1 Last lecture, and plan for today's lecture

Last lecture, we saw how to construct a PRG, given a PRF. This also gives us (in theory) an encryption scheme satisfying No-Query-Semantic-Security. Today, we will see a direct construction that gives a more efficient encryption scheme, and prove security for this construction. Next, we will discuss if the PRF can be replaced with a PRP (this is relevant for practitioners who would like to use AES for encrypting messages). Finally, we will discuss how to build PRFs from PRGs.

2 A practical encryption scheme from PRFs

We saw that PRFs imply the existence of PRGs, which in turn imply the existence of secure encryption schemes. However, this generic transformation is quite inefficient. For instance, suppose we want to encrypt messages of $n \cdot \ell$ bits. Last class, we saw that a PRF $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ can be used to construct a secure PRG $G : \{0,1\}^n \to \{0,1\}^{2n}$. One option is to stretch this PRG output to $n \cdot \ell$ bits via the chaining-based output-space extension idea seen in one of previous lectures. However, we can directly use F to generate $n \cdot \ell$ 'random looking' bits. Using this idea, we can construct an encryption scheme as follows:

- KeyGen: choose $k \leftarrow \{0,1\}^n$.
- $\mathsf{Enc}(m, k)$: Let $|m| = n \cdot \ell$ and parse $m = m_1 \,||\, \ldots \,||\, m_\ell$. For each $i \in [\ell]$, compute $y_i = F(\mathsf{bin}(i), k)$ where $\mathsf{bin}(i)$ is the binary representation of i using n bits. Compute $\mathsf{ct}_i = m_i \oplus y_i$ and output $\mathsf{ct} = (\mathsf{ct}_1 \,||\, \ldots \,||\, \mathsf{ct}_\ell)$.
- $\mathsf{Dec}(\mathsf{ct}, k)$: Parse $\mathsf{ct} = \mathsf{ct}_1 \,||\, \ldots \,||\, \mathsf{ct}_\ell$. For each $i \in [\ell]$, compute $y_i = F(\mathsf{bin}(i), k)$ where $\mathsf{bin}(i)$ is the binary representation of i using n bits. Compute $m_i = \mathsf{ct}_i \oplus y_i$ and output $m = (m_1 \,||\, \ldots \,||\, m_\ell)$.

A few noteworthy points about this scheme:

1. This is called the 'deterministic counter mode' of encryption. We will refer to this construction as the F-DET-CTR encryption scheme (that is, the deterministic counter mode encryption scheme derived from PRF F). The counter mode is one of the popularly used encryption modes.
2. Computing XOR is much cheaper than computing F. Using this scheme, one can pre-compute $y_i = F(\mathsf{bin}(i), k)$ for many values of i, even before seeing the message to be encrypted. Once the message to be encrypted is known, we can simply XOR the y_i values.
3. The decryption does not use F.
4. The scheme is **not two-time secure**. Given two ciphertexts, one can compute the XOR of the underlying messages.

2.1 No-query semantic security of counter mode encryption

Suppose there exists a p.p.t. adversary that breaks the No-Query-Semantic-Security security of F-DET-CTR. Then, using the two-world formulation of No-Query-Semantic-Security, the probability that $\mathcal{A}$ outputs 0 in world-0 is significantly far from the probability that $\mathcal{A}$ outputs 0 in world-1. Let p_0 (resp. p_1) denote the probability of $\mathcal{A}$ outputting 0 in world-0 (resp. world-1). We will create a few intermediate hybrid worlds for this proof. Since this is one of our first hybrid proofs using PRFs, we will be extra-careful and try not to skip any steps. Also, in the notes below, I will underline the part that changes as we go from one hybrid to the next one.

World-0

- $\mathcal{A}$ sends two messages m_0, m_1 s.t. $|m_0| = |m_1| = n \cdot \ell$. Further, let $m_0 = (m_{0,1} \,||\, \ldots \,||\, m_{0,\ell})$ and $m_1 = (m_{1,1} \,||\, \ldots \,||\, m_{1,\ell})$ where each $m_{b,i}$ is an n-bit string.
- Challenger chooses a PRF key $k \leftarrow \mathcal{K}$ and computes $y_i = F(\mathsf{bin}(i), k)$ for each i. It sets $\mathsf{ct}_i = y_i \oplus m_{0,i}$ and sends $\mathsf{ct} = (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell)$.
- Adversary sends b'

Hybrid-1 : This hybrid is similar to the world-0, except that the challenger chooses a uniformly random function $F_1 \leftarrow \mathsf{Func}[\mathcal{X}, \mathcal{X}]$ (recall, $\mathcal{X} = \{0,1\}^n$ and $\mathsf{Func}[\mathcal{X}, \mathcal{X}]$ is the set of all functions from $\mathcal{X}$ to $\mathcal{X}$).

- $\mathcal{A}$ sends two messages m_0, m_1 s.t. $|m_0| = |m_1| = n \cdot \ell$.
- Challenger chooses a random function $F_1 \leftarrow \mathsf{Func}[\mathcal{X}, \mathcal{X}]$ and computes $y_i = F_1(\mathsf{bin}(i))$ for each i. It sets $\mathsf{ct}_i = y_i \oplus m_{0,i}$ and sends $\mathsf{ct} = (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell)$.
- Adversary sends b'

Hybrid-2 : This hybrid is similar to the the previous hybrid, except that the challenger, instead of choosing a uniformly random function $F_1 \leftarrow \mathsf{Func}[\mathcal{X}, \mathcal{X}]$, chooses uniformly random strings y_i for each i.

- $\mathcal{A}$ sends two messages m_0, m_1 s.t. $|m_0| = |m_1| = n \cdot \ell$.
- Challenger chooses uniformly random strings $y_i \leftarrow \mathcal{X}$ for each i. It sets $\mathsf{ct}_i = y_i \oplus m_{0,i}$ and sends $\mathsf{ct} = (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell)$.
- Adversary sends b'

Hybrid-3 : In this hybrid, the challenger chooses uniformly random strings, XORs with m_1 components, and sends as the ciphertext components. This is similar to Hybrid-2, except m_1 is used instead of m_0. This hybrid is perfectly indistinguishable from the previous one, using security of Shannon's OTP.

- $\mathcal{A}$ sends two messages m_0, m_1 s.t. $|m_0| = |m_1| = n \cdot \ell$.
- Challenger chooses uniformly random strings $y_i \leftarrow \mathcal{X}$ for each i. It sets $\mathsf{ct}_i = y_i \oplus m_{1,i}$ and sends $\mathsf{ct} = (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell)$.
- Adversary sends b'

Hybrid-4 : In this hybrid, we will compute the y_i values using a uniformly random function $F_1 \leftarrow \mathsf{Func}[\mathcal{X}, \mathcal{X}]$. This is identical to the previous hybrid (and is analogous to Hybrid-1).

- $\mathcal{A}$ sends two messages m_0, m_1 s.t. $|m_0| = |m_1| = n \cdot \ell$.
- Challenger chooses $F_1 \leftarrow \mathsf{Func}[\mathcal{X}, \mathcal{X}]$, computes $y_i = F_1(\mathsf{bin}(i))$ for each i. It sets $\mathsf{ct}_i = y_i \oplus m_{1,i}$ and sends $\mathsf{ct} = (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell)$.
- Adversary sends b'

World-1

- $\mathcal{A}$ sends two messages m_0, m_1 s.t. $|m_0| = |m_1| = n \cdot \ell$.
- Challenger chooses a key $k \leftarrow \mathcal{K}$ and computes $y_i = F(\mathsf{bin}(i), k)$ for each i. It sets $\mathsf{ct}_i = y_i \oplus m_{1,i}$ and sends $\mathsf{ct} = (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell)$.
- Adversary sends b'

2.1.1 Analysis

Let p_0 (resp. p_1) denote the probability that $\mathcal{A}$ outputs 0 in world-0 (resp. world-1), and let $p_{\text{hyb},j}$ denote the probability that $\mathcal{A}$ outputs 0 in hybrid world-j. We will show the following:

1. $p_0 - p_{\text{hyb},1}$ must be negligibly small, otherwise there exists a p.p.t. algorithm for breaking PRF security.
2. $p_{\text{hyb},1} = p_{\text{hyb},2}$, since the two experiments are identical in the adversary's view. In one case, it receives the outputs of a random function **on distinct inputs**, while in the other case, it receives uniformly random strings. The indistinguishability follows from definition of a random function.
3. $p_{\text{hyb},2} = p_{\text{hyb},3}$: using security of Shannon's OTP.
4. $p_{\text{hyb},3} = p_{\text{hyb},4}$: the reasoning is same as in Point 2 above.
5. $p_{\text{hyb},4} - p_1$ must be negligibly small, otherwise there exists a p.p.t. algorithm for breaking PRF security (same as in Point 1).

Note: This hybrid structure is a bit verbose and involving unnecessary steps. You might be tempted to jump from hybrid-1 to hybrid-5 directly, since the two hybrids are 'obviously' identical. However, I would recommend you to resist such temptations. In particular, just because you've used a random function in both hybrids DOES NOT IMPLY that the two hybrids are identical. Here, we are using the fact that the inputs to the function are distinct.

Claim 11.01. There exists a p.p.t. algorithm $\mathcal{B}$ that wins the PRF game with probability $1/2 + (p_0 - p_{\text{hyb},1})/2$.

Proof. The reduction algorithm interacts with adversary $\mathcal{A}$ breaking the security of F-DET-CTR encryption scheme. It receives m_0, m_1 from $\mathcal{A}$, where $|m_0| = |m_1| = n \cdot \ell$. The reduction algorithm sends ℓ queries to the PRF challenger, where the i^{th} query is $\mathsf{bin}(i)$. It receives ℓ strings in response $\{y_1, \ldots, y_\ell\}$. It sets $\mathsf{ct}_i = y_i \oplus m_{0,i}$ and sends $\mathsf{ct} = (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell)$ to $\mathcal{A}$. Finally, the adversary sends a bit b', which the reduction algorithm forwards to the challenger.

Complete the argument by showing that $\mathcal{B}$ wins the PRF security game with probability $1/2 + (p_0 - p_{\text{hyb},1})/2$. □

Claim 11.02. There exists a p.p.t. algorithm $\mathcal{B}$ that wins the PRF game with probability $1/2 + (p_{\text{hyb},4} - p_1)/2$.

Proof. This reduction is similar to the reduction in proof of Claim 11.01. The only difference is that the reduction will need to flip the adversary's bit (otherwise the probability of reduction algorithm winning PRF game will be $1/2 + (p_1 - p_{\text{hyb},4})/2$). □

As a result, if there exists a p.p.t. adversary $\mathcal{A}$ such that $p_0 - p_1$ is large, then there exists a p.p.t. reduction algorithm that breaks the PRF security of F.

2.2 Post-proof discussion

This is one of the simplest PRF-based encryption schemes. Let us discuss a few modifications of the above scheme, and see if we can prove security.

2.2.1 Replacing PRF with PRP

What happens if we replace the PRF with a PRP (such as AES)? The construction will be exactly identical. However, in the proof, we can't directly go from world-0 to hybrid-1 (and similarly, we can't go from hybrid-5 to world-1). The problem is that PRP security guarantees that the keyed function is indistinguishable from a random permutation (drawn from $\mathsf{Perm}[\mathcal{X}]$). Therefore, we will additionally need to argue that a random

permutation is indistinguishable from a random function. Clearly, if you had the entire truth table, then you can distinguish between a random function and a random permutation. But in our case, the adversary only sees the function's evaluations at a few points (at exactly ℓ locations). As a result, one can show that the following two probability distributions are very close. We will prove this in the next two lectures.

Fact 11.01. No adversary can distinguish between the following two distributions with probability greater than $1/2 + O(\ell^2/2^n)$:

$$\mathcal{D}_1 := \{\text{Choose } \ell \text{ strings } \{r_i\}_{i\in[\ell]} \text{ uniformly at random, independently}\}$$
$$\mathcal{D}_2 := \{\text{Choose } \ell \text{ strings } \{r_i\}_{i\in[\ell]} \text{ uniformly at random, without replacement }\}$$

Using this fact, we can complete our argument as follows: introduce two new hybrid worlds: hybrid 0.5 and hybrid 4.5. In both these worlds, the challenger uses a random permutation. Use PRP security to show that world-0 and hybrid-0.5 are indistinguishable. Use the above fact to argue that hybrid-0.5 and hybrid-1 are indistinguishable. Then use the proof above to conclude that hybrid-1 and hybrid-4 are indistinguishable. Again use the above fact to show that hybrid-4 and hybrid-4.5 are indistinguishable. Finally, use the PRP security to argue that hybrid-4.5 and world-1 are indistinguishable.

2.2.2 Replacing the CTR mode with other modes of encryption

The CTR mode has the simplest proof of security, since we are guaranteed that the inputs to the random function do not repeat. For other modes of encryption, one needs to argue formally that, with high probability, the inputs to the random function do not repeat. Consider the following problems related to two other popular modes of encryption: the output-feedback-mode and the cipher-block-chaining mode. We can define hybrid-1 to hybrid-4 as we did in the proof above, but one needs to be careful when going from hybrid-1 to hybrid-4. **You are strongly advised to attempt these two questions**.

3 PRG $\to$ PRF (not in course syllabus)

Last class, we saw that the existence of a secure PRF implies the existence of a secure PRG. It turns out that the other direction also holds: if PRGs exist, then so do PRFs. This implies that the existence of one-way functions implies the existence of PRFs.

From a theoretical viewpoint, this is great news! We can build PRFs from PRGs, which can in turn be built from one-way functions — the most fundamental building block of cryptography. It is also a bit surprising — given a PRG (which just stretches the input to twice the length), we can get a PRF (which, in some sense, stretches the PRF key to a huge random-looking truth table).

First, let us see a few attempts presented in class, and understand why they do not work. We are given a deterministic function $G : \{0,1\}^n \to \{0,1\}^{2n}$ which is a secure PRG. We want to construct a deterministic function $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ that is a secure PRF.

1. $F(x,k)$ defined as follows: first compute $G(k) = (k_1, k_2, \ldots, k_{2n})$. Depending on x, output a subset of the bits of $G(k)$. This is a good idea, but how to decide the subset of bits, based on the input x? One naive option is to use the last $\log(n)$ bits of x to get a number $i \in [n]$ (that is, $\mathsf{bin}(i)$ = last few bits of x). Then output $(k_i, k_{i+1}, \ldots, k_{n+i-1})$. Unfortunately, this construction is not a secure PRF. You can query the PRF challenger with two inputs x_1 and x_2 such that the last $\log(n)$ bits are the same. In that case, the PRF output will be the same on x_1 and x_2 (irrespective of the key chosen), but the random function will give different output with high probability.

2. $F(x,k) = G(x \oplus k)_{[1,n]}$: In this construction, we compute $G(x \oplus k)$, and then output the first n bits of the output. The problem with this construction is that it assumes that output of G will be random, even if the inputs are *related*. Recall, in Assignment 0, we saw that PRG security does not imply security against related-input attacks. In particular, it is possible that G is a secure PRG, but

$G(s) = G(s \oplus 1^n)$ for every input s. As a result, if this happens, evaluation of PRF at 0^n and 1^n will be identical.

3. $F(x,k) = \Big(G(x) \oplus G(k)\Big)_{[1,n]}$: Compute $G(x)$ and $G(k)$ separately, XOR them, and output the first n bits. This is problematic, because if we consider $F(x,k) \oplus F(x',k)$, this is equal to the first n bits of $G(x) \oplus G(x')$. As a result, here is an adversary that breaks the PRF security for this construction: the adversary queries on 0^n, receives y_1, then queries on 1^n, receives y_2, and checks if $y_1 \oplus y_2 = \Big(G(0^n) \oplus G(1^n)\Big)_{[1,n]}$. If y_1 and y_2 are outputs from $F(\cdot,k)$, then $y_1 \oplus y_2$ will indeed be $\Big(G(0^n) \oplus G(1^n)\Big)_{[1,n]}$. If the challenger chose a random function, then with high probability, $y_1 \oplus y_2$ will not be $\Big(G(0^n) \oplus G(1^n)\Big)_{[1,n]}$.

Hopefully, you see that even coming up with a construction is not so trivial. Here is a hint towards the construction. Given a length-doubling PRG G, we can construct a new PRG G' that maps n bits to $4n$ bits as follows:

$$G'(s) = s_{00} \,||\, s_{01} \,||\, s_{10} \,||\, s_{11} \text{ where}$$
$$G(s) = s_0 \,||\, s_1$$
$$G(s_b) = s_{b0} \,||\, s_{b1} \text{ for each } b \in \{0,1\}$$

Next, extend this to n levels. Do you see a way to define $F(x,s)$ for every $x \in \{0,1\}^n$ following this approach?

4 Lecture summary, plan for next lecture, additional resources

Summary: We saw that given a secure pseudorandom function, we can construct an encryption scheme that satisfies No-Query-Semantic-Security. We also discussed how to adapt the proof if we are given a secure PRP (instead of a secure PRF). Finally, we discussed how to construct PRFs from PRGs.

Next Lecture: We will complete the construction of a PRF, given a PRG (this part is not in the course syllabus). Next, we will discuss the relation between PRPs and PRFs.

Relevant sections from textbook [Boneh-Shoup]: Section 4.4.3 discusses why a PRP is also a secure PRF (Theorem 4.4 of the textbook presents a formal argument for indistinguishability of hybrid-0.5 and hybrid-1 from Section 2.2.1). Section 4.6 contains the PRG $\rightarrow$ PRF transformation.

5 Questions

Question 11.01. The output-feedback-mode (OFB) is another popular encryption mode using a PRF/PRP. Let $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ be a secure PRP. Consider the following encryption scheme F-DET-OFB described below:

- KeyGen : choose a uniformly random key $k \leftarrow \{0,1\}^n$.
- $\mathsf{Enc}(m, k)$: Let $m = m_1 \,||\, \ldots \,||\, m_\ell$ where each $m_i \in \{0,1\}^n$. Let $y_1 = F(0^n, k)$, and for each $i > 1$, $y_i = F(y_{i-1}, k)$. Set $\mathsf{ct}_i = m_i \oplus y_i$ and output $\mathsf{ct} = (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell)$.

1. Describe the decryption algorithm that takes as input $\mathsf{ct} = (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell)$ and a key k.
2. Suppose there exists a p.p.t. adversary $\mathcal{A}$ that breaks the No-Query-Semantic-Security security of F-DET-OFB. Show that there exists a p.p.t. reduction algorithm that breaks the PRF security of F. Describe the hybrids involved in your argument. Show that for each pair of consecutive hybrids, either the probabilities of $\mathcal{A}$ outputting 0 are identical, or argue why they should be close.

Hint: The hybrid structure will be similar to that in Section 2.1. However, one needs to be careful in the transition from Hybrid-1 to Hybrid-2 (and similarly, Hybrid-3 to Hybrid-4). For a random function f, what is the probability that the following sequence S_f contains non-distinct elements:

$$S_f = (y_1, y_2, \ldots, y_\ell), \text{ where } y_0 = 0^n, y_i = f(y_{i-1}).$$

Relate this probability to the difference of p_1 and p_2.