Lecture 13

In this lecture, we start with a construction of a pseudorandom generator using a hard core predicate for a one-way function. We then define the concept of a pseudorandom function families and give a construction of one. Finally we show how these concepts could be used to build a secure private key cryptosystem.

1 A Pseudorandom generator

In the last lecture, we showed that if $f : \{0,1\}^n \to \{0,1\}^n$ is a one-way permutation and $B : \{0,1\}^n \to \{0,1\}$ is a hard core predicate for f, then $G : \{0,1\}^n \to \{0,1\}^{n+1}$ defined by

$$G(x) = f(x)B(x)$$

is a pseudorandom generator. We now extend that construction to get a pseudorandom generator $G^k : \{0,1\}^n \to \{0,1\}^{n+k}$ (see fig. 1), defined as

$$G^k(x) = B(x)B(f(x)) \dots B(f^{k-1}(x)).f^k(x)$$

Theorem 1
G_k defined as above is a pseudorandom generator.

Proof: We use proof by contradiction. Assume the contrary, i.e. suppose there is an algorithm D' such that

$$\left| \mathbf{Pr}_r[D'(r) = 1] - \mathbf{Pr}_x[D'(G^k(x)) = 1] \right| > \epsilon \tag{1}$$

then, as shown in the last lecture, there is a D of similar complexity such that

$$\mathbf{Pr}_r[D(r) = 1] - \mathbf{Pr}_x[D(G^k(x)) = 1] > \epsilon \tag{2}$$

Now we use a technique called the hybrid argument, originally due to Goldwasser, Micali and first used in the context of pseudorandom generators by Blum and Micali. We define a series of distributions $H_0, H_1 \dots, H_k$ such that H_i and H_{i+1} are close and the first and the last are the two distribtions distinguished by D above.

Pick $x \in \{0,1\}^n, (r_1, \dots, r_n) \in \{0,1\}^k$

$$\begin{aligned}
H_0 &= B(x)B(f(x)) \dots B(f^{k-1}(x)).f^k(x) && \approx \text{output of} G^k \\
H_1 &= r_1 B(f(x)) \dots B(f^{k-1}(x)).f^k(x) \\
H_2 &= r_1 r_2 \dots B(f^{k-1}(x)).f^k(x) \\
&\vdots \\
H_k &= r_1 r_2 \dots r_k.f^k(x) && \approx \text{uniform distribution}
\end{aligned}$$

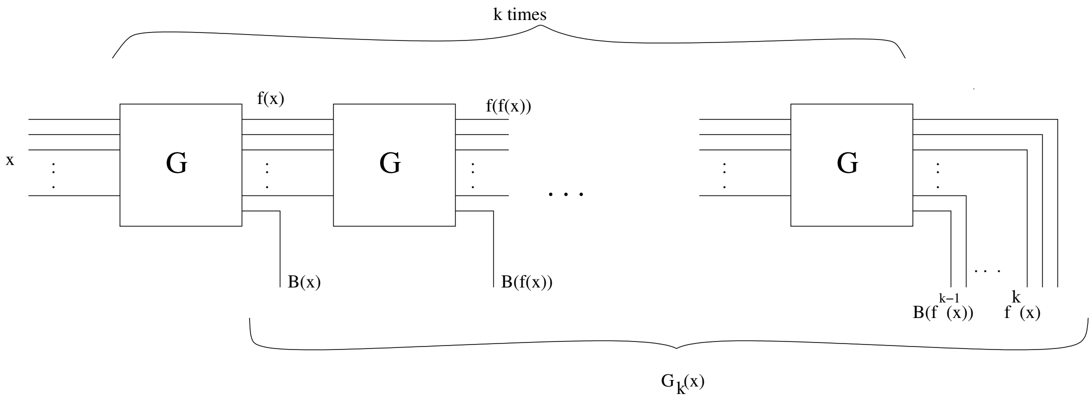

Figure 1: Pseudorandom generator

Using (2) and the definition of H_i's above,

$$\mathbf{Pr}[D(H_0) = 1] - \mathbf{Pr}[D(H_k) = 1] > \epsilon \tag{3}$$

If we define $p_i = \mathbf{Pr}[D(H_i) = 1]$, then (3) gives

$$\begin{aligned} \epsilon &= p_0 - p_k & (4) \\ &= (p_0 - p_1) + (p_1 - p_2) + \ldots + (p_{k-1} - p_k) & (5) \end{aligned}$$

Thus, there exists an i such that

$$\mathbf{Pr}[D(H_{i-1}) = 1] - \mathbf{Pr}[D(H_i) = 1] > \frac{\epsilon}{k} \tag{6}$$

Now recollect that

$$\begin{aligned} H_{i-1} &= r_1 r_2 \ldots r_{i-1} B(f^{i-1}(x)) B(f^i(x)) \ldots B(f^{k-1}x)).f^k(x) & (7) \\ H_i &= r_1 r_2 \ldots r_{i-1} r_i B(f^i(x)) \ldots B(f^{k-1}x)).f^k(x) & (8) \end{aligned}$$

Using D, we now construct an algorithm A that distinguishes $G(x)$ from uniform.

input: $y.b // y = f(z)$for a random $z, b \in \begin{cases} \text{a random bit} \\ B(z) \end{cases}$

pick at random $r_1, \ldots, r_{i-1}$.
output $D(r_1 \ldots r_{i-1} b B(y) B(f(y)) \ldots B(f^{k-i-1}(y)) f^{k-i}(y))$

To show that A distinguishes $f(z)B(z)$ from uniform, we show that

- If A recieves $f(z)B(z)$ as input, then A feeds H_{i-1} into D.
- If A recieves a random input, then A feeds H_i into D.

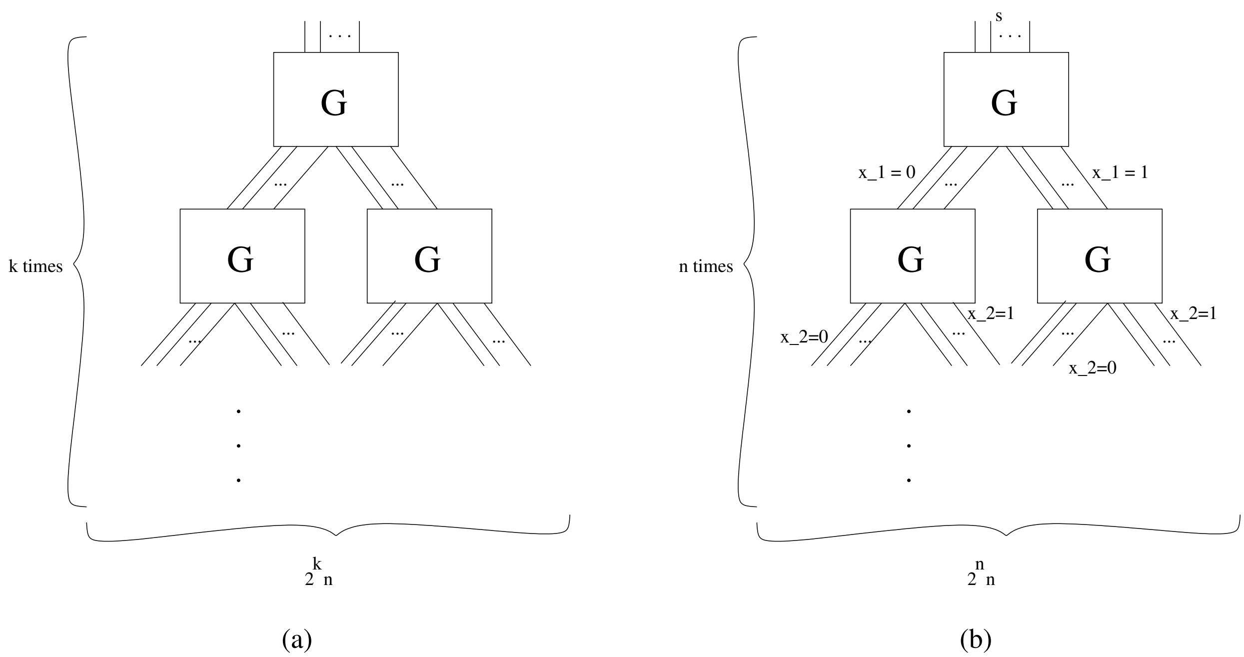

Figure 2: (a) Parallel composition of pseudorandom generators. (b) Pseudorandom function family

Note that input to D in the first case is

$$r_1 \ldots r_{i-1} B(z) B(f(z)) B(f^2(z)) \ldots B(f^{k-i}(z)) f^{k-i+1}(z)$$

Since $f^{i-1}(x)$ has the same distribution as z when x and z are random and f is a permutation, the first part of the claim holds. The input to D in the second case is

$$r_1 \ldots r_{i-1} b B(y) B(f(y)) \ldots B(f^{k-i-1}(y)) f^{k-i}(y)$$

Again since $f^i(x)$ has the same distribution as y when x and y are random and f is a permutation, and since b is random, the claim holds. Thus if we have a D that ϵ-distinguishes $G^k(x)$ from uniform, then we have an A that $\frac{\epsilon}{k}$-distinguishes $f(z)B(z)$ from uniform. However since B was a hard core predicate, there is no such A. Thus the theorem holds. □

2 Pseudorandom function families

The construction described in the last section was a sequential composition of generators. We can also have a parallel composition as in Figure 2(a).

We observe an interesting property of this pseudorandom generators. Let G be computable in time $O(n^c)$. Set $k \geq c \log n$. Then each bit of the output is computable in time $k \cdot O(n^c) = O(n^c \log n)$. The length of the output of the generator is $2^k n = O(n^{c+1})$. Thus

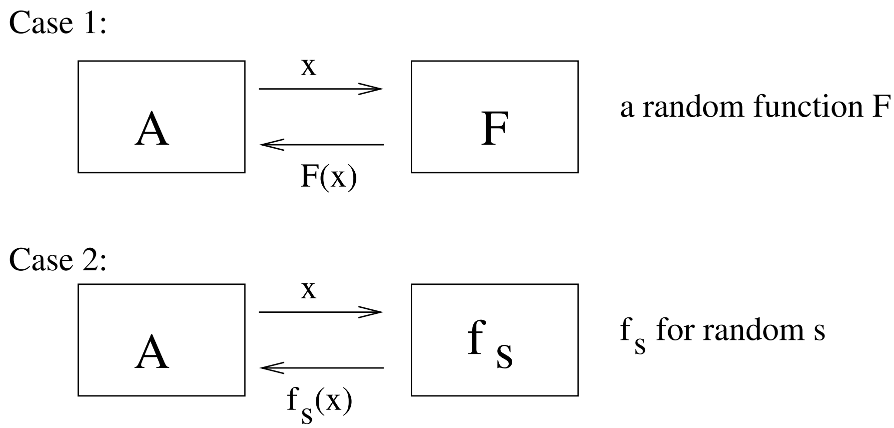

Figure 3: Oracle model

we have a long sequence of pseudorandom bits such that any one of them can be evaluated in time sublinear in the length of the string. Note that this cannot be the case for a sequential composition of pseudorandom generators.

Now consider the case with $k = n$. This construction does not satisfy the traditional definition of a pseudorandom generator since the output size is exponential in the input size and hence it cannot run in polynomial time. We now define the concept of a pseudorandom function family. Let $\mathcal{F} = \{f_s\}_{s \in \{0,1\}^k}$ be a family of functions such that $f_s : \{0,1\}^n \to \{0,1\}^n$ for every s. Let A be any algorithm that queries an oracle on some set of inputs (see figure 3). The oracle can be one of the following. It either computes the function $f_s(x)$ for some s chosen uniformly at random from $\mathcal{F}$, or computes a function $F(x)$ chosen uniformly at random from the set of all functions from $\{0,1\}^n \to \{0,1\}^n$. If no small sized circuit A can differentiate these two cases, then $\mathcal{F}$ is called a pseudorandom function family. More formally,

Definition 1 *$\{f_s\}_{s \in \{0,1\}^k}, f_s : \{0,1\}^n \to \{0,1\}^n$ is called a family of (S, ϵ) pseudorandom functions if for every* oracle circuit *A of size S,*

$$\left| \mathbf{Pr}_{F:\{0,1\}^n \to \{0,1\}^n}[A^F = 1] - \mathbf{Pr}_{s \in \{0,1\}^k}[A^{f_s} = 1] \right| \leq \epsilon$$

Asymptotically,

Definition 2 *Let $\{f_s\}_{s \in \{0,1\}^k}, f_s : \{0,1\}^n \to \{0,1\}^n$, where $k = O(n^c)$ for some c. Further let $f_s(x)$ be computable in time polynomial in n, and let $\{f_s\}$ be an $(S(n), \epsilon(n))$ pseudorandom function family, for super-polynomial functions S and $\frac{1}{\epsilon}$. Then $\{f_s\}_{s \in \{0,1\}^k}$ is a family of pseudorandom functions.*

If $G : \{0,1\}^n \to \{0,1\}^{2n}$ is a pseudorandom generator, then the tree construction gives a pseudorandom function family in the following way. The seed s is the input to the first pseudorandom generator. The input x specifies a path in the tree (See figure 2(b)). More

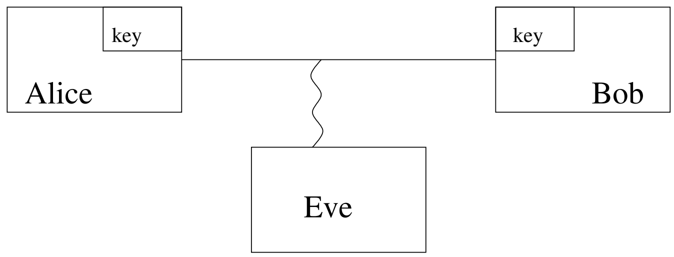

Figure 4: Communication model

formally, define

$$G_0(s) = \text{first } n \text{ bitsof } G(s) \text{ and}$$

$$G_1(s) = \text{bits } n+1, n+2, \dots, 2n \text{ of } G(s)$$

then, $f_s : \{0,1\}^n \to \{0,1\}^n$ for any $s \in \{0,1\}^n$, is defined as

$$f_s(x) = G_{x_n}(\dots G_{x_2}(G_{x_1}(s)) \dots)$$

This construction is due to Goldreich, Goldwasser and Micali.

3 Cryptography

We now apply the concepts of pseudorandomness to private and secure communication. Consider the scenario in figure 4. Alice and Bob want to communicate using an insecure channel. An eavesdropper Eve can listen, possibly corrupt and send malicious messages. We want

Privacy : Eve must get no information about the messages being exchanged.

Integrity : Eve should not be able to modify messages or to impersonate Alice/Bob.

We first consider a simple threat model where Eve can only read the channel, but not modify/write anything, and give a protocol that guarantees privacy. We then consider a stronger threat model and give a protocol that guarantees both privacy an integrity.

Idealized world

We first look at an idealized scenario where Alice and Bob share a completely random function $F : \{0,1\}^n \to \{0,1\}^n$ (see figure 5(a)). Note that both Alice and Bob share this *private key*, which is communicated using some other mechanism which is assumed to be

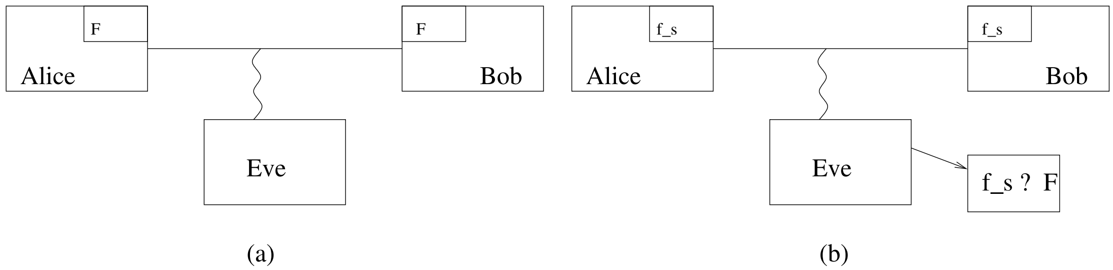

Figure 5: (a) Ideal world model (b) Real world

secure. The protocol for communication is the following

Alice	Bob
message $m \in \{0,1\}^n$	
picks an $r \in \{0,1\}^n$ randomly	
sends $r.(F(r) \oplus m)$	
	recieves $r.z$
	computes $m = F(r) \oplus z$

From the point of view of Eve, what she sees is a random string: r is uniform by choice. Since the function F was chosen uniformly at random, $F(r)$ is uniformly distributed for any arbitrary r, and hence $F(r) \oplus m$ is also uniformly distributed. Thus any two messages with distinct r appear completely random to Eve, and give no information. When an r gets repeated, he can get the xor of the two messages. However, if Alice and Bob exchange polynomially many messages, the probabilty of that happening is exponentially small.

Real world

In a real scenario, the random function F is replaced by a function f_s from a pseudorandom function family, and r and s are pseudorandom strings (figure 5(b)). We claim that Eve cannot distinguish what it sees from uniform, and hence get no information about the message. This must be the case because if Eve could distinguish f_s from F, we can simulate the scenario to construct an algorithm A that distinguishes f_s from F.

Stronger threat model

We now consider a stronger adversary who has access to a decryption procedure. The decryption oracle D returns the decryption of any message that has not been communicated between Alice and Bob (see figure 6). Note that this model captures adaptive chosen ciphertext attacks. In this model, the previous protocol fails immediately, since Eve on reading $r.z$ from the channel, can ask for decryption of $r.z'$ where z' differs from z only in the last bit, and get back m' that is same as m with the last bit flipped.

A simple fix is to send $r.(F(r) \oplus m).F(F(r) \oplus m)$. Now the probability that a message constructed by Eve is properly formatted is exponentially small. Thus with little error, we can simulate the new protocol with one where D always answers "Not well formatted", and this model is equivalent to the one without D. Also note that since a message constructed

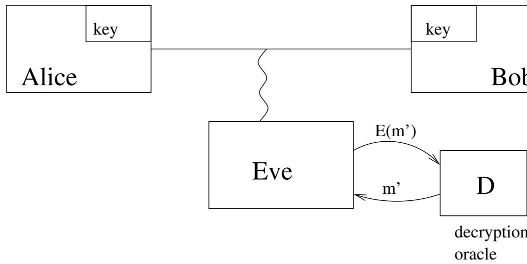

Figure 6: Stronger threat model

by Eve is invalid with very high probability, Eve cannot write valid message on the channel, and hence integrity is also guaranteed.

Thus this construction is secure against adaptive chosen ciphertext attack. Similary, we can define a protocol that is secure against even stronger threat models like chosen ciphertext and chosen plaintext attacks.