the Theorem. An interesting feature of this theorem is that we start with a weak assumption and yet we get a strong conclusion. In particular, even though the adversary gets only at most 1 query to SE, the adversary of HyE gets at most q queries.

Proof Idea. Our goal is to show that $\mathrm{HyE}[\mathrm{AE}_{pk}, \mathrm{SE}]$ is computationally indistinguishable from $\mathrm{HyE}[\mathrm{AE}_{pk}, \mathrm{SE}] \circ \$$ in the real-or-random sense. To do so, first we note that $\mathrm{HyE}[\mathrm{AE}_{pk}, \mathrm{SE}]$ is $(q\epsilon')$-computationally indistinguishable from $\mathrm{HyE}[\mathrm{AE}_{pk} \circ \$, \mathrm{SE}]$. This can be seen through a straightforward application of the data-processing inequality. Next we note that $\mathrm{HyE}[\mathrm{AE}_{pk} \circ \$, \mathrm{SE}]$ is computationally indistinguishable from $\mathrm{HyE}[\mathrm{AE}_{pk} \circ \$, \mathrm{SE} \circ \$]$. Finally, we note that $\mathrm{HyE}[\mathrm{AE}_{pk} \circ \$, \mathrm{SE} \circ \$] \equiv \mathrm{HyE}[\mathrm{AE}_{pk}, \mathrm{SE}] \circ \$$.