CSCI 5440: Cryptography
The Chinese University of Hong Kong

Lecture 4
3 October 2012

1 Construction of pseudorandom functions

We will now show how to construct a pseudorandom function from a pseudorandom generator. Let's start with a pseudorandom function that takes *one* bit of input. In other words, we want a family of functions $F_K\colon \{0,1\} \to \{0,1\}^k$ whose output is indistinguishable from the output of a random function.[1]

In this case the solution is really simple: The pseudorandom function is fully described by the pair of values $(F_K(0), F_K(1))$, and so it is sufficient that this pair be indistinguishable from a truly random pair. But we can interpret the pair $(F_K(0), F_K(1))$ as a pseudorandom string of length $2k$, which suggest the following construction: Take a pseudorandom generator $G\colon \{0,1\}^k \to \{0,1\}^{2k}$ and let $F_K(0) = G_0(K)$, $F_K(1) = G_1(K)$, where G_0 and G_1 denote the first m and last m bits of the output of G respectively.

How about a pseudorandom function on two bits $F_K\colon \{0,1\}^2 \to \{0,1\}^k$? We can certainly do the same trick – take a pseudorandom generator with $4k$ bits of output, which we divide into four blocks $F_K(00), F_K(01), F_K(10), F_K(11)$. But in fact it suffices to use a pseudorandom generator $G\colon \{0,1\}^k \to \{0,1\}^{2k}$ and set:

$$F_K(00) = G_0(G_0(K)) \quad F_K(01) = G_1(G_0(K)) \quad F_K(10) = G_0(G_1(K)) \quad F_K 11 = G_1(G_1(K)).$$

We can view this as a two-level construction. The first input of F_K determines if we take the left or the right part of the output of G. Next, we use this part as a seed and choose the left or the right part as output depending on the value of the second input.

How do we argue that F_K is pseudorandom? We can do it in two stages: First, we replace the inner application of G by a truly random string S_0S_1 of length $2k$ and argue that

$$\begin{array}{rcccc} H = & (G_0(S_0), & G_1(S_0), & G_0(S_1), & G_1(S_1)) \\ \text{is indistinguishable from} & (G_0(G_0(K)), & G_1(G_0(K)), & G_0(G_1(K)), & G_1(G_1(K)). \end{array}$$

But now we have a distribution H that is of the form $(G(S_0), G(S_1))$, where S_0 and S_1 are independent seeds, so by a hybrid argument its output will be indistinguishable from random. (It is a good exercise to complete the missing steps in this proof.)

This suggests the following general construction of a pseudorandom function $F_K\colon \{0,1\}^n \to \{0,1\}^k$ from a pseudorandom generator $G\colon \{0,1\}^k \to \{0,1\}^{2k}$:

$$\boxed{F_K(x_1x_2\dots x_n) = G_{x_n}(G_{x_{n-1}}(\dots G_{x_1}(K)\dots)).}$$

where G_0 and G_1 are the first k and last k bits of the output of G, respectively.

[1] Here we will assume for simplicity that $m = k$; extending the construction to work for larger values of m will be straightforward.

Theorem 1. *If G is a pseudorandom generator against size s and bias ε, then $\{F_K\}$ is an $(\Omega(s/tn), sn\varepsilon)$ pseudorandom function family, where t is the circuit size of G.*

To prove this theorem, it will be convenient to use an alternative characterization of pseudorandom generator: Here, we give the distinguisher oracle access to the output of G.

Lemma 2. *If $G : \{0,1\}^k \to \{0,1\}^{2k}$ is a pseudorandom against size s and bias ε, then for every circuit A of size s:*

$$\left|\Pr[A^{G(R_k)} = 1] - \Pr[A^{R_{2k}} = 1]\right| \leq s\varepsilon,$$

where R_n is an oracle that returns a random string of length n on every invocation.

Proof. Suppose there for some $A^?$ of size at most s (and therefore makes at most s oracle queries)

$$\left|\Pr[A^{G(R_k)} = 1] - \Pr[A^{R_{2k}} = 1]\right| > s\varepsilon.$$

We apply a hybrid argument. Consider the hybrid oracle H_i that answers its first i queries as $G(R_k)$ and the other $s - i$ queries as R_{2k}. Then there must exist some i such that

$$\left|\Pr[A^{H_{i-1}} = 1] - \Pr[A^{H_i} = 1]\right| > \varepsilon.$$

Since the oracle answers are independent, the following circuit B is a distinguisher for G:

> B: On input z, simulate A by answering its first $i - 1$ queries as $G(X_j)$ for a random string $X_j, 1 \leq j \leq i - 1$, its ith query by z, and its last $s - i$ queries as Y_j for a random string $Y_j, i + 1 \leq j \leq s$.

Then $B(G(X))$ is identically distributed with $A^{H_{i-1}}$, while $B(Y)$ is identically distributed with A^{H_i}, and so

$$\left|\Pr[B(G(X)) = 1] - \Pr[B(Y) = 1]\right| \geq \varepsilon.$$

By fixing the optimal choices of X_j and Y_j and hardwiring them into B, we can get a circuit B of size s that performs the distinguishing. □

We can now prove Theorem 1.

Proof of Theorem 1. Suppose that for some A of size $s' = \Omega(s/tn)$,

$$\left|\Pr\left[A^{F_K} = 1\right] - \Pr\left[A^{R} = 1\right]\right| \geq sn\varepsilon.$$

Consider the following family of hybrid functions $H_0, \ldots, H_n$:

> $H_i(x) = G_{x_n}(\cdots G_{x_{i+1}}(R(x_i \ldots x_1)) \cdots)$, where R: $\{0,1\}^i \to \{0,1\}^k$ is a random function.

Notice that H_0 is exactly the distribution F_K, while H_n is a random function from $\{0,1\}^n$ to $\{0,1\}^k$.

By the hybrid argument, there must exist an index i such that

$$\left|\Pr[A^{H_{i-1}} = 1] - \Pr[A^{H_i} = 1]\right| > s\varepsilon.$$

By Lemma 2, to show that G is not pseudorandom it is sufficient to construct a circuit $B^?$ of size s so that

$$\left|\Pr[B^{G(R_k)} = 1] - \Pr[B^{R_{2k}} = 1]\right| > s\varepsilon$$

To do this, notice that the functions H_{i-1} and H_i differ only in what happens at level i. In H_{i-1}, the inputs chosen at this level look like the outputs of G, while in H_i they look random. Intuitively, if we can distinguish between H_{i-1} and H_i, we should be able to distinguish random and pseudorandom strings of length $2k$.

The distinguisher B will do the following:

```
$B^O$: Simulate the circuit $A$. When $A$ makes its $j$th query $x$,
          If this is the first query of $A$ with prefix $x_1 \dots x_{i-1}$,
               Query the oracle $O$ to get a string $z_0 z_1 \in \{0,1\}^{2k}$.
               Answer $A$'s query by $G_{x_n}(\cdots G_{x_{i+1}}(z_{x_i})\cdots)$
               and memorize the pair $(x_1 \dots x_{i-1}, z_0 z_1)$.
          Otherwise,
               Find the previously memorized pair $(x_1 \dots x_{i-1}, z_0 z_1)$.
               Answer $A$'s query by $G_{x_n}(\cdots G_{x_{i+1}}(z_{x_i})\cdots)$.
     Return the output of $A$.
```

As this simulation goes along, B^O dynamically builds a random function $F\colon \{0,1\}^n \to \{0,1\}^k$. By construction, if O is the oracle $G(R_k)$, then F is distributed like H_{i-1}, and if O is the oracle R_{2k}, then F is distributed like H_i. It follows that

$$|\Pr[B^{G(R_k)} = 1] - \Pr[B^{R_{2k}} = 1]| = |\Pr[A^{H_{i-1}} = 1] - \Pr[A^{H_i} = 1]| > s\varepsilon.$$

Notice that the size of B is at most $O(tn)$ times the size of A (every time A calls its oracle, B performs at most n evaluations of G, each of which takes circuit size t), which is at most s by our choice of parameters. By Lemma 2, G is not pseudorandom against size s and bias ε. □

2 The chosen ciphertext attack

Recall that in the chosen plaintext attack, Eve can obtain encryptions of messages of her choice. We saw that (assuming pseudorandom generators exist) an encryption scheme can be secure against such attacks, obtaining arbitrary encryptions gives Eve little help in distinguishing between the encryptions of any two messages. We imagined some scenarios where this type of security may be desirable, say if Eve is an active adversary that may influence Alice in producing various encryptions to her benefit.

It is also possible to imagine scenarios where Eve may have active access to the decryption side of the channel. Suppose every time Alice sends a valid ciphertext C to Bob, Bob replies with an acknowledgment; however if C is some gibberish Bob informs Alice that there was an error. Now say Eve intercepts a valid ciphertext C going from Alice to Bob, but instead of letting C pass through the channel, she sends some modified version C' of the ciphertext, say $C' = C + 1$. Depending on how Bob reacts to this (acknowledgment or error), Eve may be able to deduce information about Alice's message.

A very general model of this kind of adversary is the *chosen ciphertext attack*. Recall that the way we modeled the chosen plaintext attack is by giving Eve access to an encryption oracle. The security requirement (for CPA security) was that even in the presence of such an oracle, Eve cannot distinguish between the encryptions of any two challenge messages M and M'. In a chosen ciphertext attack, Eve can also obtain *decryptions* of almost every message of her choice. Obviously, we cannot allow Eve to obtain decryptions of *every* message, because she could then ask the oracle to decrypt the challenge ciphertext and see if it equals M or M'. However we can do the next best thing: We will allow Eve access to an oracle that decrypts any ciphertext except for the challenge ciphertext.

Definition 3. A private key encryption scheme (Enc, Dec) is (s, ε) *secure against chosen ciphertext attack* (CCA-secure) if for every oracle circuit $A^?$ of size s and every pair of messages M and M',

$$\left|\Pr[A^{Enc(K,\cdot), Dec_{-C}(K,\cdot)}(C) = 1] - \Pr[A^{Enc(K,\cdot), Dec_{-C'}(K,\cdot)}(C') = 1]\right| \leq \varepsilon$$

where $C = Enc(K, M)$, $C' = Enc(K, M')$, and $Dec_{-C^*}(K, \cdot)$ is an oracle that returns $Dec(K, C)$ whenever $C \neq C^*$, and the special symbol `error` otherwise.

To gain some intuition, let us examine the CPA-secure encryption scheme from last lecture:

$$Enc(K, M) = (S, F_K(S) + M) \qquad Dec(K, (S, C)) = (F_K(S) + C),$$

where S is a random string. This scheme is *not* CCA-secure; in fact, we can always use the ciphertext oracle to decrypt the challenge ciphertext. To see this, notice that the if (S, C) is the encryption of M, then $(S, C + 1)$ is the encryption of $M + 1$. While we cannot ask the decryption oracle for the decryption of the challenge ciphertext (S, C) directly, we can trick it into revealing the decryption of $(S, C + 1)$, and by subtracting 1 from the answer, recover the message M.

In fact, this type of attack can be quite problematic. Suppose Alice outputs an encryption (S, C) of a message M. Now suppose Eve captures this ciphertext, replaces it with $(S, C + M + M')$, where M' is a message of her choice, and forwards this to Bob, impersonating Alice. Then Alice would be under the impression that Bob had sent her an encryption of M' instead of an encryption of M.

In the attack we just described, Eve gets hold of a ciphertext $Enc(K, M)$ obtained from some message M and her objective is to produce an encryption of a related message M'. A requirement that would prevent this sort of attack is the following: Whatever meaningful ciphertext $Enc(K, M')$ Eve can produce from $Enc(K, M)$ she should also be able produce without access to $Enc(K, M)$. This is a semantic notion of security, and encryption schemes that satisfy it are called *non-malleable*.[2] Under the proper definition, one can show that if an encryption scheme is CCA secure, then it is non-malleable (and vice versa). We won't give a proof of this. You can look at section 5.4.5 of Goldreich's book for a more detailed discussion.

How do we go about designing a CCA-secure private-key encryption scheme? To begin with, we must at least thwart the type of attack we just described. In this attack, Eve manages to impersonate Alice by producing a forged ciphertext, and so it makes sense to design a mechanism that prevents forgeries of ciphertexts. Let us start with the simpler problem of preventing the forgery of plaintexts, or authentication.

[2] This terminology is inspired by material science: A metal or other material is malleable if it can be hammered or pressed in a way that distorts its shape.

3 Message authentication codes

Informally, a message authentication code (MAC) is a scheme for sending messages (from Alice to Bob) that ensures the *integrity* of messages: After receiving the MAC of a message, Bob is convinced that the message is indeed the one that Alice intended to send him and not some other message. There is no privacy requirement here: the objective of message authentication is not to preserve secrecy but to prevent tampering. Message authentication codes are usually implemented by appending some verification information to the message – a *tag* – that certifies the authenticity and integrity of the message.

We begin by defining the functionality requirement for message authentication codes:

Definition 4. A *message authentication code* (MAC) with key length k, message length m, and tag length t is a pair of (possibly randomized) algorithms (Tag, Ver), where $Tag\colon \{0,1\}^k \times \{0,1\}^m \to \{0,1\}^t$ and $Ver\colon \{0,1\}^k \times \{0,1\}^m \times \{0,1\}^t \to \{0,1\}$ so that for every key $K \in \{0,1\}^k$ and message $M \in \{0,1\}^m$,

$$Ver(K, M, Tag(K, M)) = 1.$$

(The condition should hold with probability 1 if Tag and Ver are randomized.)

We now define security. The security requirement should postulate that upon seeing the tag of a message M, the adversary should not be able to produce a *forgery* $M' \neq M$ together with a tag for M'. The notion of security we will define is quite strong: The adversary can query a *tagging oracle* that can tag messages of its choice, and its task will be to come up with a forged tag of any message that it has not previously queried. While we could consider more restricted notions of security (passive adversaries, single message), we won't bother doing so since we already have the tools to achieve this strong requirement.

Definition 5. A MAC (Tag, Ver) is (s, ε) *secure against chosen message attack* if for every oracle circuit $A^?$ of size at most s,

$$\Pr[A^{Tag(K,\cdot)} \text{ produces a forgery}] \leq \varepsilon$$

where a forgery is a pair (M, T) such that (1) M is different from all the queries A makes to its oracle and (2) $Ver(K, M, T) = 1$. Here, the probability is taken over the choice of K and the randomness of Tag and Ver.

Intuitively, to satisfy this definition, the tag should be some property of the message that Alice and Bob can easily compute, but should appear random to Eve, even if she has observed the tags of other messages. This brings to mind (pseudo)random functions. Indeed, we will obtain a secure MAC by applying a pseudorandom function to the message. Let $F_K\colon \{0,1\}^m \to \{0,1\}^t$ be a family of pseudorandom functions. We consider the following MAC (Tag, Ver):

$$Tag(K, M) = F_K(M) \qquad Ver(K, M, T) = \begin{cases} 1, & \text{if } T = F_K(M) \\ 0, & \text{otherwise.} \end{cases}$$

Claim 6. *If $\{F_K\}$ is an (s, ε) pseudorandom function family, then (Tag, Ver) is $(s/m, \varepsilon + 2^{-t})$ secure against chosen message attack.*

Proof. For contradiction, suppose there exists a circuit A of size s/m such that

$$\Pr[A^{F_K(\cdot)} \text{ produces a forgery}] > \varepsilon + 2^{-t}.$$

As usual, we want to replace the pseudorandom function family with a truly random function $R\colon \{0,1\}^m \to \{0,1\}^t$. So let B^F be a circuit that simulates A^F until A^F produces its potential forgery (M,T), and outputs 1 if (1) M is different from all queries A made to its oracle and (2) $F(M) = T$. Then $B^?$ has size at most s, and

$$\left|\Pr[A^{F_K} \text{ produces a forgery}] - \Pr[A^R \text{ produces a forgery}]\right| = |\Pr[B^{F_K} = 1] - \Pr[B^R = 1]| \leq \varepsilon$$

because F_K is an (s,ε) pseudorandom family. Therefore

$$\Pr[A^{R(\cdot)} \text{ produces a forgery}] > 2^{-t}.$$

However, the probability of A^R producing a forgery (M,T) can be at most 2^{-t}: Conditioned on M being different from all the queries made by A, T is statistically independent of $R(M)$, so $\Pr[T = R(M)] = 2^{-t}$. This contradicts the assumption that (Tag, Ver) is insecure. □