1 Pseudorandom Permutations

In the first part of this lecture we will discuss pseudorandom *permutations*. A function $f\colon \{0,1\}^n \to \{0,1\}^n$ is said to be a permutation if and only if it is a bijection. Note that the PRFs considered in the previous lecture may not be permutations. (Exercise: construct a secure PRF family where the functions are *not* permutations.)

1.1 Definition

Definition 1.1 (Pseudorandom permutation)**.** A family of permutations $\{f_s\colon \{0,1\}^n \to \{0,1\}^n\}$ is a *strong PRP family* (also known as "block cipher") if it is:

- *Efficiently computable*: there is a deterministic polynomial-time algorithm F such that $F(s,x) = f_s(x)$ for all choices of seed s and input $x \in \{0,1\}^n$.
- *Pseudorandom*: For every n.u.p.p.t. $\mathcal{A}$,

  $$\left| \Pr_{f \leftarrow \{f_s\}} \left[\mathcal{A}^{f,f^{-1}}(1^n) = 1\right] - \Pr_{F \leftarrow \mathcal{P}(\{0,1\}^n)} \left[\mathcal{A}^{F,F^{-1}}(1^n) = 1\right] \right| = \text{negl}(n).$$

  Here $\mathcal{P}(\{0,1\}^n)$ is the set of *all* permutations from $\{0,1\}^n$ to $\{0,1\}^n$, and $\mathcal{A}^{f,f^{-1}}$ denotes that the algorithm $\mathcal{A}$ has oracle access to both the function f and its inverse.

  Written more succinctly using oracle indistinguishability,

  $$\left\langle f \leftarrow \{f_s\} : f, f^{-1} \right\rangle \overset{c}{\approx} \left\langle F \leftarrow \mathcal{P}(\{0,1\}^n) : F, F^{-1} \right\rangle.$$

We say that the family is a *weak* PRP family if it is pseudorandom in the forward direction, i.e., if the pseudorandomness condition holds when the adversary is given oracle access to only f (or F), and not their inverses.

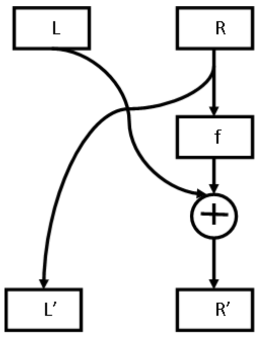
1.2 Feistel Construction

Perhaps surprisingly, a PRP family can be obtained from any PRF family (even though the functions in the PRF family may not themselves be permutations!). To accomplish this, we first show a way to obtain a permutation from *any* (length-preserving) function f; this process is called a *Feistel round*, after its inventor. (One can think of f as being drawn from a PRF family, though the Feistel construction makes no complexity assumption on f.)

Definition 1.2. For a function $f\colon \{0,1\}^n \to \{0,1\}^n$, the Feistel function $\mathcal{D}_f : \{0,1\}^{2n} \to \{0,1\}^{2n}$ is defined as follows:

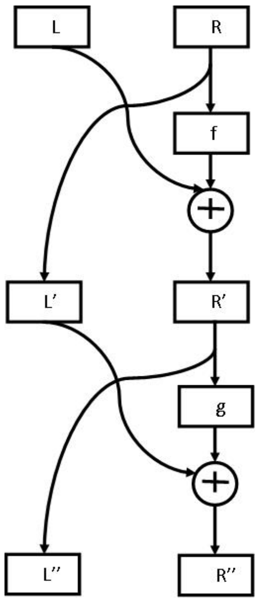
$$D_f(L,R) = (R, L \oplus f(R)).$$

Figure 1.2 gives a pictorial representation of this construction.



Note that for L' and R' as defined above, we have $L = f(L') \oplus R'$ and $R = L'$. Thus the function D_f is invertible, and hence defines a bijection. Formally, we can write $D_f^{-1} = \rho \circ D_f \circ \rho$, where $\rho(L, R) = (R, L)$.

However, D_f by itself does *not* define a PRF. This is because LEFT$(D_f(L, R)) = R$ for any L, R, which is rarely be the case for a truly random permutation. But what happens if we use *two* Fesitel rounds, each with a (possibly different) round function (see Figure 1.2)?



Unfortunately, two Feistel rounds also does not yield a PRP family. Consider arbitrary $L_1, L_2, R \in \{0,1\}^n$ with $L_1 \neq L_2$, and any function $f : \{0,1\}^n \to \{0,1\}^n$. We have $\textsc{Right}(D_f(L_1, R)) = L_1 \oplus f(R)$ and $\textsc{Right}(D_f(L_2, R)) = L_2 \oplus f(R)$. Taking the exclusive-or of both equations, we get

$$\textsc{Right}(D_f(L_1, R)) \oplus \textsc{Right}(D_f(L_2, R)) = L_1 \oplus L_2.$$

On the other hand, for a truly random permutation $\mathcal{P}$, $\mathcal{P}(L_1, R) \oplus \mathcal{P}(L_2, R)$ is the exclusive-or of two distinct random strings and its right half would equal $L_1 \oplus L_2$ with only negligible probability.

Fortunately, all is not lost: we have the following theorem of Luby and Rackoff.

Theorem 1.3 (Luby-Rackoff Theorem)**.** *Three rounds of the Feistel construction, each with a round function drawn independently from a PRF family, yields a* weak *PRP family. Moreover, four rounds yields a* strong *PRP family.*

1.3 Historical Notes

The Data Encryption Standard (DES) from 1977 is a Feistel construction, but with dependent and only "weakly" pseudorandom round functions. The Feistel idea predates any rigorous definitions or analysis. A large fraction of fast block ciphers use a Feistel network, though more recently designers have started using other structures. For example, the Advanced Encryption Standard (AES) is not based on a Feistel network.

2 Security for Shared-Key Encryption

We now take the first steps towards building a secure shared-key (symmetric) encryption system. Before we proceed, recall the model for a shared-key cryptosystem, which is represented by a triple of algorithms $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$, where

- Gen is a randomized algorithm that takes no input (aside from the implicit security parameter 1^n), and outputs a key k in some (finite) set $\mathcal{K}$.
- $\mathsf{Enc}_k(m) = \mathsf{Enc}(k, m)$ takes a key $k \in \mathcal{K}$ and message $m \in \mathcal{M}$, and outputs a ciphertext $c \in \mathcal{C}$.
- $\mathsf{Dec}_k(c) = \mathsf{Dec}(k, c)$ takes a key $k \in \mathcal{K}$ and ciphertext $c \in \mathcal{C}$, and outputs a message $m \in \mathcal{M}$.

Also recall the definition of perfect secrecy.

Definition 2.1. A shared-key encryption scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ with message space $\mathcal{M}$ and ciphertext space $\mathcal{C}$ is *perfectly secret* if for all $m_0, m_1 \in \mathcal{M}$ and all $\bar{c} \in \mathcal{C}$,

$$\Pr_{k \leftarrow Gen}[\mathsf{Enc}_k(m_0) = \bar{c}] = \Pr_{k \leftarrow Gen}[\mathsf{Enc}_k(m_1) = \bar{c}] .$$

That is, the distributions of $\mathsf{Enc}_k(m_0)$ and $\mathsf{Enc}_k(m_1)$ are *identical*, over the choice of the key $k \leftarrow \mathsf{Gen}$.

2.1 A Computational Analogue of Perfect Secrecy

The definition of perfect secrecy suggests a natural analogue for the computational setting.

Definition 2.2 (Single-message indistinguishability)**.** An encryption scheme satisfies *single-message indistinguishability* if for all messages $m_0, m_1 \in \mathcal{M}$,

$$\{k \leftarrow \mathsf{Gen} : \mathsf{Enc}_k(m_0)\} \overset{c}{\approx} \{k \leftarrow \mathsf{Gen} : \mathsf{Enc}_k(m_1)\} .$$

The following lemma follows immediately.

Lemma 2.3. *For a symmetric encryption scheme SKC, if* $\{k \leftarrow \mathsf{Gen} : \mathsf{Enc}_k(m)\}$ *is pseudorandom over* $\mathcal{C}$ *for every* $m \in \mathcal{M}$*, then the scheme is single-message indistinguishable.*

Proof. Fix any two messages $m_0, m_1 \in \mathcal{M}$. Using the definition of pseudorandomness, we have

$$\{\mathsf{Enc}_k(m_0)\} \overset{c}{\approx} \{U(\mathcal{C})\} \overset{c}{\approx} \{\mathsf{Enc}_k(m_1)\} .$$

By the *hybrid lemma* we have $\{\mathsf{Enc}_k(m_0)\} \overset{c}{\approx} \{\mathsf{Enc}_k(m_1)\}$, as desired. □

Now we describe a symmetric encryption scheme, and use Lemma 2.3 to prove its one-message security. Let $\mathcal{K} = \{0,1\}^n$ be the keyspace, and let $\mathcal{M} = \{0,1\}^{\ell(n)}$ be the message space, where $\ell(n)$ is a polynomial in n. Let G be a PRG with output length $\ell(n)$. The following algorithms specify the encryption scheme.

- Gen outputs a uniformly random $k \leftarrow \{0,1\}^n$.
- $\mathsf{Enc}_k(m) = \mathsf{Enc}(k, m)$ outputs $c = m \oplus G(k)$.
- $\mathsf{Dec}_k(c) = \mathsf{Dec}(k, c)$ outputs $m = c \oplus G(k)$.

Claim 2.4. *For any $m \in \{0,1\}^{\ell(n)}$, the distribution of ciphertexts $\{k \leftarrow \mathsf{Gen} : \mathsf{Enc}_k(m)\}$ produced by the above scheme is pseudorandom.*

Proof.

$$\begin{aligned} \{k \leftarrow \mathsf{Gen} : \mathsf{Enc}_k(m)\} &\equiv \{k \leftarrow \{0,1\}^n : m \oplus G(k)\} && (2.1) \\ &\overset{c}{\approx} \left\{x \leftarrow U_{\ell(n)} : m \oplus x\right\} && (2.2) \\ &\equiv \left\{U_{\ell(n)}\right\}. && (2.3) \end{aligned}$$

Equation (2.1) is by definition of the scheme. Equation (2.2) is true because G is a PRG, and by a trivial simulation. Equation (2.3) is by basic probability. □

Note that the scheme constructed above is good for encrypting only *one* message; it would fail completely if we encrypt two different messages using the same key. For example, consider $m_0, m_1 \in \mathcal{M}$ that are encrypted using the above scheme with the same key $k \in \mathcal{K}$. Then we have $\mathsf{Enc}_k(m_0) = m_0 \oplus G(k)$ and $\mathsf{Enc}_k(m_1) = m_1 \oplus G(k)$. Taking the exclusive-or of these two ciphertexts, we get $m_0 \oplus m_1$. In many cases (such as when the two messages are English prose), this is enough information to recover the two messages completely!

A possible way to counter the problem is to encrypt every message with a different key. But this would require the two communicating parties to have a large set of predecided secret keys. Alternatively, we might modify the encryption algorithm to append to its message a fresh key that would be used to encrypt the subsequent message (note that this is possible because the message length is longer than the key length, which is something we could not achieve with perfect secrecy). But this method also suffers from a drawback: the encryption and decryption algorithms are *stateful*. If even a single message fails to reach the receiver, it might cause the entire scheme to become out of sync.

More fundamentally, though, before thinking about ways to securely encrypt multiple messages, we first need a *definition* of security that captures this intended mode of usage.

2.2 Multiple-Message Indistinguishability

Definition 2.5 (Multi-message indistinguishability)**.** An encryption scheme is multi-message indistinguishable if for all $q = \mathrm{poly}(n)$ and all message tuples $(m_0, m_1 \cdots m_q) \in \mathcal{M}^q$ and $\left(m_0', m_1' \cdots m_q'\right) \in \mathcal{M}^q$, we have

$$(\mathsf{Enc}_k(m_0), \mathsf{Enc}_k(m_1) \cdots \mathsf{Enc}_k(m_q)) \overset{c}{\approx} \left(\mathsf{Enc}_k(m_0'), \mathsf{Enc}_k(m_1') \cdots \mathsf{Enc}_k(m_q')\right),$$

where both experiments are over $k \leftarrow \mathsf{Gen}$ and any randomness of Enc.

Remark 2.6. For multi-message indistinguishability, Enc cannot be *deterministic*. Otherwise, it would be trivial to distinguish between the encryptions of a tuple with identical messages and one without, since $\mathsf{Enc}_k(m)$ would always produce the same ciphertext given the same message. Therefore, Enc must either be *randomized* or *stateful*.

Unfortunately, we are still not done. In order to make our definition completely water-tight, we need to capture some additional (realistic) power of the adversary: that it may be able to influence the choice of messages that are encrypted. Specifically, the adversary may have the power to **adaptively** choose these encrypted messages, based on its view of the prior ciphertexts.

2.3 Indistinguishability under (Adaptive) Chosen-Plaintext Attack

The above considerations prompt us to adopt *indistinguishability under (adaptive) chosen-plaintext attack (IND-CPA)* as our notion of security. Before giving a precise definition of security, we list some of the aspects that the definition should capture.

- We allow the adversary $\mathcal{A}$ to *adaptively* query the encryption oracle, based on the ciphertexts it has already seen.
- We still want indistinguishability between the encryptions of two messages (also chosen by $\mathcal{A}$).

Now we present a formal definition satisfying the above goals.

Definition 2.7 (IND-CPA security)**.** An encryption scheme is said to be IND-CPA secure if the following pairs of oracles are computationally indistinguishable:

$$\left\langle k \leftarrow \mathsf{Gen} : C_k^0(\cdot,\cdot)\right\rangle \overset{c}{\approx} \left\langle k \leftarrow \mathsf{Gen} : C_k^1(\cdot,\cdot)\right\rangle .$$

Here $C_k^b(m_0, m_1)$, called the *challenge oracle*, simply runs $\mathsf{Enc}_k(m_b)$ and outputs the result, using fresh randomness (or keeping any state) with each invocation.

Observe that the only difference between the two oracles is in the bit b used by the challenger, i.e., which message is encrypted to produce the challenge ciphertext. Also observe that given access to either $C_k^b(\cdot,\cdot)$, we have implicit access to $\mathsf{Enc}_k(\cdot)$ just by querying C_k^b on two equal messages.

Exercise 2.8. Show that IND-CPA security implies multi-message security, but the converse is false. *Hint:* Define a “pathological” scheme that gives away its secret key via a certain sequence of adaptive queries.

We now describe a PRF-based encryption scheme enjoying IND-CPA security. Define $\mathcal{K} = \mathcal{M} = \{0,1\}^n$ for the encryption scheme, and assume that $\{f_k \colon \{0,1\}^n \to \{0,1\}^n\}_{k\in\{0,1\}^n}$ is a PRF family.

- Gen: output $k \leftarrow \{0,1\}^n$.
- $\mathsf{Enc}_k(m)$: choose $r \leftarrow U_n$ and output $(r, f_k(r) \oplus m)$ as the cipher text.
- $\mathsf{Dec}_k(r,c)$: output $f_k(r) \oplus c$ as the decrypted message.

It is easy to observe that the scheme is correct, i.e. $\mathsf{Dec}_k(\mathsf{Enc}_k(m)) = f_k(r) \oplus f_k(r) \oplus m = m$. In the next lecture we will prove that this scheme is IND-CPA secure.