1 Introduction to Cryptography

The purpose of these notes is to provide concise statements of essential facts. A full set of notes consists of these handouts together with material given in class.

The recommended text for this part of the course is Buchmann, *Introduction to Cryptography*, and will be referred to throughout the handouts. An excellent reference, which happens to be free, is *Notes on Cryptography* by Peter Cameron. A copy of these notes is on the MATH220 course page.

Not surprisingly, there are numerous books on cryptography. A great non-technical read is *The Code Book: The Science of Secrecy from Ancient Egypt to Quantum Cryptography* by Simon Singh.

The traditional setup for cryptography reads like a 1960s spy novel. There are three main characters. *Alice* who wishes to communicate a message secretly to *Bob*, and the enemy *Eve* who is waiting in the wings to intercept the message. The art of cryptography is to convert the message into a form that Bob can read and understand but that Eve cannot.

Here are three basic terms:

Plaintext Essentially, this is the message being sent. At this stage nothing is hidden.

Ciphertext This is the message that is actually sent. We assume that Eve can get her hands on the ciphertext, but hope that Eve cannot recover the plaintext from the ciphertext.

Key This is the information used to *encrypt* the plaintext to ciphertext and to *decrypt* the ciphertext to plaintext.

Historically, it was assumed that a key for encryption is the same as a key for decryption and, for the first cryptosystems we look at, this will be the case.

Eve's job–and some may say that it is the most interesting one–is to recover the plaintext from the ciphertext without access to the key. Eve is a *cryptanalyst*. The harder Eve's job, the better the cryptosystem. We begin by looking at an example where Eve's job is not too hard.

1.1 Substitution Ciphers

Typically we ignore punctuation and spaces. We also do not distinguish between upper and lower case letters. Thus we truly have an alphabet of 26 symbols.

In a substitution cipher, we take a permutation (reordering) of the alphabet and apply that to the plaintext. We decrypt by applying the inverse of the permutation to the ciphertext.

Example 1.1.

Since there are

$$26! = 403291461126605635584000000$$

permutations, it's plausible that a substitution cipher could be hard to break–this is a huge number even for the latest and biggest computers. But many of you would have done the Codebreaker in the morning newspapers and you know the trick–you take advantage of frequency analyses of letters in the appropriate language.

Of course, frequencies depend on the source. Compare the frequency of a and k in different languages. Nevertheless, there is a commonality of frequencies across a broad range of sources. The differences are not that great.

Advantage can also be taken of frequencies of letter pairs, that is *digrams*, and triples, that is *trigrams*, and these can be helpful as well. *th* and *he* are the most common digrams in English, and *the* is the most common *trigram*. So looking at the most common character in the ciphertext should reveal e, and from there you should be able to find t and h by looking at digrams. In fact, you do this intuitively. For example, if

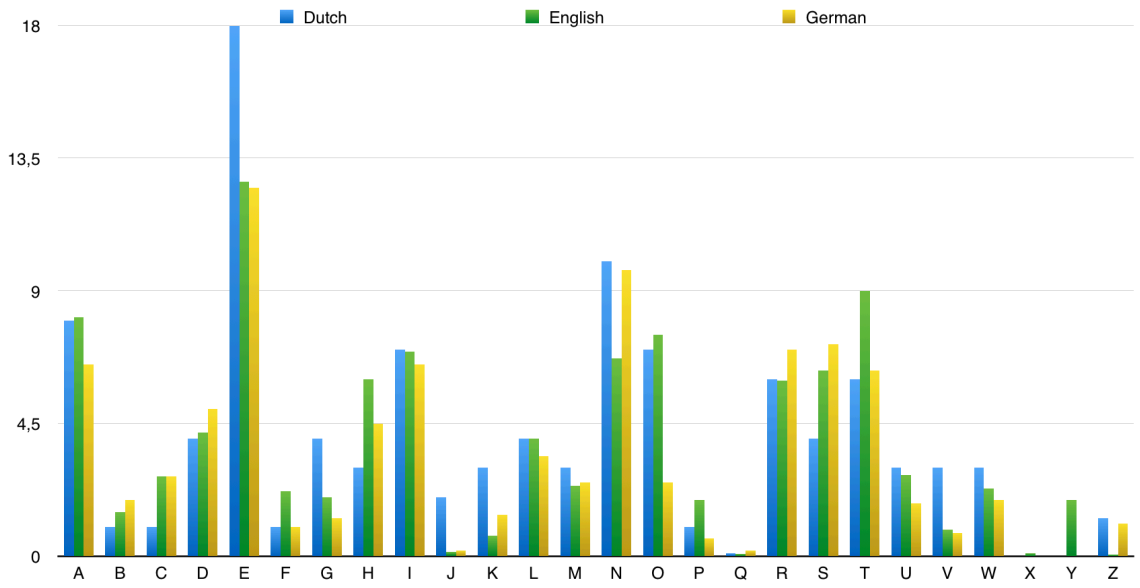


Figure 1.1: Letter frequencies in English, Dutch and German. (Wouter Maes `https://creativecommons.org/licenses/by-sa/4.0`)

you think you have broken a substitution cipher but end up decrypting a triple to xze, there is clear evidence that you have made a mistake. I cannot think of a single case where this trigram occurs in English.

See Cameron p. 14 for a good discussion on the subtle mix of science and art involved in breaking a substitution cipher.

1.2 Caesar Ciphers

In general, a substitution cipher requires Alice and Bob to each keep a copy of the key, in this case, a permutation of the alphabet. Given my fading memory, there is no way that I could memorise an arbitrary 26-letter permutation. How can we simplify the key and still have a "reasonably secure" substitution cipher?

Named after Julius Caesar, a particularly simple substitution cipher is the *Caesar Cipher* in which a constant k is chosen and each letter is replaced by the letter k places after it.

Example 1.2.

The Caesar cipher is every child's first cipher—at least my day you could often get a free code wheel in boxes of cereal. Given that the Romans successfully ran a large empire for quite some time, I find it hard to believe that they would have used such an elementary cipher.

Computers work best with numbers rather than letters. So we usually change letters to numbers first and work *modulo* 26. For example, we might set

$$a \to 0,\ b \to 1,\ \ldots,\ y \to 24,\ z \to 25.$$

Example 1.3.

1.3 Vignère cipher

The Vigenère is a polyalphabetic substitution cipher, an extension of the Caesar and (simple) substitution ciphers. The key is a sequence of letters. E.g. K = BOTTLE-OFRUM The key is repeated until it is as long as the message, and corresponding message characters and key characters are added (modulo 26).

Example 1.4.

The key is easier to remember than a random permutation, but since a different shift is applied at each key position, simple frequency analysis is broken. It can still be broken if the length of the key is known. Here the length is 11. If we take every 11th letter of the ciphertext, there are all encoded with the same shift, so we can apply frequency analysis. However we will need much more ciphertext for this to be accurate than we would for a monoalphabetic substitution cipher.

1.3.1 Autokey Vigenère cipher

Idea: Break the frequency analysis by having a non-repeating key. The key is a sequence of letters. E.g. K = BOTTLEOFRUM The key is repeated only once. The key is then augmented with the start of the message.

Example 1.5.

The Vigenère cipher was widely used and considered unbreakable for nearly 300 years. Then in 1863, a Prussian major discovered that the the length of the keyword can be accurately determined by looking at the separation of repeated bigrams. Once the key length is known, frequency analysis can be used.

1.4 Affine Ciphers

The Caesar Cipher is an example of an *affine cipher*. Here one chooses two integers α and β between 0 and 25 (the key) and encodes by the *affine* function

$$f(x) = \alpha x + \beta \bmod 26.$$

The Caesar Cipher always chooses $\alpha = 1$.

Example 1.6.

Will any two integers α and β produce a substitution cipher? The answer is no! It will only work if f is a permutation of the set $\{0, 1, 2, \ldots, 25\}$. How can we guarantee this? Again, more on this later in the handouts.

There are a variety of tricks for making substitution ciphers more secure, see Cameron p. 19. There are also related types of ciphers such as the *Playfair cipher*, Cameron p. 21. Essentially such ciphers are substitution ciphers based on the 26^2 digrams of the English alphabet.

Example 1.7. Playfair cipher. Invented in 1854 by Sir Charles Wheatstone, it was used as a field cipher by the British in the Boer War and WWI, and as an emergency back-up cipher in WWII.

Key.

Encryption.

Continue this process for each pair of letters. See Tutorial 1 for what happens if the two letters are in the same row or column.

1.5 Better Symmetric Ciphers

A *symmetric cipher* is a cipher in which the encryption key is also the decryption key (or at least the decryption key can be easily computed from the encryption key). The Playfair cipher and, more generally, substitution ciphers are examples of symmetric ciphers.

While lots of tricks can be added to substitution ciphers to make them more secure, in the end, they are all vulnerable to attack. This is because knowledge of new tricks soon spreads through the intelligence community and once Eve knows about them, she can adapt to them. This leads to a general principle, known as *Kerckhoff's Principle* (1882):

Kerckhoff's Principle.

A simple way to improve on substitution ciphers is to make different changes to message symbols according to their position in the message stream.

Using a book. One way to design a cipher is for Alice and Bob to both have copies of a book, say War and Peace. Alice encrypts by adding the n-th letter of War and Peace to the n-th letter of the message using modular (clock) arithmetic (modulo 26). Obviously it is going to be harder in general to break such a cipher than a substitution cipher. But using Kerckhoff's Principle, we can see that Eve has strategies. What are they?

The one-time pad. This cipher system offers *perfect security* no matter what resources Eve has at her disposal. It is very simple in concept but it has a major drawback; the length of the key must be at least as long as the total length of all messages sent under that key. In practice, this makes it unusable except in very special circumstances (for example, nuclear weapon launch codes).

The system itself is very simple and illustrates well the use of modular arithmetic which we will study in some depth.

One-time pad.

The one-time pad has fundamental difficulties:

(i) What does it mean to have a "random" sequence of symbols? And how do you generate such a sequence? In practice, we need to generate a so-called "psuedo random" sequence (see `http://www.random.org/`). We will return to the issue of randomness later.

(ii) The key is as long as the message stream and each recipient needs to have a copy of it.

The second difficulty would once have been a major disincentive for using a one-time pad although not so much these days. It would be perfectly feasible to store, say, a 1 gigabyte one-time pad on any modern computer, and that would handle a lot of messages. It's probable that one-time pads are used at the highest levels of security. However, it is certainly impractical for day-to-day communications like online banking.

Something to think about.

1.6 General Principles

There are a set of guidelines which should be kept in mind when creating a new cryptosystem or evaluating an existing one. These are usually attributed to Kerckhoff (born Jean-Guillaume-Hubert-Victor-François-Alexandre-Auguste Kerckhoff von Nieuwenhof in 1835). Published in his book *La Cryptographie Militaire* in 1882, they were clearly written for an earlier period, but their basic ideas are still sound.

(i) The system should be unbreakable in practice if not theoretically unbreakable.

(ii) Compromise of the system should not inconvenience the correspondents. This is Kerckhoff's Principle, assume that your opponent knows everything about the system except the key.

(iii) The method for choosing the particular key of the system should be easy to memorise and change.

(iv) Ciphertext should be transmittable by telegraph.

(v) The apparatus should be portable. Think of the famous Enigma machines which produced Enigma ciphers.

(vi) Using the system should not require a long list of rules or mental strain.

Today, some of these should be updated allowing for the use of computers.

In devising or evaluating a cryptosystem, it should be assumed that the opposing cryptanalysts are

(i) totally familiar with the particular system being used,

(ii) in possession of unlimited amount of ciphertext,

(iii) (worse!) in possession of some ciphertext with corresponding plaintext,

(iv) (catastrophic!!) in a position to obtain ciphertext corresponding to chosen pieces of plaintext.

1.7 Shannon's Theorem

Shannon (1940s) proved that the one-time pad offered "perfect security". What does this mean? Informally, it says the following.

Theorem 1.1. A one-time pad is secure against statistical attack.

The above is not really precise enough to be properly called a theorem. The idea is that if Alice has a one-time pad and Eve has access to the ciphertext, then Eve has absolutely no information about the message text. Let's make the theorem more like a theorem.

Consider a one-time pad. (It may help to look back through our example in Section 1.5.) We have three *random variables*: p for the plaintext, z for the ciphertext, and k for the key.

We say $p = p_0$ for the *event* that the message string is a particular string p_0 and $\mathbb{P}(p = p_0)$ for the *probability* of the event $p = p_0$.

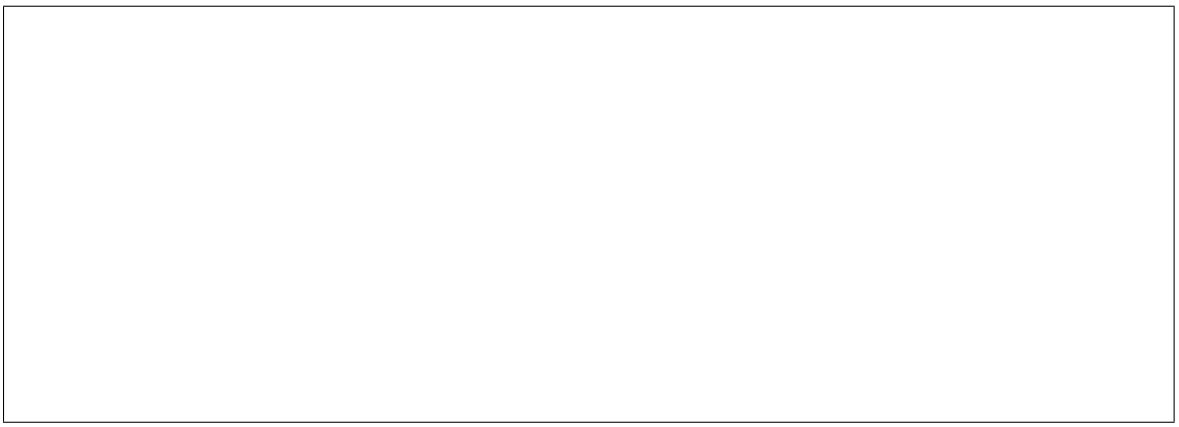
For events A and B, the *conditional probability* $\mathbb{P}(A|B)$ denotes the probability that A occurs given that B occurs, and is defined as

$$\mathbb{P}(A|B) = \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(B)}.$$

Furthermore, A and B are *independent* if

$$\mathbb{P}(A \cap B) = \mathbb{P}(A) \cdot \mathbb{P}(B).$$

Intuitively, two events are independent if the occurrence of one of the events does not effect the probability that the other occurs.



More formally Shannon's Theorem says the following:

Theorem 1.2. For any plaintext p_0, we have

$$\mathbb{P}(p = p_0 | z = z_0) = \mathbb{P}(p = p_0),$$

that is, Eve gains no new information about the message (plaintext) from the ciphertext.

Proof. As for the earlier example, we prove the result for when the alphabet consists of two symbols. For an alphabet of q symbols simply replace '2' with 'q'.

Assume the message has length n and that the alphabet has 2 symbols. We obtain the ciphertext z_0 by adding the key k_0 to the plaintext p_0 using clock arithmetic modulo 2.

By definition,

$$\mathbb{P}(p = p_0 | z = z_0) = \frac{\mathbb{P}(p = p_0 \text{ and } z = z_0)}{\mathbb{P}(z = z_0)}.$$

Now the event $p = p_0$ and $z = z_0$ is the same as the event $p = p_0$ and $k = k_0$, as they uniquely determine each other. (Why?) Since k_0 is a random string of length n from an alphabet of 2 symbols, we have

$$\mathbb{P}(k = k_0) = \underbrace{\tfrac{1}{2} \cdot \tfrac{1}{2} \cdots \tfrac{1}{2}}_{n \text{ times}} = \tfrac{1}{2^n}.$$

Moreover, the plaintext and the key are independent. Thus

$$\mathbb{P}(p = p_0 \text{ and } k = k_0) = \mathbb{P}(p = p_0) \cdot \mathbb{P}(k = k_0) = \frac{1}{2^n}\mathbb{P}(p = p_0).$$

Now, what is $\mathbb{P}(z = z_0)$? For each choice of plaintext p_i, there is a unique key k_i such that p_i encrypts to z_0. So

$$\mathbb{P}(z = z_0 | p = p_i) = \frac{1}{2^n}.$$

Therefore, using the 'Theorem of Total Probability' and summing over all i, we have

$$\begin{aligned}
\mathbb{P}(z = z_0) &= \sum \mathbb{P}(z = z_0 | p = p_i) \cdot \mathbb{P}(p = p_i) \\
&= \sum \frac{1}{2^n} \mathbb{P}(p = p_i) \\
&= \frac{1}{2^n} \sum \mathbb{P}(p = p_i) \\
&= \frac{1}{2^n},
\end{aligned}$$

as $\sum \mathbb{P}(p = p_i) = 1$. It now follows that

$$\mathbb{P}(p = p_0 | z = z_0) = \frac{\mathbb{P}(p = p_0)/2^n}{1/2^n} = \mathbb{P}(p = p_0),$$

as required. □