MATH2068/2988
Number Theory and Cryptography
Week 3 Lecture 1

Robert Howlett, ed. Anthony Henderson, Dzmitry Badziahin

The University of Sydney

14th August 2017

Why do we study classical cryptography?

Our main objective in the cryptography part of the unit is to understand the RSA, Elgamal and Rabin cryptosystems, which use some interesting number theory.

But it makes no sense to study these modern cryptosystems in ignorance of their predecessors. In particular, we need to understand some of the deficiencies of classical cryptosystems to see the advantages of RSA etc.

Alphabets

An *alphabet* is a set of symbols (of any kind).
The elements of this set are called the *letters* of the alphabet.
A *message* is a sequence of letters from some alphabet.

In most of our examples we will use the alphabet
$\{A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z\}$
consisting of the upper-case letters of English.

Alternatively, we will sometimes use the numbers
$0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25$
to represent these 26 letters, in situations where it is more convenient to work with numbers. However, this is an arbitrary choice: others might start counting from 1, or use the ASCII codes for the letters instead.

Encryption and Decryption

The basic situation envisaged in cryptography is that someone (traditionally “Alice”) wishes to send a message to someone else (traditionally “Bob”), in such a way that no-one else who may intercept the message will be able to understand it.

To achieve this, Alice applies an *encryption* or *enciphering* process to convert the original message (the *plaintext*) into another message (the *ciphertext*), which is sent instead. The *source alphabet* in which the plaintext is written could conceivably be different from the *target alphabet* in which the ciphertext is written.

Then the recipient Bob applies the *decryption* or *deciphering* process, converting the ciphertext back into the plaintext.

The encryption and decryption processes together constitute a *cryptosystem* or *cipher*.

Codes vs Ciphers

From a mathematical viewpoint, encryption is just a function from one set of messages (the conceivable plaintexts) to another set of messages (the ciphertexts resulting from these plaintexts). The function should be invertible, and decryption is the inverse function.

In everyday English, the words *encoding* and *decoding* are often used for such an inverse pair of functions, instead of *encryption* and *decryption*. But it is useful to maintain a distinction between codes and ciphers, where the word *code* is reserved for situations where there is not meant to be any secret about how to encode or decode messages.

For example, Morse code converts messages from one alphabet (ordinary letters) to another (dots and dashes), but purely for ease of transmission: it is not in itself a way of keeping messages secret.

Keys

There is always a tension between the security of a cryptosystem, i.e. how hard it is for an unintended recipient of an encrypted message to decrypt it, and the ease of use for the intended users. A cryptosystem could be made totally secure if Alice and Bob could agree beforehand on a random invertible encryption function on the set of all conceivable messages. But this is probably not practical.

Instead, we generally consider cryptosystems which are of a standard type, but depend on some secret parameter called the *key*, which takes less memory (human or computer) to record than the whole encryption function does. There may be separate keys for encryption and decryption. The security requirement is that even if eavesdroppers can work out the type of cryptosystem being used, they should not be easily able to work out the decryption key.

Caesar's cipher

Here is a very simple cipher based on one used by Julius Caesar. Represent the letters A – Z by the numbers 0 – 25.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

Caesar's cipher consists of replacing each letter i by $i+3$, reduced modulo 26 if $i+3 \geq 26$.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

So

CROSSINGRUBICONTOMORROW

becomes

FURVVLQJUXELFRQWRPRUURZ

Translation ciphers

Of course, Caesar could have used any number k in place of 3.

A *translation cipher*, or *alphabetic shift*, is a cipher in which, to encrypt a message, each letter i is replaced by $i+k$ (reduced mod 26), for some fixed k. To decrypt the ciphertext, each letter j is replaced by $j-k$ (reduced mod 26).

The key in this case is just the number k (or rather, its residue modulo 26). Anyone who knows the key knows both how to encrypt and how to decrypt messages.

Of course, this is about the least secure system imaginable! There are only 26 possible keys, and an enemy who intercepted the message could just try them all.

Such a method of attack is known as an *exhaustive key search*. A basic requirement of a cryptosystem is that the number of possible keys should be so enormous that an exhaustive key search is not feasible.

Simple substitution ciphers

Translation ciphers are examples of *simple substitution ciphers*, where a message is encrypted by applying the same pre-determined substitution rule to each letter. The encryption key is just this substitution rule, which could be any invertible function f from the source alphabet to the target alphabet. The decryption key is f^{-1}.

Thus, in a simple substitution cipher, if the plaintext is $x_1 x_2 \dots x_N$, the ciphertext is $f(x_1)f(x_2)\dots f(x_N)$.

For an arbitrary simple substitution cipher on the alphabet A – Z, the key is a permutation of the 26 letters. So the number of possible keys is
$26! = 26 \times 25 \times \cdots \times 2 \times 1 = 403291461126605635840000000$,
which certainly makes an exhaustive key search impossible.

Homophonic substitution ciphers

These systems are slightly less simple: the target alphabet has more letters than the source alphabet, and each letter of the source alphabet is associated with a set of letters of the target alphabet.

Example: 26 letter source alphabet A – Z; 64 letter target alphabet consisting of 8 term sequences of 0's and 1's.

$$\begin{aligned} \text{A} &\to \{01101100, 11110000\} \\ \text{B} &\to \{10101010, 00011110\} \\ &\dots \\ \text{E} &\to \{11011011, 00110011, 01010101\} \\ &\dots \end{aligned}$$

An A in the plaintext can be encrypted as 01101100 or as 11110000, an E as 11011011, 00110011 or 01010101, etc..

This makes frequency analysis harder, but not impossible.

Frequency analysis

However, simple substitution ciphers have an obvious weakness when applied to actual text in English (or another similar language): one can make a good guess at which ciphertext letters represent which plaintext letters by examining the frequencies with which the letters occur in the ciphertext.

The most common letters in English text are E ($\approx$ 12%), T ($\approx$ 9%), A ($\approx$ 8.5%), O, N, I, S, R, H (all about 6% to 7.5%), followed by D ($\approx$ 5%) and L ($\approx$ 4%).

So, given a ciphertext which is long enough for the frequencies of letters to be meaningful, it is probably easy to identify what letters represent E, T and A. The next six most frequently occurring letters in the ciphertext are likely to represent O, N, I, S, R, H in some order, and there are only $6! = 720$ different orders to try. When you try the right order there will be so many recognizable words that the rest will be easy.

Playfair Square

An example of a *polygram substitution cipher*, where the letters of the plaintext are grouped into blocks (or "polygrams"); then blocks are substituted by other blocks according to some rule.

The Playfair Square requires a 25 letter source alphabet. This can be achieved by identifying J's and I's. The key is an arrangement of the 25 letters in a 5 by 5 square. For example,

T H E Q U
I C K B R
O W N F X
M P S V L
A Z Y D G

The letters of the plaintext are grouped into two-letter blocks. Double letter blocks are not allowed; if one occurs an X is inserted. So "Meet me tomorrow" is first written as

ME ET ME TO MO RX RO WX.

(We had to stick an X on the end to make the length even.)

Playfair Square (continued)

T H E Q U
I C K B R
O W N F X
M P S V L
A Z Y D G

Plaintext: ME ET ME TO MO RX RO WX.

Ciphertext: STQHSTIMAMXLIXNO.

If the letters in a block are not in the same row or the same column, they are replaced by the letters at the other two corners of the rectangle they determine.

Two letters in the same row are shifted one step to the right (wrapping if necessary). Two letters in the same column are shifted one step down (wrapping if necessary).

The intended recipient, knowing the key, can easily reverse this to decrypt the message. Attackers could try to use frequency analysis on two-letter blocks ("digraphs").

Simple affine ciphers

These are simple substitution ciphers which are a slight improvement on translation ciphers. Instead of the substitution rule $i \mapsto i + k \pmod{26}$, we use $i \mapsto mi + k \pmod{26}$ instead. Now the key is the pair of numbers (m, k).

The multiplier m has to be coprime to 26 for $i \mapsto mi + k$ to yield a permutation of the numbers 0 to 25. This follows easily from the "coprime cancellation" property proved last week.

Simple affine ciphers are not much more secure than translation ciphers. You can use frequency analysis to work out which letters represent E and T, which means you know $4m + k$ and $19m + k \pmod{26}$. Then you can work out m and k.

Polygram affine ciphers

It is much better to split the plaintext into blocks of length n (for some n). Then, regarding each block as an n-component column vector, encrypt by multiplying by some fixed $n \times n$ matrix followed by adding on some fixed n-component vector. The determinant of the multiplier matrix has to be coprime to 26.

For example, with $n = 3$, one could use the rule

$$\begin{pmatrix} i_1 \\ i_2 \\ i_3 \end{pmatrix} \mapsto \begin{pmatrix} 3 & 5 & 2 \\ 1 & 2 & 7 \\ 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} i_1 \\ i_2 \\ i_3 \end{pmatrix} + \begin{pmatrix} 4 \\ 8 \\ 12 \end{pmatrix}$$

An attacker would probably have to use polygram frequency analysis, made harder if the block size n is unknown.

Polyalphabetic substitution ciphers

"Polyalphabetic" means "many alphabets", but its use in this context is based on a usage of the word "alphabet" that is different from ours, in which it means essentially the same thing as "substitution key": a permutation of the letters A – Z.

So polyalphabetic ciphers employ m different substitution keys, for some number m. This is to make frequency analysis harder.

The encryption process applies the first substitution key to the first letter of the plaintext, the second key to the second, and so on. For the $(m+1)$-st letter of the plaintext one returns to the first key, and so on.

The Vigenère cipher

A Vigenère cipher is a polyalphabetic translation cipher. That is, m alphabetic shifts are used, for some m known as the *period*.

The key for a Vigenère cipher is the m-letter word giving the images of the letter A under the m translations.

Example: Suppose $m = 5$ and the encryption key is MATHS. Converting letters to numbers, the key becomes 12 0 19 7 18.

S	O	M	E	W	H	E	R	E	O	V	E	R	T	H	E	R	A	I	N	B	O	W
18	14	12	4	22	7	4	17	4	14	21	4	17	19	7	4	17	0	8	13	1	14	22
12	0	19	7	18	12	0	19	7	18	12	0	19	7	18	12	0	19	7	18	12	0	19
4	14	5	11	14	19	4	10	11	6	7	4	10	0	25	16	17	19	15	5	13	14	15
E	O	F	L	O	T	E	K	L	G	H	E	K	A	Z	Q	R	T	P	F	N	O	P

The Vigenère cipher (continued)

The plaintext M can be thought of as a sequence of residues modulo 26. So $M = c_1 c_2 c_3 \dots c_N$, where each c_i is a natural number less than 26, and N is the length of the message.

The keyword $K = k_1 k_2 \dots k_m$ is also a sequence of residues modulo 26. Here m is the period.

Define $k_{m+1} = k_1$, $k_{m+2} = k_2$, etc. More precisely, for each $i \in \mathbb{Z}$ let $k_i = k_r$, where r is the residue of i mod m.

The ciphertext is $M' = c'_1 c'_2 c'_3 \dots c'_N$, where the i-th term c'_i is the mod 26 residue of $c_i + k_i$.

In particular the sequence $c'_1 c'_{m+1} c'_{2m+1} \dots$ is simply an "alphabetic shift" of the sequence $c_1 c_{m+1} c_{2m+1} \dots$. To get c'_{am+1} you just add k_1 to c_{am+1} (mod 26).

We define the *decimation of M with period m and index r* to be the sequence $\mathrm{Dec}(M, m, r) = c_r c_{m+r} c_{2m+r} c_{3m+r} \dots$.

Finding the period using coincidence index

The *coincidence index* of a piece of text is the probability that two randomly chosen letters are the same. That is, if the relative frequencies of the 26 letters are $p_0, p_1, \dots, p_{25}$ then the coincidence index is $\sum_{i=0}^{25} p_i^2$.

By the Cauchy–Schwarz inequality, the coincidence index is always at least $1/26 = 0.0385\dots$. The more skewed the distribution of frequencies is, the higher the coincidence index will be. For English text the coincidence index is usually about 0.065.

An alphabetic shift (or any simple substitution cipher) does not change the coincidence index. In particular, if a Vigenère cipher has period m, the decimations $\mathrm{Dec}(M', m, i)$ will have coincidence index about 0.065.

This provides a convenient way to find m: compute the coincidence index of $\mathrm{Dec}(M', m, 1)$ for $m = 1, 2, 3, \dots$ until we find an m that gives a value greater than about 0.06.

Finding the key once the period is known

Suppose that the period is m. Then

$$\mathrm{Dec}(M', m, 1) = c'_1 c'_{m+1} c'_{2m+1} c'_{3m+1} \dots$$

is the same as

$$\mathrm{Dec}(M, m, 1) = c_1 c_{m+1} c_{2m+1} c_{2m+1} \dots$$

alphabetically shifted by k_1, where k_1 is the first term of the key.

We can find k_1 by examining letter frequencies in $\mathrm{Dec}(M', m, 1)$: whatever is the most frequent letter in $\mathrm{Dec}(M', m, 1)$ is probably the shift of E by k_1, and we can check that guess easily using the next most frequent letters.

And we can similarly find $k_2, k_3, \dots, k_m$ by examining letter frequencies in $\mathrm{Dec}(M', m, 2)$, $\mathrm{Dec}(M', m, 3)$, $\dots$, $\mathrm{Dec}(M', m, m)$.

For the above methods to work one needs a piece of ciphertext of length many times the length of the period m. Otherwise the decimations $\mathrm{Dec}(M', m, i)$ will not be long enough to give meaningful frequency distributions. In the extreme case that m is greater than the length of the ciphertext, it is impossible to decrypt the message without knowing the key.