YALE UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE

CPSC 467a: Cryptography and Computer Security *Notes 2 (rev. 2)*
Professor M. J. Fischer *September 12, 2006*

Lecture Notes 2

6 Symmetric cryptosystems

A *symmetric cryptosystem* consists of a set $\mathcal{M}$ of *plaintext messages*, a set $\mathcal{C}$ of *ciphertexts*, a set $\mathcal{K}$ of keys, and two functions, an *encryption* function $E : \mathcal{K} \times \mathcal{M} \to \mathcal{C}$ and a *decryption* function $D : \mathcal{K} \times \mathcal{C} \to \mathcal{M}$. We often write $E_k(m) = E(k, m)$ and $D_k(c) = D(k, c)$. One may view a cryptosystem as consisting of a family of encryption and decryption functions indexed by k, but this should not obscure the fact that in order to be useful, it must be possible to compute the functions given an arbitrary key k.

We would like a cryptosystem to have three properties:

Decipherability $\forall m \in \mathcal{M}, \forall k \in \mathcal{K}, D_k(E_k(m)) = m$. In other words, D_k is the (one-sided) inverse of E_k.

Feasibility E and D, regarded as functions of two arguments, should be computable using a feasible amount of time and storage.

Security (weak) It should be difficult to find m given $c = E_k(m)$ without knowing k.

7 Attack scenarios

While these properties are always required of a cryptosystem, we generally want m to remain secret even when Eve has access to various kinds of additional information. This leads us to define several *attack scenarios*.

Ciphertext-only attack This is the situation covered by weak security, where Eve knows only c and tries to recover m.

Known plaintext attack Here we assume that Eve knows a sequence of plaintext-ciphertext pairs $(m_1, c_1), \ldots, (m_r, c_r)$. Now she obtains a new ciphertext $c \notin \{c_1, \ldots, c_r\}$ and wants to recover the corresponding message m.

Chosen plaintext attack This is like a known plaintext attack, except that before getting c, Eve gets to choose messages $m_1, \ldots, m_r$ and somehow get Alice (or Bob) to encrypt them for her and supply her with the corresponding ciphertexts $c_1, \ldots, c_r$.

Chosen ciphertext attack This is like a known plaintext attack, except that before getting c, Eve gets to choose ciphertexts $c_1, \ldots, c_r$ and somehow get Alice (or Bob) to decrypt them for her and supply her with the corresponding plaintext messages $m_1, \ldots, m_r$.

Still stronger versions of these attacks are possible. For example, in a chosen plaintext attack, Eve might be able to choose the messages m_i one at a time rather than all at once so that m_2 depends on the value of c_1, m_3 depends on both c_1 and c_2, and so forth. Such an attack is called *adaptive*. Adaptive chosen ciphertext attacks are of course also possible, where each new ciphertext chosen depends on the plaintexts corresponding to the previously-chosen ciphertexts. One can have mixed chosen plaintext and chosen ciphertext attacks where Eve can choose some plaintexts and some ciphertexts and get the corresponding decryptions or encryptions of same. Adaptive versions of mixed attacks are also possible.

One might question whether such attack scenarios are feasible in practice. Why would Alice cooperate with Eve in this way? It all depends on the situation in which the cryptosystem is used. "Alice" might be an Internet server, not a person, who encrypts/decrypts certain messages received in the course of carrying out more complicated cryptographic protocols. We will see such protocols later in the course. One could also imagine that an attacker gets hold of Alice's encryption device and is able to play with it for awhile even though she can't extract the secret key that is stored inside. For example, Eve might sit down at Alice's computer when Alice is out to lunch, or she might slip Alice's cryptographic smart card out of her purse when Alice isn't looking. Of course, if Alice discovers the intrusion or learns that her card is missing, she'll change her key and Eve's efforts would be for naught, but if Eve puts everything back in order before Alice returns, Alice might be none the wiser.

8 Compromising a cryptosystem

We've already mentioned one kind of compromise of a cryptosystem—learning the secret message m. But actually there are many different degrees of compromise, some more serious than others, depending on the application. Here is a list of some of the kinds of compromise which one should be aware of and ideally be able to prevent:

Complete break Eve finds the key k. Eve now has the same knowledge as both Alice and Bob, so she can impersonate Alice when talking to Bob and impersonate Bob when talking to Alice. She can also passively read all communications between Alice and Bob.

Complete message recovery Eve can recover any message m even without knowing k. She can thus passively read all communications between Alice and Bob, but she can't impersonate one to the other since she is unable to produce encrypted messages herself.

Selected message recovery Eve is only able to recover certain messages from their ciphertexts. Namely, there is a set $M \subseteq \mathcal{M}$ of messages which Eve is able to read; others remain secure. The larger the set M is, the more serious this kind of compromise.

Uncertain message recovery Eve is able to recover m from c only when certain keys $k \in K \subseteq \mathcal{K}$ are in use. The larger the set K is, the more serious this kind of compromise. Because the key is generally chosen randomly and uniformly from $\mathcal{K}$, Eve has probably $|K|/|\mathcal{K}|$ of compromising Alice's messages.

Partial message recovery In each of the above kinds of compromise, Eve either succeeds or not in recovering m. In partial message recovery, she obtains partial information about m. Initially, Eve knows only the probability distribution from which m is drawn. After seeing the ciphertext $c = E_k(m)$, Eve may be able to determine that m lies in a subset $M \subseteq \mathcal{M}$. If $|M| = 1$, then Eve has narrowed down the possibilities for m to a single message, so she has learned m. But if $|M| \geq 2$, she has obtained only partial information about M. Whether or not that partial information is useful to Eve depends on what partial information she receives and of course on the application.

Information-theoretic or total security This is the other extreme in which Eve learns nothing at all about m from seeing the ciphertext c. Whatever she knew about m before seeing c she still knows, but she learns nothing new from seeing c. (Note that if Eve already knew m beforehand, then she would learn nothing new about m from seeing c, so in that special case, all cryptosystems are totally secure.)

9 Probabilistic model

When we start talking about the probability of Even learning the message, we must ask, "Where does randomness enter into the discussion of attacks on a cryptosystem?" We assume there are three such places:

1. The message itself is drawn at random from some probability distribution over the message space $\mathcal{M}$. That distribution is not necessarily uniform or even anything close to uniform[1]. However, whatever the distribution is, we assume that it is part of Eve's *a priori* knowledge.
2. The secret key is chosen uniformly from the key space $\mathcal{K}$.
3. The attacker (Eve) has access to a source of randomness which she may use while attempting to break the system. For example, Eve can choose an element $k' \in \mathcal{K}$ at random. With probability $p = 1/|\mathcal{K}|$, her element k' is actually the correct key k.

We further assume that these three sources of randomness are *statistically independent*. This means that Eve's random numbers do not depend on (nor give any information about) the message or key used by Alice. It also means that Alice's key does not depend on the particular message or vice versa.

These multiple sources of randomness give rise to a *joint probability distribution* that assigns a well-defined probability (i.e., a real number in the interval $[0, 1]$) to each triple (m, k, z), where m is a message, k a key, and z is the result of the random choices Eve makes during her computation. The independence requirement simply says that the probability of the triple (m, k, z) is precisely the product of $p_m \times p_k \times p_z$, where p_m is the probability that m is the chosen message, p_k is the probability that k is the chosen key, and p_z is the probability that z represents the random choices made by Eve during the course of her computation.

When talking about random choices, we often use the term *random variable* to denote the experiment of choosing an element from a set according to a particular probability distribution. For example, we sometimes use m to denote the random variable that describes the experiment of Alice choosing a message $m \in \mathcal{M}$ according to the assumed message distribution. Ambiguously, we often use m to denote a particular element in the set $\mathcal{M}$, perhaps the outcome of the experiment described by the random variable m. While these two uses may sometimes be confusing, we will try to make which usage is intended clear from context.

For example, if we let p_m denote the probability that the message distribution assigns to the particular message $m \in \mathcal{M}$, then there is no experiment under consideration and it is clear that m is being used to denote an element of $\mathcal{M}$. On the other hand, if we talk about the event $m = m_0$, then we are using m in the sense of a random variable, and the event $m = m_0$ means that the outcome of the experiment described by m is m_0. We denote the probability of that event occurring by $\mathrm{prob}[m = m_0]$, which in this case is simply p_{m_0}. See section 15.1 of Trapp and Washington for a more thorough review of basic probability definitions.

[1]The *uniform* distribution over a finite set Ω assigns probability $1/|\Omega|$ to each point $x \in \Omega$. In other words, each element $x \in \Omega$ has equal probability of being chosen.

10 Statistical independence

Information-theoretic security is defined formally in terms of statistical independence. In our probabilistic model, we can consider both m and k to be random variables. Together, they induce a probability distribution on the ciphertext $c = E(k, m)$. Here we see the power of the random variable notation, for c itself can be considered to be a random variable which is a function of k and m. To define c formally, we must define the probability of each of the events $c = c_0$ for $c_0 \in \mathcal{C}$. These are easily derived from the random variables k and m as follows:

$$\text{prob}[c = c_0] = \sum_{k_0, m_0 : c_0 = E(k_0, m_0)} \text{prob}[k = k_0 \wedge m = m_0].$$

Because of the assumed independence of k and m, this is the same as

$$\text{prob}[c = c_0] = \sum_{k_0, m_0 : c_0 = E(k_0, m_0)} \text{prob}[k = k_0] \times \text{prob}[m = m_0].$$

The reason for the summation is that the same ciphertext can arise from many different key-message pairs.

Now, what it means for the cryptosystem to be information-theoretically secure is that the random variables c and m are statistically independent, that is,

$$\text{prob}[m = m_0 \wedge c = c_0] = \text{prob}[m = m_0] \times \text{prob}[c = c_0]$$

for all $m_0 \in \mathcal{M}$ and $c_0 \in \mathcal{C}$.

This independence is often expressed more intuitively in terms of conditional probabilities. We define

$$\text{prob}[m = m_0 \mid c = c_0] = \frac{\text{prob}[m = m_0 \wedge c = c_0]}{\text{prob}[c = c_0]}$$

assuming the denominator of the fraction is non-zero. This is the probability that $m = m_0$ given that $c = c_0$. We then say that c and m are statistically independent if

$$\text{prob}[m = m_0 \mid c = c_0] = \text{prob}[m = m_0]$$

for all $m_0 \in \mathcal{M}$ and $c_0 \in \mathcal{C}$ such that $\text{prob}[c = c_0] \neq 0$. Hence, even after Eve receives the ciphertext c_0, her opinion of the likelihood of each message m_0 is the same as it was initially.

11 Caeser cipher

To make these ideas more concrete, we look at the Caeser cipher, which is said to go back to Roman times. It is an alphabetic cipher, that is, used to encode the 26 letters of the Roman alphabet $A, B, \ldots, Z$. For convenience, we will represent the letters by the numbers $0, 1, \ldots, 25$, where $A = 0$, $B = 1$, …, $Z = 25$.

11.1 Encrypting single letters

The base cipher handles just single letters. The message space is $\mathcal{M} = \{0, \ldots, 25\}$, and the ciphertext space and key space are the same, so $\mathcal{M} = \mathcal{C} = \mathcal{K}$.

To encrypt, E_k replaces each letter m by the letter k places further on in the alphabet when read circularly, with A following Z. To decrypt, D_k replaces each letter c by the letter k places earlier in the alphabet when read circularly. Mathematically, the encryption and decryption functions are given by

$$E_k(m) = (m + k) \bmod 26$$
$$D_k(c) = (m - k) \bmod 26.$$

(Notation: $x \bmod 26$ is the remainder of x divided by 26.) Thus, if $k = 3$, then $E_k(P) = S$ and $D_k(S) = P$.

11.2 Encrypting strings

Of course, one is generally interested in encrypting more than single letter messages, so we need a way to extend the encryption and decryption functions to strings of letters. The simplest way of extending is to encrypt each letter separately using the same key k. If $m_1 \ldots m_r$ is an r-letter message, then its encryption is $c_1 \ldots c_r$, where $c_i = E_k(m_i) = (m_i + k) \bmod 26$ for each $i = 1, \ldots, r$. Decryption works similarly, letter by letter.

What we have really done by the above is to build a new cryptosystem from the base cryptosystem on single letters. In the new system, the message space and ciphertext space are

$$\mathcal{M}^r = \mathcal{C}^r = \underbrace{\mathcal{M} \times \mathcal{M} \times \ldots \mathcal{M}}_{r},$$

that is, length-r sequences of numbers from $\{0, \ldots, 25\}$. The encryption and decryption functions are

$$E_k^r(m_1 \ldots m_r) = E_k(m_1) \ldots E_k(m_r)$$
$$D_k^r(c_1 \ldots c_r) = D_k(c_1) \ldots D_k(c_r).$$

Here, we use the superscript r to distinguish these functions from the base functions $E_k()$ and $D_k()$.