Cryptography

Kálmán Liptai

Cryptography

Kálmán Liptai

Publication date 2011

Table of Contents

Acknowledgements

Cryptography is an infinitely exciting and fascinating chapter of human thinking. Historical events and prominent members of human thinking have contributed to its development. Wars and conflicts fastened its advancement, which is a sad fact. On the other hand, the joining of outstanding scientists into the world of encryption enriched this field. Newer and newer disciplines helped and still help to secure flow and store information in the 21th century.

This is a very complex field of the science and draws on several resources. Cryptography has become curriculum in higher education all around the world, so as in Hungary. To create and complete the syllabus which can be read and studied in the followings, I have received a lot of help from my colleagues and students. Probably without their interest and sensitivity to this issue I would not have undertaken this job.

I would like to express my thanks hereby to Péter Olajos Phd. and Tibor Tómács Phd. for their countless and patient help that I always received from them. The natural curiosity of my students and their works have also given the level of the final work a great push. I would express my thanks to Kiss Norbert and Gábor Mészáros for the demonstrative program of AES, to György Győrfi and Ádám Csintalan for the Playfair program, to István Csonka and Viktor Trombitás for demonstrating the DES. I owe my gratitude to Tivadar Radácsy, Tamás Vass, Beáta Mátéfi, Judit Kovácsa and to many of my thesis writers as they showed me that we are at the gate of inexhaustible interest, when we are studying cryptography.

In this part I would say special thanks to Attila Egri–Nagy Phd. and Csaba Vreczenár that the work could be made in English too.

Finally I would like to thank my family that they endured the feverish job which the creation of the course book demanded.

Chapter 1. Historical Overview

1. Introduction

The history of cryptography is at least as complicated and intricate as the history of mankind itself. This statement would be hard to prove formally, but if we consider the past centuries, all the decisive moments seem to be important from a cryptographical viewpoint as well. In order to see the path leading to today's cryptographical applications we need to go through the basic notions.

The term cryptography originates form the ancient Greek words „kryptos" (hidden) and „grápho" (write) word. In plain English we could say secret messaging, but cryptography itself is now well-understood terminology.

It is easy to state the basic problem. How can we send a message in a way that the receiver can easily read it but for a third party it is nearly impossible, or it takes a huge amount of time to decipher the message. Later we will determine precisely what does "huge amount of time" exactly mean, but for the time being the everyday interpretation is enough.

The meaning of the actual text (or the lack of it) is not important to us, since most of the time we encrypt text that is already encoded, therefore it is already just a sequence of letters and numbers. In the following by encoding we mean the process of substituting letters (symbols) by numbers. An example is the usual coding of characters by their ordinal number in the alphabet's order.

In historic times the encrypted message consisted of letters, nowadays these texts are mainly bit-sequences.

In following chapters we can distinguish between two parts, that differentiate both historically and in principle. The first part is usually called classical cryptography and it lasted until the middle of 20th century. In this part clever ideas without mathematical methods dominate, so a method consists of a series of innovative tricks, that is kept top secret. These moments of ingenuity are usually connected to historical periods or events. We will talk about statistical methods that can be used to crack these older encryptions with today's computers.

The second part is public key cryptography. These methods make the encryption method and encryption keys public. Of course a "secret trapdoor" is set up and we keep the decryption keys in order to attain privacy. These methods are based on mathematical theorems and breaking the encryption would need enormous computational time and power.

We have another distinction between the encryption methods beyond the classic/public key classification. Those methods that use the same keys and encryption method for both the sender and the receiver are called *symmetric-key encryptions*. All classical methods are of this kind but there are also contemporary algorithms using symmetric-keys. For instance the DES or AES methods.

It was thought impossible that we can have a method for secret communication without sharing some common secret by the communicating parties, or such that we know the encryption key but we are still unable to decrypt the message. In the 20th century this riddle was solved, and we call these methods *asymmetric-key encryptions*. The prime example is the RSA method.

2. Basic Notions

Independent on the text and the type of the encryption we can determine a logical sequence of secret information exchange. We start with a plaintext T written in natural language. First we have to encode then encrypt the message getting the $C_T = E_k(T)$ encrypted text, this result will be referred to as the cryptotext. This message can be transmitted through an open channel, if we used a good method. The receiver can decrypt the message by using the decryption key D_k, thus getting $D_k(C_T)$. After decoding we get back the original plaintext. Clearly the notation comes from the corresponding English words: T text, E_k encrypt, D_k decrypt, k is the applied key (see [17]). In the literature the terms "cleartext" and "ciphertext" or briefly "cipher" are often used instead of "plaintext" and "cryptotext". The verbs for translation are in the case "encipher" and "decipher".

Sir Francis Bacon (1561-1626), who was active in politics and philosophy, thought about the question 'What makes a good a cryptosystem?' as well. In his opinion E_k and D_k methods should be simple and the encrypted

text should look innocent. Today, at the age of computers every bit-sequence looks innocent so this requirement is not a problem, but the others are still valid guidelines.

Sir Francis Bacon

Probably it is clear for everyone that no one can work on encryption without trying to break the code. We can test our methods by playing the role of the illegal intruder and trying to crack the system. In fact, it is often more exciting trying to crack existing methods rather than devising new ones. At the same time we can learn a lot from these attempts.

From now on we assume that we know the encryption method and the main task is to decrypt, to figure out the original message.

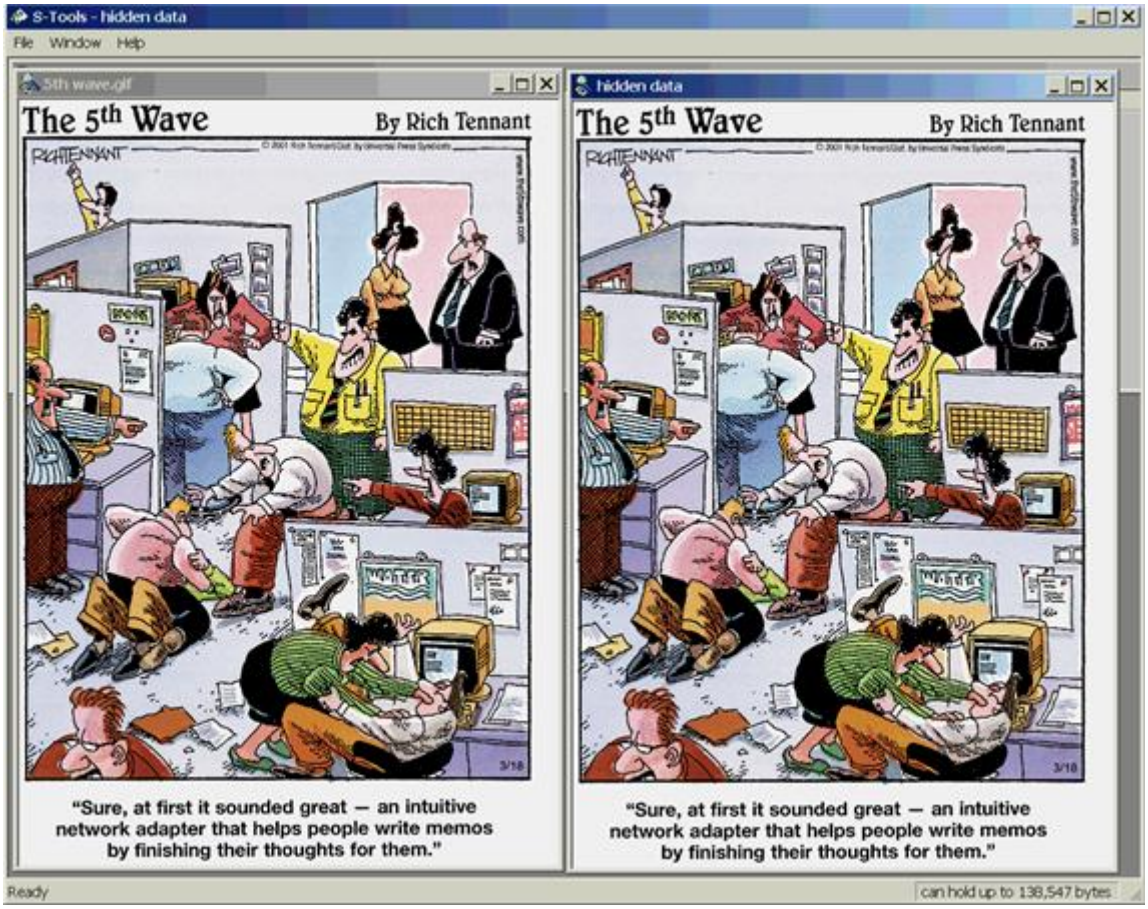
The main question is whether the decryption is possible at all or not? We have several cases:

1. Let's suppose that we know some encrypted text that is quite long. In this case we can try to crack the classical cryptosystems if we know some statistical information on the given language.

2. We also have chance If we know some $(T, E_k(T))$ pairs.

3. If the intruder is skillful enough to pretend to be a legal user then $(T, E_k(T))$ pairs can be obtained of his/her choice. This also significantly increases the chances.

We should mention here that since we are interested mainly in the mathematical aspects of cryptography we do not discuss some historically important cryptosystems. Such an example is encrypting with a *Codebook*, the "aristocracy of cryptosystems", where both parties use their own dictionaries. Also, hiding the message with invisible ink or writing it on a shaved head on which the hair grows eventually. These later methods belong to steganography. The main difference between cryptography and steganography is that the first aims to hinder third parties to be able to read the message, while the later aims to hide even the existence of the message. Digital images give ample opportunities for applying steganography in 21st century. If we change only a bit in the color information of a pixel, the change is not really perceivable (using not too many points), but for the insider the changes code useful information. The same applies for digital sound recording as well.



steganography.zip

Chapter 2. Monoalphabetic substitution

In this chapter the classical encrypting methods will be examined (the work of Simon Singh provides an excellent summary of the topic, see [18]). Here we describe, or rather break, the cryptography of ancient times, noting that these were hidden from uninitiated eyes contrary to the present-days' modern public-key algorithms.

Simon Singh

The first known cipher, the scytale from Sparta, had already been used in the 7th century B. C.

Aeneas Tacticus, Greek author, lists several methods in his military documents, written around 360 B. C. The classical methods were mostly used at wartimes. Therefore the work of Aeneas Tacticus also deals with castle defense. Although we would not claim that this was the only reason of cryptography. Diplomacy, public

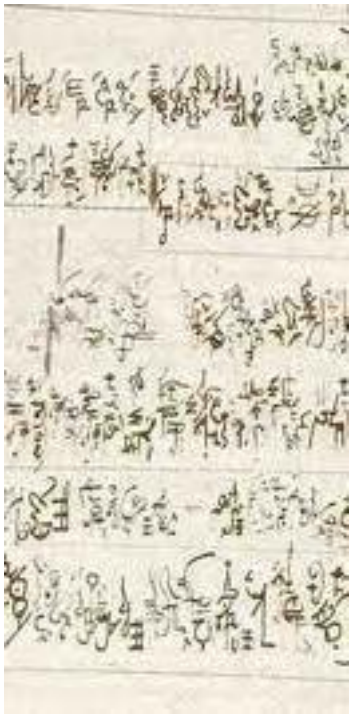
administration, science and the desire for private life could give the reasons of cryptography in ancient times as well.

There is a very special interest with Hungarian reference, namely the diary of Géza Gárdonyi. Gárdonyi developed a unique cipher for himself that consisted of strangely shaped symbols. He mastered the use of it so well that he managed to write as fast as normal handwriting. To hide his thoughts even deeper, the cover of the diary was titled as "Tibetan grammar". However this odd writing is not Tibetan nor Chinese, Korean or Indian. These symbols are used nowhere else on earth. These are Gárdonyi's own inventions but indeed recall the image of some exotic writings.

Géza Gárdonyi

The secret diary remained unsolved from his death in 1922 until even 1965. Then the Géza Gárdonyi Memorial of Eger announced a competition for decrypting the cipher. Gábor Gilicze, a university student and pretty officer Ottó Gyürk had solved the problem independently of each other and the entire secret diary was published.

Diary of Géza Gárdonyi



The statistics of the relation of characters and letters examined by linguists are as helpful in encrypting classical ciphers as computers. It is not known, who recognized first that if the frequency of the letters is known, it can be used for decryption, but we do know who had documented this method first. Yaqub ibn Ishaq al-Kindi, philosopher of the Arabs, did it in the 9th century. His most important thesis is entitled “Secret messages”, and had only been discovered in 1987 in the Ottoman Archive.

The use of the statistical method should be imagined in the following way. The encrypted text is being examined statistically, that is we explore the frequency of occurrence of certain letters, letter pars or in some cases even letter groups. Having the frequency we compare it to the known frequencies of the natural language, in order to find possible matches. In easy cases the system can be broken by finding one single letter, but for more complicated systems it is much more difficult.

The first serious frequency analysis in modern age was performed in English and it is based on 100362 letters altogether. It was created by H. Beker and F. Piper and was first published in their work titled “Cipher Systems: The Protection of Communication”. The following tables contain their data.

a	b	c	d	e	f	g	h	i	j
8,2	1,5	2,8	4,3	12,7	2,2	2,0	6,1	7,0	0,2

k	l	m	n	o	p	q	r	s	t	u
),8	4,0	2,4	6,7	7,5	1,9	0,1	6,0	6,3	9,1	2,8

v	w	x	y	z
1,0	2,4	0,2	2,0	0,1

The statistical features of the Hungarian language are also well known. The most common vowels and consonants are ’a’, ’e’ and ’l’, ’n’.

Certainly, the statistical mapping not only covers letters but also letter pars and triplets. On the other hand not just its words and syntax characterize a given language but its set of characters too. Certain languages contain letters that are missing from other languages even if the way of writing is mostly the same. It usually turns out from such monitoring which language is of our business.

In most cases it is presumable that the given language is known, moreover it is well charted in terms of frequency. Rare language families can be a more difficult task and could give a hard time even to legal decoders, as very few people may understand the given language.

One well known example of the use of non-charted languages was the Navajo language used in World War II. The language of one of the most populous but rather illiterate Native American tribes was especially suitable for sending each other oral messages on the frontlines.

The messages had not been translated to Navajo as the encrypted text used substitute expressions. They had created a very complicated system, in which each military expression in English had been replaced with a Navajo word. Although the appropriate word had some logical relations with its English counterpart (like potato meant grenade) in order to make it easier to memorize, but it was not a direct translation. Therefore the messages did not make any sense for outsider Navajos.

Navajo code talkers also took part in the Korean and Vietnamese wars. (Just for the record we would like to mention that because of its secrecy the participating soldiers had not been awarded at all until 1982. Then President Reagan officially thanked Navajos and pronounced 14th August the Navajo Code Talkers Day).

Without the help of computers even the classical ciphers can be difficult to code and decode thus simple support programs have been created in behalf of demonstration. Henceforward for being pragmatic we agree in using letters without accent and in case of English language letter J is taken off due to its rare occurrence. Now suppose that our alphabet contains 25 letters.

At first we are going to deal with the so called monoalphabetic substitution, which means that the alternatives of certain letters do not change during encryption. This makes them quite easy to decode, so obviously they are not used any more, but their historical significance is worth mentioning.

1. Ceasar cipher

The first encrypting method being examined, is the Caesar cipher that is made up of a simple slip of the Alphabet. The use of substitution as an encrypting method for military purposes had been documented in the "Commentaries on the Gallic War" by Julius Caesar. Caesar had recourse to use cryptography so often that Valeris Probus wrote an entire thesis of the code used by him, but unfortunately it had not survived.

Julius Caesar

However due to Suetonius, who wrote his work "The twelve Caesars", we can get a detailed description of replacement algorithm used by Caesar. He changed every letter of the Alphabet for the third following ones.

A	B	C	D	...	W	X	Y	Z
D	E	F	G	...	Z	A	B	C

Obviously, if the scale of shift that is the alternate of a letter is revealed, the algorithm becomes easily solvable. Using statistic method we can find the substitute letter.

Ceasar.zip

2. Caesar shift cipher

It is based on the same principles like the simple Caesar cipher, but here we use a keyword to shift the Alphabet. In choosing the key, we need to pay attention (now and also later) that such word to choose, that consists of different letters.

Now encrypt the word: cryptography. Let the keyword be SOMA.

A	B	C	D	E	F	G	H	I	J	K	L	M	...
S	O	M	A	B	C	D	E	F	G	H	I	J	...

CRYPTOGRAPHY = MQYNTLDQSNEY

A little bit complicated version of Caesar cipher, when the text is divided into units of letters and the scale of shift is different per letters within the certain units. But we can score here as well if we are able to find out the scale of shift within these units, that is after how many letters is the scale similar. Simple statistical reviews get us to attain our goal soon, with this easy cipher, just like with any other classical ciphers.

3. Polybius square cipher

The next truly ancient cipher is called Polybius. Polybius was an adviser of the great general of the 3rd Punic War Cornelius Scipio. We may encrypt a text by using the following chart, where each letter is replaced with a pair of letters.

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

In this case any letters can be encrypted by finding the appropriate index of the rows and columns. Every pair stands for one letter, for example IA hides letter K and IU letter O.

The choice of the index is of course optional from the world of letters or maybe from any other signs. The letters of the Alphabet are replaced by vowel pairs, and these can easily be hidden in words. You may read an encrypted text hereby: LOOK! WHAT'S THAT UNDER THAT USUAL POOR? To decrypt the text all the vowels have to be graded in pairs.

Then we have the following pairs: OO AA UE AU UA OO. Now we can decode the message by using the chart above. The secret message is: Save us.

Like the previous ones, the encrypted can be resolved by using statistical methods and paying attention that pairs stand for single letters.

4. Hill method

Lester S. Hill developed this method in 1929 that was named after him and uses matrixes.

Lester S. Hill

It is also capable of encrypting any blocks regardless of its length. To use the Hill cipher we create a simple coding first, in which we replace the letters of the Alphabet with ordinal numbers, that is:

	B	C	D	..
	2	3	4	..

After this substitution we consider each $(\mathrm{mod}\ 25)$. For encryption we use an optional $n \times n$ type invertible matrix and write its elements $(\mathrm{mod}\ 25)$.

The words to be coded are written down without spaces, and divided into units containing n letters. Then we decode these units and create T_i n dimensional column vectors from them. After executing the operations mentioned above, the formula can be provided by MT_i matrix multiplication. The operation results T_i' column vectors, which reveal the message after decryption.

For example, encrypt the word MINDIG by using a 2×2 matrix.

$$M = \begin{pmatrix} 3 & 5 \\ 2 & 4 \end{pmatrix}, \quad T_1 = \begin{pmatrix} M \\ I \end{pmatrix} = \begin{pmatrix} 13 \\ 9 \end{pmatrix},$$

$$T_2 = \begin{pmatrix} N \\ D \end{pmatrix} = \begin{pmatrix} 14 \\ 4 \end{pmatrix}, \quad T_3 = \begin{pmatrix} I \\ G \end{pmatrix} = \begin{pmatrix} 9 \\ 7 \end{pmatrix}.$$

Now compose T_1', T_2', T_3' vectors according to the given rule.

$$MT_1 = \begin{pmatrix} 57 \\ 101 \end{pmatrix}, \quad MT_2 = \begin{pmatrix} 50 \\ 86 \end{pmatrix}, \quad MT_3 = \begin{pmatrix} 41 \\ 73 \end{pmatrix}.$$

Take the elements of these matrixes $(\mathrm{mod}\ 25)$ and we get the following matrixes.

$$T_1{}' = \begin{pmatrix} 7 \\ 1 \end{pmatrix}, \quad T_2{}' = \begin{pmatrix} 0 \\ 11 \end{pmatrix}, \quad T_3{}' = \begin{pmatrix} 16 \\ 23 \end{pmatrix}.$$

Thus we have gained the encrypted word, HBALRY.

Decryption is obviously easy if matrix M is known, because if we have chosen properly, the matrix is invertible and the $M^{-1}T_i'$ product (taking the result $(\mathrm{mod}\ 25)$) provides the codes of the letters of the original text.

Any illegal intruder must be aware of the image of two pairs. In order to determine it, we have to examine the distribution of the letter pairs first. After identifying the most common pairs, we may have a good chance of decryption.

Suppose that, the images of T_1, T_2 matrixes are known. Then the chosen M matrix comes from the following matrix multiplication.

$$\begin{pmatrix} 57 & 50 \\ 101 & 86 \end{pmatrix} \begin{pmatrix} 13 & 14 \\ 9 & 4 \end{pmatrix}^{-1}$$

However, we might need a few luck as well to succeed at once, as the given matrix may not be invertible. In this case, we look for another suitable pair.

Remark. It also comes after some easy calculations, that the invert matrix will be the following

$$\begin{pmatrix} 2 & -\frac{5}{2} \\ -1 & \end{pmatrix}$$

We also note that if the given text cannot be divided into units with length n, then we either add some extra letters which do not change the meaning, or we consciously make some grammatical mistakes. This method proved to be very effective at the time of its invention, as it is quite labor-intensive, but with appearance of computers both the encryption and the decryption became obvious.

5. Affin cipher

The affin cryptosystem is the next one in our row of description. Suppose that a and b are such positive integers, that $0 \leq a, b \leq 24$ and $(a, 25) = 1$. Then after using the previously introduced and by this time habitual coding each α letters is replaced with the result of $a\alpha + b \pmod{25}$. Finally the given result is decoded to provide the original message.

We also note that assuming that $(a, 25) = 1$ is necessary for the assignment $a\alpha + b \pmod{25}$ that provides the final result, to be bijective. Otherwise it could be possible, that different letters have the same image. Because if we encrypt α and α_1 their generated image would be $a\alpha + b \pmod{25}$ and $a\alpha_1 + b \pmod{25}$. These determine the same numbers if $a(\alpha - \alpha_1) \equiv 0 \pmod{25}$ is fulfilled and according to the condition it can only happen if α and α_1 are the same numbers.

The decryption of this system uses statistical methods. After finding two letters the system collapses.

6. Exercises

1.

   Encrypt the term ”The die has been cast” with Caesar cipher, by using the word CRYPTO as the key.

2.

   Use affin cipher to encrypt the following phrase ”Sapienti sat” where $a = 6$ and $b = 2$.

3.

   Encrypt the quote ”Be great in act, as you have been in though” (W. Shakespeare) with the help of the Hill cipher.

   $$M = \begin{pmatrix} 6 & 5 \\ 2 & 3 \end{pmatrix}.$$

4.

   Design Polybius cipher by using geometrical formations.

5.

Decrypt the document in the file szidd2.txt with the help of the attached statistic maker program stat.exe. The encryption has been made by Caesar cipher and the original text is from Herman Hesse’s book, Siddharta.

Chapter 3. Polyalphabetic substitution

It turns out during a more detailed examination of the Hill cipher that the images of identical letters are not always identical. For example, if we use a 2x2 matrix for encryption, letter group ”VE” might have different images in the words UNIVERSITY and VERSA.

These encrypting methods are called monoalphabetic substitutions in a broader sense. This leads us to the polyalphabetic substitutions, mentioned in the title, where the substitution of the identical text sequences are different during the encrypting process.

1. Playfair cipher

The first method of its kind is the so-called Playfair. This is a symmetric cipher, Charles Wheatstone invented it in 1854.

Charles Wheatstone

Lord Playfair promoted the use of this method. Taking advantage of the reduction mentioned above, we place 25 letters of the Alphabet in a 5x5 square. We form the text in a way that it contains an even number of letters. In case of odd numbers of letters we may make a grammatical mistake or double a character. Then we divide the text into blocks containing two letters, without placing identical letters in one box (the previous tricks can be applied if necessary).

If the resulted letter pair is not set in identical column or row, then considering the letters as the two opposite vertexes of an imagined square, the letters in the other two vertexes provide the encrypted image. If they are set in identical column or row, then according to agreement we shift the letter pair up or down, left or right and so gain letters that gives us the encrypted image.

Y	D	W
I	P	U
C	A	X
N	O	G
K	M	J

The above-mentioned encrypting methods can be read from the illustration. For example the image of AE pair is FO, the encrypted version of HA is CX while it is IN for CK.

Using the previous method, the encryption does not change if we perform a cyclical change of row and column. We can apply the use of a keyword here, as well. Let the compound KEYWORDS be the key, then list all the missing letters, without repeating any.

Decryption is more difficult than in the previous cases. Examination and statistical processing of letter pairs, triplets and quartets lead to the result. Then the resulted data has to be compared to the rules of the given language. In case there is a keyword, resolving the length of it provide the decryption of the method, as the letters come in alphabetical row after the keyword. However the coder has several possibilities to make the decoding process more difficult. Every letter can be encrypted differently or even translated to a foreign language.

You can try this method using the program Playfair.exe.

2. Vigenére cryptosystem

Although the system is titled as Virgenére cipher, more creators contributed to the system. Its origin can be dated back to Leon Batista an Italian philosopher and polymath from the 15th century. The scientist was born in 1404 and was a prominent figure of the renaissance, besides many outstanding works, his most significant one is the Trevi Fountain. He was the first one who thought about a system which replaces the monoalphabetic cryptology by using more than one Alphabet.

Unfortunately it was left unfinished, so others could be victorious. The first one was a German abbot Johannes Trithemius, born in 1462 then he was followed by the Italian scientist Giambattista della Porta, born in 1535 and finally a French diplomat, Blaise de Vigenére, who was brought forth in 1523.

Blaise de Vigenére

Vigenére got acquainted with the works of Alberti, Trithemius and Porta at the age of 26 during a two year long commission in Rome. At first his interest turned towards cryptography only for practical reasons and in connection with his tasks as a diplomat. Later, after leaving his career, he forged their thoughts to a brand new, unified and strong cryptosystem. The work of Blaise de Vigenére culminated in his thesis titled Traicté des

Chiffres (Discourse of cryptography) and published in 1586. However the system was quoted as “le chiffre indéchiffrable” (unbreakable code), it had been forgotten for a long time.

Let us see the following table.

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

Encrypt the following proverb, ”The ball is in your court”. Let the chosen key be the word MARS. Write the key periodically above the text to be encrypted.

M	A	R	S	M	A	R	S	M	A	R	S	R	S	M
N	E	M	M	I	N	D	A	R	A	N	Y	A	M	

Then the alternate of T, call it F, is going to be the Mth unit of the Tth row. The alternate of H is going to be the Ath unit of the Hth row, H. Serial repetition of these steps leads us to the encrypted text.

A similar square can be made, with the only difference that the order of the letters is the opposite. This one is called Beaufort square after its creator Rear - Admiral Sir Francis Beaufort. A wind speed measure also possesses the Admiral’s name.

The Vigenére system is a typical example of the encrypting method, when a keyword is repeated periodically and the encryption is based upon it. As being a polialphabetic system, obviously the long standing statistical

method cannot be used. However, if the length of the keyword is known, it can be reduced to a monoalphabetic system.

Suppose that the length of the keyword is known currently it is four characters long. Place the text to be encrypted into four columns in the following way:

	b	c	
	2	3	
	6	7	
	10	11	1
	⋮	⋮	

The numbers indicate the position of the letters in the encrypted text. The same letter in the same column represents identical letter from the original text. This means that if we had a good method to determine the length of the keyword, we could use the long standing statistical method after having made this arrangement.

Frideric Kasiski, a German cryptographer, developed a method in the 1860s which can help us to find out the length of the keyword. The Kassiski method, named after him, was published in 1863 and essentially it is no more than the examination of the repeating occurrence of the identical letter groups in the encrypted text. We observe the distance of these occurrences that is how many letters there are between them.

For example, suppose that a computer program hit upon the repetition of letter group RUNS. The occurrence of such letter group can be accidental, but the longer the group is that we can examine, the more possible it is that the sender encrypted an identical part of the text. If the occurrence of the specific follows the forthcoming pattern:

. . . RUNS 28 letters RUNS 44 letters RUNS 68 letters RUNS . . .

Then we assume that the length of the keyword equals the greatest common factor which is four in this case. If we examine the occurrence of more than one letter groups, it lies in our power to check our assumptions. If we are lucky these make the length of the keyword unambiguous. Otherwise it turns out only after the columns' division and the application of statistical methods, which variant is the correct.

It is also true, as it has been previously that the method is rather time-consuming without using computers, in our case we can obviously hit the target fast.

We note, it seems that independently from Kasiski , Charles Babbage had also materialized this idea back in 1846.

Charles Babbage

3. Autoclave system

The autoclave system is an encrypted method of the Vigenére method that was invented by the famous mathematician Gerolamo Cardano (1501-1576). In this system we use the source text as the key with the help of a certain shift in the text.

Gerolamo Cardano

Let the measure of the shift be 4 letters and encrypt the well known Latin proverb: ”VERITASVINCIT”, then the encrypted text is the following:

Source: $VERITASVINCIT$

Key: $TASVINCITVERI$

Encrypted text: $OEJDBNUDBIGZB$

The use of the key is the same as in the Vigenére system. The remaining part can be filled with the end of the source text as we have just seen, or we may figure out a suitable keyword. At present, the name JACK is an appropriate choice, so we can define the encrypted text.

Source: $VERITASVINCIT$

Key: $JACKVERITASVI$

Encrypted text: $EETSOEJDBNUDB$

The legal decoder obviously has an easy job, as by knowing the keyword he also gets the first few letters of the original text, which mean the further decrypting key.

Another variation may also be used. We choose a key for encryption again, but contrary to the other method, it is not the source text that gives the key but the letters of the encrypted text.

Source: $VERITASVINCIT$

Key: $JACKSEETLEWOT$

Encrypted text: $SYUMWPULDTVQ$

The main aim of the illegal decoder is to determine the length of the key. The previously detailed Kasiki method provides an opportunity to find out the length of the keyword here as well. However, we may notice that the method is not as strong as it was in the previous cases, because the possibility that a specific letter group encrypt the same group is only acceptable in sufficiently long texts.

The original method also requires the cognition of the keyword. We choose an optional letter with the help of a frequency chart (there are 25 possibilities). This letter together with the first letter of the encrypted text determines the first letter of the source text. As we had used the letters of the source text for encryption, we were able to find a new letter of the key.

In our original example, where the key contained 4 letters, we may find the 5th letter of the key. Continuing this process we may also determine the letters of the source text in positions $9, 13, 17, \ldots$. If the frequency of these letters is contradictory to the results, we try a new letter. The determination of the remaining letters of the keyword follows this pattern.

In the first chapter we summarized some old encrypting methods. We could observe that our primal help is the examination of the statistical occurrence of the letters. Therefore the decoder of the encrypted text must have accurate knowledge of the given language that has been encrypted. Obviously the senders figure out all kinds of methods to make the job of the illegal decoders harder. One of the most popular tricks is that the text is translated from the certain well known language to a rare, statistically unmapped language. Here the main motto of cryptography gains its importance: ”Never underestimate the coder.” With this remark we have floundered to an area beyond cryptography which is called the world of politics, intelligence and conspiracy and this would lead us far from our interest.

4. Exercises

1.

Encrypt the world ”probability” with the Playfair method, introduced above.

2.

Use the Vigenére system to encrypt the English proverb ”All roads lead to Rome”. Use the word ”versa” as the key.

3.

Use the Autoclave system to encrypt the name of its creator Gerolamo Cardano. Let the keyword be the word ”math”.

4.

Repeat the previous encryption in a way that after using the keyword let the encrypted text to be the keyword.

5.

Using the Playfair method and the word ”playfire” as the key decrypt the following text: ”ypvieirddnizspyvtsarlypxneztftftnevyajykrpdv”

Chapter 4. Mathematical Preliminaries

1. Divisibility

Next we discuss the mathematical foundation indispensable for understanding the upcoming chapters. Here we do not introduce elliptic curves, this will be done separately.

Definition 4.1. We say that b natural number is *divisible* by natural number a if there exists a natural number x such that $b = ax$.

For divisibility we use the $a \mid b$ notation. In case a is not divisible by b we use $a \nmid b$.

Here we mention a few important properties of divisibility.

Theorem 4.2.

1. $a|b$ implies $a|bc$ for all integer c;
2. $a|b$ and $b|c$ imply $a|c$;
3. $a|b$ and $a|c$ imply that

   $a|(bx + cy)$

   for all integers x, y
4. if $m \neq 0$, then $a|b$ and $ma|mb$ are equivalent

Theorem 4.3 (Division with remainder property).

For arbitrary $a > 0$ and b integers there exist unique q and r integers such that

$$b = aq + r, \quad 0 \leq r < a.$$

Definition 4.4. The *greatest common divisor* of a and b (at least one of them is nonzero) is the greatest element of the set of their common divisors and it is denoted by (a, b).

Theorem 4.5. If g is the greatest common divisor of integers b and c, then there exist x_0 and y_0 integers such that

$$g = (b, c) = bx_0 + cy_0.$$

Theorem 4.6. $g = (a, b)$ can be characterized in the following two different ways:

1. g is the smallest positive value of the form $ax + by$, x and y arbitrary integers

2.

g is a common divisor of a and b such that it can be divided by all common divisors of a and b

Theorem 4.7. For all positive integer m

$$(ma, mb) = m(a, b).$$

Theorem 4.8. If $d|a$, $d|b$ and $d > 0$, then

$$\left(\frac{a}{d}, \frac{b}{d}\right) = \frac{1}{d}(a, b).$$

If $(a, b) = g$, then

$$\left(\frac{a}{g}, \frac{b}{g}\right) = 1.$$

Definition 4.9. We say that a and b are relative primes if $(a, b) = 1$.

Theorem 4.10. For all x

$$(a, b) = (a, b + ax).$$

After introducing these basic properties we give a theorem for determining the greatest common divisor. It is named after the ancient Greek mathematician Euclid.

Euclid

Euclid's famous textbook, The Elements, is said to be the second most printed work after The Bible. However, the following algorithm is likely to be a a result obtained by mathematicians before Euclid, so it is not his own.

Theorem 4.11 (Euclid's Algorithm).

We apply the division with remainder property to given integers b and $c > 0$, thus we get the following sequence of equations:

$$\begin{aligned}
b &= cq_1 + r_1, \quad 0 < r_1 < c, \\
c &= r_1 q_2 + r_2, \quad 0 < r_2 < r_1, \\
r_1 &= r_2 q_3 + r_3, \quad 0 < r_3 < r_2, \\
&\vdots \\
r_{j-2} &= r_{j-1} q_j + r_j, \quad 0 < r_j < r_{j-1}, \\
r_{j-1} &= r_j q_{j+1}.
\end{aligned}$$

The greatest common divisor (b, c) of numbers b and c számok (b, c) is r_j, the last nonzero remainder of the division algorithm.

2. Primes

Primes, jus like atoms in the material world, play a very important role in number theory and in cryptography as well.

Definition 4.12. An integer number $p > 1$ is called a *prime* if p does not have a divisor d such that $1 < d < p$. If an integer is not a prime then it is called a *composite number*.

Theorem 4.13 (Fundamental Theory of Arithmetic, Gauss 1801.). Any integer integer number $n > 1$ can be written as a unique product (up to ordering of the factors) of prime numbers.

This theorem is from Carl Friedrich Gauss (1777-1855) who is often called "the Prince of Mathematics".

Carl Friedrich Gauss

His outstanding talent became obvious early in his childhood, there are many anecdotes on the young Gauss. The Disquisitiones Arithmeticae, written at the age of 24, is a foundational work of number theory and it contains the above theorem.

Remarks on factorization

Next we show that for an arbitrary composite number its smallest factor is smaller than $\sqrt{n}$. Let

$$n = fa \quad \text{and} \quad f \leq a.$$

In this case

$$a = \frac{n}{f} \Rightarrow f \leq \frac{n}{f} \Rightarrow f^2 < n$$
$$f < \sqrt{n}.$$

The previous result makes an interesting thought experiment possible. This indicates the mysterious properties of primes and their applicability in cryptography.

For a number with 100 digits

$$n > 10^{100} \Rightarrow \sqrt{n} > 10^{50}$$

For simplicity we assume that our computer performs 10^{10} steps per second. This is can be considered to be a good approximation of today's available computational power. Then 10^{40} seconds, approx. 10^{31} years are needed to find the smallest prime factor with exhaustive search. In order to get the feeling how much time this is it is enough to know that the estimated age of the universe is $2 \cdot 10^{11}$ years.

Since the number of primes and their distribution is very important for cryptographical applicability we need to study a bit more number theory.

Theorem 4.14 (Euclid). The number of primes is infinite.

Theorem 4.15. In the sequence of primes there are arbitrary big gaps, i.e. for arbitrary positive integer k there exist k consecutive composite numbers.

Georg Friedrich Bernhard Riemann (1826-1866) was an excellent mathematician who died at a very young age.

Georg Friedrich Bernhard Riemann

He made extraordinary contributions to analysis, differential geometry, and analytic number theory. His conjecture (Riemann conjecture) is one of the seven Millenium Problems. The Clay Institute of Mathematics founded a million-dollar prize for solving any of these problems. Riemann gave this definition in his work on the behavior of prime numbers.

Definition 4.16. Let $\pi(x)$ denote for all real x the number of primes not greater than x.

Pafnuty Lvovich Chebyshev (1821-1894) Russian mathematician succeeded to prove that between any natural number and its double there exists a prime number. The following theorem is from his work in number theory.

Pafnuty Lvovich Chebyshev

Theorem 4.17 (Chebyshev). There exist a and b positive constants such that

$$a\frac{x}{\log x} < \pi(x) < b\frac{x}{\log x}.$$

The most famous mathematical problem of 19th century was the Prime Number Theorem. It was solved independently by Jacques Hadamard and de la Vallée Poussin in 1896.

Jacques Hadamard

de la Vallée Poussin

Theorem 4.18 (Prime Number Theorem 1896.).

$$\lim_{x \to \infty} \frac{\pi(x) \log x}{x} = 1.$$

Next we mention some interesting properties of primes and some classical problems.

Theorem 4.19. All prime numbers p can be given as the sum of four square numbers.

Theorem 4.20. Given an $f(x)$ polynomial, there are infinitely many positive m for which $f(m)$ is composite.

As we will see later finding primes, in case of big numbers, is not easy. It was always a dream for mathematicians to construct an expression that will produce prime numbers given some parameters. We mention two such attempts that are historically important.

Definition 4.21. We call the numbers of the form $M(n) = 2^n - 1$ *Mersenne-numbers*, where n is a nonnegative integer.

Marin Mersenne (1588-1648) was a French theologian, mathematician and physicist.

Marin Mersenne

It is worth noting that he attended the same Jesuit college where later René Descartes was also a student. We call *Mersenne-primes* those Mersenne-numbers with prime exponent n.

In order to justify the appearance of Mersenne-numbers it is worth taking a small detour into the realm of perfect numbers. If a number is the sum of its divisors (not including itself) then it is called a *perfect number*. For instance 6 is a perfect number since $6 = 1 + 2 + 3$.

Euclid recognized that the first 4 perfect numbers are of the form

$$2^{n-1}(2^n - 1)$$

where $2^n - 1$ is a prime. In these cases $n = 2, 3, 5, 7$. The conjecture that all perfect numbers have this form was proved by Leonhard Euler some 1500 years later.

Leonhard Euler

In Mersenne's Cogitata Physica-Mathematica (1644) he wrote the false statement that for $n = 2, 3, 5, 7, 13, 17, 19, 31, 67, 127, 257$ we get prime numbers, but for $n < 257$ we get composite numbers. Later Leonhard Euler (1707-1783) Swiss mathematician showed that $n = 31$ indeed produces a prime. This number was for more than one hundred years the greatest known prime. Later it turned out the following list is correct: $n = 2, 3, 5, 7, 13, 17, 19, 31, 61, 89, 107, 127$.

Up to now 47 Mersenne-primes were found. The last one was found in April 2009, where $n = 42643801$ and the number consists of 12837064 digits. There is a world-wide collaboration involving many computers for finding further Mersenne-primes.

(For further details please visit: http://www.mersenne.org).

Further interesting numbers are the Fermat-numbers.

Definition 4.22. Primes of the form $F(n) = 2^{2^n} + 1$, where n is a nonnegative integer, are called Fermat-primes.

Pierre de Fermat (1601-1665), French lawyer, did mathematics as a pastime activity with considerable result.

Pierre de Fermat

The above problem is interesting enough but he is famous for these lines: “it is impossible to separate a cube into two cubes, or a fourth power into two fourth powers, or in general, any power higher than the second, into two like powers. I have discovered a truly marvelous proof of this, which this margin is too narrow to contain.” This short proof is still sought-after, but in 1995 Princeton Professor Andrew Wiles proved the conjecture, on more than 100 pages.

Fermat did not put emphasis on proofs, so his conjecture that numbers of the form $2^{2^n}+1$ are always primes, remained only a conjecture. In fact Euler in 1732 showed that 641 is a divisor of F_5.

There are many open questions in this field. We still do not know whether there are infinitely many Mersenne-primes and Fermat-primes or not. Is there any odd perfect number?

3. Congruences

The theory of congruences in its present form was worked out by Carl Friedrich Gauss in his Disquisitiones Arithmeticae.

Definition 4.23. If a nonzero integer m divides the difference $a-b$, then a and b are congruent *congruent* modulo m. Notation: $a \equiv b \pmod{m}$.

Theorem 4.24. Let a, b, c, d, x and y integer numbers.

(a) If $a \equiv b \pmod{m}$ and $b \equiv c \pmod{m}$, then $a \equiv c \pmod{m}$.

(b) If $a \equiv b \pmod{m}$ and $c \equiv d \pmod{m}$, then $ax+cy \equiv bx+dy \pmod{m}$.

(c) If $a \equiv b \pmod{m}$ and $c \equiv d \pmod{m}$, then $ac \equiv bd \pmod{m}$.

Theorem 4.25. Let f be a polynomial with integer coefficients. If $a \equiv b \pmod{m}$, then $f(a) \equiv f(b) \pmod{m}$.

Theorem 4.26. $ax \equiv ay \pmod{m}$ if and only if $x \equiv y \pmod{\frac{m}{(a,m)}}$.

Theorem 4.27. If $ax \equiv ay \pmod{m}$ and $(a,m)=1$ then $x \equiv y \pmod{m}$.

Definition 4.28. If $x \equiv y \pmod{m}$, then we call y the remainder of x dividing by m. The $x_1, \ldots, x_m$ set of numbers form a *complete remainder system* modulo m, if for arbitrary integer y there exists exactly one x_j such that $y \equiv x_j \pmod{m}$.

Definition 4.29. The r set of integer numbers is a reduced remainder system modulo m, if $(r_i, m) = 1$; $r_i \not\equiv r_j \pmod{m}$ $i \neq j$, where $x \equiv r_i \pmod{m}$ and for arbitrary m and for x relative prime we can find an from the set such that .

Notation: All reduced remainder systems $\pmod{m}$ contains the same number of elements. This number is denoted by $\phi(m)$ and we call it the *Euler's* ϕ function.

Theorem 4.30. $\phi(m)$ is the number of those positive integers that are not greater than m and are relative primes to m.

Theorem 4.31 (Euler). If $(a, m) = 1$, then

$$a^{\phi(m)} \equiv 1 \pmod{m}.$$

Theorem 4.32 (Fermat). Let p be prime and assume that $(a, p) = 1$. Then

$$a^{p-1} \equiv 1 \pmod{p}.$$

Theorem 4.33. Let $g = (a, m)$. If $g \nmid b$, then the $ax \equiv b \pmod{m}$ congruence does not have any solution. However if $g \mid b$, then the congruence has g solutions. These solutions are the values

$$x \equiv \frac{b}{g} x_0 + t\frac{m}{g} \pmod{m},\ t = 0, 1, \ldots g - 1$$

where x_0 is an arbitrary solution of

$$\frac{a}{g} x \equiv 1 \pmod{\frac{m}{g}}.$$

Example 4.34. Let's solve the following linear congruence!

$$15x \equiv 25 \pmod{35}$$

Since $(15, 35) = 5$ and $5 \mid 25$ the congruence can be solved.

$$3x \equiv 1 \pmod{7} \Rightarrow x_0 = 5$$

Solution: $x \equiv 25 + 7t \pmod{35}$ and $t = 0, 1, 2, 3, 4$.

The next statement is about simultaneous congruence systems consisting of more than one congruence system. This was known by a Chinese mathematician named Sun Tzu more than 2000 years ago, hence the name of the theorem.

Theorem 4.35 (Chinese Remainder Theorem). If $m_1, m_2, \ldots, m_r$ are pairwise relative prime positive integers and $a_1, a_2, \ldots, a_r$ are arbitrary integers, then the congruences

$$x \equiv a_i \pmod{m_i}\ i = 1, 2, \ldots r$$

have a common solution. In this case any two solutions are congruent modulo $m_1 m_2 \cdots m_r$.

Method: $m = m_1 m_2 \cdots m_r$

$$\frac{m}{m_j} b_j \equiv 1 \pmod{m_j}$$

$$x_0 = \sum_{j=1}^{r} \frac{m}{m_j} b_j a_j$$

Example 4.36. Choose a number smaller than 60, divide it by 3,4 and 5 and give the remainders.

The chosen number is the remainder of $40a + 45b + 36c$ divided by 60, assuming that the corresponding remainder values are a, b, c . Choosing 29 we get $40 \cdot 2 + 45 \cdot 1 + 36 \cdot 4 = 269$.

Solution.

$20b_1 \equiv 1 \pmod 3$

$15b_2 \equiv 1 \pmod 4$

$12b_3 \equiv 1 \pmod 5$

Then $b_1 = 2, b_2 = 3, b_3 = 3$.

$x_0 = 20 \cdot 2 \cdot 2 + 15 \cdot 3 \cdot 1 + 12 \cdot 3 \cdot 4 \pmod{60} = 29$

4. Finite fields

The work of Evariste Galois (1811–1832) was the starting point for the development of the theory of finite fields. In the recent years it is strongly developed and it became a very important part for code theory or cryptography.

Now we give a short introduction to the theory of finite fields.

Definition 4.37. A group is a set G, together with an operation $\star$ (called the group law) that combines two elements and the operation must satisfy three requirements

1. The operation $\star$ is associative,
2. The set G has an identity element, that is there exists an element e, such that for every element a of G satisfy the equation $e * a = a * e = a$,
3. For each element a of G there exists an element b of G such that $a * b = b * a = e$. The element b is the inverse element of a, it is denoted by a^{-1}.

Definition 4.38. If the group operation is commutative, that is for every a, b elements of G the equation $a * b = b * a$ is satisfied, then the group is called commutative group or abelian group.

Definition 4.39. A multiplicative group G is called cyclic group, if there exists an element of $a \in G$, such that for every elements of $b \in G$ there exists an integer j, such that $b = a^j$. Such elements a of cylic group is called generator.

Definition 4.40. Let $+$ and $.$ be binary operations on a set R, they are called addition and multiplication. A set R is called a ring, if the two operations satisfy the following requirements, known as the ring axioms,

1. $(R; +)$ is an abelian group,
2.

distributive $(a+b)\cdot c = a\cdot c + b\cdot c$ is satisfied, that is $(a+b)\cdot c = a\cdot c + b\cdot c$ and for all $a, b, c \in R$,

3. For all a, b and c elements of R the equation $(a\cdot b)\cdot c = a\cdot(b\cdot c)$ holds, that is the multiplication is associative.

Definition 4.41. Rings which do have multiplicative identities, (and thus satisfy all of the axioms above) is called unital rings. The multiplicative identies is denoted by 1 and the equation $a\cdot 1 = 1\cdot a = a$ is satisfied for all $a \in R$.

Definition 4.42. A ring R is called commutative ring, if for all a and b the equation $a\cdot b = b\cdot a$ is satisfied, that is the multiplication is commutative.

Definition 4.43. A nonzero element a of a ring R is called a left zero divisor, if there is a nonzero element b, such that $ab = 0$. Similarly can be defined the notation of right zero divisor. A ring R which has no left or right zero divisor is called a domain.

Definition 4.44. A commutative domain with a multiplicative identity is called integral domain.

Definition 4.45. A field is a commutative ring whose nonzero elements form a group under multiplication.

Theorem 4.46. All finite integral domain F is a field.

Theorem 4.47. $\mathbb{Z}_n$ is a field if and only if n is a prime.

For example Z_2, Z_3 and Z_5 are finite fields, but Z_9 is not, since there is no multiplicative inverse of residue class 3 in Z_9. The finite fields with p^n elements are denoted by $\mathbb{GF}(p^n)$, where the notation $\mathbb{GF}$ is in honor of Évariste Galois.

Theorem 4.48. For all prime p and natural number n there exist a finite field with $q = p^n$ elements.

Definition 4.49. The characteristic of a field F is the smallest natural number m such that

$$ma = \sum_{i=1}^{m} a = a + a + \cdots + a = 0,$$

for all elements $a \in F$ (this is the addition in the field). If there is no such a number m, then the characteristic of the field is 0.

We remark that the characteristic of the ring $(\mathbb{Z}_3, +, \cdot)$ is 3, the characteristic of the ring $(\mathbb{Z}_4, +, \cdot)$ is 4 and the characteristic of the ring $(\mathbb{Z}_n, +, \cdot)$ is n. The characteristic of the rings $(\mathbb{Z}, +, \cdot)$ and $(\mathbb{Q}, +, \cdot)$ is 0.

Easy to see that if F is a field with characteristic p, then there is a subfield $\mathbb{GF}(p)$ in the field F with p elements, the elements

$$1, 1+1, \ldots, \sum_{i=1}^{m} a = 0$$

are in the subfield. These elements are different, this set is closed under the multiplication and addition, for all elements have additive inverse and all elements except zero have multiplicative inverse. There exists isomorphism between the subfield with p elements and Z_p, this way we can say that every finite fields with characteristic p is Z_p. The field $\mathbb{GF}(p)$ with characteristic p is the prime field of the finite field F.

F^* is denoted the elements of the field F except 0.

Definition 4.50. An element α of F^* is called primitive, if all elements of the field F except 0 can be written uniquely as positive integer powers of α.

Theorem 4.51. Let F be a finite filed with q elements, then for all element α the equation $\alpha^q = \alpha$ is satisfied, so all elements of the field F is a root the polynomial $f(x) = x^q - x$.

The structure of finite fields can ve seen in the following theorem.

Theorem 4.52. The multiplicative group of $\mathbb{GF}(q)$ is a cyclic group.

Theorem 4.53. There is a primitive element of all fields $F = \mathbb{GF}(q)$.

Theorem 4.54. All finite field is a vector space over Z_p, if the vector space is an n-dimensional vector space, then the number of the elements of the field is p^n, where p is a prime.

Definition 4.55. Let F be a finite field, the number of elements is called the order of the field.

5. Exercises

1.

Solve the following congruences $25x \equiv 15 \pmod{29}$ and $5x \equiv 2 \pmod{26}$.

2.

If we break eggs from a basket 2,3,4,5,6 by, it remains in turn 1,2,3,4,5, eggs. But if we removed the eggs 7 by, none remains in the basket. Give the number of the eggs in the basket. (Brahmagupta i.sz. VII.sz.)

3.

Find the residue of 4444^{4444} dividing by 9.

4.

Solve the following diophantine equations using congruence $4x + 51y = 9$, and $5x - 53y = 17$.

5.

Determine the greatest common divisor of 12543 and 29447.

6.

We have a 12 and 51 liter barrel. Can we fill up with these barrels a 5211 liter tank, if we always fill up the barrels and we have to spill into the tank the whole contents of the barrels, and if the water don't brim over the tank.

7.

Prove that for all integer a, b we have $(a^2, b^2) = (a, b)^2$.

8.

Calculate the greatest common divisor of $a = 255$ and $b = 111$ using the Euclid's algorithm, further determine the coefficients x and y for $g = ax + by$.

9.

Is it a complete residue system or not $7, 22, 37, 52, \ldots, 11632, 11647 \pmod{777}$?

10.

Reduced residue system or not $5, 15, 25, 35, 45, 55, \ldots, 155 \pmod{32}$?

11.

Solve the following linear congruences system using the Chinese remainder theorem

$$5x \equiv 1 \pmod{7}$$
$$4x \equiv 1 \pmod{9}$$
$$8x \equiv 1 \pmod{13}$$

Chapter 5. DES

Wholly until the 2000 the DES (Data Encryption Standard) had been the most widely used algorithm in cryptography.

IBM began to run a research project at the end of the 1960s to develop a symmetrical encryption system that uses secret keys. With the lead of Horst Feistal an algorithm had been developed by 1971 which was called LUCIFER at that time. It divided the open text to 128 bites blocks and also used a 128 bites long key for encryption.

LUCIFER was sold to the Lloyd Insurance Company in London that used it with a cash - distributing system also developed by IBM. Carl Meyer and Walter Tuchman wanted to implement the hardware, operating the LUCIFER algorithm, on one single chip for what they modified the algorithm a little bit.

Around the middle of the 1970 the NSA (National Security Agency) announced a competition to create an algorithm that can be standardized. For this competition presented Carl Meyer and Walter Tuchman from IBM their invented method which had been far the best from all the other tenders, henceforth it was standardized as DES in 1977.

The system suited the tremendously developing word of data processing well. It provided a high level of security yet that came from a simple structure. The hardware solutions are better than by software as DES handles a huge amount of operations on the level of bites.

The algorithm possesses the so-called avalanche effect which means that a minor change in input means a major difference in the output.

1. Feistel cipher

The DES algorithm can be appointed with the name Feistel cipher as well. This algorithm is a 64 bites block algorithm that is a 64 bites encrypted block is assigned to a 64 bites block of the open text. The assignment only depends on the key in use.

Every step uses the result of the previous step namely in an identical way but depending on the key. This step is called a round and the parameter of the algorithm is the number of these rounds.

Let t be the length of the block. Let f_K be the coding function of K key, which is called subkey and do not have to be invertible. Fix a number $r \geq 3$ (in case of Feistel cipher this is an even number) to the sequence, the $\mathcal{K}$ key space and a method so that we may generate a $K_1, \ldots, K_r$ key-sequence to any $k \in \mathcal{K}$ keys.

The coding function E_k operates in the following way.Let p be the $2t$ long part of the open text space. Cut it two t long parts, that is $p = (L_0, R_0)$, where L_0 is the left, R_0 is the right side. Then the sequence

$$(L_i, R_i) = (R_{i-1}, L_{i-1} \oplus f_{K_i}(R_{i-1})), \quad 1 \leq i \leq r$$

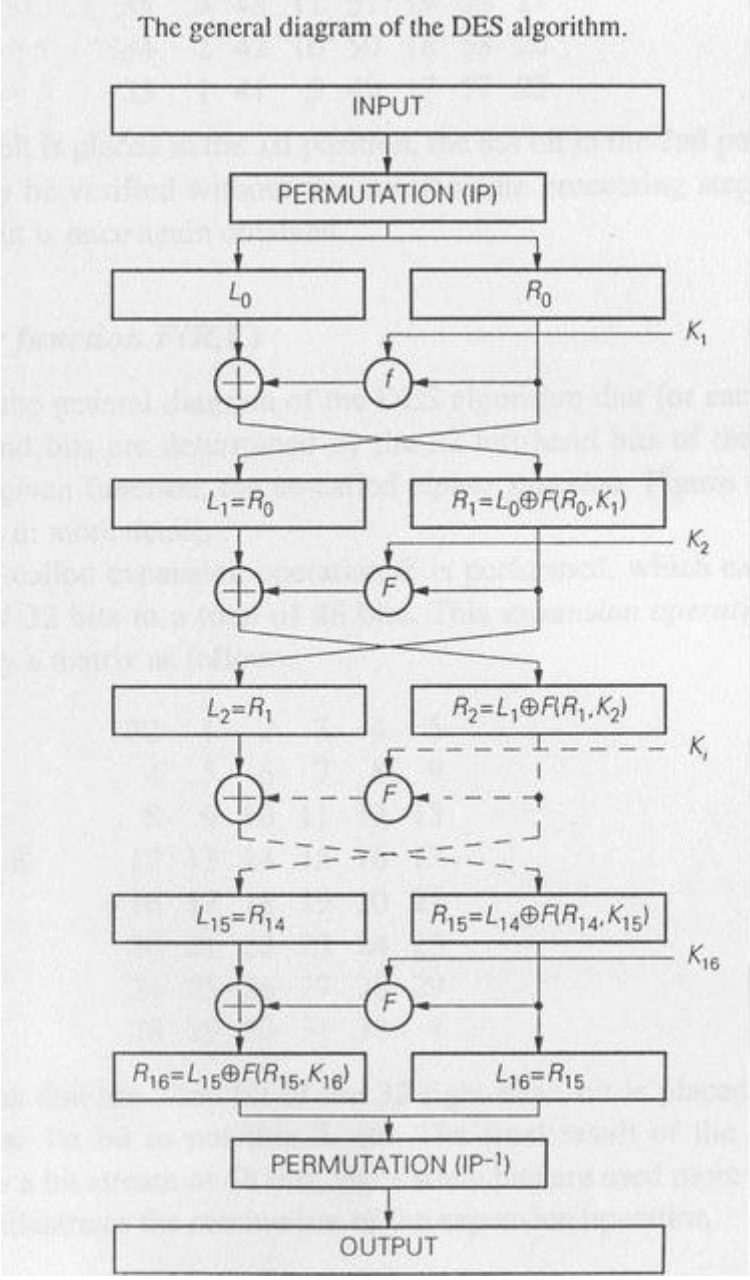
comes in this way and

$$E_k(L_0, R_0) = (R_r, L_r).$$

The $\oplus$ operation that has been applied means the usual XOR operation. The level of security may be heightened by rising the number of circles. The decoding process is the following:

$$(R_{i-1}, L_{i-1}) = (L_i, R_i \oplus f_{K_i}(L_i)), \quad 1 \leq i \leq r.$$

Using this r times with the $K_r, \ldots, K_1$ key-sequence we get back the (L_0, R_0) original text from (R_r, L_r).



2. The DES algorithm

The key-size of DES is 64 bites every eight is not included in use. The left bites are used for checking so the real key-size is 56 bites. The number of DES keys is $2^{56} \approx 7.2 \cdot 1016$.

For example a valid DES key in hexadecimal form could be the following

$$133457799BBCDFF1$$

or in binary exposition as the next chart presents:

0	1	0	0
1	1	0	1
0	1	0	1
1	1	1	0
0	1	1	0
1	1	1	1
0	1	1	1
1	1	0	0

In the first step we mix the bites of the input while in the last step we apply the invert of this process. In DES we call it Initial Permutation (IP).

This is a bit-permutation for 64 bites vectors that is it is independent from the chosen key. The IP and its inverse can be seen in the chart underneath. The interpretation of the chart is the following: If $p \in \{0,1\}^{64}$, $p = p_1 p_2 p_3 \dots p_{64}$, then $\mathrm{IP}(p) = p_{58} p_{50} p_{42} \dots p_7$.

IP								IP^{-1}							
58	50	42	34	26	18	10	2	40	8	48	16	56	24	64	32
60	52	44	36	28	20	12	4	39	7	47	15	55	23	63	31
62	54	46	38	30	22	14	6	38	6	46	14	54	22	62	30
64	56	48	40	32	24	16	8	37	5	45	13	53	21	61	29
57	49	41	33	25	17	9	1	36	4	44	12	52	20	60	28
59	51	43	35	27	19	11	3	35	3	43	11	51	19	59	27
61	53	45	37	29	21	13	5	34	2	42	10	50	18	58	26
63	55	47	39	31	23	15	7	33	1	41	9	49	17	57	25

Then we apply 16 round Feistel coding for the permutated open text. Finally the invert of IP provides the encrypted text that is:

$c = \mathrm{IP}^{-1}(R_{16}, L_{16})$.

3. Coding the inner block

The alphabet is $\{0,1\}$, the length of the block is 32 and the key-space is $\{0,1\}^{48}$. Then we use a coding function $f_K\colon \{0,1\}^{32} \to \{0,1\}^{32}$ with a key $K \in \{0,1\}^{48}$. The $R \in \{0,1\}^{32}$ part is expanded with a function $\mathrm{E}\colon \{0,1\}^{32} \to \{0,1\}^{48}$. This function is stated on the following illustration.

E					R		
1	2	3	4	5	16	7	10
5	6	7	8	9	29	12	28
9	10	11	12	13	1	15	23
13	14	15	16	17	5	18	31
17	18	19	20	21	2	8	24
21	22	23	24	25	32	27	3
25	26	27	28	29	19	13	30
29	30	31	32	1	22	11	4

If $R = R_1R_2R_3\ldots R_{32}$, then $\mathrm{E}(R) = R_{32}R_1R_2\ldots R_{32}R_1$. The next step is the reckoning of $\mathrm{E}(R) \oplus K$ and the division of result to 8 blocks. Let them be signed by B_i ($1 \leq i \leq 8$). The length of these is 6 that is

$$\mathrm{E}(R) \oplus K = B_1B_2B_3B_4B_5B_6B_7B_8.$$

Hereinafter we use S_i

$$S_i\colon \{0,1\}^6 \to \{0,1\}^4, \quad 1 \leq i \leq 8$$

functions (or S-boxes as they are called sometimes).

With the help of these functions we get the string

$$C = C_1C_2C_3C_4C_5C_6C_7C_8,$$

where $C_i = S_i(B_i)$. The length of these is 32. We also apply the P permutation on them so we get the $f_K(R)$.

4. S-boxes

The heart of the DES algorithm is composed of these boxes as they are (very) unlinear. Every single S-box can be represented by a table that consists of 4 rows and 16 columns. In case of every $B_i = b_1b_2b_3b_4b_5b_6$ string $S_i(B_i)$ can be calculated in the following way.

That integer, which binary form is b_1b_6, will be the row-index. The integer that stands for the $b_2b_3b_4b_5$ binary number will be used as the index of the column. We look for the appropriate value, state its binary form and if necessary we may add some extra 0 for the length to be 4. Hereby we get $S_i(B_i)$.

For example determine $S_1(001011)$. Then 01 marks the row-index and 0101 provides the column-index. These just mean the $S_1(001011) = 0010$ and 5 integers. In the first S_1-box the proper cell value is 2 which binary form is 10 that is due to the length 4 .

5. Keys

The final part is about how to generate keys. Let $k \in \{0,1\}^{64}$ be a DES key. From this we generate the K_i, $1 \leq i \leq 16$ which are 48 long. We define the ν_i value in the following way:

$$\nu_i = \begin{cases} 1, & \text{if } i \in \{1,2,9,16\} \\ 2, & \text{if } i \notin \{1,2,9,16\}. \end{cases}$$

The next algorithm and functions provide the key:

$$\mathrm{PC1}\colon \{0,1\}^{64} \to \{0,1\}^{28} \times \{0,1\}^{28},$$
$$\mathrm{PC2}\colon \{0,1\}^{28} \times \{0,1\}^{28} \to \{0,1\}^{48},$$

where the above functions are set according to the followings:

PC1						
57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

PC2					
14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

The algorithm:

1. Let $(C_0, D_0) = \mathrm{PC1}(k)$.
2. Suppose that for every $1 \leq i \leq 16$
   a. Let C_i be the string, that comes from C_{i-1} by a cyclical shift to the left position ν_i.
   b. Let D_i the string, that comes from D_{i-1} by a cyclical shift to the left with position ν_i.
   c. Let us define $K_i = \mathrm{PC2}(C_i, D_i)$.

The PC1 function C and D, gives back two 28 long strings from the 64 long key. This can be seen well from the table. For example if $PC1(b_1 b_2 \ldots b_{64}) = b_{57} b_{49} \ldots b_4$ then $PC1(0,1,0,\ldots,1,0) = (1,1,1,\ldots,0,0)$. The PC2 function provides a 48 long string from a pair. For example $PC2(b_1 b_2 \ldots b_{56}) = b_{14} b_{17} \ldots b_{32}$.

We can decrypt a text that has been encrypted on the bases of DES by applying a reverse key-sequence coding process.

Let us see the steps in the next figures.

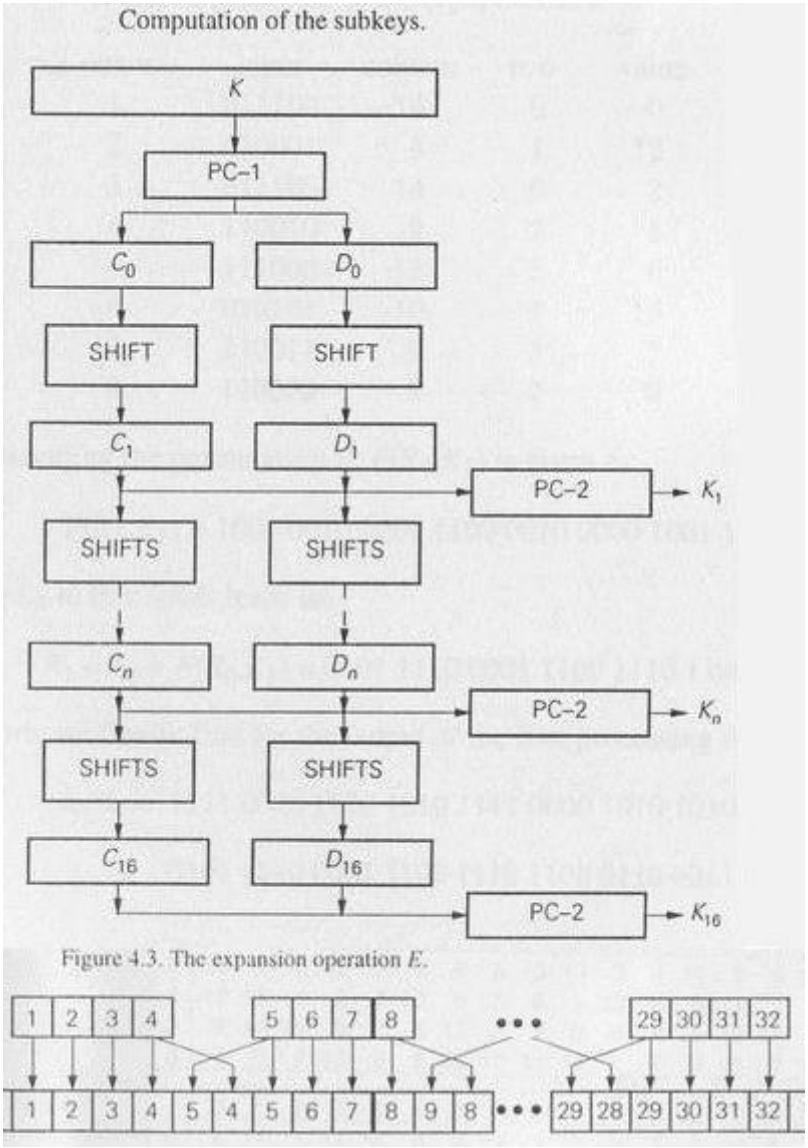
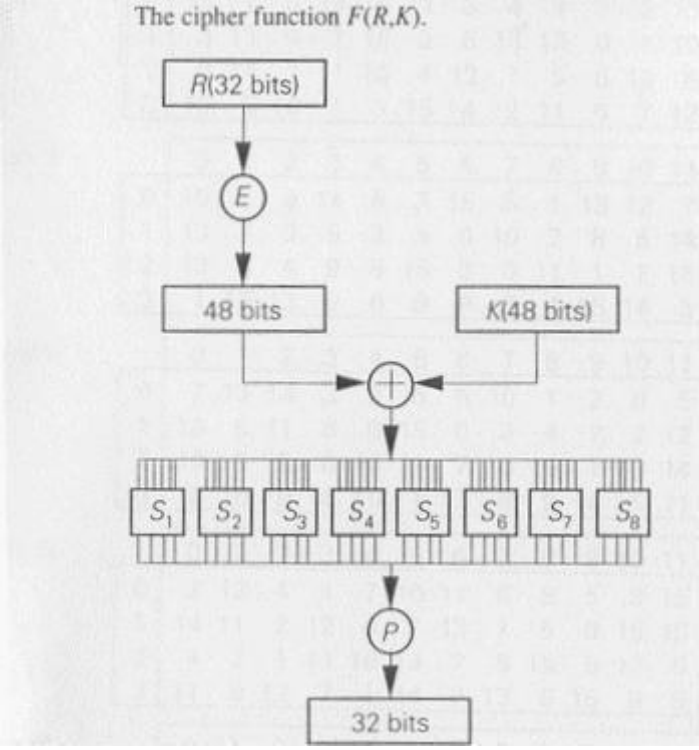


Figure 4.3. The expansion operation E.



6. Example for DES

Create the encryption of the $p = 0123456789ABCDEF$ open text with the help of DES. Its binary form is:

0	0	0	0
1	0	0	0
0	0	0	1
1	0	0	1
0	0	1	0
1	0	1	0
0	0	1	1
1	0	1	1

Apply the previously seen IP permutation to gain

0	0	1	1
0	0	0	0
0	0	1	1
1	1	1	1
1	1	0	0
1	0	1	0
1	1	0	0
1	0	1	0

that is

$L_0 = 11001100000000001100110011111111,$
$R_0 = 11110000101010101111000010101010.$

Use the DES key that became acquainted previously that is in our case $133457799BBCDFF1$ which binary form is

0	1	0	0
1	1	0	1
0	1	0	1
1	1	1	0
0	1	1	0
1	1	1	1
0	1	1	1
1	1	0	0

Let us figure the first key:

$C_0 = 11110000110011001010101111,$
$D_0 = 01010101011001100111100011111,$
$C_1 = 11100001100110010101011111,$
$D_1 = 10101010110011001111100011110.$

We get the key

$K_1 = 00011011000000101110111111111100011100000011100 10.$

Using the previous result we have

$\mathrm{E}(R_0) \oplus K_1 = 01100001000101111011101010000110011001010010011 1,$

and

$f_{K_1}(R_0) = 00000011010010111010100110111011,$

finally

$R_1 = 11001111010010110110010101000100.$

The remaining turns can be calculated similarly.

Our program DES.exe will show you every details in the DES method.

7. The security of DES

Since it had been invented the security of DES was examined. The DES had been attacked by special techniques but no algorithm has been found until today that can break the system without possessing the key.

On the other hand, as the range of key scale is limited, with today's calculating capacity the system is powerless against the so-called brute-force attacks. In these cases we just simply check every possible key of decryption. We can harden decryption by repeating the algorithm in succession. These methods are called TripleDES or 3DES. With the first one we use three different keys while in the second one two are identical from the three applied keys.

As a regard of these it is essential to note that DES is not a group. This means if we have k_1 and k_2 keys we have no k_3 keys in a way that $\mathrm{DES}_{k_1} \circ \mathrm{DES}_{k_2} = \mathrm{DES}_{k_3}$.

Obviously otherwise the repetition of encrypting would not enlarge security. Finally we also have to mention that not only the extreme growth of calculating capacity works against the DES but also our growing knowledge of shared-systems. In cases where the certain task can be separated to parts, joint operation of computers lead to the result in a very period of time.

Chapter 6. AES crypto-system

The word of Information Technology has changed a lot since the announcement of DES in 1976. The network data traffic had grown so as the speed of computers and it was becoming obvious for professionals that it could not provide the level of security as it used to do.

Now in the beginning of the 21th century we reckon that the lifetime of a crypto-system is around 20 years and we also expect from this system to keep our secrets for another 10-50 years after its suspension.

In 1996 in the United States the National Institute for Standards and Technology had begun the preparations of a new cryptographic algorithm. The expectations were published in 1997 and the algorithm got the name AES (Advanced Encryption Standard).

The following expectations had been conceived of the new system:

1. be a symmetric key block algorithm,
2. use 128 long blocks,
3. work with 128–192–256 bites key–size,
4. be faster than 3DES and provide better security,
5. use the resources of the computer effectively,
6. be flexible in adapting the possibilities of different platforms.

Great companies and research groups took part in the challenge such as IBM, RSA laboratories, Nippon Telegraph and Telephone Corporation. The final winner was announced three conferences, several examinations and breaking attempts on 3rd October in 2000. The winner of the competition was the Rijndael symmetric key algorithm that originated its name after the creators Vincent Rijmen and Joan Daemen.

Vincent Rijmen

Joan Daemen

The algorithm fulfilled all the above mentioned expectations. it is worth mentioning that the algorithm is not under copyright laws. It is very stable and resists all kinds of present day attacks. The only way is to try all the possibilities or in other words brute-force attack.

1. Basics

The Rijndael method combines substitution and linear transformations. Its main advantage is that the creation of roundkeys is fast and they can be multi-processed that is obviously an or advantage as far as speed is concerned. The repeating circle functions consists of four independent transformations which from this point are called layers and defined hereby.

1.

   The linear mixing layer provides a high intense mixture of the boxes. The MixColumns layer (combined in the level of columns) while the ShiftRows operation can be combined on the level of rows.

2.

   The non linear layer uses only a single S-box and the SubBytes layer can be combined on the level of bytes.

3.

   The key addition layer makes the final result depend on the key. The method uses a simple XOR operation and in each round a different Roundkey. The AddRoundKey layer can be combined on the level of bytes. We also note that the other layers are independent from the key.

In case of AES 128, being examined by us, the round functions have to be repeated ten times. From the first one 9 rounds must be done, while only one from the second. The in- and output data is stored in a so-called State structure.

1. Round (State, Roundkey)
   a. SubBytes (State)
   b. ShiftRows(State)
   c. MixColumns(State)
   d. AddRoundKey(State, RoundKey)
2. FinalRound (State, RoundKey)
   a. SubBytes(State)
   b. ShiftRows(State)
   c. AddRoundKey(State, RoundKey)

The last round is a little bit different from the others as one layer is left out. In the brackets we can see when to use the key and when not.

In order to understand the algorithm some expressions may be needed. Word means 32 bits, block size (N_B) means the size of the blocks expressed by words in this case $N_B = 4$.. Key-size (N_K) stands for the size of the key also expressed by words, $N_K = 4$ again. The number of rounds depends on the block- and key size as well, in our case, as mentioned above, it means 10 rounds $(N_R = 10)$.

2. Layers of rounds

2.1. The State

The state structure can be illustrated by a 4x4 square where each square represents one byte.

,0	$s_{0,1}$	$s_{0,2}$	$s($
,0	$s_{1,1}$	$s_{1,2}$	s_1
,0	$s_{2,1}$	$s_{2,2}$	s_2
,0	$s_{3,1}$	$s_{3,2}$	s_3

When upload the state structure, the key and the encrypted document, the way to follow is from up to down from right to the left. The column vectors of the state structure may be considered as words.

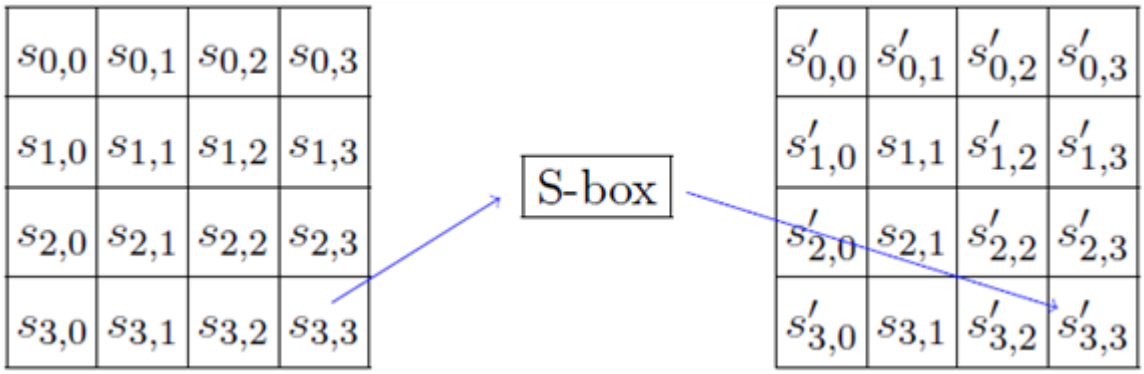
2.2. SubBytes transformation

The SubBytes transformation uses a non-linear invertible S-box, each byte is replaced with the same S-box. The following presents the rules of operations, where b_i means the ith bit of the given byte and c_i is the ith bit of the $C = 01100011$ binary digit, where $0 \leq i \leq 7$.

In the following equality operations are defined on the level of bits, where the numbering of bits is the usual right to left.

$$b'_i = b_i \oplus b_{(i+4) \pmod 8} \oplus b_{(i+5) \pmod 8} \oplus b_{(i+6) \pmod 8} \oplus b_{(i+7) \pmod 8} \oplus C_i$$

In every case the letter marked with a comma provides the varied value. The values can be calculated in advance, the used S-box can be found in hexadecimal form in the FIPS notice [4]. The process can be imagined this way:



The chosen letter pair is in one row. Obviously the InvShift Rows invert operation contains the very same steps in reverse order.

2.3. ShiftRows transformation

The ShiftRows transformation is the simplest layer. In the ShiftRows transformation, the bytes in the last three rows of the State are cyclically shifted over different numbers of bytes. The first row, $r = 0$, is not shifted, the second row, $r = 1$, the third row, $r = 2$ and the forth, $r = 3$.

These steps are similar to the steps of Playfair cipher. Obviously easy to see the InvShiftRows transformation.

2.4. MixColumns transformation

The less difficult the ShiftRows transformation has been, the more complicated the MixColumns transformation is, which of course makes us happy as an outstanding symmetrical method should deploy brave ideas.

In order to understand the essence of these layers, some mathematical knowledge has to be acquired. As a matter of fact the working of AES is based on operations on byte level that has been seen by the previous layers. Let the bits of B byte be $B_7B_6B_5B_4B_3B_2B_1B_0$ and corresponds polynomial is the following

$$b(x) = B_7x^7 + B_6x^6 + B_5x^5 + B_4x^4 + B_3x^3 + B_2x^2 + B_1x + B_0x^0.$$

The coefficients of these polynomial are either 0 or 1 so such polynomial equivalent to any 8 term long bit sequence.

For example the {10000011} bit sequence, the {10000011} hexadecimal number and the $x^7 + x + 1$ polynomial are equivalent.

In these cases addition between the polynomials is corresponds the exponents of the identical powers are added ($\mod 2$).

For example if we add the $x^6 + x^4 + x^2 + x + 1$ polynomial to the previous one, we get the $x^7 + x^6 + x^4 + x^2$ polynomial. Using the binary notation we get the following equality

$$\{10000011\} \oplus \{01010111\} = \{11010100\}.$$

The result is the same if we do the addition on byte level with hexadecimal numbers $\{83\} \oplus \{57\} = \{d4\}$. In other words the AES algorithm uses the $\mathbb{GF}(2^8)$finite field to define the MixColumns layer.

Now let's see the multiplication. We may need the irreducible polynomial that was used by AES algorithm $m(x) = x^8 + x^4 + x^3 + x + 1$ (see [4]). In hexadecimal mode, defined by us, this would be the following $m(x) = \{01\}\{1B\}$. Henceforward the $(\bullet)$ operation means the remainder of the product of two polynomials dividing by $m(x)$.

Applying the above mentioned method and using hexadecimal style we get that $\{57\} \bullet \{83\} = \{c1\}$. Its truth can be proved by performing the usual multiplication first. Naturally we take care of executing the additions in the defined way,

$$(x^6 + x^4 + x^2 + x + 1)(x^7 + x + 1) = x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1.$$

In the next step do the division with remainder with an irreducible polynome that is used in AES algorithm, where we gain the foreseen result

$$x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1 \pmod{x^8 + x^4 + x^3 + x + 1} = x^7 + x^6 + 1.$$

We already know that modulus creation had been an outstanding way to confuse the regularity in the vector in Knapsack method. There is no difference here and no other binary operation that would provide the gained result, so it is an excellent idea.

We note that the result after division is at maximum 7th degree so the coefficients can exceedingly be illustrated on one byte.

The $(\bullet)$ operation is associative and the identity element in the structure is $\{01\}$. The invert of any binary polynomial under 8th degree can be determined by the expanded Eucledian Algorithm.

Now only one thing is missing from our discernment. Let's see what happens if $b(x)$ polynomial is multiplied by the $x^1 = \{02\}$ polynomial. At first multiply with x to gain:

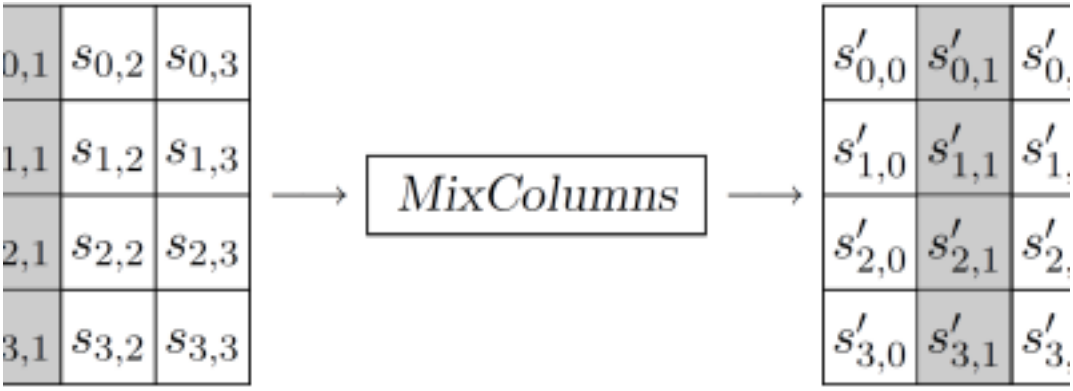
$$B_7x^8 + B_6x^7 + B_5x^6 + B_4x^5 + B_3x^4 + B_2x^3 + B_1x^2 + B_0x^1.$$

The $(\bullet)$ operation orders the configuration of $\bmod m(x)$ modulus with the resulted polynome. If $B_7 = 0$ we have no job as the modulus configuration $m(x)$ changes nothing. If $B_7 = 1$, subtract the $m(x)$ polynomial from the given polynome or simply XOR it with .

We can see that with the $\{02\}$ polynomial the multiplication is simple, the coefficients of the presented $b(x)$ polynomials are shifted one place left and if the value from the byte is 1 we XOR the number with {1b}. This method is called xtime() operation in the documentation of Rijndael system. The operations with higher powers can be done easily by possessing the acquired knowledge.

The MixColumns transformation converts the bytes of state structure in all cases in a way that the bytes are multiplied by the predefined polynomials as above introduced. Each new byte depends on all bytes in the column of the original byte. Evidently any minor changes in one byte results in a major change of the entire picture. The following equations defines the columns:

$$s'_{0,c} = (\{02\} \bullet s_{0,c}) \oplus (\{03\} \bullet s_{1,c}) \oplus s_{2,c} \oplus s_{3,c}$$
$$s'_{1,c} = s_{0,c} \oplus (\{02\} \bullet s_{1,c}) \oplus (\{03\} \bullet s_{2,c}) \oplus s_{3,c}$$
$$s'_{2,c} = s_{0,c} \oplus s_{1,c} \oplus (\{02\} \bullet s_{2,c}) \oplus (\{03\} \bullet s_{3,c})$$
$$s'_{3,c} = (\{03\} \bullet s_{0,c}) \oplus s_{1,c} \oplus s_{2,c} \oplus (\{02\} \bullet s_{3,c})$$



Equations, similar to the ones above, defines the InvMixColumns command that has to be used in case of legal decryption. This operation, as it turns out from its name, is the invert of the MixColumns operation. The $(\bullet)$ operation defined hereby equals the previously introduced operation.

$$s'_{0,c} = (\{0e\} \bullet s_{0,c}) \oplus (\{0b\} \bullet s_{1,c}) \oplus (\{0d\} \bullet s_{2,c}) \oplus (\{09\} \bullet s_{3,c})$$
$$s'_{1,c} = (\{09\} \bullet s_{0,c}) \oplus (\{0e\} \bullet s_{1,c}) \oplus (\{0b\} \bullet s_{2,c}) \oplus (\{0d\} \bullet s_{3,c})$$
$$s'_{2,c} = (\{0d\} \bullet s_{0,c}) \oplus (\{09\} \bullet s_{1,c}) \oplus (\{0e\} \bullet s_{2,c}) \oplus (\{0b\} \bullet s_{3,c})$$
$$s'_{3,c} = (\{0b\} \bullet s_{0,c}) \oplus (\{0d\} \bullet s_{1,c}) \oplus (\{09\} \bullet s_{2,c}) \oplus (\{0e\} \bullet s_{3,c})$$

2.5. AddRoundKey transformation

This layer makes our encrypting method depend on the key. The operation itself is far easier than the formerly reviewed MixColumns. The operation is a simple addition (XOR) between the pre-set structure and the bytes of the roundkey.

From the provided secret key the algorithm makes a long, a so-called expanded key. At the beginning of the expanded key stands a copy of the original secret key then every other words can be originated from the previous words.

A roundkey contains N_B words, in our case 4, and we need $N_R + 1$ roundkeys including the secret key. In case of AES- 128 the length of the roundkey is $N_B(N_R + 1)$ that is 44 words.

In all cases the expanded key has to be divided into the same size of pieces as the N_B state-structure. The ordering process has to be handled carefully because the first roundkey that is the first N_B number of words belongs to 0th circle while the second roundkey that is the second number of words to 1th circle.

Continuing this implicitly we gain the further correspondents. In the roundkey the words belongs to first, second, third and fourth columns in an order which also sets the order of performing the XOR operation. The steps are described by the following equality:

$$[s'_{0,c}, s'_{1,c}, s'_{2,c}, s'_{3,c}] = [s_{0,c}, s_{1,c}, s_{2,c}, s_{3,c}] \oplus w_{round*N_B+c},\ 0 \leq c < N_B,$$

where the expression round means the number of the actual round and the w vector will be explained soon.

To fully understand the process of generating roundkey we need to introduce two more functions. Both the input and the output of the SubWord function is a 4 bytes long word. We imply the S-box of SubBytes on every four input byte. The output of the RotWord function is also a 4 bytes long word that changes the (a, b, c, d) letter order into (b, c, d, a).

An invariant (Rcon[i]) belongs to each round which, according to the previously introduced forms, is determined by the $[x^{i-1}, \{00\}, \{00\}, \{00\}]$ term, where raising the power occurs according to the way introduced in this chapter.

Keeping the hexadecimal signs x is marked by $\{02\}$. the starting value of i index is 1.

The first N_k words of the expanded key contain the secret key, every further word, let it be marked by w_i, is provided by the XOR operation executed between the preceding w_{i-1}, and the N_k times earlier w_{i-N_k} words.

In case of those words which position is the multiples of N_k, the result is given by a XOR operation between w_{i-1} and Rcon[i] of which henceforward the SubWord and SubBytes operations will be applied.

The above introduced w_i is a 4 bytes long word where $0 \leq i < N_B(N_R + 1)$.

Now every detail has been cleared. We set a secret key in the Rijndael encrypting algorithm then create the expanded key. After that we engage in the $N_R - 1$th round the Round function, introduced at the beginning of the chapter then we establish the encrypted image by a FinalRound function.

aes.msi

During decryption we use the inverts of the encrypting algorithm. In order to manifest the propriety of the order clearly, we write down the order, applied during encryption, again.

1. AddRoundKey (State, 0. roundkey)
   (a) SubBytes (State)
   (b) ShiftRows(State)
   (c) MixColumns(State)
   (d) AddRoundKey(State, 1. roundkey)

⋮

9. AddRoundKey (State, 8. roundkey)
   (a) SubBytes (State)
   (b) ShiftRows(State)
   (c) MixColumns(State)
   (d) AddRoundKey(State, 9. roundkey)

10. FinalRound (State, RoundKey)
    (a) SubBytes(State)
    (b) ShiftRows(State)
    (c) AddRoundKey(State, 10. roundkey)

And here is the inverse order

1. AddRoundKey (10. roundkey)
    (a) InvShiftRows (State)
    (b) InvSubBytes(State)
    (c) AddRoundKey(State, 9. roundkey)
    (d) InvMixColumns(State)

⋮

10. FinalRound (State)
    (a) InvShiftRows(State)
    (b) InvSubBytes(State)
    (c) AddRoundKey(State, 0. roundkey)

It is also obvious from the description that AddRoundKey layer is its own invert.

As it seems the AES deserves the place in the word of cryptography that was intended for. It works well on different platforms and the level of encryption provided by AES also hits the expected standards. It appears that at present there is no better option to break it than brutal force, a systematical check of possibilities.

3. Secret communication

The AES algorithm, introduced in this chapter, is a symmetrical algorithm that contains a key-dependent part which is also the token of security. On the other hand the keys have to be shared with the participants, which is not always an easy task.

If we have the possibility to work with a physically stable channel the task is considered quite simple as the intruder can only access our secret key by abusing the channel. This method can exceedingly be resolved in case of small distances but for large distances it is not worth to be chosen as safety cannot be guaranteed and it is even expensive.

The secured channel has no physical protection so the attacker may contact the channel and also endanger the safety of data processing. Several different protocols are used for communication that guarantees the invulnerability of our data.

It is easy to see that in case of n number of communicating partners we need $\frac{n(n-1)}{2}$ keys if we would like to provide a key to every possible pair. This makes a considerably large numbers of keys necessary to generate in case of large number of partners and when using the symmetric method every participants has to agree each other. The obviously complicated negotiations are unnecessary due to the use of asymmetric keys.

Worthy of note that we can agree to sue a common key without a pre-arranged key change, an example for this is 3-Way cryptography.

Imagine the process as two participants, Alice and Bob, would like to communicate. The message is meant to be sent in a box and both of them possess a padlock and naturally a key.

Firstly Alice puts the message into the box and locks it with her own padlock. Bob also puts his padlock onto the box and sends it back to Alice. Alice takes off her own padlock and sends the box back to Bob. Finally Bob takes off his padlock and gets the message.

The message was sent securely in both cases as it had a padlock on it and the participants did not need to send the keys. There was only one condition for the method to work that the encrypting and decrypting methods must have been compatible that is:

$E_k(D_k(T) = D_k(E_k(T)).$

4. Exercises

1.

Determine which polynomials belongs to {2A} and {75} hexadecimal numbers according to the representation introduced in the chapter.

2.

Determine the value of the operation xtime({57}).

3.

Let $a(x) = x^3 + x^2 + 1$ and $b(x) = x^6 + x^4 + x^3 + x$ are given polinomials. Determine the value of $a(x) + b(x)$ and $a(x) \bullet b(x) \pmod{m(x)}$!

4.

Let $s_{0,1} = 00111000$, $s_{1,1} = 10011000$ $s_{2,1} = 10011101$ and $s_{3,1} = 00000111$ elements of a column. Give the result of MixColumns operation.

5.

The (Rcon[i]), applied in AES and using the previously introduced style, can be written as $[x^{i-1}, \{00\}, \{00\}, \{00\}]$. Keeping the hexadecimal form x is represented by $\{02\}$. Determine the values in cases $i = 1$, $i = 2$ and $i = 3$.

Chapter 7. Knapsack

We saw in the previous section that classical cryptosystems are vulnerable to skillfull and sometimes lucky codebreakers. If we know the method and cipher key then we can easily get to the original message. Therefore the research for finding better methods continued in order to pose greater difficulties for the illegal intruder.

Therefore the research must have continued in order to find cipher methods that pose bigger challenge for the illegal intruder.

In 1970s Ralph Merkle, Whitfield Diffie and Martin Hellman suggested a new type of encryption method, the so called public key cryptography.

Ralph Merkle

Whitfield Diffie

Martin Hellman

They wanted a cipher that is very difficult to break even if the encryption key and the method is known. Of course the method is not really a big hit if decryption is also demanding for the intended receiver. The solution is that we keep a tiny bit of information hidden from outsiders, and that extra information enables quick decryption.

For the mathematically versed readers it is immediate that the term “very difficult” is not a well-defined notion. If we want to make this precise then we have to go into the problems of complexity theory a bit deeper. There are many good books in this topic (for instance [6], [10]).

Next we take a small detour to get some impressions on what is meant by “easy” and “difficult”. The *time complexity* of an algorithm is a function of the length of the input data. Roughly speaking, the given algorithm has time complexity $f(n)$ if for an arbitrary input of length n the computation finishes after at most $f(n)$ steps.

Following J. Edmonds and A. Cobham, we say that there are two main classes of algorithms. The “good” algorithms are when the previous time complexity function is polynomial, and the “bad” ones where it is exponential.

In the first case $f(n)$ is of the form $c_1 n^s$, while in the second case it is $c_2 a^n$, where c_1, c_2, a, s are appropriate constant values. (It is important to note that the there can be “good” algorithms according to the definition that are practically incomputable, for instance $f(n) = 10^{300} n^{50}$, and vice-versa $f(n) = \exp(10^{-500} n)$ can be computed quickly for many n though it is “bad” by definition).

Let’s denote the class of problems that can be solved in polynomial time by P. For the algorithms with exponential time complexity we use the notation NP. This comes from the fact these difficult problems can be solved in polynomial time if we use a hypothetical machine that computes all possible branches of the algorithm at the same time (nondeterministic).

It is still an unsolved problem whether $P = NP$ or $P \subset NP$. The intuition favours the second case but a proof has been elusive so far. It is clear that problems in P are in NP as well. For the other direction our conjecture is that it is not true. However, in order to prove that we have to show for every difficult problem that it actually cannot be solved by an efficient algorithm.

There is further trouble, since those problems in NP, that we think they are not in P, usually have the same level of difficulty.

A problem in NP is NP-complete if solving it in polynomial time would imply that all problems in NP can be solved in polynomial time. It turned out that all the problems we can consider are NP-complete. (Note that integer factorization is thought not to be NP-complete.) Some of this type of NP problems:

1.

(Packing Problem) To put a given set of differently sized and shaped items into the least possible number of fixed size boxes.

2.

(Travelling Salesman Problem) Given a set of cities, we would like to visit them all. What is the shortest/cheapest route?

3.

(Assignment Problem) Given sets of courses, students and lecturers construct a timetable with no collisions.

After these considerations we can imagine the design of a cryptosystem.

Choose a "difficult" problem, i.e. a problem not in P. Take a base problem of this problem (denoted by P_k), meaning that it can be solved in linear time.

Then with some mathematical method we "shift" this problem to problem P'_k, which resembles to the original difficult problem. We publish P'_k and hide the way of getting back to P_k as a secret trapdoor.

Thus the intended receiver needs to solve an easy problem, why the illegal user faces a difficult one.

Typically we know very little about the underlying problems of public cryptosystems. Probably these problems are NP-complete or more complex, for example, integer factorization, determining primality, or finding primes of certain size.

1. The Knapsack Problem

The first problem that makes the public encryption method possible is the so called *Knapsack Problem*.

Mathematicians are interested in this problem even without its cryptography applications. The task is to put many different small items into a knapsack but we would like to have it as full as possible so we need an algorithmic method for deciding what to put into the knapsack.

It is clear even for the first reading that if we have many items and a large knapsack we will need long time for coming up with an optimal solution.

Formally, let $A := (a_1, a_2, \ldots, a_n)$ be a vector containing positive integers and k also a positive integer. We need to determine a set of a_i values such that their sum is k. We can surely get the result by the trial and error method so we need at most 2^n attempts. In case of 10 items it takes only 1024 checks. However, for 300 pieces we already face a "difficult" problem.

For cryptography applications we definitely need a case of the Knapsack Problem for which the packing is simple. We can imagine such a convenient situation.

Let each item be of such a size that the sum of the preceding items in the list fit in but does not fully fill. In this case the packing is simple. We need to take the biggest possible item that fits into the knapsack. It is clear that if we leave out this one, all the remaining can fill up only less amount of space. Then for the remaining available space we take again the biggest available item and we continue similarly. In a few steps it becomes whether an exact packing is possible or not.

In 1982 Ralph Merkle constructed a public key cipher method based on the Knapsack problem. He bet 1000 dollars that the code is unbreakable.

For the more exact treatment we need the definition of *super-increasing vectors*. In the followings we assume that the lowercase letters indicate positive integers.

Let the vector $A := (a_1, a_2, \ldots, a_n)$ be *super-increasing*, if all elements of the vector is greater than the sum of preceeding elements, i.e.

$$a_j > \sum_{i=1}^{j-1} a_i \qquad j = 2, 3, \ldots, n.$$

Let's see now the underlying idea. For the sake of simplicity we assume that we would like to encrypt the ab pairs of symbols and we have a super-increasing vector of size 10, $A := (c_1, c_2, \ldots, c_{10})$.

Let symbol a correspond to 00001 and b to 00010. Similarly for other symbols of the alphabet we use the 5 bit binary representation of the code number of the letter. Thus for the ab pair we have the

$$X_{ab} = (0, 0, 0, 0, 1, 0, 0, 0, 1, 0)$$

vector.

Next we calculate the dot product CX_{ab} of A and X_{ab}. Clearly, only those components of the super-increasing vector contribute to the product that have a corresponding 1 in the other vector. Therefore we get an integer number, from which only those could get back the encrypted message, who know which components of the A vector are in the sum. Then we can give a bitvector, in which we have a 1 if the corresponding element of A is in the sum, 0 otherwise. After that, reconstructing the message is just a simple decoding step.

For super-increasing vectors this task is a simple one. Following the previously mentioned method in the general case we can do the following. Given an $A := (a_1, a_2, \ldots, a_n)$ super-increasing vector and assuming that the coded message (consisting of 0s and 1s) is exactly an n dimensional X rowvector. Let's denote the value of the AX dot product by k. Deciphering then goes this way: we check whether the inequality $k \geq a_n$ holds or not. If yes, then last component of the X vector is a 1, if not, then that component is a 0. Then k_1 is defined by

$$k_1 := \begin{cases} k & , \text{if } k < a_n, \\ k - a_n & , \text{if } k \geq a_n. \end{cases}$$

Continuing similarly to a_1 we get the 1s and 0s. These are going to be the coded symbols. Decoding can be done easily by a lookup table.

Clearly, if the A vector is public then encryption is very easy. From now on we talk about the solution of the (A, α) problem when we decompose an α number as a sum of the components in the A vector.

The trouble is that the encryption is also very easy for the illegal user as well, and that contradicts our goals.

We have to investigate how we can "damage" the vector A in a way that it does not look super-increasing any more, but we can reconstruct the original vector. This would fulfill the aim of public key coding since decomposing a number over an arbitrary number is indeed a very difficult problem, given that the components are big enough and there are sufficiently many of them.

These informally defined notions mean that we need to choose a vector such that selecting the components for the sum would take disproportionally long time. For the illegal user it is difficult, but for the person knowing the super-increasing vector it is very is easy to calculate the solution by using the previous method.

Let choose a natural number m such that

$$m > \sum_{i=1}^{n} a_i.$$

This m is obviously much bigger than any a_i since A was a super-increasing vector. Let t be a natural number such that $(t, m) = 1$. We call m the modulus and t the multiplier.

From the choice of t it follows that there exists a natural number t^{-1} such that

$$tt^{-1} \equiv 1 \pmod{m}.$$

After these choices we calculate $ta_i, \ (i = 1, 2, \ldots, n)$ products and reduce them $\pmod{m}$, thus we get values $b_1, b_2, \ldots, b_n$. The resulting $B = (b_1, b_2, \ldots, b_n)$ vector is then a public key. B is not super-increasing, therefore despite knowing the public key decryption is still troublesome.

The series of operations on the A vector is called *strong modular multiplication* with respect to m and t. The secret trapdoor consists of the t, t^{-1} and m values. With these the legal user can easily get the A vector from B. At the same time the α "preimage" can also be calculated from the β value yielded by the encryption process. This then can be deciphered based on A. To show the correctness and the usage of the method we have the following theorem:

Theorem 7.1. Let $A = (a_1, a_2, \ldots, a_n)$ be a super-increasing vector and B derived from A by strong modular multiplication with respect to m and t. Furthermore let $u \equiv t^{-1} \pmod{m}$, β arbitrary positive integer and

$$\alpha \equiv u\beta \pmod{m}, \quad 0 < \alpha < m.$$

Then the following statements hold

1. The (A, α) Knapsack problem can be solved in linear time. If a solution exists, then it is unique.

2. The (B, β) Knapsack problem has at most 1 solution.

3. If there is a solution for the (B, β) problem, then it is the same as the solution for (A, α).

Proof. The first statement are obvious from the algorithm given above.

For the third statement let's suppose that there exists an n-bit D vector which is a solution for the (B, β) problem, i.e. $BD = \beta$. Thus

$$\alpha \equiv u\beta = uBD \equiv u(tA)D \equiv AD \pmod{m}.$$

Since m is greater than the component sum of A the inequality $AD < m$ holds. Moreover $\alpha < m$ also holds due to the definition of α. Clearly $AD = \alpha$ follows, which means that D is the same as the unique solution of the (A, α) problem. This proves the second statement as well. □

Ralph Merkle lost the 1000 dollar bet. Adi Shamir immediately broke one version of the code, but this was not enough for winning.

Adi Shamir

In 1985 Ernest Brickell succeeded to find a quick algorithm for the Knapsack problem, therefore he succeeded to break the previously described cipher method.

Chapter 8. RSA

In the 1970s computer engineers were more and more involved in the issue of key-sharing. The evolution of computer networks had begun and foreseers recognized that sharing the keys would be the most burning question in the future of information technology.

Only a few scientists took part in this utopian challenge as Withfield Diffie, Martin Hellman and a little bit later, Ralph Merke. They tried to find such functions which were called one-direct functions that is where it is easy to count from one direction but almost impossible from the other. To be more accurate, for proceeding backwards we need some extra information. As a matter of fact their ideas had created the basics of asymmetric key cryptography. They developed the Diffie–Hellman key exchange method which did work although not perfectly. They continued their researches on Stanford University, were still looking for that one-direct function which would make asymmetric key cryptography become a reality. His dedicated job can be the best described by a sentence from Martin Hellman. “God rewards the fools”

A well - respected figure of number theory, G. H. Hardy (1877 - 1947) wrote the following about his work: “I have never done anything ’useful’. No discovery of mine has made, or is likely to make, directly or indirectly, for good or ill, the least difference to the amenity of the world. I took part in the qualification of new mathematicians, mathematicians like me, and their job – or the part of it that can be ascribed to my help – has been so far as pointless as mine. Judged by all practical standards, the value of my mathematical life is nil; and outside mathematics it is trivial anyhow.”

G. H. Hardy

There may be several aspects in this outstanding mathematician’s viewpoints to argue with, but in the distance of half a century it would not be wise to do so. The reason why these lines were eager to appear is the interesting fact that the materialization of public key cryptography led the scientists to ’the queen of mathematics’, number theory and made them benefit from the ’useless’ science of Hardy.

1. RSA

Researches continued for the sake of public key cryptography and the new ideas were originated from the world of primes. We may consider the factorization of a composite number to be an easy task. This thought tends to be correct if the given number is not large. But the summary of complexity theory from the theoretical mathematical part point out that this concept becomes invalid if the number is large enough. In other words, no algorithm is known that can operate factorization fast. Hendrik W. Lenstra Jr. dropped the following funny remark: “Suppose that the house - keeper accidentally threw out the p and q numbers, but the pq product was left. How can we regain the factors? We can only read it as the defeat of mathematics that the most appropriate way to scavenge the junkyard and use memo- hypnotic techniques.”

Ted Rivest, Adi Shamir and Leonard Adleman worked in the IT laboratory of MIT and knew the researches of Diffie, Hellman and Merkle and they would have been eager to create the one-direct function dreamed by

others. After a fine celebration of Easter, in April 1977, Rivest found the answer and published it with his co-workers which opened brand new ways of cryptography. After the capitals of their names the method is called RSA. We are able to hide the key from uninitiated eyes with the help of Fermat's little theorem. Hereby we detail the encrypting method.

Ted Rivest, Adi Shamir and Leonard Adleman

Let p and q be different prime numbers, in general we choose decimal numbers with a hundred or more digits. Let $\phi(n)$ be the number of positive integers, smaller or equal to n and relatively prime to n.

Then if $n = pq$, the following equation is true

$$\phi(n) = (p-1)(q-1)$$

n is called modulus in the followings. Choose an integer $d > 1$ in a way that $(d, \phi(n) = 1$ and determine the integer e where $1 < e < \phi(n)$ and e fulfills the congruence

$$ed \equiv 1 \pmod{\phi(n)}.$$

After having these values, we code the chosen text and encrypt the given T value. the encrypted text C is defined by the next equation

$$C = T^q \pmod{n}.$$

(We note that decimal numbers are used in general for coding. The gained number sequences are divided into blocks and encrypted separately. We usually use i long blocks where $10^{i-1} < n < 10^i$.)

After finishing the encrypting process we are engaging the issue of decryption. The next theorem shows the way of decryption.

Theorem 8.1. Using the previous notation, the following congruence is fulfilled

$$T \equiv C^d \pmod{n}.$$

Hence, if decryption is unique then $T = C^d, \pmod{n}$.

The theorem is proved with the help of Euler's theorem.

Proof.

According to the choice of the previously defined d, such j exists, where

$$ed = j\phi(n) + 1.$$

First we suppose that neither p nor q divides T. According to Euler's theorem the following congruences are true,

$$T^{\phi(n)} \equiv 1 \pmod{n}$$
$$T^{ed-1} \equiv 1 \pmod{n}.$$

So we have

$$C^d \equiv (T^e)^d \equiv T \pmod{n}.$$

If exactly one of p and q, say p divides T, then we obtain that

$$T^{q-1} \equiv 1 \pmod{q}.$$

Finally we have

$$T^{\phi(n)} \equiv 1 \pmod{q}, \quad T^{j\phi(n)} \equiv 1 \pmod{q}, \quad T^{ed} \equiv T \pmod{q}.$$

Since this last congruence is valid $\pmod{p}$ we have proved this case.

Similarly we can prove the case when p and q divide T. □

So the theorem, proved by us, shows that if we rise the encrypted text to the dth power, then reduce it $\pmod{n}$, we gain the original text.

We note that by designing the system, it is essential to determine the relative primality e and $\phi(n)$. This can be done in linear time with using the Euclidean algorithm. Our next task is to determine the value of d satisfying the congruence

$$ed \equiv 1 \pmod{\phi(n)}.$$

It is easy to recognize that such d decrypting exponent exists. As $(e, \phi(n)) = 1$ there exist x_0 and y_0 integers such that

$$ex_0 + \phi(n)y_0 = 1.$$

From this equation we get that

$$ex_0 \equiv 1 \pmod{\phi(n)}$$

that is $x_0 = d$.

A simple example can present how the RSA works. Let Alice choose the prime, $p = 11$ and $q = 23$ numbers. Then her modulus will be $pq = 253$ and in our case $\phi(n) = (p-1)(q-1) = 220$. Let $e = 17$ be the encrypting exponent from which $d = 13$ the decrypting exponent can be determined. Bob's similar choices are the following, $p = 13, q = 19$, so $n = 247$, and $\phi(n) = 216, e = 65$ and $d = 113$.

Imagine that Alice would like to send a message to Bob, at present the word ”TITOK”. He applies the RSA method on each letters that is she defines the value

$$C = (T^e, (\mathrm{mod}\ n)).$$

The ASCII code of letter T is 84, Bob’s public key is 65 and his modulus is 247 which is also public. So Alice calculates the value

$$84^{65} \equiv 145 \pmod{247}$$

and the encrypted image of T will be 145.

We follow the same pattern in case of every letters. the results are indicted in the chart below.

ASCII	(
84	1
73	4
84	1
79	1
75	!

Now as the method has been introduced the question arise what is public and what has to remain hidden. We can publish the modulus n and the encrypting exponent e by this system. The numbers $p, q, \phi(n)$ and d represents the hidden trap-door which means, either of them is known the system is broken.

The RSA algorithm was under copyright law until 2000 in the United States. Today, anybody can create software or hardware tools, operating with RSA algorithm, without paying licence fees.

The PGP (Pretty Good Privacy) system was invented by Philiph R. Zimmermann using the RSA method and nowadays everybody can use (see [18], http://www.pgpi.org/).

The RSA attains safely the dream of Diffie and Hellman, the key change, as the role of e and d can be reversed.

RSA is the most known asymmetric encrypting method nowadays and as DES also a block algorithm. Under the length of the RSA’s key we always mean the length of n (usually $1024 - 3072$ bits). For the sake of security it is advisable to generate the key from approximately equally large primes. Some pragmatically essential details will be mentioned later.

It is essential to note that RSA becomes breakable if we can factorize the n number. This can obviously be done as it requires only enough time and calculation capacity, but with today’s mathematical knowledge and calculation capacity this is a huge demand and decryption is not possible within reasonable time.

Interestingly, after the announcement of RSA in 1977 Martin Gardner, American mathematician and author of popularizing popular sciences, published in the mathematical games heading of Scientific America journal an article titled, “A new kind of cipher that would take millions of years to break”.

Martin Gardner

Here he explained the method of public key cryptography to the readers and provided an n modulus which he used to encrypt a text. In his case n = 114 381 625 757 888 867 669 235 779 976 146 612 010 218 296 721 242 362 562 561 842 935 706 935 245 733 897 830 597 123 563 958 705 058 989 075 147 599 290 026 879 543 541.

The task of the readers was to factorize n and decrypt the text. He offered a 100 dollars price to the winner. Gardner suggested that to understand the RSA the participants should turn to the IT lab of MIT. We can imagine the surprise of Rivest, Shamir and Adleman when they received more than 3000 letters.

Gardner example had only been solved 17 years later. In 26th April, 1994 a group of 600 volunteers announced that the factors of N are the following, q = 3 490 529 510 847 650 949 147 849 619 903 898 133 417 764 638 493 387 843 990 820 577 and p = 32 769 132 993 266 709 549 961 988 190 834 461 413 177 642 967 992 942 539 798 299 533.

To tell the story as a whole, the decrypted text was: “The magic words are squeamish ossifrage.”

The factorization task was divided on the computer network using every free capacity. We note that the search for Mersanne primes follows the same pattern.

17 years may seem to be a short period of time but we have to understand that in case of Gardner we only used a modulus of the order of 10^{129} which is far smaller than the currently suggested modulus where we have to consider billions of years.

Anyway, it is just a nice story and has nothing to do with banks or military secrets yet illustrates well our remark about time limit. To break the RSA the intruders would need a fast way of factorization. We do not have such method for the time being. The algorithm works well but stands on weak legs in the sense that it is not proven whether a polynomial time factorizing algorithm exist.

Moreover it is also unknown if such algorithm exists that can break RSA without factorization.

The fact that the encrypting and decrypting key can be reversed makes the Diffie–Hellman method possible to realize. Let us suppose that Alice and Bob would like to choose a common key so they publicly agree in choosing a modulus n and a generator g which are 195–512 bits long.

The term, generator means that all the numbers smaller than n have to be generated by the formula $g^x \mod n$. After that, both of them choose a random number smaller than $n-1$. Let them be x and y. Then the following steps occur:

1.

Alice send Bob the value $k_1 = (g^x, (\mathrm{mod}\ n))$,

2.

Bob sends back $k_2 = (g^y, (\mathrm{mod}\ n))$ in his answer,

3.

Alice calculates $k_3 = (g^y, (\mathrm{mod}\ n))^x\ \ (\mathrm{mod}\ n)$,

4.

Bob calculates $k_4 = (g^x, (\mathrm{mod}\ n))^y\ \ (\mathrm{mod}\ n)$,

5.

Their common key is $(g^{xy}, (\mathrm{mod}\ n))$.

It is known that the equation $a^x = y$ is solvable over real number field using the logarithm function. But we have problems when we involve computing in modular arithmetic in RSA public-key cryptosystem. We have a small chance to determine x.

More generally let g and y elements of the finite group G. If $x \in G$ is a solution of the equation $g^x = y$, then x is called a discrete logarithm to the base g of h in the group G.

Obviously every element $y \in G$ discrete logarithm to the base g if and only if G cyclic group and g is a generator. The discrete logarithm problem is an NP problem.

A modified version of Diffie–Hellman method is the ElGamal method, which was published by Taher Elgmal in 1984.

Let q and the generator g of $F^*(q)$ is known for the two participiants. Then A chose secretly an integer $0 < m_A < q - 1$ and publish g^{m_A} (obviously it is element of $F(q)$).

We send our message w to A in the following form (g^k, wg^{km_A}), where k is an arbitrary positive integer number. Obviously since a discrete logarithm problem is difficult the illegal intruder can not obtain valuable information.

When A get the message, can be calculated g^{-km_A} and in this way w also.

The discovery of RSA is a serious feat of arms, a beautiful result of human mind and the ability of working together. On the other hand it is also interesting in its history. For the sake of completeness we would mention that according to the British government, public key cryptography was firstly invented in a top secret institution, built after WWII, in Cheltenham, in the so called Government Communication Headquarters (GCHQ).

We could learn subsequently that British scientists, James Ellis, Clifford Cocks and Malcolm Williamson had developed all the basic theorems of public key cryptography by 1975, but they were ordered to stay quiet.

These events demonstrate well that we are challenging an exotic border of sciences, where the new inventions are kept in secret, because the secret knowledge means steps forward for the given government. Unfortunately in these cases, the fate of humans becomes secondary.

2. Pragmatic comments

At first we have to face with the issue of finding primes with a 100 digit. Now the problem of searching and determining the primality also arises. The search for large primes is in process at the moment and they are possibly not published because of being classified.

The searching process operates in a way that we choose an appropriately large odd number (in or case 100 digits) then with the help of some primality test, which will be introduced later, we decide whether the given natural number is a prime.

If the answer is no, we give a try to the following odd number. According to the Prime number theorem there are about

$$10^{100}/\ln 10^{100} - 10^{99}/\ln 10^{99}$$

primes with 100 digit.

This implies that in case of any odd number, the chance of a successful test is 0,00868.

The next problem is choosing d. After selecting p and q we have to set d. It is important that d must not be small as it may lead to the break of our system.

The chosen d can be tested with the Euclidean algorithm. If our choice was good and d satisfies the $(d, \phi(n)) = 1$ condition, we have found the appropriate one so the number e can be read from the equations of the Euclidean algorithm.

We need to define the $(a^r \pmod n)$ term for both decryption and encryption. We may do this operations a lot faster if instead of multiplying a with itself and then reduce it, we follow the so-called successive squaring method and after each operation we reduce the given number $\text{modulo}\, n$. The method is the following.

At first we regard the binary representation of r, in our case

$$r = \sum_{j=0}^{k} x_j 2^j, \quad x_j = 0, 1; \quad k = [log_2 r] + 1.$$

Using successive squaring we can easily determine the values

$$(a^{2^j} \pmod n), \quad 0 \leq j \leq k.$$

From this the expression $(a^r \bmod n)$, which is necessary to us can easily be calculated.

At maximum $k - 1$ multiplication and reduction is required for the calculation to be done.

Let's see an example of successive squaring. Give the value of $7^{83} \pmod{61}$. According to Fermat's theorem we have

$$7^{60} \equiv 1 \pmod{61}.$$

So it is enough to calculate the value of $7^{23} \pmod{61}$. For the successive squaring process we pre–create those powers of 7 where the exponent is in the form 2^j and the final result is got by $\pmod{61}$. The results are summed in a table:

0	1	2	3
7	49	22	57

Now, the wished result comes from an easy calculation if we now that the binary form of 23 is 10111 we have

$$7^{23} \pmod{61} \equiv (16(22(49 \cdot 7)) \pmod{61}) = 17.$$

Let's see how it works in practice. Code the letters after their ordinal numbers unlike previously where the code was in accordance to ASCII value. Encrypt the letter pair SA which equals 1901 and the UN letter pair which is 2114 and suppose that the encrypting exponent is 17. The following chart shows the successive squaring step-by-step.

	1901	21
	582	16
	418	17
	25	22
3	625	4
7	1281	16

Further attention is required if we would like to create an ambitious system which is difficult to decrypt. We must keep clear p and q of being close to each other. If p and q are close then $(p-q)/2$ is small and $(p+q)/2$ is not much larger than $\sqrt{n}$. Moreover, the left side of the next equation is a full square

$$\frac{(p+q)^2}{4} - n = \frac{(p-q)^2}{4}.$$

With the help of this information we may factorize n by testing such x where $x > \sqrt{n}$ and continue the process until $x^2 - n$ is not a full square.

If we sign this full square with y, the equations $p = x + y$ and $q = x - y$ give the factors.

During planning we have to pay attention to the behavior of $\phi(n)$. If the greatest common divisor of $p-1$ and $q-1$ is large, their least common multiple, let us denote by u, is small compared to $\phi(n)$.

In this case, all inverse of $e \operatorname{modulo} u$ can be used as decrypting exponent. In this case it is easier to find d so we have to keep in mind that $(p-1, q-1)$ should not be large.

To avoid the above mentioned problems we generally use so so called strong primes which features are the followings:

1.

   the chosen prime p is large, at least 400-500 bit long,

2.

   the greatest prime divisor of $p-1$ is large,

3.

the greatest prime divisor of $p-2$ is large,

4.

the greatest prime divisor of $p+1$ is large.

However, the researches of R. Rivest and R. Silverman proves ([16] that some new factorizing methods (for example Lenstra's method based on Elliptic curves) can be efficient in case of strong primes as well. So the strong primes do not solve all the problems either, but apart from this RSA provides a reliable level of secrecy nowadays.

In this part of the chapter we discuss practical issues, so let's drop a few words about every day use. Although successive squaring fastens the execution of the chosen mathematical operations, the speed of RSA continues to be inappropriate in every - day life. Therefore, the whole text is rarely be encrypted with open key algorithm in practice, especially if the text is long, but with traditional symmetric algorithms which are hundred times faster than RSA.

So called Hybrid cryptosystems which uses public key cryptography are also often used. The common process is that the text is encrypted by a fast secret key algorithm and the randomly generated key of it is encrypted by public key method and these two are sent together. Such occasionally or only once used keys are called session keys. Naturally, in this case the public key algorithm protects only the key so can provide help in key sharing. So when attaining the hybrid system we must pay attention to the symmetric algorithm because if it is breakable we protected our key for nothing.

3. Digital signiture

Digital signature is a very commonly used expression nowadays, yet it is not well known what it means. Digital signature was originally created to replace the traditional hand-written signatures but also fulfill the present day standards of IT.

The digital signature itself is a number that strongly depends on the private key (which is also a number) of the signing party. It also depends on some public parameters. It is essential for a digital signature to be verifiable that is an objective third member could apparently prove without knowing the private key of the signing party that the signature was indeed made by that entity.

Asymmetric encrypting methods can be well applied to create digital signature. In this case all members have a private and a public key. The signing one always keeps his/her private key in secret. It can never be revealed for the sake of his/her own safety.

Contrary to this the public key may be published for anybody. In most cases it is necessary as the digital signature regarding a given message and a person, can be validated by the public key. There is a crucial aspect that if A acquires the digital signature of B concerning a message, then A should not be able to use this to attend other messages with the signature of B. Digital signature has several fields of use nowadays,

1.

Data Integrity (to make sure that the data have not been changed by unreliable participants),

2.

verification of the data's resource (to prove that the data is indeed originated from where it should be.),

3.

protection against denial (to make sure that a given participant could not deny the signatures made by him/her)

We can use the now introduced RSA algorithm, the techniques based on discreet logarithm, or the elliptic curves which will be introduced later for creating digital signature patterns.

Let us see a digital signature based on the RSA algorithm. This method is really simple, although far from being the safest one, the point of the method can be understood well. Our names are still Alice and Bob. Alice calculates the $C = m^{d_A} \pmod n$ value with her secret key d_A, where m means the message. Then she sends it to B who decode it using Alice's public key e_A.

$$C^{e_A} \pmod n = (m^{d_A})^{e_A} \pmod n = m.$$

If the result is the message that is m is a meaningful text, he can be sure that it was from Alice. Here we have no encryption, as knowing the public key, anybody can decode the message.

Encrypting the whole message with open key algorithms is quite problematic as it can cost a lot of time. Even with the different fastening methods the RSA is till slow. Therefore, not the entire text uses to be encrypted but an extract of it. This extract is called message digest, MD. From these the two most well known ones are SHA-1 or MD5. These are very exotic algorithms. They make an fix long bit sequence from an optional long one. (This length is 160 bit in case of SHA-1 and 128 with MD5). Hereafter this relatively short bit sequence represents the content of the documents.

In this case the process of signing is the following. We calculate the so called monitoring value $s = (MD(m))^d \pmod n$ and send the $\{m, s\}$ pair so fasten the process of signing. We note that the $\{e, n\}$ pair is public so it can be used to check.

4. Exercises

1. Let $p = 2609$ and $q = 3023$ be given prime numbers. Determine the other required parameters of RSA.

2. Determine the value of $2109^{157} \pmod{2773}$ using the successive square method.

3. Let $n = 2773$ and $e = 17$ be given integers, encrypt the word SZAUNA. Use the alphabet for coding. (For example 19 belong to S, 01 belong to A)

4. Decrypt the number 1281, if we know that $e = 17$, $n = 2773$ and $\phi(n) = 2668$.

5. Suppose that we spoiled the choice of n and it have three divisors instead of two, in out case $n = 7 \cdot 13 \cdot 19$. Determine the required parameters of RSA.

Chapter 9. Primality tests and factorization

1. Primality tests

We could see above that the efficiency of the method depends on a good choice of two large primes. But we do not know such algorithm that can decide in polynomial time, in case of any positive integer, whether the given number is a prime. Therefore we would need an algorithm that operates with a low possibility of errors. Obviously, no mathematician is happy to say that the number is most likely a prime. In these cases it is worth to use other primality tests or apply longer calculation time for PCs for the purpose.

Before we begin the testing, we exclude some numbers which are obviously not primes. Such easily verifiable method is the division with the elements of set A, where $A = 2, 3, 5$ and the square numbers can also be excluded. We usually test some divisions with previously fixed primes before we begin to apply the methods. The previous screenings are necessary because the following tests need huge resources in terms of computers and calculation time.

1.1. Euler–Fermat primality test

A trivial consequence of the Euler–Fermat theorem, introduced in the mathematical chapter, that if $(w, m) = 1$ and

$$w^{m-1} \not\equiv 1 \pmod{m},$$

then m is a composite number.

This would stand as a primality test as the true or false state of the congruence would decide the issue of primality. The problem is that such numbers exist which can slip through the test. For any w exists such m composite number with $(w, m) = 1$ and the congruence

$$w^{m-1} \equiv 1 \pmod{m}$$

is true.

These m numbers are called pseudoprimes to base w. For example 91 is a pseudoprime to base 3 as it can be easily proved that $3^{90} \equiv 1 \pmod{91}$ but $91 = 3 \cdot 17$.

The next theorem contains a probalistic statement, mentioned in the introduction. Let us call an integer w with $(w, m) = 1$ and satisfying the Euler–Fermat congruence a witness for the primality of m.

> **Theorem 9.1.** Either all or at most half of integers w with
>
> $$1 \leq w < m, (w, m) = 1,$$
>
> are witness for the primality of m.

We can now base a simple prime searching method, a probalistic algorithm on this theorem.

At first, we randomly choose an integer w for m, where $1 \leq w < m$.

Then we determine the greatest common divisor of w and m with the help of the Euclidean algorithm. If $(w, m) > 1$ than m is composite.

Otherwise, we can begin the testing. We calculate the value of $u = (w^{m-1}, (\mathrm{mod}\ m))$. If $u \neq 1$ we conclude that m is composite. If $u = 1$, w is a witness for the primality of m and we have some evidence that m could be prime.

When we have found k witnesses, then the probality of m being composite is at most 2^{-k}, except in that unlucy case that all number with $1 \leq w < m$ and $(w, m) = 1$ are witnesses.

These primes are called Carmichael numbers. The smallest number with these features is $561 = 3 \cdot 11 \cdot 17$. W. R. Alford, A. Granville and C. Pomerance (see [2]) proved that similarly to pseudoprimes, the number of these is also infinite.

If number n is a Carmichael number, n is not square, has at least 3 prime factors and if p is a prime divisor, than p divides $n - 1$. We can see that this method needs to be refined.

1.2. Solovay–Strassen primality test

To understand the Solovay–Strassen test we need some mathematical notation. The Legendre–symbol, denoted by $\left(\frac{a}{p}\right)$, for an integer a and prime $p > 2$ is defined by

$$\left(\frac{a}{p}\right) = \begin{cases} 0, & \text{if } a \equiv 0 \pmod{p} \\ 1, & \text{if } a \not\equiv 0 \pmod{p} \text{ and } \exists x \in \mathbb{Z}, a \equiv x^2 \pmod{p} \\ -1, & \text{if } a \not\equiv 0 \pmod{p} \text{ and } \nexists x \in \mathbb{Z},\ a \equiv x^2 \pmod{p} \end{cases}$$

If $\left(\frac{a}{p}\right) = 1$ then a is a quadratic residue $\pmod{p}$. If $\left(\frac{a}{p}\right) = -1$ then a is a quadratic nonresidue $\pmod{p}$.

The basic result concerning the Legendre symbol is

Theorem 9.2. If m an odd prime, then for every w

$$w^{\frac{m-1}{2}} \equiv \left(\frac{w}{m}\right) \pmod{m}.$$

The Jacobi symbol is a generalization of the Legendre–symbol. Let $3 \leq n$ and $n = p_1^{e_1} p_2^{e_2} \cdots {}^{e_k}_{k}$, then the Jacobi symbol is defined to be the product of corresponding Legendre symbols, that is

$$\left(\frac{a}{n}\right) = \left(\frac{a}{p_1}\right)^{e_1} \left(\frac{a}{p_2}\right)^{e_2} \cdots \left(\frac{a}{p_k}\right)^{e_k}.$$

In this case we find an other type pseudoprime number. Odd composite numbers m satisfying the congruence 9.2 for some w with $(w, m) = 1$ are called Euler pseudoprime to the base w.

Theorem 9.3. If an integer m is Euler pseudoprime to the base w then it is pseudoprime to the base w.

We are very glad since there is no analogues of the Carmichael–number. We can say that this algorithm is "stronger".

Theorem 9.4. If m is an odd composite numbers, then at most half of the integers w with $1 < w < m$ and $(w, m) = 1$ satisfy the congruence 9.2.

We can use a similar algorithm to Eular–Fermat test. If the congruence 9.2 is not valid, m is composite. Otherwise we regard w as a witness for the primality of m. Choose another random integer less than m and repeat the procedure. After find k witnesses we may conclude that the probality of m being composite is at most 2^{-k}.

However the estimates can not be improved, there are Euler pseudoprimes to exactly half of all possible bases.

This test is called Solovay-Strassen primality test.

We remark that using the following theorem to determine the values of Jacobi symbols is easy.

Theorem 9.5. (The Law of Quadratic Reciprocity). Let p and q are different odd primes then

$$\left(\frac{p}{q}\right)\left(\frac{q}{p}\right) = (-1)^{\frac{p-1}{2}\frac{q-1}{2}}.$$

1.3. Miller–Rabin primality test

In this section we describe a useful test, known as Miller–Rabin primality test.

Theorem 9.6. Let p be an odd prime and $p-1 = 2^s r$ where r is odd. If $(a,n) = 1$ and $1 < a < n$ then

$$a^r \equiv 1 \pmod{n} \quad \text{or} \quad a^{2^{s'}r} \equiv -1 \pmod{n}$$

for some $0 < s' < s$.

Our method is based on the Theorem 9.6.

First steps

Let us choose an arbitrary integer n and a natural number $1 < a < n$. If $(a,n) \neq 1$, then n is composite, if $(a,n) = 1$, then $(n-1)$ can be written in the form $n - 1 = 2^s r$, where r is odd.

Extracting square roots

Let us test the satisfying of the congruence $a^{2^s r} \equiv 1 \pmod{n}$. Then let us extract square roots.

Test

After the first extraction we have three possibilities.

- If $a^{2^{s-1}r} \not\equiv \pm 1 \pmod{n}$, then n is composite.
- If $a^{2^{s-1}r} \equiv 1 \pmod{n}$, then we continue the extraction.
- If $a^{2^{s-1}r} \equiv -1 \pmod{n}$ satisfy, then a is called the witness for the primality of n.

Further extracting square roots

As far as the continuation of the previous algorithm is possible we operate other extractions of root.

The end of the test

Finally if at the end of the extractions the congruence $a^r \equiv 1 \pmod{n}$ is satisfied, we also say that a is a witness for the primality n.

If a composite number passes the previous steps, we call that a strong pseudoprime.

Theorem 9.7. If n is a strong pseudoprime to the base a, then Euler pseudoprime to the base a.

If the test fails, then m is composite. Otherwise we regard w as a witness for the primality, it can be proved that the probality of m being composite is at most the $1/4$.

It means that after executing k tests the probability that the found number is not prime is $(\frac{1}{4})^k$.

We also mention, if $n < 25109$, no composite numbers pass the Miller–Rabin primality test if we apply the test for the set $\{2,5,7,13\}$ as chosen a values.

1.4. AKS algorithm

This is a deterministic primality algorithm, which was republished (see [1]) in 2002 and 2004 by three Indian mathematician Manindra Agrawal, Neeraj Kayal and Nitin Saxena. It is the first process which is deterministic, has polynomial running time and not based on any hypothesis. After publishing, Lenstra and Pomerance revised the running time of the original algorithm in their thesis in 2005 (see [9]]).

The implementation of the algorithm has had several open questions since then. The algorithm is based on a well-known identity, which says that n is a prime if and only if the next congruence is true

$$(x-a)^n \equiv (x^n - a) \pmod{n},$$

where the division algorithm has to be applied on the coefficients of the polynomial.

For the sake of better understanding we show an easy example.

Example 9.8. Prove that 5 is a prime number!

The following

$$(x-1)^5 = x^5 + 5x^4 + 10x^3 + 10x^2 + 5x + 1 \equiv (x^5 - 1) \pmod{5}$$

congruence is true, since every coefficients is 0 after dividing with 5, but the first and last coefficients.

Further we need the notation of the order.

Definition 9.9. For some natural number r the least natural number t is called the order of n if

$$n^t \equiv 1 \pmod{r}.$$

It is denoted by $ord_r(n)$.

The AKS method is based on the following theorem.

Theorem 9.10. Let $n \geq 2$ be a given integer number and let r be a positive integer $r < n$ and $ord_r(n) > \log_2(n)$. The number n is prime if and only if the following conditions are satisfied:

1. n is not a perfect power,
2. n has no prime factor which is equal to or smaller than r,
3. $(x-a)^n \equiv x^n - a \pmod{x^r - 1, n}$, for every integer a, where $1 \leq a \leq \sqrt{r}\log n$.

The congruence $(x-a)^n \equiv x^n - a \pmod{x^r - 1, n}$ in the theorem means that we determine the remainders of the polynomial dividing by $x^r - 1$. Then the coefficients are taken $\pmod{n}$.

2. Factorization of integers

The familiarized RSA algorithm is based on the fact that the factorization of integers is considered a difficult task in mathematics that is we do not know a good algorithm to determine the factors. In this part of the chapter we introduce some algorithms that may give us a chance to get the factors. In other words, this means that the developers of RSA have to be careful with these breaking methods.

2.1. Fermat factorization

First we look at a case which can be used when the composite number n can be written as the difference of two square numbers and one of the square numbers is small.

Theorem 9.11. Let n be an odd positive integers. There is a 1-to-1 correspondence between factorization of natural number n in the form $n = ab$, where $a \geq b > 0$, and representation of n in the form $n = t^2 - s^2$ where t and s nonnegative integers.

Proof.

Given such a factorization, we can write n in the following form:

$$n = ab = \left(\frac{a+b}{2}\right)^2 - \left(\frac{a-b}{2}\right)^2.$$

Conversely, given the equation

$$n = t^2 - s^2 = (t+s)(t-s).$$

Our theorem is proved. □

If $n = ab$ and a and b are close to each other, then $\frac{a-b}{2}$ "small", so t is close to $\sqrt{n}$.

Obviously the word "small" is not well defined and also strange in a mathematic book, but can be understood well after some attempts.

That is to find t we begin the attempts with $[\sqrt{n}] + 1$, then we enlarge the numbers by one at a time and we watch when $t^2 - n = s^2$ is realized. Our method will be more understandable through an example.

Example 9.12. Factorize 200819.

In our case $[\sqrt{200819}] + 1 = 449$. Then $449^2 - 200819 = 782$, which is not a perfect square. Our next attempt $t = 450$, then $450^2 - 200819 = 1681 = 41^2$. Here we managed to divide the factorizing number to the difference of two square numbers so,

$$200819 = 450^2 - 41^2 = (450+41)(450-41) = 491 \cdot 409.$$

It is obvious, that when planning RSA it is not worth to use primes close to each other. But a modified method of the Fermat algorithm can be of help in these cases as well.

In this case choose a small value k and t in the following way $[kn] + 1, [kn] + 2, [kn] + 3$ .... After choosing t let's examine the realization of $t^2 - kn = s^2$ equation. Then $(t+s)(t-s) = kn$ and $t+s$ has a non trivial common divisor with n, that is $(t+s, n)$ provides the wished result. As it turned out previously, the Euclidean algorithm can do it easily.

Example 9.13. Factorize 141467.

It turns out soon, that we cannot score fast with the basic method as we have to begin the attempts with 377.

Let $k = 3$ and try the values $[\sqrt{3n}] + 1, [\sqrt{3n}] + 2 \ldots$, that is $t = 652, 653, \ldots$. After some attempts we get that

$$655^2 - 3 \cdot 141467 = 68^2.$$

Using the Euclidean algorithm we have

$$655^2 - 3 \cdot 141467 = 68^2.$$

Finally we get the equation $141467 = 241 \cdot 587$.

Examining the result carefully, we may notice that one factor is the triple of the other, which can also justify the $k = 3$ choice.

In case of the previous methods, we can set a generalization. If we we can give a congruence

$$t^2 \equiv s^2 \pmod{n},$$

where $t \not\equiv \pm s \pmod{n}$, then we can determine a factor of n calculating $(t+s,n)$ or $(t-s,n)$.

2.2. Pollard's ρ factorization algorithm

The title method was published by John Pollard in 1975 [13]. It can be applied to determine the prime divisors of any integer, where the integer cannot be a power of a prime, and the prime divisors should be small.

The algorithm that is also called as Monte–Carlo method works in the following way, supposing that we would like to determine the prime factors of a number n.

- Let us choose a polynomial with integer coefficients, which should be simple enough for further calculations (for example $f(x) = x^2 + 1$),
- Let us choose a starting point x_0 or generate it randomly (for example $x_0 = 1$ or $x_0 = 2$),
- We calculate the next iteration,

  $$x_1 = f(x_0) \pmod{n},\ x_2 = f(f(x_0)) \pmod{n},\ \ldots$$

  that is $x_{j+1} = f(x_j) \pmod{n}\ j = 1, 2, \ldots,$
- the x_i values are compared, and we look for such values, which belong to different groups $\pmod{n}$, but to the same for $\pmod{r}$. That is we test the values $(x_i - x_j, n)$ until we get a proper divisor of n.

We note, that after some iteration we are going to discover repetition.

We assume that the polynomial f maps on itself of Z/nZ quite randomly that is all the remainders should occur in different orders.

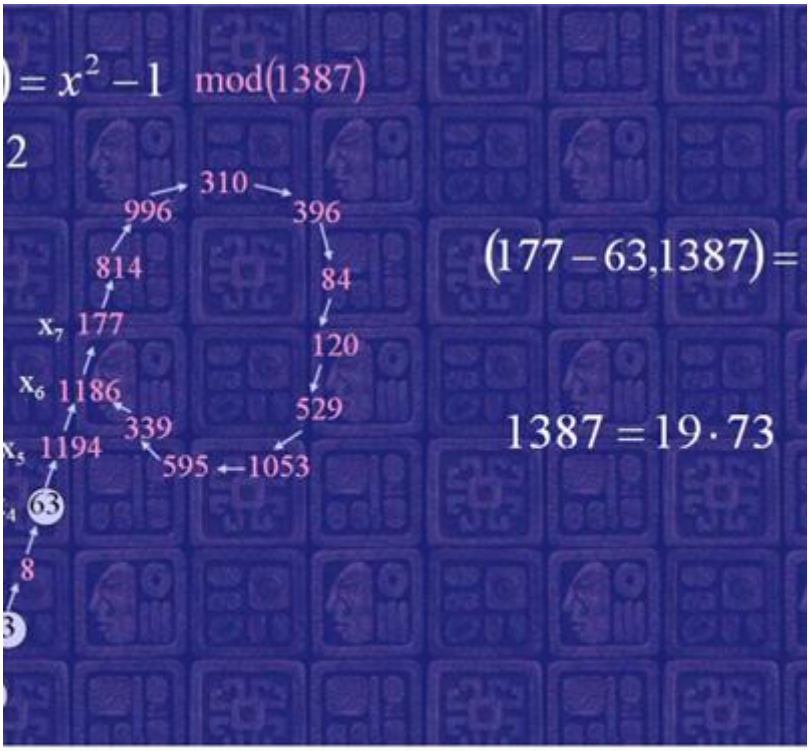
Let us see an example.

Example 9.14. Factorize the number 1387 using the Pollard's ρ method.

Let us use the polynomial $f(x) = x^2 - 1$ and the point $x_0 = 2$. The next table contains the iterations. Observe that after the 17th iteration we get back the point x_6 point so the iteration will follow this cycle hereinafter. The method got its name after the strange noose that locks in itself.

2	x_{10}	
3	x_{11}	
8	x_{12}	
63	x_{13}	
1194	x_{14}	
1186	x_{15}	
177	x_{16}	
811	x_{17}	
996	x_{18}	

Using the equation $(x_7 - x_4, n) = (177 - 63, 1387) = 19$ we get that 19 is a factor of 1387, that is $1387 = 19 \cdot 73$.



It is obviously interesting for us, how long we should search from the values $(x_i - x_j, n)$, until we get a nontrivial result. If r is a nontrivial divisor of n, we are interested in that considering all mappings of Z/nZ on itself and all the possible values x_0, to which i value exist such j in average, that $x_i \equiv x_j \pmod r$. In other words from which iteration begins the above mentioned repetition. N. Koblitz proved the following theorem concerning with it [7].

Theorem 9.15. Let S be a set of r elements. Given a map f from S to S and an element $x_0 \in S$. Let $x_{i+1} = f(x_i)$ $i = 0, 1, 2, \cdots$, let $\lambda > 0$ be a positive real number, and let $l = 1 + [\sqrt{2\lambda r}]$. Then the proportion of pairs f, x_0 for which $x_0, x_1, \ldots x_l$ are distinct, where f runs over all maps from S to S and x_0 runs over all elements of S, is less then $e^{-\lambda}$.

2.3. Quadratic sieve algorithm

The Quadratic sieve method, was first published by Carl Pomerance (see [15]).

Carl Pomerance

It rates as one of the fastest factorizing algorithm. There is only one condition for the n number to be factorized namely that no prime divisors can be larger than $\sqrt{n}$. The algorithm finds those x and y numbers which fulfills the followings

$$x^2 \equiv y^2 \pmod{n},$$
$$x \not\equiv y \pmod{n},$$
$$x \not\equiv (-y) \pmod{n}.$$

We get a factor of n by calculating $(x-y, n)$.

The algorithm uses a polynomial $f(x) = (m+x)^2 - n$, where $m = [\sqrt{n}]$ and $x = 0, \pm 1, \pm 2, \ldots$. The algorithm both determines the values of $f(x)$ and their factorizing composition.

The algorithm establishes a B threshold value and a list S, which is going to contain those p primes of which the followings are satisfied, $\left(\frac{n}{p}\right) = 1$ and $p \leq B$. In our case $\left(\frac{n}{p}\right)$ represents the Legendre symbol.

From the calculated values of $f(x)$ only those will be stored, in which factorization there is no prime factor which would not be in the S list. In the mathematics these elements are called B–smooth ([15]). The suggested value to define B is

$$B = e^{\sqrt{\ln n \ln \ln n}}.$$

If the number of list of S is k and the factors of $f(x_i)$ is in the form

$$\prod_{j=1}^{k} {p_j}^{e_{i,j}},$$

then the number of defined $f(x_i)$ has to be at least more than k by one. In each prime factorization we can order a k dimensional vector to the exponent e_i in the following way

$$v_i = (v_{i1}, v_{i2}, \ldots v_{ik}),$$

where $v_{ij} \equiv e_{ij} \pmod{2},\ j = 1, 2, \ldots k.$

Then we have to choose those vectors, which sum is $0 \pmod{2}$. The inventor of the method ensures us in this way that if we multiply these $f(x_i)$ values we get a perfect square, in our case y^2.

By multiplying the values $m + x_i$ belonging to y, we also get $m + x_i$. Now we only have to check the conditions. The next example illustrates the method well.

Example 9.16. Determine the divisors of $n = 25387$.

Let $m = \sqrt{n} = 159$ and $S = \{-1, 2, 3, 23, 41, 43, 47\}$. Apply the function $f(x) = (m+x)^2 - n$ as introduced above. We show the factorizations and the vectors v_i.

i	x	$f(x)$	$m+x$	factorization of $f(x)$	v_i
1	-2	-738	157	$(-1)\cdot 2\cdot 3^2\cdot 41$	(1 1 0 0 1 0 0
2	-6	-1978	153	$-1\cdot 2\cdot 23\cdot 43$	(1 1 0 1 0 1 0
3	10	3174	169	$2\cdot 3\cdot 23^2$	(0 1 1 0 0 0 0
4	-11	-3483	148	$(-1)\cdot 3^4\cdot 43$	(1 0 0 0 0 1 0
5	17	5589	176	$3^5\cdot 23$	(0 0 1 1 0 0 0
6	-29	-8487	130	$(-1)\cdot 3^2\cdot 23\cdot 41$	(1 0 0 1 1 0 0
7	32	11094	191	$2\cdot 3\cdot 43^2$	(0 1 1 0 0 0 0
8	80	31734	239	$2\cdot 3^2\cdot 41\cdot 43$	(0 1 0 0 1 1 0

It is easy to check that $v_4+v_5+v_6+v_7+v_8=0$. After multiplying the proper factors of $f(x)$ we get a perfect square, it is denoted by y^2, so

$$y^2=((-1)\cdot 3^4\cdot 43)\cdot(3^5\cdot 23)\cdot((-1)\cdot 3^2\cdot 23)\cdot(2\cdot 3\cdot 43^2)\cdot(2\cdot 3^2\cdot 41\cdot 43).$$

We get the following values of x and y

$$y\equiv(-1)\cdot 2\cdot 3^7\cdot 23\dot{4}1\cdot 43^2\equiv 22426 \pmod{25387},$$
$$x\equiv 130\cdot 148\cdot 176\cdot 191\cdot 239\equiv 22426 \pmod{25387}.$$

In this case we get that $x\equiv y \pmod n$, which means that x and y do not fit our aim. Let us find other x and y.

In our case $v_3+v_7=0$, after finding the proper values of $f(x)$ we have

$$y^2=(2\cdot 3\cdot 23^2)\cdot(2\cdot 3\cdot 43^2)=2^2\cdot 3^2\dot{2}3^2\cdot 43^2.$$

Now it comes easily that

$$y\equiv 2\cdot 3\cdot 23\cdot 43\equiv 5934 \pmod{25387},$$
$$x\equiv 169\cdot 191\equiv 6892 \pmod{25387}.$$

It follows that

$$x\not\equiv y \pmod n,$$
$$x\not\equiv(-y) \pmod n.$$

After that we determine the greatest common divisors, using the Euclidean algorithm, the values of $(x-y,n)$ and $(x+y,n)$. Finally we get the factors of n, that is

$$(6892-5934,25387)=479\ (6892+5934,25387)=53.$$

3. Exercises

1.

Determine a factor of 517 using the Fermat factorization.

2.

Determine the factors of 2041 using the modified Fermat factorization.

3.

Determine a factor of 25661 using the Pollard's heuristic ρ–method. Use the polynomial $f(x) = x^2 + 1$ and the point $x_0 = 2$.

4.

Determine a factor of 4087 using the Pollard's heuristic ρ–method. Use the polynomial $f(x) = x^2 + x + 1$ and the point $x_0 = 2$.

5.

Decide using a known method that 2701 is a prime or not.

6.

Determine a smallest pseudoprime to the base 5.

7.

Prove that 65 is a strong pseudoprime to the bases 8 and 18, but it not to the base 10, which is the product of 8 and 18$\pmod{65}$.

8.

Prove that 17 is a prime using the AKS algorithm.

9.

Prove that 1729 is a Carmichael number!

10.

Determine the factors of 20473 using the Quadratic sieve.

Chapter 10. Elliptic Curves

Nowadays we see the ECC abbreviation more and more often. It resolves to Elliptic Curve Cryptosystem, which is a public key cryptosystem based on elliptic curves. The advantages are that this method uses smaller keys than RSA for the same level of security and it is significantly faster.

The theory of elliptic curves dates back to the 17th century, when Isaac Newton (1642-1727) and Gottfried Wilhelm Leibniz (1646-1716) independently worked out the theory of differentiation and integrals.

Isaac Newton

Gottfried Wilhelm Leibniz

Scientists of that age happily applied these new mathematical tools for physical problems that required geometrical treatment.

The task was to determine those curves that is the trajectory of a certain “particle” under some external forces. Jakob Bernoulli (1654-1705) suggested the following problem: what is the curve, on which a rolling body travels equal amounts of distance in equally long time periods. He found the curve $(x^2+y^2)^2 = 2a^2(x^2-y^2)$, similar to a rotated 8 digit, and called it lemniscus (stripe in ancient Greek).

The curve of the above equation is usually called the *Bernoulli lemniscate*. Calculating its arc length yields the integral

$$\int_0^1 \frac{1}{\sqrt{1-x^4}}dx$$

which is called *elliptic integral* as it relates to the arc length of the ellipse. The inverse functions of this type of functions called *elliptic curves*.

Giulio Carlo Fagnano (1682-1766) italian mathematician continued this line of research, and later Leonhard Euler (1707-1783) laid the foundations for the theory of elliptic curves. The next stage of development is marked by the works of Adrien-Marie Legendre (1752-1833), Niels Henrik Abel (1802-1829) and Carl Gustav Jakob Jacobi.

The cryptographic applications of elliptic curves were first suggested independently by Neal Koblitz (University of Washington) and Victor Miller (IBM).

To motivate the mathematical study let's compare the previous 3 cryptosystems' key sizes in general and according to their standard. The values in one row have approximately the same strength ([19]).

nodulus	AES	RSA modulus	RSA
12	56	512	
61	80	1024	
56	128	3072	1
84	192	7680	2
12	256	15630	3

The table clearly shows that cryptosystems based on elliptic curves have definite advantages. However, the mathematics behind the elliptic cryptosystems is lot more complicated, therefore next we need to give some background for understanding.

1. Elliptic Curves

Here we present only the bare minimum needed for understanding the theory of elliptic curves. Deeper exposition can be found for example in [3, 8].

Definition 10.1. Let K be a field of characteristic different from 2 and 3 and let

$$x^3 + ax + b, \quad a, b \in K$$

be a cubic polynom without multiple roots. An elliptic curve over a field K is a set of points $P(x, y)$ such that the $x, y \in K$ coordinates are solutions for the equation

$$y^2 = x^3 + ax + b$$

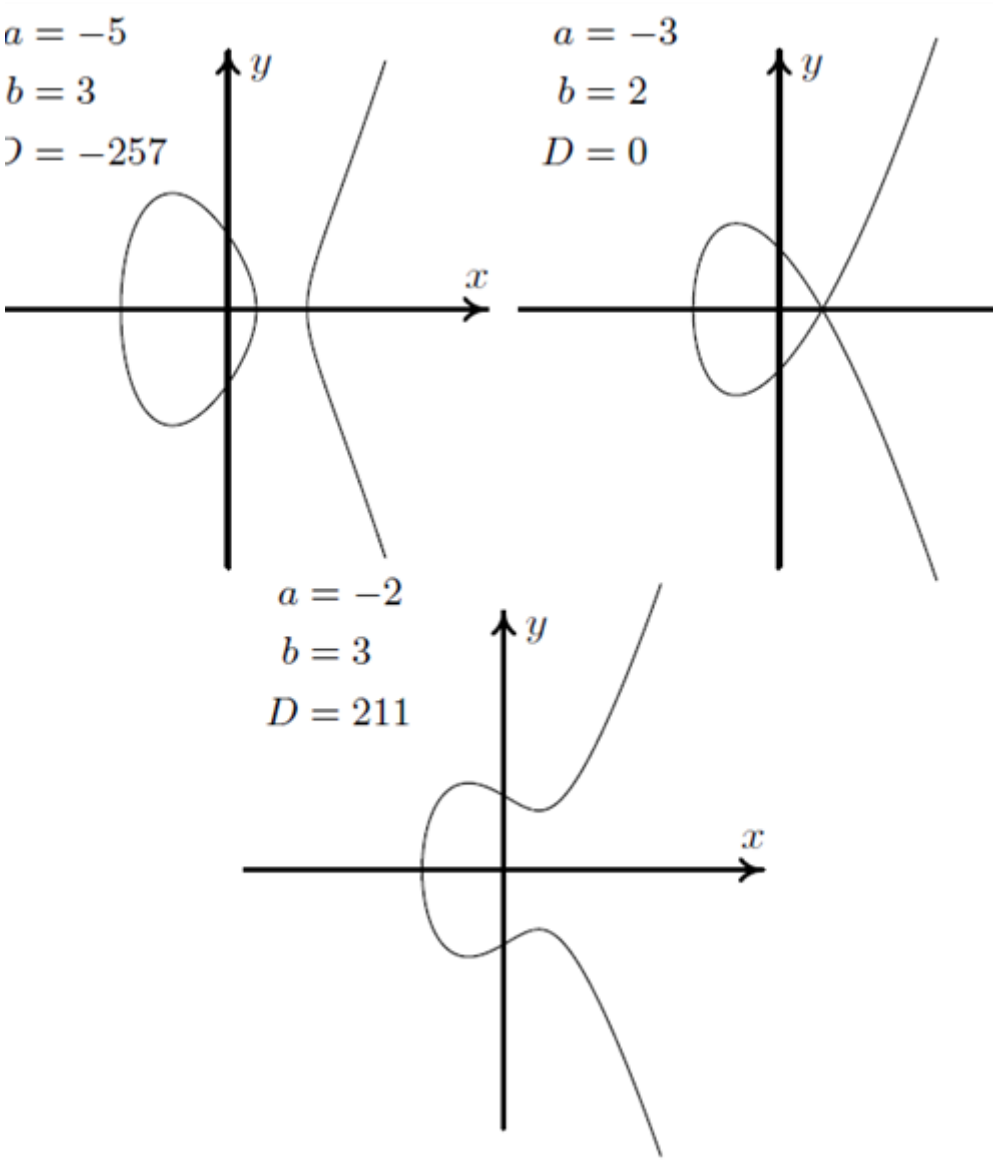
and O "the point at infinity".

The *discriminant* of an elliptic curve is the expression $D = -16(4a^3 + 27b^2)$. The discriminant is nonzero if $x^3 + ax + b$ has 3 different roots, just like here.

In case K is the field of real numbers, we can give some geometric interpretation of the discriminant. If $D \neq 0$, then the elliptic curve is nonsingular (the genus of the curve is 1). If $D = 0$ and $a = 0$, then the curve has one tangent at the singular point (cusp singularity). If $D = 0$ and $a \neq 0$, then the singular point of a curve is called a node and it has 2 distinct tangents. In this case the curve crosses itself as you can see on the examples below. In the following we do not consider the singular cases but we mention that for these curves the genus is 0.

Next we draw three elliptic curves with different discriminant values. In cryptography we need elliptic curves with no singularity, i.e. $D \neq 0$.

Figure 10.1. Elliptic curves with different discriminant values



2. Operations on Curve Points

For the previously defined nonsingular curves we define some operations.

1.

   Additive inverse of a point P

   The *additive inverse* of a point $P(x, y)$ is the point $-P$ which is the image of P through a reflection along the x axis. The image is also on the curve with coordinates $(x, -y)$.

2.

   Adding points

   Let P and Q two distinct points on the curve and denote the *sum* of them by $R = P + Q$. The operation consists of the following steps:

   1. Draw a line connecting P and Q.

   2. The line intersects the curve in a third point denoted by $-R$.

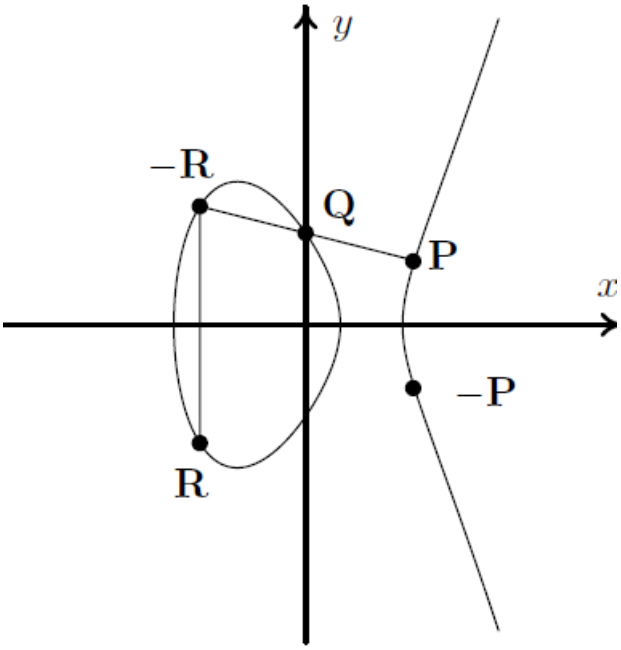
3. We take the the additive inverse of the intersection point, i.e. reflecting along the x axis, to get the point R.

3.

Doubling a point

Determining $2P$ is done similarly to method b) above, but we take the tangent at point P instead of the connecting line. The intersection determines $-2P$. As in a) we take the additive inverse.

Figure 10.2. Operations



It is worth noting that addition gives one point on the curve, except the case when we add (x, y) and $(x, -y)$. In this case we get the point in infinity which by definition belongs to the elliptic curve. This addition of the points of the curve was first suggested by Carl Gustav Jacob Jacobi in 1835.

The addition can also be done algebraically, since we just need to find the intersection points of lines and the elliptic curves. We have already seen the additive inverse, here are the other cases.

1.

Adding distinct points

If points $P(x_1, y_1)$ and $Q(x_2, y_2)$ are not additive inverses of each other, then the coordinates of the point $R = P + Q$ can be given by using the expression $s = (y_1 - y_2)/(x_1 - x_2)$.

$$x_R = s^2 - x_1 - x_2,$$
$$y_R = s(x_1 - x_R) - y_1.$$

2.

Doubling a point

Using the previous notation the coordinates of $R = 2P$ can be calculated in the following way

$$x_R = \left(\frac{3{x_1}^2 + a}{2y_1}\right)^2 - 2x_1,$$
$$y_R = -y_1 + \left(\frac{3{x_1}^2 + a}{2y_1}(x_1 - x_R)\right).$$

It can be shown that the points of the curve together with the point at infinity form an abelian group under addition. The point at the infinity behaves as the additive zero element.

So far we trusted our imagination for grasping the point at infinity, but for deeper understanding we have to mention the notion of the projective plane.

By *projective plane* we mean equivalence classes of triples of numbers (X, Y, Z) (not all components are zero), where two triples are equivalent if they can be derived from each other by scalar multiplication. We call such an equivalence class a *projective point*. If $Z \neq 0$ then there is exactly one point equivalent to $(x, y, 1)$. It is easy to see that in this case we can have a correspondence between the projective and the "normal" points of the Euclidean plane. The $Z = 0$ projective points are on the *line at infinity*. Substituting $x = \frac{X}{Z}$ and $y = \frac{Y}{Z}$ we get the equation

$$Y^2 Z = X^3 + aXZ^2 + bZ^3$$

. Letting Z to be zero we get $X = 0$. Thus there is only one point on the elliptic curve whose Z coordinate is zero, the $(0, 1, 0)$ equivalence class. This point we call the *point at infinity* and is denoted by O.

3. Elliptic curves over the field of rational numbers

In case the field K is the field of rational numbers, i.e. the coefficients a and b are rational numbers and $x, y \in Q$, we know more about the curve. In 1921 Louis Mordell proved the following theorem.

> **Theorem 10.2.** The points of an elliptic curve over the field of rational numbers form a finitely generated abelien group.

For understanding the theorem we need another definition.

> **Definition 10.3.** The *order* of a point P on an elliptic curve is the smallest natural number N such that $NP = O$.

Note that the existence of such point N is not necessary. The question, whether there exist points of finite order on an elliptic curve, is of great importance for mathematicians and cryptographers, especially over the rational field.

We even know the structure of the abelian group mentioned in Mordell's theorem. The group consists of a finitely generated torsion subgroup (points of finite order) and a subgroup of finitely many points of infinite order. This means that there exists r points $P_1, P_2, \dots P_r$ of infinite order, and $Q_1, Q_2, \dots Q_s$ points with prime power order such that all rational P points on the elliptic curve can be written as

$$P = n_1 P_1 + n_2 P_2 \cdots n_r P_r + m_1 Q_1 + m_2 Q_2 \cdots m_s Q_s,$$

where $n_i \in \mathbb{Z}$ and $m_i \in \mathbb{Z}/{p_i}^{e_i}\mathbb{Z}$. The *rank* of an elliptic curve is the number of points of infinite order.

4. Elliptic curve over finite fields

The theory of finite fields started with the works of Evariste Galois (1811-1832). In recent years finite fields have become really important due to their powerful applications (algebraic codes, cryptography). Here are two important theorems stated without proofs.

> **Theorem 10.4.** $\mathbb{Z}_n$ is a field if and only if n is a prime.

Theorem 10.5. For all prime p and all n natural numbers there exists a finite field with $q = p^n$ elements.

From now on let's denote a finite field by F_q and let E be an elliptic curve over this field. It is not difficult to see that E has at most $2q + 1$ points. This means that there are $2q$ points with (x, y) coordinate pairs and the point at infinity. We observe that there are at most two y value for each x.

However, we usually do not know how many there are actually on the elliptic curve over the finite field F_q. Helmut Hasse (1898-1979) gave an estimate.

Theorem 10.6 (Hasse). Let N be the number of F_q-points of the elliptic curve over the finite field F_q. Then

$$|N - (q + 1)| \leq 2\sqrt{q}$$

For easier understanding we consider the elliptic curve over $\mathbb{Z}_p$, so we will use the rules of modular arithmetic.

Let

$$y^2 \equiv x^3 + ax + b \pmod{p},$$

where p is prime.

We can say that if both sides of the equation give the same remainder when dividing by p then $P(x, y)$ is a point on the curve.

For the points of the curve and for the discriminant the followings hold:

1. $0 \leq x \leq p - 1$ and $0 \leq y \leq p - 1$
2. $(4a^3 + 27^2) \pmod{p} \neq 0$.

First this may seem daunting but with a closer look we can discover interesting properties and these may be helpful.

1. calculating with real numbers is slow and inaccurate, modular arithmetic is fast and accurate as it works only with integers
2. the "real" curve has infinitely many points, the modular one has far less
3. in modular arithmetic the domain of an operation is bounded, since the operand and the result are between 0 and $p - 1$,
4. modular arithmetic increases the number of cryptographic solutions.

These curves are different from the well-known real curves. While symmetry remains but in many cases not along the x axis. The next figure shows the "curve" defined by the parameters $p = 11$, $a = 1$ and $b = 0$ ([19]). We observe that

1.

the curve has 11 points,

2.

there is one point in the origin (since $b = 0$),

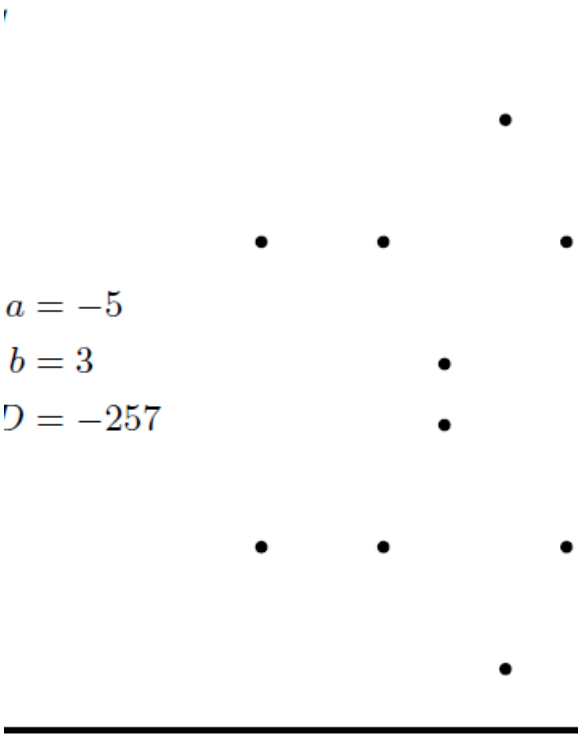
3.

10 points look scattered somewhat randomly but symmetric about the point $y = 5,5$, therefore

4.

for all x values we have 2 y values.

Figure 10.3. Az $E : y^2 \equiv x^3 + 1x + 0 \pmod{11}$ "curve"



For the sake of accuracy we list the points of the curve: $(0,0), (5,3), (7,3), (8,5), (9,1), (10,3), (5,8), (7,8), (8,6), (9,10), (10,8)$.

Now the number of points of the curve (also called the cardinality or the order of the curve denoted by $\#E(p)$) is 11 for E in the above example. This is just a coincidence. Actually, those curves that has p number of points called *anomalous curves* and banned practically by all standards since there exists an efficient attack method for ECC systems using these curves.

5. Modular operations on curve points

1.

Additive inverse

On real curves the additive inverse of $P(x, y)$ was simply $(x, -y)$. The situation is the same now, but we have to take into account the modulus. By $-y$ we mean the $p - y$ point. Looking at the previous figure we can see that the sum of y coordinates of opposite points is always p. For instance $y_R = y_P \pmod p$ and $R = -P$. Therefore we can calculate an $x_R = x_P$ point by and .

2.

Addition

The previous method of "connnecting" two points with a line and find the intersection obviously breaks down in the modular case. Point doubling also does not work in the same way. However, following the algebraic method and calculating according to the modular arithmetic is doable. Thus

$$s = (y_P - y_Q)(x_P - x_Q)^{-1} \pmod p$$
$$x_R = s^2 - x_P - x_Q \pmod p$$
$$y_R = s(x_P - x_R) - y_P \pmod p$$

6. Discrete logarithm

When introducing the knapsack problem we mentioned that each public key cryptosystem is based on a practically unsolvable method. This means that the solution would require so much time which is not in proportion with time period we have to acquire the information. The same is true for the ECC based methods. the underlying problem is called *Elliptic Curve Discrete Logarithm Problem*, ECDLP.

In 1991 few researchers worked out the RSA algorithm based on elliptic curves, but a few years later it was shown that the *ECC-like RSA* has no significant advantages. ECRSA still uses factorization as the underlying hard problem.

So far we defined three operations on the curve, addition, doubling of points and additive inverse. If we imagine the sequence

$$R, R + R, 2R + R, 3R + R$$

then we discover that we can do multiplication as well. The point we get $Q = nR$ is called the *scalar multiple*. It is easy to see that determining the natural number n is not an easy task, especially if the curve is over a $\mathbb{Z}_p$ field.

> **Definition 10.7.** Let E be an elliptic curve over the field Z_p and R a point on the curve. Then we talk about the *discrete logarithm problem* (with base R), if for a given $Q \in E$ point we want to find the natural number n such that $nR = Q$, in case such an n exists. We call n the *discrete logarithm* of Q with repect to base R.

The discrete logarithm's previously defined multiplication is essentially the same as addition on the elliptic curve.

We note, that most systems based on ECDLP are signature or key agreement based, since for fast encryption this moethod is not suitable. Next we describe some existing systems.

6.1. ECDH - Elliptic Curve Diffie - Hellman key agreement

The original Diffie-Hellman encryption algorithm solved the key agreement problem of symmetric key cipher systems. Both the sender and the reciever perform the same operations with the same public and differing private keys, but they get the same result which they could use as a key. The ECDH works the same way but it uses operations on elliptic curves instead of modular exponentiation.

> **Example 10.8.** Let's demonstrate this in an example:

Alice and Bob agrees on an elliptic curve E and the point G on the curve, called the base point. These are the public parameters of the system. Alice chooses a random number a (smaller that the order of G) and Bob acts similarly and chooses b. These choices are kept secret. As the next step of the key exchange Alice computes the point aG and sends it to Bob. Similarly Bob calculates bG and sends it to Alice. Finally Alice multiplies b (received from Bob) by a getting the abG point. In the same way Bob multiplies y by b and thus get the same result: baG. Some property of the common point (e.g. the x or y coordinate, $x+y$, or XOR $x \oplus y$, etc.) can be used for a key. Curious Eve would have to calculate abG from aG and bG, but only knows aG, bG and G and not the secret a and b numbers.

We can follow these steps in the following numerical example:

Public parameters	Let $E : y^2 \equiv x^3 + 5x + 8 \pmod{23}$, $G(8, 10)$	
0. secret parameters:	Alice chooses: $a = 7$	Bob chooses: $b = 3$
1. Preparing keys:	Alice computes: $1G(8, 10)$ $2G(13, 4)$ $3G(20, 9)$ $4G(22, 18)$ $5G(6, 1)$ $6G(2, 18)$ $7G(7, 15)$ $\mathbf{aG = (7, 15)}$	Bob computes: $1G(8, 10)$ $2G(3, 4)$ $3G(20, 9)$ $\mathbf{bG = (20, 9)}$
2.Communication:	Alice's result: $aG \rightarrow$	Bob's result: $\leftarrow bG$
3. Shared key:	$1bG(20, 9)$ $2bG(12, 18)$ $3bG(7, 8)$ $4bG(22, 5)$ $5bG(8, 13)$ $6bG(13, 4)$ $7bG(6, 1)$ $\mathbf{abG(6, 1)}$	$1aG(7, 15)$ $2aG(13, 19)$ $3aG(6, 1)$ $\mathbf{baG(6, 1)}$

6.2. EC ElGamal encryption

As the original ElGamal cipher is based on the Diffie-Hellman's problem, the elliptic ElGamal is built on ECDH:

1.

Alice and Bob choose a curve E and a base point G.

2.

They both choose random numbers a and b as private keys.

3.

Alice sends point aG to Bob as public key.

4.

Bob sends point bG to Alice as public key.

5.

If Alice wants to send a message, she maps the message to a point M (or points) of the curve and generates a random k as ephemeral key. Then she sends the message pair $(kG, M + k(bG))$ to Bob.

6.

Bob reads the message with the following method: he multiplies the first half of the message with his secret b , thus getting bkG that can be simply subtracted from the second half of the message.

6.3. ECDSA-Elliptic Curve Digital Signature Algorithm

For sending a message M signed Alice needs the following tools and parameters:

1.

an elliptic over $(\mathrm{mod}\ q)$ (public key),

2.

base point G of order n (public key, $n \geq 160$ bit),

3.

a random number d, $(1 \leq d \leq n-1)$ and a point $Q = dG$. Alice's keypair (d, Q), where d is private and Q is the public key.

7. The signing algorithm

- Alice chooses a number k between 1 and $n-1$.
- She calculates the $kG = (x_1, y_1)$ pointot and $r = x_1 \pmod n$. If the x coordinate is zero $(x_1 = 0)$, then she chooses a new k. The x coordinate of the will be one component of the signature, therefore we denote it by r.
- She computes the multiplicative inverse of k, $(k^{-1} \pmod n)$.
- She calculates the stamp of the message. For this the standard recommends the SHA-1 algorithm. Let $e = \mathrm{SHA\text{-}1}(M)$ (interpreted as a number)!
- Two other components of the signature: $s = k^{-1}(e + dr) \pmod n$. In the unfortunate case, when $s = 0$, we have to restart the whole algorithm. Here we can see that in the 2. step why r cannot be zero, since the signature would not contain any private key!
- Alice's part of the signature belonging to message M: (r, s).

8. Exercises

1.

1. Let $P = (-3, 9)$ and $Q = (-2, 8)$ be points on the elliptic curve $y^2 = x^3 - 36x$. Calculate $P + Q$ and $2P$!

2. Determine the order of point $P = (2, 3)$ on the $y^2 = x^3 + 1$ elliptic curve!

3. Consider the real elliptic curve $Y^2 - 2Y = X^3 - X^2$ and the point $P = (0, 0)$. Compute the coordinates of $2P$!

4. Given the elliptic curve $Y^2 = X^3 + X + 5 \pmod{11}$ and point $P = (0, 7)$, What are $2P, 3P, 4P, 5P, 6P, 7P, 8P, 9P, 10P$?

5. Using the elliptic curve from the previous exercise send a message using the instructions of EC-ElGamal! The message is $M = (2, 9)$, the base point $G = (0, 7)$, $b = 3$ and $k = 6$.

Bibliography

[1] M. Agrawal, N. Kayal, N. Saxena Primes is in P., Annals of Mathematics 160 (2004), 781–793.
[2] W. R. Alford, A. Granville, C. Pomerance, There are Infinitely Many Carmichael Numbers, Annals of Mathematics 140 (1994), 703–722.
[3] I. Blake, G. Seoussi, N. Smart, Elliptic curves in Cryptography, Cambridge University Press, 1999.
[4] Data Encryption Standard, Federal Information Processing Standards Publication, FIPS PUB 46-3, 1999. (http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf)
[5] D. Husemöller, Elliptic curves, Springer-Verlag, 1987.
[6] Iványi A. (szerk), Informatikai algoritmusok 1., ELTE, Eötvös Kiadó, 2004.
[7] N. Koblitz, A course in number theory and cryptography, Springer-Verlag, 1987.
[8] N. Koblitz, Introduction to Elliptic Curves and Modular Forms, Springer-Verlag, 1984.
[9] H. W. Lenstra, Jr., C. Pomerance, Primality Testing with Gaussian Periods, 2005.
[10] H. Lewis, C. Papadimitriou Elements of the Theory of Computation, Prentice-Hall, 1981.
[11] Jan C. A. Van Der Lubbe, Basic Methods of Cryptography, Cambridge University Press, 1998.
[12] A. Menezes, P. van Oorschot, and S. Vanstone Handbook of Applied Cryptography, CRC Press, 1996.
[13] J. M. Pollard, A Monte Carlo method for factorization, BIT Numerical Mathematics 15, (1975), 331–334.
[14] Márton Gyöngyvér, Kriptográfiai alapismeretek, Sciencia Kiadó Kolozsvár, 2008.
[15] C. Pomerance, A tale of two sieves, Notices Amer. Math. Soc. 43, (1996), 1473–1485.
[16] R. Rivest, R. Silverman, Are 'Strong' Primes Needed for RSA, Cryptology ePrint Archive: Report 2001/007.
[17] A. Salomaa, Public-key cryptography, Springer-Verlag, 1990.
[18] S. Singh, Kódkönyv, Park Könyvkiadó, 2007.
[19] Virasztó T., Titkosítás és adatrejtés, NetAcademia Kft., 2004.