1 Last lecture, and plan for today's lecture

Last lecture, we formally defined what it means for a keyed function/permutation to be a *pseudorandom function/permutation*. A popular cryptographic building block, the AES circuit, is believed to be a secure PRP.

2 Pseudorandom Functions (PRFs)/Pseudorandom Permutations (PRPs): Discussion

Here are a few points related to the PRF/PRP security game that were discussed at the beginning of today's lecture. Let $F : \mathcal{X} \times \mathcal{K} \to \mathcal{Y}$ be a keyed function. Below, we assume the key space, input space and output space are all $\{0,1\}^n$. Strictly speaking, we have a separate key space $\mathcal{K}_n$, input space $\mathcal{X}_n$ and output space $\mathcal{Y}_n$ for each security parameter n. However, for simplicity of notation, we will skip the dependence on n.

1. In the PRF security game, the challenger either chooses a PRF key $k \leftarrow \mathcal{K}$, or a uniformly random function F_1 from $\mathsf{Func}[\mathcal{X}, \mathcal{Y}]$ — the set of all functions mapping $\mathcal{X}$ to $\mathcal{Y}$. The challenger also chooses a bit b. If $b = 0$, it sets function $F_0 = F(\cdot, k)$. After this, the adversary is allowed to make polynomially many adaptive queries. For every query x, it receives $F_b(x)$. Note that if an adversary repeats a query, it gets back the same response. Finally, after polynomially many queries, it sends its guess b'.

2. For a PRP, there are three major differences:

   (a) First, the function $F : \mathcal{X} \times \mathcal{K} \to \mathcal{X}$. Moreover, for every key k, $F(\cdot, k) : \mathcal{X} \to \mathcal{X}$ must be a permutation.

   (b) Second, there exists an inverse function $F^{-1} : \mathcal{X} \times \mathcal{K} \to \mathcal{X}$ such that, for every $x \in \mathcal{X}$ and key $k \in \mathcal{K}$, $F^{-1}(F(x,k),k) = x$.

   (c) Finally, in the security game, if $b = 1$, the challenger chooses a random permutation from the set of all permutations $\mathsf{Perm}[\mathcal{X}]$.

3. There is some similarity between the PRF security game and the PRG security game. Suppose $G : \{0,1\}^n \to \{0,1\}^{2n}$ is a deterministic function. In the PRG security game, the challenger either chooses from a 2^n-sized subset of $\{0,1\}^{2n}$ (this is the subset corresponding to all outputs of G), or it chooses a uniformly random element from the entire 2^{2n}-sized set.

   In the case of PRFs also, we have something similar happening. If $b = 0$, the challenger chooses $k \leftarrow \mathcal{K}$ (which, in turn, fixes a function $F(\cdot, k) \in \mathsf{Func}[\mathcal{X}, \mathcal{Y}]$). In this case, the challenger is choosing a random function from a subset of size $|\mathcal{K}| = 2^n$. If $b = 1$, the challenger chooses a uniformly random function from $\mathsf{Func}[\mathcal{X}, \mathcal{Y}]$, which is a set of size 2^{n2^n}.

4. **Lazy sampling by the challenger:** Note that if the challenger chooses $b = 1$, then it does not need to choose the entire random function. Instead, it can start with an empty table T. For each query x, the challenger checks if there exists an (x, y) pair in T. If so, it outputs y. Otherwise, it samples a uniformly random $y \leftarrow \mathcal{Y}$, sends y to the adversary, and adds (x, y) to T. This is identical to choosing a uniformly random function. Note that maintaining a table is important, so that the challenger sends the same output for the same input query.

Question 10.01 (*). In the PRF/PRP security game, we allow the adversary to make unbounded (but polynomial) number of queries to the challenger. One can similarly modify the PRG security game, where we allow the adversary to make unbounded (but polynomially many) queries to the PRG challenger. At the start of the game, the challenger chooses a random bit $b \leftarrow \{0,1\}$. For each query, the challenger chooses a fresh string s and if $b = 0$, it sends $G(s)$, otherwise it sends a uniformly random string. Show that this seemingly stronger definition is equivalent to the PRG definition we saw in class.

Qn: Why is the adversary allowed to query the PRF on any input of its choice? What if the challenger chose the inputs from some distribution?

Ans: We can consider weaker notions where, for each query, the challenger chooses input x uniformly at random from $\mathcal{X}$, then sends $(x, F_b(x))$. These are also well studied in literature, and are referred to as *weak pseudorandom functions*. Weak PRFs might not suffice for certain applications. For instance, we will see the construction of PRGs from PRFs. You should check that weak PRFs will not suffice for this application. We can also consider a definition where the adversary sends a distribution, and the challenge chooses the inputs from this distribution. This is also weaker than general PRFs.

3 Using PRFs/PRPs for encryption

Last class, we saw a few means of using a PRP to construct an encrytion scheme:

3.1 ECB mode with a counter

Since the ECB mode is insecure, consider the following variant of ECB mode that uses a pseudorandom permutation $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$.

- KeyGen : choose a random $k \leftarrow \{0,1\}^n$.
- $\mathsf{Enc}(m,k)$: Given input message $m \in \{0,1\}^{n \cdot \ell}$, break the message m into 2ℓ chunks $m_1 \,||\, \ldots \,||\, m_{2\ell}$ where each $m_i \in \{0,1\}^{n/2}$. For each $i \in [2\ell]$, compute $\mathsf{ct}_i = F(\mathsf{bin}(i) \,||\, m_i, k)$ where $\mathsf{bin}(i)$ is the binary representation of i using $n/2$ bits. The final ciphertext is $\mathsf{ct} = (\mathsf{ct}_1, \mathsf{ct}_2, \ldots, \mathsf{ct}_{2\ell})$
- $\mathsf{Dec}(\mathsf{ct},k)$: Let $\mathsf{ct} = (\mathsf{ct}_1, \mathsf{ct}_2, \ldots, \mathsf{ct}_{2\ell})$. Decrypt each ciphertext component separately; that is, for each $i \in [2\ell]$, compute $z_i = F^{-1}(\mathsf{ct}_i, k)$. Parse $z_i = (t_i \,||\, m_i)$, and output $m = (m_1 \,||\, \ldots \,||\, m_{2\ell})$.

A few points to note about this construction:

1. the ciphertext's size is double the message's size. This can be optimized by keeping fewer bits for the counter.
2. the decryption uses F^{-1}.
3. the 'ECB mode attack' does not work — each 'block' is now distinct, even if some $m_i = m_j$ (for $i \neq j$).

Question 10.02. Assuming F is a secure PRF, show that the above encryption scheme satisfies No-Query-Semantic-Security.

3.2 Chaining based encryption

Last class, we saw the following proposal for using AES to obtain a secure encryption scheme. Let us recall that construction (using the PRP abstraction).

- $\mathsf{Enc}(m,k)$: Parse $m = (m_1 \,||\, \ldots \,||\, m_\ell)$. Let $\mathsf{ct}_1 = F(m_1, k)$. For each $i > 1$, compute $\mathsf{ct}_i = F(m_i, \mathsf{ct}_{i-1} \oplus k)$.
- $\mathsf{Dec}(\mathsf{ct},k)$: Parse $\mathsf{ct} = (\mathsf{ct}_1 \,||\, \ldots \,||\, \mathsf{ct}_\ell)$. Let $m_1 = F^{-1}(\mathsf{ct}_1, k)$. For each $i > 1$, compute $m_i = F^{-1}(\mathsf{ct}_i, \mathsf{ct}_{i-1} \oplus k)$.

This construction is not immediately broken. However, each component uses a 'related key', and this may be problematic. In particular, it is plausible that this scheme is secure if we use AES, however we can't prove that the scheme is secure by relying on just PRP security.

Qn: Why is it fine to use the same key with related messages, but not safe to use related keys?

Ans: The answer lies in the PRP/PRF security game. Note that the security games allows the adversary to query on any inputs of its choice. In particular, the adversary's queries can be related to each other. However, in the security game, the same key is used for all queries. As a result, we can't be sure of security if different but related keys are used.

However, there is a simple modification that will allow us to prove security (assuming PRP is secure). This is closely related to a popular encryption mode called the *cipher block chaining*.

- KeyGen : choose a random key $k \leftarrow \{0,1\}^n$.
- $\mathsf{Enc}(m,k)$: Parse $m = (m_1 \mid\mid \ldots \mid\mid m_\ell)$. Let $\mathsf{ct}_1 = F(m_1,k)$. For each $i > 1$, compute $\mathsf{ct}_i = F(m_i \oplus \mathsf{ct}_{i-1}, k)$.
- $\mathsf{Dec}(\mathsf{ct},k)$: Parse $\mathsf{ct} = (\mathsf{ct}_1 \mid\mid \ldots \mid\mid \mathsf{ct}_\ell)$. Let $m_1 = F^{-1}(\mathsf{ct}_1,k)$. For each $i > 1$, compute $m_i = F^{-1}(\mathsf{ct}_i,k) \oplus \mathsf{ct}_{i-1}$.

A few points to note about this construction:

1. the decryption uses F^{-1}.

Compared to the previous scheme, proving security for this scheme is slightly more challenging, and you are encouraged to try it to appreciate the subtleties involved.

Question 10.03 (*). Assuming F is a secure PRF, show that the above encryption scheme satisfies No-Query-Semantic-Security.

Question 10.04. Modify the above construction so that in the resulting scheme, the decryption algorithm also uses only F (instead of F^{-1}).

3.3 PRF $\rightarrow$ PRG $\rightarrow$ Secure encryption scheme

The final approach was to use the PRF to first construct a PRG, and then use the PRG to expand the key. After expanding the key, we can use each individual key component separately for encryption. Assuming we have a secure PRG $G : \{0,1\}^n \rightarrow \{0,1\}^{n \cdot \ell}$, we can do the following:

- KeyGen : choose key $k \leftarrow \{0,1\}^n$
- $\mathsf{Enc}(m,k)$: compute $G(k) = (k_1 \mid\mid \ldots \mid\mid k_\ell)$. Parse message $m = (m_1 \mid\mid \ldots \mid\mid m_\ell)$, and compute $\mathsf{ct}_i = F(m_i, k_i)$.
- $\mathsf{Dec}(\mathsf{ct},k)$: compute $G(k) = (k_1 \mid\mid \ldots \mid\mid k_\ell)$. Parse ciphertext $\mathsf{ct} = (\mathsf{ct}_1 \mid\mid \ldots \mid\mid \mathsf{ct}_\ell)$, and compute $m_i = F^{-1}(\mathsf{ct}_i, k_i)$.

The main missing component here is the secure PRG, which we address in the next section. Also, the above construction is inefficient (from a practical perspective). Next lecture, we will discuss how to make this pratically efficient.

4 Secure PRF $\implies$ Secure PRG

We will now see a construction of PRGs from PRFs. Intuitively, this should not be so surprising, given Point 3 in our discussion above. For simplicity, let us assume $F : \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}^n$ is a secure PRF (satisfying Definition 09.02). We want to construct a secure PRG G that maps n bits to $2n$ bits. First, let us see a few interesting attempts that **do not** work.

1. $G(s) = F(s, 0^n) \,||\, F(s, 1^n)$: here, we are using the input $s \in \{0,1\}^n$ as an input for the PRF with two different keys — 0^n and 1^n. Intuitively, this may not be a secure PRG. This is because PRF security is only guaranteed when the key is random. Here, we are using 0^n and 1^n as the keys. There may exist secure PRFs where $F(s, 0^n)$ and $F(s, 1^n)$ are not random at all!

   A secure PRF where the above constructed G is not a secure PRG:
   Let $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ be a secure PRF (as per Definition 09.01). Consider the function $F' : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ defined as follows:

$$F'(x,k) = \begin{cases} F(x,k) \text{ if } k \notin \{0^n, 1^n\} \\ x \text{ if } k \in \{0^n, 1^n\} \end{cases}$$

   Show that F' is a secure PRF, but the resulting PRG (as per construction attempt 1) is not a secure PRG.

   Proof: First, check that $G(s) = s \,||\, s$ (if we follow the construction attempt 1). This is clearly distinguishable from a random $2n$ bit number. However, F' is a secure PRF, assuming F is. This is an easy reduction that is left as an exercise.

2. $G(s) = F\Big(0^{n/2-1} \mid 0 \mid s_{[1,n/2]}, 0^n\Big) \,\Big|\Big|\, F\Big(0^{n/2-1} \mid 1 \mid s_{n/2+1,n}, 0^n\Big)$: this attempt is a bit more involved, but runs into the same issue as attempt 1 — we are using 0^n as the PRF key. There may exist secure PRFs where this construction will not give us a secure PRG.

3. $G(s) = F(s, s) \,||\, F(\overline{s}, s)$: this attempt is interesting, since it is using s as a key for the PRF. However, it is using s as the PRF input also. PRF security guarantee does not say anything about the PRF output *when the input is related to the key*!

(☕) **Challenge Question:** Let F be a secure PRF. Construct a new keyed function F' such that F' is a secure PRF (assuming F is), but if F' is used to construct a PRG using attempt 3, then it results in an insecure PRG.

4.1 A secure construction

Let us now see a construction with provable guarantees.

$$G(s) = F(0^n, s) \,||\, F(1^n, s)$$

Claim 10.01. Suppose there exists a p.p.t. adversary $\mathcal{A}$ that breaks the PRG security of G. Then there exists a reduction algorithm $\mathcal{B}$ that breaks the PRF security of F.

Proof. The reduction algorithm queries the PRF challenger on inputs 0^n and 1^n. It receives two strings y_1 and y_2. It sends $y_1 \,||\, y_2$ to the adversary $\mathcal{A}$. If the PRF challenger used $F(\cdot, k)$ for a random key k, then the adversary $\mathcal{A}$ received PRG output $G(k) = y_1 \,||\, y_2$. If the PRF challenger used a uniformly random function F_1, then both $F_1(0^n)$ and $F_1(1^n)$ uniformly random, independent strings. As a result, the adversary $\mathcal{A}$ receives a uniformly random $2n$ bit string. Complete the proof by showing probability of $\mathcal{B}$ winning the PRF security game is equal to the probability of $\mathcal{A}$ winning PRG game.

□

Question 10.05. Suppose $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ is a secure PRF. Use F to construct a secure PRG $G : \{0,1\}^n \to \{0,1\}^\ell$.

Question 10.06. Check why the reduction fails for attempts 1, 2 and 3.

5 Lecture summary, plan for next lecture, additional resources

Summary In this lecture, we saw how to use a PRF to build a PRG. This also shows us how to use a PRF to build an encryption scheme satisfying No-Query-Semantic-Security.

Next Lecture: Next lecture, we will see a more direct construction of an encryption scheme from PRFs. We will also propose a stronger security definition for encryption schemes that handles all 'read-only' adversaries.

Relevant sections from textbook [Boneh-Shoup]: Section 4.4.4.