**Week 2** **CS6160: Cryptology**

*Computational Security, Pseudorandom Generators, Stream Ciphers*

# 1 Drawbacks of Perfect Security and the One-Time Pad

We have seen one definition of security, namely perfect security. We have also seen an example of a perectly secure cryptosystem, namely the One-Time Pad.

However, in any perfectly secure the key-space must be at least as large as the message space, equivalently, the key must be as long as the message. This is because of the following result of Shannon.

**Theorem 1.** *Let $\Pi = (M, C, K)$ is a cryptosystem that is perfectly secure, then we must have $|K| \geq |M$.*

Instead of a formal proof, we work through an example and suggest how the same idea can be used for a general proof.

**Example:** Let $M = \{00, 01, 10, 11\}, C = M, K = \{01, 10, 11\}$ and let $Enc(m, k) = m \oplus k$. We show that this cryptosystem is not secure by showing two messages that are distinguishable. Let $m_0 = 00, m_1 = 01$. The possible ciphertexts for $m_0$ are $\{01, 10, 11, \}$, while the possible ciphertexts for $m_1$ are $\{00, 10, 11\}$. In the indistinguishablity experiment, an adversary, on observing 00, outputs 1 (indicating that the message is $m_1$) and on observing 01, outputs 0. Otherwise they make a random guess. Then the probability that their guess is correct (over the random choice of key and message) is $\frac{2}{3}$.

It is even easier to show that the distribution of ciphertexts is NOT independent of the message. To prove Shannon's theorem, we will be able to find a triple $m_0, m_1, c$ such that $Enc(m_0, k) \neq c$ for every $k$ while $Enc(m_1, k) = c$ for some key $k$. This means that $Pr[Enc(m_0, k) = c] = 0$ while $Pr[Enc(m_1, k) = c] \neq 0$. You are invited to think about why such a triple must exist if $|K| < |M|$.

Having keys that are as large as messages is impractical, as messages may include files that are several GBs large. Further how can such a large key be distributed securely between the communicating parties in the first place?

The One-Time Pad in particular, also has the following other drawbacks:

- It cannot be used many times. Given two ciphertexts $c_1, c_2$ that are encrypted by the One-Time Pad using the same key, we can find $c_1 \oplus c_2$ which is also $m_1 \oplus m_2$ and obtain partial information about $m_1, m_2$.
- It is completely insecure against Chosen Plaintext Attacks. In such attacks, it is assumed that the adversary has access to some pair $(m, c)$ where $c = Enc(m, k)$, and may even be able to ask for the ciphertext corresponding to a message of their choice. However, knowing $m, c$, the adversary can immediately find $k$.
- It is vulnerable to tampering in a predictable way. That is, if an attacker changes the ciphertext, they know the effect on the plaintext (although they don't know the plaintext itself).

# 2 Computational Security

We will relax the definition of perfect security in two ways: by focusing only on efficient adversaries, and by modifying the requirement in the indistinguishability experiment so that the adversary may be able to guess with a probability slightly larger than $1/2$, but not much larger. The first relaxation is practical because if an inefficient adversary would not be able to break the cryptosystem in reasonable time, eg: if the time needed to gain information is 1000 years, then we can consider the cryptosystem to be practically safe.

We now make these ideas precise.

**Definition 1.** *An algorithm is said to run in probabilistic polynomial time (PPT) if its running time is bounded by some polynomial $p(n)$ (where $n$ is the size of the input), and can make random choices.*

Thus, we consider an adversary to be efficient if they are PPT.

**Definition 2.** *We say that a function $f(n)$ is negligible if for every polynomial $p(n)$, it is the case that $|f(n)| < \frac{1}{p(n)}$ (for all sufficiently large $n$).*

Why is a negligible function have the above definition? Recall the last exercise from Class 3. We can generalize it as follows.

Suppose that an adversary is able to guess some secret information correctly with probability $\varepsilon$. If the adversary makes $\frac{3}{\varepsilon}$ such guesses independently, then the probability that at least one guess is correct is greater than 0.9.

In particular, suppose that a PPT adversary can, in time $p(n)$, guess some information correctly with probability $\frac{1}{f(n)}$, where $p(n), f(n)$ are polynomials. Then the adversary can, in time $3p(n)f(n)$, guess that information correctly with probability 0.9, which is something to be avoided. This is a reason why a negilible probability must be less than $\frac{1}{f(n)}$ for every polynomial function $f(n)$.

# 3 Pseudo Random Generators

Informally, a Pseudo Random Generator(PRG) is an algorithm that generates a random-looking sequence of bits. Usually a PRG takes a short sequence of random bits, called the seed, and produces a longer sequence of bits.

Formally, we have the following definition:

**Definition 3.** *A function $G : \{0,1\}^* \to \{0,1\}^*$ is a PseudoRandomGenerator (PRG) if it can be computed in polynomial time and satisfies the following two properties:*

*(i) If $G$ takes an input of length $n$ and produces an output of length $l(n)$, then $l(n) \geq n+1$. This property ensures that the PRG produces more random bits that it is given as input; otherwise, it could, for example, just produce its input itself as an output.*

*(ii) The output of $G()$ on a random seed from $\{0,1\}^n$ is indistinguishable from a random string in $\{0,1\}^{l(n)}$ in the sense of Definition 4 or 5 of indistinguishability of two probability distributions (see below).*

**Definition 4.** *[Indistinguishability for probability distributions $\pi_1, \pi_2$]*

*Let $\pi_1, \pi_2$ be two probability distributions on $\{0,1\}^m$. Let $s_1, s_2$ be random strings chosen according to $\pi_1, \pi_2$ respectively, and let $b$ be chosen randomly*

*chosen from* $\{1,2\}$. *Given the string* $s_b$, *an adversary* $A$ *outputs a value* $b' \in \{1,2\}$. *The adversary wins if* $b' = b$. *We say that* $\pi_1, \pi_2$ *are indistinguishable, if for every PPT adversary* $A$, *the probability that* $A$ *wins the above experiment is at most* $\frac{1}{2} + negl(n)$, *where* $negl(n)$ *denotes some negligible function.*

**Definition 5.** *[Statistical Indistinguishability for probability distributions* $\pi_1, \pi_2$*]*

*A statistical test is a PPT algorithm that accepts a binary string and ouputs a value in* $\{0,1\}$. *We say that two probability distributions* $\pi_1, \pi_2$ *on* $\{0,1\}^n$ *are statistically indistinguishable if for every statistical test* $T$, *we have:*

$$|Pr_{\pi_1}[T(x) = 1] - Pr_{\pi_2}[T(x) = 1]| = negl(n).$$

An example of a class of PRGs, which however are not indistinguishable, is Linear Feedback Shift Registers (LFSRs). A LFSR uses a recurrence of the form $x_n = \sum_{i \in I} c_i x_{n-i} \bmod 2$ where $I$ is some constant set and $c_i \in \{0,1\}$; the positions of $I$ are referred to as taps. For example, if $I = \{1,4\}$, then recurrence is $x_n = x_{n-1} + x_{n-4} \pmod 2$. Given a seed $s$, for example, $s = 101100$, the recurrence is used to compute the successive bits, in this case, the successive bits are $1, 0, 1, 1, \ldots$. It turns out that given $2n$ successive bits in the output of a LFSR, it is possible to reconstruct the LFSR, and hence predict future bits of the sequence.

# 4 Stream Ciphers

We consider stream ciphers based on PRGs.

Given a PRG $G : \{0,1\}^n \to \{0,1\}^{l(n)}$, we define the stream cipher $\Pi_G = (Gen, Enc, Dec)$ as follows. The message space and cipher space are $M = C = \{0,1\}^{l(n)}$ and the key space is $K = \{0,1\}^n$.

- $Gen()$ takes as input $1^n$ and ouputs a random string $k \in \{0,1\}^n$.
- $Enc(m,k) = m \oplus G(k)$.
- $Dec(c,k) = c \oplus G(k)$.

**Theorem 2.** *Given a PRG* $G : \{0,1\}^n \to \{0,1\}^{l(n)}$ *satisfying Definition 3, the cryptosystem* $\Pi_G$ *is indistinguishable in the presence of an eavesdropper.*

*Proof.* We prove the contrapositive, i.e. suppose that $\Pi_G$ is not secure. Let $A$ be a PPT adversary that, given $1^n$, can produce two messages $m_0, m_1$ of length $l(n)$ and win the indistinguishability game with probability $\frac{1}{2} + f(n)$, where $f(n)$ is non-negligible.

We construct a PPT algorithm $B$ which can distinguish the output of $G$ from a random string in $\{0,1\}^{l(n)}$. Let $k \in \{0,1\}^n$ and $s_1 \in \{0,1\}^{l(n)}$ be chosen at random; let $s_0 = G(k)$. A random string $s \in \{s_0, s_1\}$ is given to $B$ and the goal is to show that $B$ can predict whether $s = s_0$ or $s = s_1$ with probability non-negligibly more than $1/2$.

Given the string $s$, $B$ computes two strings $c_0 = m_0 \oplus s$ and $c_1 = m_1 \oplus s$. Notice that if $s = s_0$, then the two strings $c_0, c_1$ are the ciphertexts corresponding to $m_0, m_1$ whereas if $s = s_1$, then $c_0, c_1$ are two random strings.

The main idea is that in one case (when $s = s_0$), $A$, when given one of $c_0, c_1$ will be able to identify whether it corresponds to $m_0$ or $m_1$ with probability larger than $1/2$. In the other case, when $s = s_1$, every algorithm (and in particular $A$) can only make the identification with probability equal to $1/2$.

Thus, $B$ does the following: pick $c \in \{c_0, c_1\}$ uniformly at random and invoke $A$ with $c$ as input. If $A$ correctly identifies whether $c = c_0$ or $c = c_1$, then $B$ outputs that $s = s_0$, otherwise $B$ outputs that $s = s_1$.

The probability that $B$'s answer is correct is: $\frac{1}{2}\left(\frac{1}{2} + f(n)\right) + \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{2} + \frac{f(n)}{2}$. This completes the proof of Theorem 2

□