Notes for Lecture 5 (Draft)

Summary

Having introduced the notion of CPA security in the past lecture, we shall now see constructions that achieve it. Such constructions shall require either *pseudorandom functions* or *pseudorandom permutations*. We shall see later how to construct such objects.

1 Pseudorandom Functions

To achieve CPA security, we shall use very powerful primitives called *pseudorandom functions* and *pseudorandom permutations*. We'll see later how to construct them.

To understand the definition, it's good to think of a pseudorandom function as a pseudorandom generator whose output is *exponentially long*, and such that each bit of the output is efficiently computable given the seed. The security is against efficient adversaries that are allowed to look at at any subset of the exponentially many output bits.

Definition 1 (Pseudorandom Function) *A function $F : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ is a (t, ϵ)-secure pseudorandom function if for every oracle algorithm T that has complexity at most t we have*

$$|\mathbb{P}_{K \in \{0,1\}^k}[T^{F_K}() = 1] - \mathbb{P}_{R:\{0,1\}^n \to \{0,1\}^n}[T^R() = 1]| \leq \epsilon$$

Typical parameters are $k = n = 128$, in which case security as high as $(2^{60}, 2^{-40})$ is conjectured to be possible.

As usual, it is possible to give an asymptotic definition, in which $\epsilon(k)$ is required to be negligible, $t(k), q(k)$ are allowed to be any polynomial, and F is required to be computable in polynomial time.

2 Encryption Using Pseudorandom Functions

Suppose $F : \{0,1\}^k \times \{0,1\}^m \to \{0,1\}^m$ is a pseudorandom permutation. We define the following encryption scheme.

- $Enc(K, M)$: pick a random $r \in \{0,1\}^m$, output $(r, F_K(r) \oplus M)$
- $Dec(K, (C_0, C_1)) := F_K(C_0) \oplus C_1$

This construction achieves CPA security.

Theorem 2 *Suppose F is a (t, ϵ) secure pseudorandom function. Then the above scheme is $(t - O(m), 2\epsilon + t \cdot 2^{-m})$-secure against CPA.*

The proof of Theorem 2 will introduce another key idea that will often reappear in this course: to first pretend that our pseudorandom object is truly random, and perform our analysis accordingly. Then extend the analysis from the pseudorandom case to the truly random case.