COM S 6830 – Cryptography Oct 1, 2009

Lecture 11: Pseudorandom functions

Instructor: Rafael Pass *Scribe: Stefano Ermon*

1 Recap

Definition 1 *(Gen, Enc, Dec) is a single message secure encryption scheme if for all nuPPT $\mathcal{A}$, there exists a negligible function $\epsilon(\cdot)$ such that $\forall\ n \in \mathbb{N}$ and for all $m, m' \in \{0,1\}^n$, $\mathcal{A}$ distinguishes*

$$\{k \leftarrow Gen(1^n) : Enc_k(m)\}$$

$$\{k \leftarrow Gen(1^n) : Enc_k(m')\}$$

with probability $\leq \epsilon(n)$

This definition of security is similar to the Shannon's one, except that here the ensembles of probability distributions need to be indistinguishable instead of identical.
We proved that the encryption scheme $Enc_k(m) = m \oplus G(k)$ is secure if G is a double lengthening PRG, but is it secure if the same key is used to encrypt many messages?

2 Multi message security

Definition 2 (Multi-message secure encryption) *(Gen, Enc, Dec) is a multi-message secure encryption scheme if for all nuPPT $\mathcal{A}$, for all polynomial $q(\cdot)$ there exists a negligible function $\epsilon(\cdot)$ such that $\forall\ n \in \mathbb{N}$ and for all pairs of sequences of messages $m_0, m_1, \ldots, m_{q(n)}$, $m'_0, m'_1, \ldots, m'_{q(n)} \in \{0,1\}^n$, $\mathcal{A}$ distinguishes*

$$\{k \leftarrow Gen(1^n) : Enc_k(m_0), \ldots, Enc_k(m_{q(n)})\}$$

$$\{k \leftarrow Gen(1^n) : Enc_k(m'_0), \ldots, Enc_k(m'_{q(n)})\}$$

with probability at most $\epsilon(n)$.

According to this definition the encryption scheme $Enc_k(m) = m \oplus G(k)$ introduced before is not multi-message secure, and more generally:

Theorem 1 *There is no deterministic stateless multi-message secure encryption scheme.*

Proof. Consider two messages m_0, m_1, with $m_0 \neq m_1$ and the sequences $m_0 m_0$ and m_0, m_1. Since the scheme is stateless and deterministic the encryption of the first sequence is $Enc_k(m_0), Enc_k(m_0)$. The second one encrypts to $Enc_k(m_0), Enc_k(m_1)$, where $Enc_k(m_0) \neq Enc_k(m_1)$, so that the sequences can be trivially distinguished with high probability in polynomial time.

2.1 Stateful and deterministic scheme

If we allow an encryption scheme to be stateful, it is easy to build a multi-message secure scheme. In fact given a key of fixed length it is possible to generate an arbitrarily long string of pseudorandom bits with a PRG, and then XOR each message in the sequence with a portion of this larger key. In this case state is used to keep track of how many bits have been already used. The problem of this approach is that Alice and Bob need to be synchronized, so that they always know which portion of the larger key has been used to encrypt a certain message.

2.2 Stateless and non deterministic scheme

One possible idea to build a stateless and randomized scheme is to generate a long pseudorandom string of bits from a key k with a PRG G, then pick an index i at random and let

$$Enc_k(m) = i || m \oplus G(k)[i]$$

where $G(k)[i]$ represents the i-th block of the string generated with the PRG. The problem with this approach is that PRGs can expand only polynomially, so that i would be $O(\log n)$ and the same index would be chosen more than once with reasonably high probability, so that the scheme would not be multi-message secure.
The idea to solve this problem is to introduce a *pseudorandom function* that allows us to index exponentially many bits in polynomial time, so that i can be of order n. Intuitively this object should have a short description, but should be able to emulate an exponentially long string of random bits.

3 Pseudorandom functions

Definition 3 *A random function $F : \{0,1\}^n \to \{0,1\}^n$ is a map that associates at each $x \in \{0,1\}^n$ a random string $y = F(x) \in \{0,1\}^n$.*

This object can be completely described by an array of 2^n entries that stores the image of each possible input through F. Since each entry is n bits long, $n2^n$ bits are needed to store the entire table, and for any n there are 2^{n2^n} possible functions of this type.
A random function can be also interpreted in an algorithmic view, as a machine that works as follows. Given an input x, if it has not been seen before, the machine outputs $y \leftarrow \{0,1\}^n$ and stores the pair $(x, y = F(x))$ in a table. If x has been seen before, then it outputs the pair $(x, F(x))$ stored in the table. It is easy to see that a polynomial number of queries to the machine can be answered in polynomial time.

3.1 Pseudorandom functions

Intuitively we would like a pseudorandom function (PRF) to look like a random function to any nuPPT adversary, even if the PRF starts only with small n bit seed. In other

words, we would like a way to compress exponentially ($exp(n)$) many bits into n bits, similarly as we did with PRGs.
To define this concept formally, we will need a new notion of indistinguishability. In fact a computationally bounded adversary would not be able to effectively compare something to a random function, because it has an exponentially long description. For this reason we will consider a new class of adversaries that have *oracle* access to a black box that can be either a PRF or a truly random function, and they are supposed to decide which one they are interacting with.

Definition 4 (Oracle indistinguishability) *Let $\{O_n\}_{n\in\mathbb{N}}$, $\{O'_n\}_{n\in\mathbb{N}}$ be ensembles of probability distributions, where O_n and O'_n are distributions over functions $\{0,1\}^{l_1(n)} \to \{0,1\}^{l_2(n)}$ and l_1 and l_2 are polynomials. We say that $\{O_n\}_{n\in\mathbb{N}}$, $\{O'_n\}_{n\in\mathbb{N}}$ are computationally indistinguishable if for all oracle nuPPT $\mathcal{D}$, there exists a negligible function $\epsilon(\cdot)$ such that $\forall n \in \mathbb{N}$*

$$|Pr[F \leftarrow O_n : \mathcal{D}^F(1^n) = 1] - Pr[F \leftarrow O'_n : \mathcal{D}^F(1^n) = 1]| \leq \epsilon(n)$$

In this definition D^F is an *oracle Turing machine*, that is a Turing machine augmented with a component called an *oracle* that is used to sample F.
It can be proved that the notion of *oracle indistinguishability* satisfies the 3 lemmas previously proved for standard indistinguishability (efficient operations, the Hybrid Lemma, and the Prediction Lemma).
We are now ready to define *pseudorandom functions.* Let RF_n be the distribution that picks one of the 2^{n2^n} functions mapping $\{0,1\}^n \to \{0,1\}^n$ uniformly at random.

Definition 5 (Pseudorandom function) *A family of functions $F = \{f_s : \{0,1\}^{l(|s|)} \to \{0,1\}^{l(|s|)}\}_{s\in\{0,1\}^*}$ is a family of pseudorandom functions if*

- *(Easy to compute): Given $s \in \{0,1\}^n$ and $x \in \{0,1\}^{l(n)}$, $f_s(x)$ can be efficiently computed (in p.p.t time).*
- *(Pseudorandom): $\{s \leftarrow \{0,1\}^n : f_s\}_{n\in\mathbb{N}}$ is computationally indistinguishable from $\{F \leftarrow RF_{l(n)} : F\}_{n\in\mathbb{N}}$*

Notice that to get indistinguishability it is fundamental that the seed s is not revealed to the adversary. Otherwise it would be easy to distinguish them by querying the oracle for any value x and check whether the response is equal to $f_s(x)$.

4 Existence of Pseudorandom functions

We will show that the existence of a pseudorandom generator (PRG) implies the existence of a pseudorandom function (PRF). By using previously proved results we have that $OWP \Rightarrow PRG \Rightarrow PRF$ where OWP stands for the existence of one way permutations.

It is also possible to prove that $OWF \Rightarrow PRG \Rightarrow PRF$, where OWF stands for the existence of one way functions. Moreover it is possible to see that the existence of PRF implies the existence of PRG (a PRG is obtained by calling the PRF a sufficient number of times in order to get expansion).

Theorem 2 *If there exists a pseudorandom generator, then there exists a pseudorandom function.*

Proof. Let without loss of generality $G(x) = G_0(x)||G_1(x)$ be a length doubling PRG, so that $|G_0(x)| = |G_1(x)| = |x|$. We define the candidate pseudorandom function

$$f_s(b_1, b_2, \ldots, b_n) = G_{b_n}(G_{b_{n-1}}(\ldots G_{b_2}(G_{b_1}(s))\ldots))$$

It is easy to see that f keeps only one half of the output of the pseudorandom generator at each of the n calls, so that the recursive calls to G_i can be represented as a tree, where the leafs are the possible final outputs of f.

$$\begin{tikzcd}
 & & s \arrow[dl] \arrow[dr] & & \\
 & s_0 = G_0(s) \arrow[dl] \arrow[d] & & s_1 = G_1(s) \arrow[d] \arrow[dr] & \\
s_{00} = G_0(s_0) & s_{01} = G_1(s_0) & & s_{10} = G_0(s_1) & s_{11} = G_1(s_1)
\end{tikzcd}$$

We need to show that f is a PRF. By contradiction, assume there exists a distinguisher $\mathcal{D}$ and a polynomial $p(\cdot)$ such that $\mathcal{D}$ distinguishes $\{s \leftarrow \{0,1\}^n : f_s\}$ from $\{F \leftarrow RF_n : F\}$ with probability $\geq \frac{1}{p(n)}$ for infinitely many n.
One possible approach here is to use the hybrid lemma, building hybrids by successively replacing each leaf with a truly random distribution. This approach does not work because there are too many (exponentially many) hybrids and therefore the lemma is not useful in this case. Instead we define a family of hybrids HF_n^i, where the i-th hybrid is constructed by picking the first i layers of the tree uniformly at random and then applying the tree construction as before. In this way

$$HF_n^1 = \{s \leftarrow \{0,1\}^n : f_s\} \text{ (only the seed is chosen at random)}$$

$$HF_n^n = RF_n \text{ (all the leaves are chosen at random)}$$

Notice that each hybrid HF_n^i can be efficiently emulated (as we did before for the random function, but keeping a table of the i-th layer of the tree).
By the hybrid lemma there exists i such that $\mathcal{D}$ distinguishes HF_n^i and HF_n^{i+1} with probability $\frac{1}{np(n)}$, since there are n hybrids.

Notice that the difference between HF_n^i and HF_n^{i+1} is that level $i+1$ in HF_n^i is pseudorandom (each block is distributed as $G(U_n)$), while in HF_n^{i+1} level $i+1$ is truly random. Since the size of the layers grows exponentially, it gets difficult to effectively distinguish between the two hybrids and to complete the proof we need another set of hybrids.
Since $\mathcal{D}$ runs in polynomial time, there exists a polynomial $q()$ such that the number of queries to the oracle made by $\mathcal{D}$ is bounded by $q(n)$.
We define a new family of hybrids HHF_n^j for $j = 0, \dots, q(n)$, where HHF_n^j answers the first j unique queries consistently with HF_n^i, and the remaining ones consistently with HF_n^{i+1}. Furthermore notice that

$$HHF_n^0 = HF_n^{i+1}$$

$$HHF_n^{q(n)} = HF_n^i$$

By using the hybrid lemma, there exists j such that $\mathcal{D}$ can distinguish HHF_n^j and HHF_n^{j+1} with probability $\frac{1}{nq(n)p(n)}$.
The only difference between HHF_n^j and HHF_n^{j+1} is that HHF_n^{j+1} answers its $(j+1)$-th query using the output of a pseudorandom generator on a randomly chosen value, while HHF_n^j answers its $(j+1)$-th query starting with a randomly chosen value.
As we noted before , queries to HHF_n^j and HHF_n^{j+1} can be efficiently emulated in probabilistic polynomial time. Then it follows by the closure under efficient operations lemma and the pseudorandomness of G that $\mathcal{D}$ cannot distinguish them.