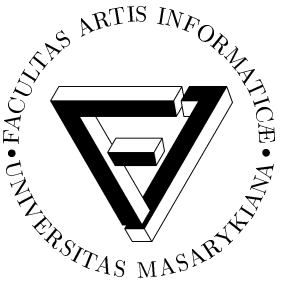
MASARYK UNIVERSITY
FACULTY OF INFORMATICS



Coding theory, cryptography and cryptographic protocols – exercises with solutions

(given in 2006)

BACHELOR THESIS

Zuzana Kuklová

Brno, Spring 2007

Declaration

Hereby I declare, that this paper is my original authorial work, which I have worked out by my own. All sources, references and literature used or excerpted during elaboration of this work are properly cited and listed in complete reference to the due source.

Advisor: prof. RNDr. Jozef Gruska, DrSc.

Acknowledgement

I would like to thank prof. RNDr. Jozef Gruska, DrSc. and Mgr. Lukáš Boháč for their inspiring comments which have essentially contributed to fulfilling of the presented work.
I am obliged to my family for understanding and furtherance.

Abstract

The main goal of this work is to present detailed solutions of exercises that have been submitted to students of the course Coding, cryptography and cryptographic protocols, given by prof. RNDr. Jozef Gruska, DrSc. in 2006 as homeworks. This way a handbook of solved exercises from coding theory and cryptography is created. Ahead of each set of new exercises we include main concepts and results from the corresponding lecture that are needed to solve exercises.

Keywords

Coding theory, code, linear code, cyclic code, cryptography, cryptosystem, cryptoanalysis, secret key cryptography, public key cryptography, digital signature, subliminal channel, elliptic curve, factorization, prime recognition, identification, authentication, bit commitment, zero knowledge proof

Contents

Introduction

The main goal of this work is to present detailed solutions of exercises that have been submitted to students of the course Coding, cryptography and cryptographic protocols, given by prof. RNDr. Jozef Gruska, DrSc. in 2006 as homeworks. The authors of exercises are Mgr. Lukáš Boháč, RNDr. Jan Bouda, Ph.D., Mgr. Ivan Fialík and Mgr. Josef Šprojcar.

This way we create a handbook of solved exercises from coding theory and cryptography that could be useful to the future students of the above course. Ahead of each set of exercises we include main concepts and results from the corresponding lecture that are needed to solve exercises. The main source of solutions presented here are solutions submitted by the students of the above course. The solutions were adopted and/or modified to achieve a uniform presentation of the exercises and of their solutions.

For some of the exercises we present not only one, but several solutions in the case sufficiently different approaches have been used in the submitted solutions. Some of the solutions are newly created. The authors of solutions are cited. The solutions, where no author is stated, were created or submitted by myself.

Ciphers and codes have been a part of human history since the time of Egyptian pharaohs. They arose from the requirement to protect secrets and messages against aliens and enemies. People were trying to protect their own secrets, as hard as they were trying to discover secrets of others. Their competition led up to invent better and better ciphers and codes that cannot be so easily broken through. And this is how the cryptography progresses till now: code makers are inventing new more sophisticated and secure ciphers and codes and code breakers try to crack them. The struggle between the code makers and the code breakers stood in the background of various historical events – it decided battles, revolts and human lives.

Today, encipherment, coding and authentication are an inseparable part of our daily life. Therefore, it is very important to know the history of ciphers, how they work and where are their weaknesses. The basics can be obtained in the course Coding, Cryptography and Cryptographic Protocols, taught at the Faculty of Informatics every year by prof. RNDr. Jozef Gruska, DrSc.

The bibliography I used as a source of information for my work and which can be useful for everyone interested in more detailed information about studied problems is listed at the end of the work. Simultaneously, there are listed some interesting web pages, where can be found more about problems, as well as some useful tools for solving exercises.

Chapter 1

Basics of Coding Theory

Coding theory has developed methods of protecting information against noise. Without coding theory and error correcting codes, there would be no deep space pictures, no satellite TV, no CD, no DVD and many more...

1.1 Definition of Code

A *code* C over an alphabet Σ is a subset of Σ^* ($C \subseteq \Sigma^*$). A q-ary code is a code over alphabet of q symbols. A binary code is a code over the alphabet $\{0, 1\}$.

The *Hamming distance* $h(x, y)$ of words x, y is the number of positions, where words x and y differs. The properties of Hamming distance are following:

1. $h(x, y) = 0 \Leftrightarrow x = y$
2. $h(x, y) = h(y, x)$
3. $h(x, z) \leq h(x, y) + h(y, z)$

An important parameter of codes is their *minimal distance* $h(C)$.

$$h(C) = \min\{h(x, y) | x, y \in C, x \neq y\},$$

$h(C)$ is the smallest number of bits needed to change one codeword into another. Code C can detect up to s errors if $h(C) \geq s + 1$. Code C can correct up to t errors if $h(C) \geq 2t + 1$.

An (n, M, d)-code C is a code such that n is the length of codewords, M is the number of codewords and d is the minimum distance of C. A good (n, M, d) code has small n and large M and d.

The main coding problem is to optimize one of the parameters n, M, d for given values of the other two. $A_q(n, d)$ is the largest M such that there is a q-ary (n, M, d)-code. It holds that

1. $A_q(n, 1) = q^n$
2. $A_q(n, n) = q$

1.2 Equivalence of Codes

Two q-ary codes are *equivalent* if one can be obtained from the other by a combination of following operations:

1. permutation of the positions of the code;

2. permutation of symbols at the fixed positions.

Any q-ary (n, M, d)-code is equivalent to an (n, M, d)-code which contains the zero codeword.

If d is odd then a binary (n, M, d)-code exists if and only if a binary $(n+1, M, d+1)$-code exists. That means that if d is odd then $A_2(n, d) = A_2(n+1, d+1)$ and if d is even then $A_2(n, d) = A_2(n-1, d-1)$.

1.3 Properties of Code

F_q^n is a set of all words of length n over alphabet $\{0, 1, \dots q-1\}$. For any codeword $u \in F_q^n$ and any integer $r \geq 0$ the sphere of radius r and center u is defined as

$$S(u, r) = \{v \in F_q^n | d(u, v) \leq r\}.$$

A sphere of radius r in F_q^n, $0 \leq r \leq n$ contains

$$\sum_{i=0}^{r} \binom{n}{i} (q-1)^i$$

words.

The sphere packing bound: If C is a q-ary $(n, M, 2t+1)$-code, then

$$M \cdot \sum_{i=0}^{t} \binom{n}{i} (q-1)^i \leq q^n. \tag{1.1}$$

A code which achieves the sphere packing bound (a code that satisfies the equality) is called a *perfect code*.

Singleton's bound: If C is a q-ary (n, M, d)-code, then

$$M \leq q^{n-d+1}. \tag{1.2}$$

Gilbert-Varshamov's bound (lower bound): For a given $d \leq n$, there exists a q-ary (n, M, d)-code with

$$M \geq \frac{q^n}{\sum_{j=0}^{d-1} \binom{n}{j} (q-1)^j} \tag{1.3}$$

and therefore

$$A_q(n, d) \geq \frac{q^n}{\sum_{j=0}^{d-1} \binom{n}{j} (q-1)^j}.$$

1.4 Entropy

Let X be a random variable (source) which takes a value x with probability $p(x)$. The *entropy* of X is defined by

$$S(X) = -\sum_x p(x) \lg p(x) \tag{1.4}$$

and it is considered to be the information content of X. Shannon's noiseless coding theorem says that in order to transmit n values of X we need to use $nS(X)$ bits. More exactly, we cannot do better and we should reach the bound $nS(X)$ as close as possible.

1.5 Exercises

Exercise 1.1

Determine $A_q(n, d)$ and write or describe the corresponding code that achieves the upper bound.

1. $A_2(8, d)$ for $d = 1$ and $d = 2$.
2. $A_2(n, 4)$ for $n = 4$, $n = 5$ and $n = 6$.
3. $A_q(4, 3)$ for $q = 2$ and $q = 3$.

Solution 1.1.1

1. $A_2(8, d)$
   (a) $d = 1$. $A_2(8, 1) = 2^8$. Code C contains all binary words of the length eight.
   (b) $d = 2$. $A_2(8, 2) = A_2(7, 1) = 2^7$. Code C contains all binary words of the length seven with the parity bit added.
2. $A_2(n, 4)$
   (a) $n = 4$. $A_2(4, 4) = 2$. $C = \{0000, 1111\}$.
   (b) $n = 5$. $A_2(5, 4) = 2$ because C contains the word 00000, then it can contain only one word with four or five ones. C cannot contain any other word because of the given minimum distance $d = 4$.
   (c) $n = 6$. $A_2(6, 4) = A_2(5, 3)$. We know from the first lesson, that $A_2(5, 3) = 4$ and one of the corresponding codes is the code $C_3 = \{00000, 01101, 10110, 11011\}$. $(6, 4, 4)$-codes exist and come from the code C_3. There is, for example, the code $C = \{000000, 011011, 101101, 110110\}$ (it is the code C_3 with a parity bit added).
3. $A_q(4, 3)$
   (a) $q = 2$. $A_2(4, 3) = 2$. Indeed, because $0000 \in C$, there is no word in C with less then three ones and there can be only one word with three or four ones in C. One of $A_2(4, 3)$-codes is code $C = \{0000, 0111\}$.
   (b) $q = 3$. $A_3(4, 3) \leq q^{n-d+1} = 3^2$ according to Singleton's bound. And we can find a ternary $(4, 9, 3)$-code $C = \{0000, 0111, 0222, 1012, 1120, 1201, 2021, 2102, 2210\}$ that reaches the Singleton's bound (1.2).

Exercise 1.2

Let $q > 1$. What is the relation ($\leq$, $\geq$ or $=$) between

1. $A_q(2n, d)$ and $A_q(n, d)$
2. $A_q(n, d)$ and $A_q(n + 2, 2d)$
3. $A_2(n, 2d - 1)$ and $A_2(n + 1, 2d)$

Solution 1.2.1

1. $A_q(2n, d) \geq A_q(n, d)$
Let $A_q(2n, d) = M_1$ and $A_q(n, d) = M_2$. We need to show that for each q, n and d it holds $M_1 \geq M_2$. To do that, we need to determine which code contains more codewords. We compute it using the Singleton bound (1.2), page 5:

$$M_1 \leq q^{2n-d+1},$$

$$M_2 \leq q^{n-d+1}.$$

We can see that

$$q^{2n-d+1} = q^n q^{n-d+1} \geq q^{n-d+1}$$

and therefore $A_q(2n, d)$ contains q^n codewords more then $A_q(n, d)$ and hence $A_q(2n, d) \geq A_q(n, d)$.

We can see that if we have two codes of different length with the same minimum distance, the code with longer codewords contains more codewords then the other code.

2. $A_q(n, d)$ and $A_q(n + 2, 2d)$ are incomparable as we can see in the following examples:
if $q = 2$, $n = 2$ and $d = 1$ then $A_2(2, 1) = 2^2 < A_2(4, 2) = A_2(3, 1) = 2^3$,
if $n = 2$ and $d = 2$ then $A_q(2, 2) = q = A_q(4, 4) = q$,
if $q = 2$, $n = 4$ and $d = 2$ then $A_2(4, 2) = A_2(3, 1) = 2^3 > A_2(6, 4) = A_2(5, 3) = 4$.

3. $A_2(n, 2d - 1) = A_2(n + 1, 2d)$ Because
$2d - 1$ is odd, we have $A_2(n, 2d - 1) = A_2(n + 1, (2d - 1) + 1) = A_2(n + 1, 2d)$.

Exercise 1.3

Consider the binary erasure channel which has two inputs (0 or 1) and three outputs (0, 1 or ?). The symbol is correctly received with probability $1 - p$ and erased with probability p. Erasure is indicated by receiving the symbol '?'.

1. Consider the nearest neighbour decoding strategy and the code

$$C = \{011, 101, 110, 000\}.$$

Calculate the probability that the received word is decoded incorrectly and the probability of error detection.

2. Consider a code C with the minimum distance $h(C) = d$. How many erasures can the code C detect and correct?

3. Consider a binary channel that has both erasures and errors. Give the lower bound for the minimum Hamming distance for a code capable of correcting all combinations of e erasures and t errors.

Solution 1.3.1

1. The received word is decoded incorrectly only if it contains two or more question marks. The probability of an erroneous decoding is:

$$3 \cdot p^2(1-p) + p^3 = 3p^2 - 3p^3 + p^3 = 3p^2 - 2p^3$$

We can detect every erasure because the question mark is not an element of the code alphabet. And we can correct one erasure in the codeword. So the probability that the received word is decoded correctly is:

$$(1-p)^3 + 3 \cdot p \cdot (1-p)^2 = 1 - 3p + 3p^2 - p^3 + 3p - 6p^2 + 3p^3 = 1 - 3p^2 + 2p^3 = 1 - (3p^2 - 2p^3)$$

2. Code C can detect every erasure – in this case we receive a symbol that cannot be sent. Code C can correct up to $d-1$ erasures in a codeword. When receiving a word with $e \leq d-1$ erased symbols, we delete positions where we received a question mark in all codewords of C. This way we get a new code C'. The length of the codewords decreases from n to $n-e$ and the $d(C)$ decreases to $d-e \geq 1$. That means, that there is still the Hamming distance $h(x,y) \geq 1$ of each two words x, y of the code C'. So we can decode the received word correctly.

3. The minimum distance for a code C capable of correcting all combinations of e erasures and t errors is $d(C) = 2t + e + 1$. When there are some (less then or equal to e) erased symbols, we transform the code C to code C' the same way as it was described above and $d(C') \geq 2t + 1$. According to the definition of Hamming distance, we can correct up to t errors in the codeword.

Exercise 1.4

You are given two dices with 6 faces. Design a binary Huffman code for encoding the sum of two dices. Compare efficiency of the proposed code with Shannon's entropy.

Solution 1.4.1

All the possible sums of two dices and their probabilities are written in the Table 1.1.

At the Figure 1.1 there you can see how to design a Huffman code for the given data. (For short, there is written 1 there instead of 1/36 and so on.)

In the Table 1.2, there are written the possible values and their codes. We can see, that it is a prefix code.

We calculate the Shannon's entropy (1.4) as follows:

$$S(X) = -\sum_x p(x) \cdot \lg p(x) \approx 3.2744$$

By Shannon's theorem, we need 3.2744 bits in average per message. Now, we calculate the efficiency E of our code:

$$E = \sum_x p(x)|code(x)| = 3 \cdot \frac{3+4+5+6+5+4}{36} + 4 \cdot \frac{2+3+2}{36} + 5 \cdot \frac{1+1}{36} \approx 3.3056$$

By using our code we need circa 0.03 bits per message (sum of two dices) more.

x	$p(x)$
2	1/36
3	2/36
4	3/36
5	4/36
6	5/36
7	6/36
8	5/36
9	4/36
10	3/36
11	2/36
12	1/36

Table 1.1: Messages and their probabilities

Figure 1.1: Design of the Huffman code

Σ	code
2	00101
3	1000
4	000
5	011
6	110
7	111
8	101
9	010
10	1001
11	0011
12	00100

Table 1.2: Messages and their codes

Exercise 1.5

You have found the belt with an ornament displayed at the Figure 1.2. It seems that the ornament is related to coding theory. Decode the hidden message.

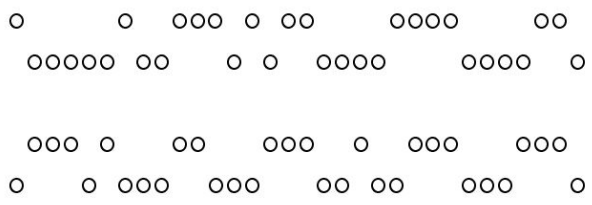


Figure 1.2: Ornament belt

Solution 1.5.1 **by Lukáš Boháč**

NRZI (Non Return to Zero, Inverted) signal encoding was used. A change of the level encodes 1, staying on the level encodes 0. The bit-string encodes the message CODE NRZI (8bit ASCII code) as you can see at the Figure 1.3.

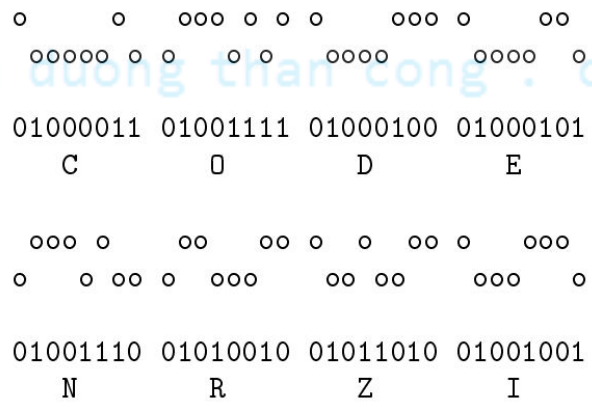


Figure 1.3: Ornament belt – the hidden message

Exercise 1.6

A single character was encoded into the following long message. Decode.

0122211020112002101101212220120012111222O1

Solution 1.6.1

The message is 42 bits long and there should be hidden only one single character. Therefore there is a strong probability that it is a graphic cipher. Our task is to form the message into a table and look for the hidden letter. The character is formed by twos when we put the message into a table with six rows and seven columns. And here we can see the letter G:

$$\begin{array}{ccccccc}
0 & 1 & 2 & 2 & 2 & 1 & 1 \\
0 & 2 & 0 & 1 & 1 & 2 & 0 \\
0 & 2 & 1 & 0 & 1 & 1 & 0 \\
1 & 2 & 1 & 2 & 2 & 2 & 0 \\
1 & 2 & 0 & 0 & 1 & 2 & 1 \\
1 & 1 & 2 & 2 & 2 & 0 & 1
\end{array}$$

The letter G is better to see when replacing the 1s and 0s by spaces:

$$\begin{array}{ccccccc}
 & 2 & 2 & 2 & & \\
2 & & & & 2 & \\
2 & & & & & \\
2 & & 2 & 2 & 2 & \\
2 & & & & 2 & \\
 & 2 & 2 & 2 & &
\end{array}$$

Chapter 2

Linear Codes

Linear codes are important because they have very concise description, very nice properties, very easy encoding and in principle quite easy decoding.

2.1 Definition of Linear Code

Linear codes are special sets of words of length n over an alphabet $\{0, \ldots, q-1\}$ where q is a power of prime.

A subset $C \subseteq V(n, q)$ is called a *linear code* if

1. for all $u, v \in C$: $u + v \in C$;
2. for all $u \in C$, $a \in GF(q)$: $au \in C$,

where $GF(q)$ is Galois field, the set $\{0, \ldots, q-1\}$ with operations $+$ and $\cdot$ taken modulo q, where q is a prime.

We can also say that a subset $C \subseteq V(n, q)$ is a linear code if one of the following conditions are satisfied:

1. C is a subspace of $V(n, q)$;
2. sum of any two codewords from C is in C (for the case $q = 2$).

If C is a k-dimensional subspace of $V(n, q)$ then C is called $[n, k]$-code and C consists of q^k codewords.

If S is a set of vectors of a vector space then $\langle S \rangle$ is the set of all linear combinations of vectors from S. For any subset S of a vector space the set $\langle S \rangle$ is a linear space that consists of the following words:

- the zero word;
- all words from S;
- all sums of two or more words from S.

The *weight of a codeword* x denoted as $w(x)$ is the number of nonzero entries of x. If $x, y \in V(n, q)$ then the Hamming distance $h(x, y) = w(x - y)$. If C is a linear code then the weight of code C, denoted as $w(C)$, is the smallest weight of all the weights of nonzero codewords from C and $w(C) = h(C)$.

If C is a linear $[n, k]$-code then it has a basis of k codewords.

A $k \times n$ matrix whose rows forms a basis of a linear $[n, k]$-code (subspace) C is said to be a *generator matrix* of C.

2.2 Equivalence of Linear Codes

Two linear codes over $GF(q)$ are *equivalent* if one can be obtained from the other by the following operations:

1. permutation of the positions of the code;
2. multiplication of symbols appearing in a fixed position by a nonzero scalar.

Two $n \times k$ matrices generate equivalent linear $[n, k]$-code over $GF(q)$ if one matrix can be obtained from the other by a sequence of the following operations:

1. permutation of the rows;
2. multiplication of a row by a nonzero scalar;
3. addition of one row to another;
4. permutation of columns;
5. multiplication of a column by a nonzero scalar.

2.3 Dual Code

If C is a linear $[n, k]$-code then the *dual code* of C, denoted as $C^{\perp}$, is defined by

$$C^{\perp} = \{v \in V(n, q) | v \cdot u = 0 \text{ if } u \in C\}.$$

We can also say that if C is a linear $[n, k]$-code with generator matrix G, then for all $v \in V(n, q)$ holds

$$v \in C^{\perp} \Leftrightarrow vG^T = 0,$$

where G^T denotes the transpose of the matrix G.

If C is a linear $[n, k]$-code over $GF(q)$ then the dual code $C^{\perp}$ is a linear $[n, n-k]$-code. A parity check matrix H of a linear $[n, k]$-code C is a generator matrix of code $C^{\perp}$.

If H is a *parity check matrix* of C then

$$C = \{x \in V(n, q) | xH^T = 0\}$$

and therefore any linear code is completely specified by its parity check matrix. The rows of a parity check matrix are parity checks on codewords.

If $G = [I_k|A]$ is the standard form of generator matrix of a linear $[n, k]$-code C, then the parity check matrix for C is $H = [-A^T|I_{n-k}]$.

2.4 Encoding with Linear Codes

Encoding of a message $u = (u_1, \ldots, u_k)$ with a linear code C with a generator matrix G is vector – matrix multiplication:

$$u \cdot G = \sum_{i-1}^{k} u_i r_i,$$

where $r_1, \ldots, r_k$ are rows of the matrix G.

If a codeword $x = x_1, \ldots, x_n$ is sent and the word $y = y_1, \ldots, y_n$ is received then $e = y - x = e_1, \ldots, e_n$ is said to be the error vector. To decode y, it is necessary to decide which x was sent or which error e occurred.

2.5 Decoding of Linear Codes

Suppose that C is a linear $[n, k]$-code over $GF(q)$ and $a \in V(n, q)$. The set

$$a + C = \{a + x | x \in C\}$$

is called a coset of C in $V(q, n)$.

If C is a linear $[n, k]$-code over $GF(q)$, then every vector of $V(n, q)$ is in some coset of C. Every coset contains exactly q^k words and every two cosets are either disjoint or identical. Each vector having the minimum weight in a coset is called a coset leader.

Nearest neighbour decoding strategy: A word y is decoded as a codeword of the first row of the column in which y occurs.

Let C be a binary $[n, k]$-code, and for $i = 0, 1, \ldots, n$ let α_i be the number of coset leaders of weight i. The probability $P_{corr}(C)$ that a received vector after decoding is the codeword which was sent is given by

$$P_{corr}(C) = \sum_{i=0}^{n} \alpha_i p^i (1-p)^{n-i}.$$

The decoder will fail to detect transmission errors if the received word y is a codeword different from the sent codeword x. Let C be a binary $[n, k]$-code and A_i be the number of codewords of C of weight i. The probability $P_{undetected}(C)$ that a an incorrect codeword is received is

$$P_{undetected}(C) = \sum_{i=0}^{n} A_i p^i (1-p)^{n-i}.$$

If H is a parity check matrix of a linear $[n, k]$-code C, then $S(y)$ is called the syndrom of y, for each $y \in V(n, q)$. The syndrom can be calculated as follows:

$$S(y) = yH^T. \tag{2.1}$$

Two words have the same syndrom if and only if they are in the same coset.

Syndrom decoding: When a word y is received, compute $S(y)$, locate the coset leader l with the same syndrom and decode y as $y - l$.

2.6 Hamming Code

An important family of simple linear codes are *Hamming codes*. Let r be an integer and H be a $r \times (2^r - 1)$ matrix whose columns are nonzero distinct words from $V(r, 2)$. The code having H as its parity check matrix is called binary Hamming code and denoted as $Ham(r, 2)$.

The Hamming code $Ham(r, 2)$ is a linear $[2^r - 1, 2^r - 1 - r]$-code, it has the minimum distance 3 and it is a perfect code. Coset leaders are words of weight less then or equal to 1. The syndrom of the word z with one at the ith position and zeroes otherwise is the transpose of the ith column of matrix H.

Decoding the Hamming codes for the case that columns of H are arranged in the order of increasing binary numbers the columns represent: when received word y compute $S(y)$, if $S(y) = 0$ then y is assumed to be the codeword sent, if $S(y) \neq 0$ then assuming a single error, $S(y)$ gives the binary position of the error.

2.7 Properties of Linear Code

Singleton bound: If C is a linear (n, k, d)-code, then

$$n - k \geq d - 1.$$

If u is a codeword of a linear code C of weight $w(u) = s$, then there is a dependence relation among s columns of any parity check matrix of C. Otherwise, any dependence relation among s columns of a parity check matrix of C yields a codeword of weight s in C.

If C is a linear code then C has *minimum weight* d if d is the largest number such that every $d - 1$ columns of any parity check matrix of C are independent.

A linear (n, k, d)-code is called *maximum distance separable* (MDS code) if $d = n - k + 1$. MDS codes are codes with maximal possible minimum weight.

2.8 Exercises

Exercise 2.1

Decide which of the following codes is linear. Find a generator matrix in standard form for linear codes.

1. 5-ary code $C_1 = \{21234, 42413, 13142, 34321, 00000\}$

2. 6-ary code $C_2 = \{201, 202, 231, 402, 403, 432, 003, 004, 033, 204, 205, 234, 405, 400, 435, 000, 001, 030, 404, 005, 200, 401, 002, 203, 433, 034, 235, 430, 031, 232, 035, 230, 431, 032, 233, 434\}$

3. Ternary code $C_3 = \{000, 201, 111, 021, 012, 120, 102, 222, 210\}$

Solution 2.1.1

1. 5-ary code $C_1 = \{21234, 42413, 13142, 34321, 00000\}$ is linear code over $GF(5)$ because for each $u, v \in C_1 : u + v \in C_1$ and for each $a \in GF(5), u \in C_1 : au \in C_1$. The generator matrix G is:

$$\begin{pmatrix} 2 & 1 & 2 & 3 & 4 \\ 4 & 2 & 4 & 1 & 3 \\ 1 & 3 & 1 & 4 & 2 \\ 3 & 4 & 3 & 2 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \rightsquigarrow \begin{pmatrix} 1 & 3 & 1 & 4 & 2 \end{pmatrix} = G$$

2. 6-ary code C_2 is not a linear code because 6 is not a power of prime.

3. Ternary code $C_3 = \{000, 201, 111, 021, 012, 120, 102, 222, 210\}$ is linear code over $GF(3)$ because for each $u, v \in C_3 : u + v \in C_3$ and for each $a \in GF(3), u \in C_3 : au \in C_3$. The

generator matrix G is:

$$\begin{pmatrix} 0 & 0 & 0 \\ 2 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 2 & 1 \\ 0 & 1 & 2 \\ 1 & 2 & 0 \\ 1 & 0 & 2 \\ 2 & 2 & 2 \\ 2 & 1 & 0 \end{pmatrix} \rightsquigarrow \begin{pmatrix} 1 & 0 & 2 \\ 0 & 1 & 2 \end{pmatrix} = G$$

Exercise 2.2

Let C be a binary code of length 6 such that for every $x_1x_2x_3 \in \{0,1\}^3 : x_1x_2x_3x_4x_5x_6 \in C$ if and only if $x_4 = x_1 + x_2$, $x_5 = x_2 + x_3$ and $x_6 = x_1 + x_2 + x_3$. Show that C is a linear code. Find a generator matrix and a parity check matrix for C.

Solution 2.2.1 by Jiří Novosad

Consider binary code C of length 6 such that for every $x_1x_2x_3 \in \{0,1\}^3 : x_1x_2x_3x_4x_5x_6 \in C \iff x_4 = x_1 + x_2, x_5 = x_2 + x_3, x_6 = x_1 + x_2 + x_3$. In the next table are shown all the codewords from the code C:

x_1	x_2	x_3	$x_1x_2x_3x_4x_5x_6$
0	0	0	000000
0	0	1	001011
0	1	0	010111
0	1	1	011100
1	0	0	100101
1	0	1	101110
1	1	0	110010
1	1	1	111001

Since C is a binary code we have to prove that $\forall x, y \in C : x+y \in C$. Let $x = x_1x_2x_3x_4x_5x_6 \in C$ and $y = y_1y_2y_3y_4y_5y_6 \in C$. If

$$z = x + y = (x_1 + y_1)(x_2 + y_2)\dots(x_6 + y_6) = z_1z_2z_3z_4z_5z_6$$

then we can see that

$$z_4 = x_4 + y_4 = x_1 + x_2 + y_1 + y_2 = z_1 + z_2$$

$$z_5 = x_5 + y_5 = x_2 + x_3 + y_2 + y_3 = z_2 + z_3$$

$$z_6 = x_6 + y_6 = x_1 + x_2 + x_3 + y_1 + y_2 + y_3 = z_1 + z_2 + z_3$$

and that means that $z \in C$ and that the code C is linear.

Code C consists of 8 codewords thus its dimension must be 3. The generator matrix for code C is:

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix} \rightsquigarrow \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} = G.$$

The parity check matrix for code C is:

$$H = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}.$$

Solution 2.2.2 by Martin Vejnár

Let $B = \{100101, 010111, 001011\}$. Then $C = \langle B \rangle$ because

$$\forall x_1, x_2, x_3 . x_1(100101) + x_2(010111) + x_3(001011) = (x_1, x_2, x_3, x_1 + x_2, x_2 + x_3, x_1 + x_2 + x_3).$$

The generator matrix for code C is

$$G = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}.$$

And the parity check matrix for code C is

$$H = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}.$$

Exercise 2.3

Find examples of a linear self-dual code of length 3 and 4. If such code does not exist, prove it.

Solution 2.3.1

There is no self-dual code C of length 3 because C must be a $[3, k]$-code where $k \in \{1, 2, 3\}$. Code $C^\perp$ must be a $[3, 3-k]$-code. But there is no k such that $k = 3 - k$.

The code $C = \{0000, 1010, 0101, 1111\}$ is a self-dual code. The generator matrix for code C is:

$$G = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}.$$

The parity check matrix H for code C is equal to matrix G. Because the generator matrix $G^\perp$ for the dual code $C^\perp$ is the parity check matrix for code C, we have $G = H = G^\perp$. So we can see that the code C is self-dual.

Exercise 2.4

Find a generator matrix and a parity check matrix for ISBN code.

Solution 2.4.1

The ISBN code is not a linear code unless we allow all position of the code to have a value from $\mathbb{Z}_{11}$ – strictly, only the last digit can be X.

The ISBN code is a 11-ary code of length 10. We use it to encode massages of length 9, so its dimension must be 9. Basically, encoding is a process of calculating the 10th position of the given message so that the following equality is fulfilled:

$$\sum_{i=1}^{10} i \cdot x_i \equiv 0 \pmod{11}$$

We can see, that the generator matrix for ISBN code is:

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 3 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 4 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 5 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 6 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 7 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 8 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 9 \end{pmatrix}.$$

The parity check matrix is following:

$$H = \begin{pmatrix} -1 & -2 & -3 & -4 & -5 & -6 & -7 & -8 & -9 & 1 \end{pmatrix} = \begin{pmatrix} 10 & 9 & 8 & 7 & 6 & 5 & 4 & 3 & 2 & 1 \end{pmatrix}.$$

Exercise 2.5

Prove that a binary Hamming code $\mathcal{H}_r$ is perfect.

Solution 2.5.1

According to the Corollary *"If C is a linear code, then C has minimum weight d, if d is the largest number such that every $d-1$ columns of any parity check matrix of C are independent."* we can see, that the minimum distance d of $\mathcal{H}_r$ is 3. Because the columns of the parity check matrix for a $\mathcal{H}_r$ consists of all non zero distinct words from $V(r,2)$, every two columns are independent. When we have words 01...1, 10...0, 1...1 of length r, we can see that the sum of the first and the second word is the third word. That means that not every 3 columns are independent and therefore the largest d is 3.

The parity check matrix H for $\mathcal{H}_r$ is a $r \times 2^r - 1$ matrix, hence the generator matrix for $\mathcal{H}_r$ is a $2^r - 1 - r \times 2^r - 1 - r + r$ matrix. That means that $\mathcal{H}_r$ is a $[2^r - 1, 2^r - 1 - r]$-code. Since the dimension of the code $\mathcal{H}_r$ is $2^r - 1 - r$, the number of codewords is 2^{2^r-1-r}. We can say that $\mathcal{H}_r$ is a $(2^r - 1, 2^{2^r-1-r}, 3)$-code.

We know that a code is perfect if it achieves the sphere packing bound (1.1), page 5. That means that the following equality must be satisfied:

$$2^{2^r-1-r}\left(\sum_{i=0}^{1}\binom{2^r-1}{i}(2-1)^i\right)=2^{2^r-1}$$

And we have:

$$2^{2^r-1-r}\left(\binom{2^r-1}{0}+\binom{2^r-1}{1}(2-1)\right)=2^{2^r-1-r}\left(1+2^r-1\right)=2^{2^r-1-r}\cdot 2^r=2^{2^r-1}$$

Now it is obvious, that $\mathcal{H}_r$ is a perfect code.

Exercise 2.6

Let $C=\{00000,10001,01010,11011,00100,10101,01110,11111\}$ be a binary linear code. List all the cosets of C. Compute a parity check matrix for C. Use syndrom decoding to decode words 00111 and 01011.

Solution 2.6.1

Code $C=\{00000,10001,01010,11011,00100,10101,01110,11111\}$ is a binary linear code because the sums of any two or more words from C falls into C. The generator matrix G of code C is

$$\begin{pmatrix}0&0&0&0&0\\1&0&0&0&1\\0&1&0&1&0\\1&1&0&1&1\\0&0&1&0&0\\1&0&1&0&1\\0&1&1&1&0\\1&1&1&1&1\end{pmatrix}\rightsquigarrow\begin{pmatrix}1&0&0&0&1\\0&1&0&1&0\\0&0&1&0&0\end{pmatrix}=G.$$

The dimension of the code is 3 and the parity check matrix H of code C is

$$H=\begin{pmatrix}0&1&0&1&0\\1&0&0&0&1\end{pmatrix}.$$

The cosets of code C are following:

- $00000+C=\{00000,10001,01010,11011,00100,10101,01110,11111\}$
- $00001+C=\{00001,10000,01011,11010,00101,10100,01111,11110\}$
- $00010+C=\{00010,10011,01000,11001,00110,10111,01100,11101\}$
- $00011+C=\{00011,10010,01001,11000,00111,10110,01101,11100\}$

There are no other cosets because there is only 2^5 binary words of length 5 and all of them are listed above.

We determine the syndrom $S(y)$ of word y as shown in (2.1), page 14. The syndromes of coset leaders are following:

coset leader $I(z)$	syndrom z
00000	00
00001	01
00010	10
00011	11

Now, we should decode words 00111 and 01011 using the syndrom decoding strategy:

- Let $y = 00111$ then $z = S(y) = 11$. The sent word was $y - I(z) = 00111 - 00011 = 00100$.
- Let $y = 01011$ then $z = S(y) = 01$. The sent word was $y - I(z) = 01011 - 00001 = 01010$.

Exercise 2.7

Let C be a binary linear code. Show that either all the codewords of C have even weight or exactly half of the codewords have even weight.

Solution 2.7.1

Let u and v be two binary words form $V(r, 2)$. If $w(u)$ and $w(v)$ are both odd or both even, the weight of their sum $w(u + v)$ is even. If $w(u)$ is even and $w(v)$ is odd (or vice versa), the weight of their sum $w(u + v)$ is odd.

That means that if there is no word $u \in C$ with odd weight, all words from C must have even weight.

In case that there is a word $u \in C$ with odd weight, the sum $x + u$ must fall into C for each $x \in C$ because C is a linear code. Now, we can define a relation α over the codewords from C so that $(x, y) \in \alpha$ if $x + y = u$. Since C is a binary code, α is symmetric relation, thus $(x, y) \in \alpha \Rightarrow (y, x) \in \alpha$. Because $w(u)$ is odd then one of $w(x)$ and $w(y)$ must be odd and the other must be even. We can easily see that $(x, y) \in \alpha$ only if $x \neq y$. In case that $x = y$ then $x + y = x + x = 0$ which is contradiction because $w(0)$ is not odd.

Because α is defined over all words from C, and two words are in relation α only if one is even and the other is odd, we can see that exactly one half of the codewords has odd weight and the other has even weight.

Solution 2.7.2 by Jiří Novosad

Let x, y be codewords. Then $x + y$ is a codeword with ones in exactly those positions, where x and y differ. If $w(x)$ and $w(y)$ are both even, then $w(x + y)$ is even too (two words with even number of ones can't differ in an odd number of positions). By the same reasoning, we get:

1. $2|w(x) \wedge 2|w(y) \Rightarrow 2|w(x + y)$
2. $2 \not| w(x) \wedge 2|w(y) \Rightarrow 2 \not| w(x + y)$
3. $2 \not| w(x) \wedge 2 \not| w(y) \Rightarrow 2|w(x + y)$

Now let us consider the three forms a generator matrix of a particular code can take (let k be the dimension of the code):

- Firstly, all the vectors in the matrix can have even weight. From (1) we get that all the generated vectors have even weight.

- If exactly one of the vectors has odd weight, vector o, then the remaining $k-1$ vectors generate a subspace E with 2^{k-1} elements, whose weight is even, from (1). The coset $E+o$ has 2^{k-1} elements with odd weight, from (2). The union of these two sets gives us the whole subspace and exactly half the vectors has odd weight and the other half has even weight.
- If there are more vectors of odd weight in the generator matrix, we can get the previous case by choosing one odd vector and adding it to all the remaining odd vectors. From (3) follows that there is only one vector with odd weight in the generator matrix.

Exercise 2.8

Let C be a binary linear code of length n. Let c_i denote the number of words of weight i in C. Suppose that $c_n = 1$. Show that $c_i = c_{n-i}$ for $i \in \{0, 1, \ldots, n\}$.

Solution 2.8.1 by Martin Vejnár

Let C be a binary linear code of length n. Let $C_i = \{c \in C | w(c) = i\}$ be a set of codewords of weight i. It holds that $c_i = |C_i|$. The only possible codeword x of weight $w(x) = n$ is $x = 1^n$. Since $c_n = 1$, $C_n = \{x\}$. Because C is a linear code, it is closed under addition and it holds $\forall c \in C.(c + x) \in C$. It can be easily observed, that $w(c + x) = n - w(c)$.

Now, we need to show that there is a bijective mapping for each $0 \leq i \leq n$. Let $f_i : C_i \to C_{n-i}$ be a mapping such that $f_i(c) = c + x$. There is also a mapping $k_i : C_{n-i} \to C_i$ such that $k_i(d) = d + x$. We can see that f_i and k_i are bijective because $f_i \circ k_i = id = k_i \circ f_i$.

Chapter 3

Cyclic Codes

Cyclic codes are of interest and importance because they posses rich algebraic structure that can be utilized in a variety of ways. They have extremely concise specifications, they can be efficiently implemented using shift registers. Many practically important codes are cyclic. In order to specify a binary cyclic code with 2^k codewords of length n it is sufficient to write down only one codeword of length n.

3.1 Definition of Cyclic Code

A code C is *cyclic* if

1. C is a linear code;
2. any cyclic shift of a codeword is also a codeword.

Comparing with linear codes, the cyclic codes are quite rare. For any field F and any integer $n \geq 3$ there are always the following trivial cyclic codes of length n over F:

- Non information code (code consisting of just one zero codeword);
- Repetition code (code consisting of codewords a^n for each $a \in F$);
- Single-parity-check code (code consisting of all codewords with parity 0);
- Non parity code (code consisting of all codewords of length n).

For some cases, there are no other cyclic codes then the four trivial cyclic codes.

3.2 Algebraic Characterization of Cyclic Codes

A codeword of a cyclic code is usually denoted as

$$a_0 a_1 \dots a_{n-1}$$

and to each such a codeword is associated the polynomial

$$a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1}.$$

A code C is cyclic if C satisfies two conditions

1. $a(x), b(x) \in C \Rightarrow a(x) + b(x) \in C$;
2. $a(x) \in C, r(x) \in R_n \Rightarrow r(x)a(x) \in C$,

where R_n is a field such that $R_n = F_q[x]/(x^n - 1)$, where $F_q[x]$ denotes the set of all polynomials over $GF(q)$.

For any $f(x) \in R_n$ the set

$$\langle f(x) \rangle = \{r(x)f(x) | r(x) \in R_n\}$$

is a cyclic code generated by the polynomial f.

Let C be a non zero cyclic code in R_n. Then there exists unique monic polynomial $g(x)$ of the smallest degree such that

- $C = \langle g(x) \rangle$
- $g(x)$ is a factor of $x^n - 1$.

If for a cyclic code C it holds

$$C = \langle g(x) \rangle,$$

then $g(x)$ is called the *generator polynomial* for code C.

The task of finding all cyclic codes of given length n is equal to the task of finding all factors of polynomial $x^n - 1$.

3.3 Generator Matrix, Parity Check Matrix and Dual Code

Suppose C s a cyclic code of length n with the generator polynomial

$$g(x) = g_0 + g_1 x + \cdots + g_r x^r.$$

Then $dim(C) = n - r$ and the generator matrix G for C is

$$G = \begin{pmatrix} g_0 & g_1 & g_2 & \cdots & g_r & 0 & 0 & 0 & \cdots & 0 \\ 0 & g_0 & g_1 & g_2 & \cdots & g_r & 0 & 0 & \cdots & 0 \\ 0 & 0 & g_0 & g_1 & g_2 & \cdots & g_r & 0 & \cdots & 0 \\ \vdots & & & & & & & & & \vdots \\ 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & g_0 & \cdots & g_r \end{pmatrix}.$$

Let C be a cyclic $[n, k]$-code with the generator polynomial $g(x)$ of order $n-k$. Polynomial $g(x)$ is a factor of $x^n - 1$. Hence

$$x^n - 1 = g(x)h(x)$$

for some polynomial $h(x)$ of degree k. Polynomial $h(x)$ is called the *check polynomial* of code C.

Let C be a cyclic code over R_n with a generator polynomial $g(x)$ and a check polynomial $h(x)$. Then any polynomial $c(x) \in R_n$ is a codeword of C if $c(x)h(x) \equiv 0 \pmod{x^n - 1}$.

Suppose C is a cyclic $[n, k]$-code with the check polynomial $h(x) = h_0 + h_1 x + \cdots + h_k x^k$, then

1. a parity check matrix for code C is

$$H = \begin{pmatrix} h_k & h_{k-1} & \cdots & h_0 & 0 & \cdots & 0 \\ 0 & h_k & \cdots & h_1 & h_0 & \cdots & 0 \\ \vdots & & & & & & \vdots \\ 0 & 0 & \cdots & 0 & h_k & \cdots & h_0 \end{pmatrix};$$

2. $C^{\perp}$ is the cyclic code generated by the reciprocal polynomial of $h(x)$:

$$\overline{h}(x) = h_k + h_{k-1}x + \cdots + h_0 x^k$$

3.4 Encoding with Cyclic Codes

Encoding using a cyclic code can be done by a multiplication of two polynomials – a message polynomial and the generator polynomial of the cyclic code.

Let C be a cyclic $[n, k]$-code over any field F with the generator polynomial $g(x) = g_0 + g_1 x + \cdots + g_r x^r$ of degree $r = n - k$. If a message vector m is represented by a polynomial $m(x)$ of degree k and m is encoded as

$$c = mG,$$

then the following relation between $m(x)$ and $c(x)$ holds

$$c(x) = m(x)g(x).$$

3.5 Hamming Code

Let r be a positive integer and H be a $r \times (2^r - 1)$ matrix whose columns are distinct nonzero vectors from $V(r, 2)$. Then the code having H as its parity check matrix is called binary Hamming code and denoted as $Ham(r, 2)$.

The binary Hamming code $Ham(r, 2)$ is equivalent to a cyclic code.

If $p(x)$ is an irreducible polynomial of degree r such that x is a primitive element of the field $F[x]/p(x)$, then $p(x)$ is called a primitive polynomial.

If $p(x)$ is a primitive polynomial over $GF(2)$ of degree r, then the cyclic code $\langle p(x) \rangle$ is the Hamming code $Ham(r, 2)$.

3.6 Exercises

Exercise 3.1

Let us consider the following definition of equivalence of two binary codes: Two binary codes are equivalent if and only if they can be transformed to each other by permutation of positions and addition of a constant vector.

Is this definition correct? Prove or disprove.

Solution 3.1.1 by Lukáš Mojžíš

The definition is correct. The standard definition of equivalence of two codes states that two codes are equivalent if and only if one can be obtained from the other by permutation of position and permutation of symbols appearing in a fixed position.

The first operation is the same in both definitions. The second operation, addition of a constant vector, can be (for binary codes) transformed into permutation. If we want to permute ith column, we add te every codeword vector $v = (v_1, v_2, \ldots, v_i, \ldots, v_n)$ where n is the length of codeword and for $i \neq j$ we have $v_i = 1$ and $v_j = 0$. Adding vector v to every codeword we achieve the permutation of ith position.

Exercise 3.2

If it holds that $C \subset C^{\perp}$, we say that C is weakly self-dual. If C is a weakly self-dual code, show that every codeword has Hamming weight divisible by 3.

Solution 3.2.1 by Zdenek Kolář

This statement is not correct.

For example code $C = \{000, 011\}$ has a dual code $C^{\perp} = \{000, 011, 111, 100\}$. We can see that $C \subset C^{\perp}$ but the only non zero codeword has Hamming weight $w(011) = 2$.

Exercise 3.3

Use the Plotkin bound and properties of $A(n, d)$ to prove that $A_2(2d, d) \leq 4d$.

Plotkin bound: Let C be a binary code with minimum distance d and length n. If $2d > n$ then

$$A(n, d) \leq 2 \left\lfloor \frac{d}{2d - n} \right\rfloor .$$

Solution 3.3.1

First we need to show that $A_q(n, d) \leq qA_q(n-1, d)$. Let $A_q(n, d) = M_1$ and $qA_q(n-1, d) = M_2$. From the Singleton's bound (1.2) (page 5) we have

$$\begin{aligned} M_1 &\leq q^{n-d+1}, \\ M_2 &\leq qq^{n-1-d+1} = q^{n-d+1}. \end{aligned}$$

We can see that the values of M_i are lower then q^{n-d+1}. Now, we need to determine the minimal values of M_i. To do that, we use the lower bound (1.3) (page 5):

$$\begin{aligned} M_{1_{min}} &= \frac{q^n}{\sum_{j=0}^{d-1} \binom{n}{j} (q-1)^j}, \\ M_{2_{min}} &= \frac{qq^{n-1}}{\sum_{j=0}^{d-1} \binom{n-1}{j} (q-1)^j}. \end{aligned}$$

It is obvious that $M_{1_{min}} \leq M_{2_{min}}$. In total, we get that $M_1 \leq M_2$, what is equal to $A_q(n, d) \leq qA_q(n-1, d)$.

Let's assume that $q = 2$ and $n = 2d$. Then we have $A_2(2d, d) \leq 2A_2(2d-1, d)$. Because $2d > 2d - 1$ we can use the Plotkin bound and we get

$$A_2(2d-1, d) \leq 2 \left\lfloor \frac{d}{2d - (2d-1)} \right\rfloor = 2d.$$

When we put this two inequalities together we get

$$A_2(2d, d) \leq 2A_2(2d-1, d) \leq 4d.$$

Exercise 3.4

Determine whether the following codes are cyclic. Explain your reasoning.

1. The ternary code $C_1 = \{0000, 1212, 2121\}$
2. The ternary code $C_2 = \{x | x \in \mathbb{Z}_3^5 \wedge w(x) \equiv 0 \pmod 3\}$
3. The ternary code $C_3 = \{x | x \in \mathbb{Z}_3^5 \wedge x_1 + x_2 + \cdots + x_5 \equiv 0 \pmod 3\}$
4. The 7-ary code $C_4 = \{x | x \in \mathbb{Z}_7^5 \wedge \sum_{i=1}^{5} i x_i \equiv 0 \pmod 7\}$

Solution 3.4.1

1. Ternary code C_1 is cyclic code. Let $c_1 = 0000$, $c_2 = 1212$ and $c_3 = 2121$, then:

   $c_1 + c_2 = c_2$, $c_1 + c_3 = c_3$, $c_2 + c_3 = c_1$, $c_2 + c_2 = c_3$, $c_3 + c_3 = c_2$
   $c_1 \to c_1$, $c_2 \to c_3 \to c_2$
   $C_1 = \langle 1 - x + x^2 - x^3 \rangle$

2. Ternary code C_2 is not cyclic. For example, words 11100 and 00111 both fall into C_2, because $w(11100) = w(00111) \equiv 0 \pmod 3$. Their sum must also fall into C_2 in case it is linear, but $w(11100 + 00111) = w(11211) \equiv 2 \pmod 3$. That is contradiction so code C_2 is not cyclic.

3. Ternary code C_3 is a cyclic code. We know, that a code is cyclic, if it is a linear code and if any cyclic shift of a codeword is also a codeword.

   Firstly, let $x = x_1x_2x_3x_4x_5$ and $y = y_1y_2y_3y_4y_5$ be words from C_3. Then $x_1 + x_2 + x_3 + x_4 + x_5 \equiv 0 \pmod 3$ and $y_1 + y_2 + y_3 + y_4 + y_5 \equiv 0 \pmod 3$, that means that $(x_1+x_2+x_3+x_4+x_5)+(y_1+y_2+y_3+y_4+y_5) \equiv 0 \pmod 3$ and because of commutativity and associativity of addition we get $(x_1+y_1)+(x_2+y_2)+(x_3+y_3)+(x_4+y_4)+(x_5+y_5) \equiv 0 \pmod 3$. And so the word $x + y = (x_1 + y_1)(x_2 + y_2)(x_3 + y_3)(x_4 + y_4)(x_5 + y_5)$ falls into C_3.

   Let $x = x_1x_2x_3x_4x_5$ be a word from code C_3 and a be a scalar < 3. We have $x_1+x_2+x_3+x_4+x_5 \equiv 0 \pmod 3$ after multiplying this equality by a we get $a(x_1+x_2+x_3+x_4+x_5) \equiv a0 \pmod 3$ because of distributivity we get $ax_1 + ax_2 + ax_3 + ax_4 + ax_5 \equiv 0 \pmod 3$ hence the word $ax = (ax_1)(ax_2)(ax_3)(ax_4)(ax_5) \in C_3$.

   Secondly, let $x = x_1x_2x_3x_4x_5$ be a word from code C_3, then $x_1 + x_2 + x_3 + x_4 + x_5 \equiv 0 \pmod 3$. And because of commutativity of addition we get $x_5 + x_1 + x_2 + x_3 + x_4 \equiv 0 \pmod 3$, that means that the word $x_5x_1x_2x_3x_4$ falls into C_3 too.

4. The 7-ary code C_4 is not cyclic. We can easily see that the word 20001 is a codeword from C_4 because $2 + 0 + 0 + 0 + 5 \equiv 0 \pmod 7$. If code C_4 is cyclic, then any cyclic shift of a codeword should be a codeword too. But the word 12000 is not a codeword because $1 + 4 + 0 + 0 + 0 \equiv 5 \pmod 7$. Hence the code C_4 cannot be a cyclic code.

Exercise 3.5

Let C_1, C_2 be q-ary cyclic codes of length n with generator polynomials $f_1(x)$, $f_2(x)$. Show that the code $C_3 = C_1 \cap C_2$ is also cyclic. Find its generator polynomial.

Solution 3.5.1

Let x and y be words from C_3. That means that $x, y \in C_1$ and $x, y \in C_2$. Because both C_1 and C_2 are cyclic, all cyclic shifts of x and y and their sums falls into C_1 so as into C_2. Hence they fall into C_3 too. And that means that C_3 is a cyclic code too.

Let $f_1(x)$ and $f_2(x)$ be the generator polynomials of C_1 and C_2, then $C_1 = \langle f_1(x) \rangle$ and $C_2 = \langle f_2(x) \rangle$. We know that $\langle f(x) \rangle = \{r(x)f(x) | r(x) \in R_n\}$ hence: $\langle f_1(x) \rangle = \{r(x)f_1(x) | r(x) \in R_n\}$ and $\langle f_2(x) \rangle = \{s(x)f_2(x) | s(x) \in R_n\}$ (multiplication is taken modulo $x^n - 1$) because $f_1(x), f_2(x) \in R_n$, there must be some $r(x) = f_2(x)$ and some $s(x) = f_1(x)$. That is the way how we get the generator polynomial of C_3, it is the polynomial $\text{lcm}(f_1(x), f_2(x))$ (modulo $x^n - 1$).

Exercise 3.6

1. Describe all binary cyclic codes of length 19.
2. How many different binary cyclic $[65, 36]$ codes are there?
3. Is it possible to find a binary cyclic $[65, 20]$ code?

Solution 3.6.1

With the help of website http://www.quickmath.com/ [6] and its factorizing tool we can get the following results:

1. There are only four trivial binary cyclic codes of length 19. Firstly, it is written in the materials, secondly, $x^{19} - 1 = (x + 1)(x^{18} + x^{17} + x^{16} + x^{15} + x^{14} + x^{13} + x^{12} + x^{11} + x^{10} + x^9 + x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + x^2 + x + 1)$ so the according cyclic codes are:
   - $\langle 1 \rangle \Rightarrow$ nonparity code
   - $\langle x + 1 \rangle \Rightarrow$ single-parity-check code
   - $\langle x^{18} + x^{17} + \cdots + x + 1 \rangle \Rightarrow$ repetition code
   - $\langle x^{19} - 1 \rangle \Rightarrow$ non-information code
2. The factors of polynomial $x^{65} - 1$ are $(x+1)(x^4+x^3+x^2+x+1)(x^{12}+x^8+x^7+x^6+x^5+x^4+1)(x^{12}+x^{10}+x^7+x^6+x^5+x^2+1)(x^{12}+x^{10}+x^9+x^8+x^6+x^4+x^3+x^2+1)(x^{12}+x^{11}+x^9+x^7+x^6+x^5+x^3+x+1)(x^{12}+x^{11}+x^{10}+x^9+x^8+x^7+x^6+x^5+x^4+x^3+x^2+x+1)$. The binary cyclic $[65, 36]$-code is generated by polynomial of degree $65 - 36 = 29$ that are factors of $x^{65} - 1$. Since $29 = 1 + 4 + 12 + 12$ we have to choose two different polynomials of degree 12, so there are $\binom{5}{2} = 10$ possibilities how can we choose them. That means that there are 10 different binary cyclic $[65, 36]$-codes.

3. The factors of polynomial $x^{65} - 1$ are written above and we can see that we get all the trivial cyclic codes, one $[65, 61]$-code, one $[65, 60]$-code, five $[65, 53]$-codes, five $[65, 52]$-codes, five $[65, 49]$-codes, five $[65, 48]$-codes, ten $[65, 41]$-codes, ten $[65, 40]$-codes, ten $[65, 37]$-codes, ten $[65, 36]$-codes, ten $[65, 29]$-codes, ten $[65, 28]$-codes, ten $[65, 25]$-codes, ten $[65, 24]$-codes, five $[65, 17]$-codes, five $[65, 16]$-codes, five $[65, 13]$-codes, five $[65, 12]$-codes, one $[65, 5]$-code and one $[65, 4]$-code. But there is no $[65, 20]$-code at all.

Exercise 3.7

Consider the polynomial $g(x) = x^3+x+1$. Show that there is a cyclic code C of length 8 over $\mathbb{F}_3$ such that $g(x)$ is its generator polynomial. Find the generator polynomial of the code $C^\perp$.

Solution 3.7.1

All cyclic codes of length 8 over field $\mathbb{F}_3$ are generated by the factors of the polynomial x^8-1. If the polynomial $g(x) = x^3 + x + 1$ is a factor of $x^8 - 1$ then it generates a $[8, 5]$-cyclic code.

First, we need to factorize the polynomial $x^8 - 1$: $x^8 - 1 = (x+1)(x-1)(x^2+x-1)(x^4 - x^3 - x - 1)$. We can see that there are two different $[8, 5]$-codes, there are their generator polynomials:

- $(x + 1)(x^2 + x - 1) = x^3 - x^2 - 1$
- $(x - 1)(x^2 + x - 1) = x^3 + x + 1 = g(x)$

Now it is obvious that $g(x)$ generates a cyclic code of length 8 over $\mathbb{F}_3$. Since $x^8 - 1 = g(x)h(x)$, the check polynomial $h(x) = (x+1)(x^4 - x^3 - x - 1) = x^5 - x^3 - x^2 + x - 1$. Hence the code $C^\perp$ is generated by the reciprocal polynomial $\overline{h}(x) = -x^5 + x^4 - x^3 - x^2 + 1$.

Exercise 3.8

Decide correctness of the following statement. Prove your decision. Let C be a code.

$$(C^\perp)^\perp = C$$

Solution 3.8.1 **by Lukáš Mojžíš**

The statement is not generally correct. For example for code $C = \{01\}$, $C^\perp = \{00, 10\}$ and then $(C^\perp)^\perp = \{00, 01\} \neq C$.

If C is a linear code then the statement is correct. We will show that if $C \subseteq F^n$ is a k-dimensional code and if $C^\perp \subseteq F^n$ is its dual code and if G is a generator matrix for C. Then $C = (C^\perp)^\perp$.

Let $g : F^n \to F^k$ is a linear map defined by the $k \times n$ generator matrix G of code C. Observe that a vector $x \in F^n$ obeys $g(x) = 0$ if and only if it is orthogonal to each of the rows of G; but the rows of G form a basis for C. Therefore $g(x) = 0$ if and only if $x \in C^\perp$. Thus $C^\perp = \ker(g)$. Since the k rows of G are linearly independent, the map g has rank k. By the Rank Nullity Theorem, the kernel has dimension $n - k$, thus $C^\perp$ is an $(n - k)$-dimensional code. Clearly $C \subseteq (C^\perp)^\perp$ since a vector $x \in C$ is orthogonal to every vector which is orthogonal to x; but C and $(C^\perp)^\perp$ both have dimension k, hence $C = (C^\perp)^\perp$.

Solution 3.8.2

This statement is true for linear and cyclic code.

First, let C be a q-ary linear $[n, k]$-code with the generator matrix G and parity check matrix H. Then the dual code $D = C^\perp$ is a linear $[n, n - k]$-code with the generator matrix $H = G'$ and the parity check matrix H'. Let $v \in V(n, q)$, $v \in D \Leftrightarrow vG^T = 0$. The dual code $E = D^\perp$ is a $[n, k]$-code with generator matrix $H' = G''$. Word $w \in E \Leftrightarrow wG'^T = 0 \Leftrightarrow wH^T = 0$.

We can easily see that if $wH^T = 0$ then $w \in C$. Because w is a word from code E then $C = E = D^\perp = (C^\perp)^\perp$.

Second, if C is a cyclic $[n, k]$-code with the generator polynomial $g(x)$ of degree $n-k$ then the check polynomial $h(x) = h_0 + h_1x + \cdots + h_kx^k$ and we know that $g(x)h(x) = x^n - 1$. We also know that the dual code $D = C^\perp$ is generated by the polynomial $\overline{h}(x) = h_k + h_{k-1}x + \cdots + h_1x^{k-1} + h_0x^k$. According to the proof of polynomial representation of dual codes we know that $\overline{h}(x) = x^k h(x^{-1})$ and $\overline{h}(x)\overline{g}(x) = 1 - x^n$. That means that $\overline{g}(x) = f(x)$ is the check polynomial (of degree $n - k$) of code D. The dual code $E = D^\perp$ is generated by the polynomial $\overline{f}(x)$. Since polynomial $f(x) = f_0 + f_1x + \cdots + f_{n-k}x^{n-k}$ then polynomial $\overline{f}(x) = f_{n-k} + f_{n-k-1}x + \cdots + f_1x^{n-k-1} + f_0x^{n-k}$. Since polynomial $g(x) = g_0 + g_1x + \cdots + g_{n-k}x^{n-k}$ and polynomial $\overline{g}(x) = g_{n-k} + g_{n-k-1}x + \cdots + g_1x^{n-k-1} + g_0x^{n-k} = f(x) = f_0 + f_1x + \cdots + f_{n-k}x^{n-k}$ we get $f_0 = g_{n-k}$, $f_1 = g_{n-k-1}$, $\ldots$,$f_{n-k} = g_0$. And now, we can easily see that $\overline{f}(x) = f_{n-k} + f_{n-k-1}x + \cdots + f_1x^{n-k-1} + f_0x^{n-k} = g_0 + g_1x + \cdots + g_{n-k-1}x^{n-k-1} + g_{n-k}x^{n-k} = g(x)$ and that means that $C = \langle g(x) \rangle = \langle \overline{f}(x) \rangle = E = D^\perp = (C^\perp)^\perp$. What we had to show.

Chapter 4

Secret Key Cryptography

Secret key cryptosystems are very old. They were primarily used in pre-computer era – secret key cryptosystems are too weak nowadays and too easy to break, especially with computers. However, they can illustrate several ideas of cryptography and cryptoanalysis.

4.1 Cryptosystem

Every cryptosystem consists of a plaintext space P (set of plaintexts over an alphabet Σ), a cryptotext space C (set of cryptotexts over an alphabet Δ) and a key space K (set of possible keys).

Each key k determines an encryption algorithm e_k and a decryption algorithm d_k such, that for any plaintext w, $e_k(w)$ is the corresponding cryptotext and it holds

$$w \in d_k(e_k(w)) \quad \text{or} \quad w = d_k(e_k(w))$$

As encryption algorithms we can also use randomized algorithms.

The philosophy of modern cryptoanalysis is embodied in the Kerckhoff's principle formulated in 1983: The security of a cryptosystem must not depend on keeping secret the encryption algorithm. The security should depend only on keeping secret the key.

The requirements for good cryptosystem according to Sir F. R. Bacon are:

1. Given e_k end a plaintext w, it should be easy to compute $c = e_k(w)$.
2. Given d_k end a cryptotext c, it should be easy to compute $w = d_k(c)$.
3. A cryptotext $e_k(w)$ should not be much longer then the plaintext w.
4. It should be unfeasible to determine w from $e_k(w)$ without knowing d_k.
5. The so called avalanche effect should hold: A small change in the plaintext, or in the key, should lead to a big change in the cryptotext.
6. The cryptosystem should not be closed under composition.
7. The set of keys should be very large.

4.2 Cryptoanalysis

The aim of cryptoanalysis is to get as much information about the plaintext or the key as possible.

Main types of cryptoanalytics attacks are: cryptotexts-only attack, known-plaintexts attack, chosen-plaintexts attack, known-encryption-algorithm attack and chosen-cryptotext attack.

Cryptotexts-only attack: The cryptoanalysts get cryptotexts $c_1 = e_k(w_1), \ldots, c_n = e_k(w_n)$ and try to infer the key k or as many of the plaintexts $w_1, \ldots, w_n$ as possible.

Known-plaintexts attack: The cryptoanalysts know some pair $w_i, e_k(w_i)$, $1 \leq i \leq n$, and try to infer key k, or at least the plaintext w_{n+1} for a new cryptotext $e_k(w_{n+1})$.

Chosen-plaintexts attack: The cryptoanalysts choose plaintexts $w_1, \ldots, w_n$ to get cryptotexts $e_k(w_1), \ldots, e_k(w_n)$, and try to infer key k or at least w_{n+1} for a new cryptotext $c_{n+1} = e_k(w_{n+1})$.

Known-encryption-algorithm attack: The encryption algorithm e_k is given and the cryptoanalysts try to get the decryption algorithm d_k.

Chosen-cryptotext attack: The cryptoanalysts know some pairs $c_i, d_k(c_i)$, $1 \leq i \leq n$, where the cryptotexts c_i have been chosen by the cryptoanalysts. The aim is to determine the key k.

The basic cryptoanalytic attack against monoalphabetic substitution cryptosystems begins with frequency analysis. Each letter in the cryptotext is counted and put into a table, so we get the frequency counts for each letter. The distribution of letters in the cryptotext is likely to be substituted for the letter with highest frequency in the plaintext language, etc. The likelihood grows with the length of cryptotext. The frequency tables can be found in the Internet.

4.3 Secret Key Cryptosystem

A cryptosystem is called *secret key cryptosystem* if some secret piece of information (the key) has to be agreed first between any two parties that want to communicate through the cryptosystem.

There are some basic types of secret key cryptosystems:

- substitution based cryptosystems – they substitute the characters of plaintext for another characters;
  - monoalphabetic cryptosystems – they use a fixed substitution, one character is always replaced with the same group of symbols;
  - polyalphabetic cryptosystems – the substitution keeps changing during the encryption;
- transposition based cryptosystems – they only transpose the characters of plaintext, for example `permission/impression`.

The cryptosystems can be also divided into block cryptosystems (cryptosystems that are used to encrypt simultaneously blocks of plaintext) and into stream cryptosystems (cryptosystems that encrypt plaintext letter by letter, the encryption may vary during the encryption process).

Stream cryptosystems are more appropriate in some applications (telecommunication), usually are simpler to implement, faster and have no error propagation. In stream cryptosystems is each block of plaintext encrypted using a different key.

In block cryptosystems, the same key is used to encrypt arbitrarily long plaintext block by block.

4.3.1 Caesar Cryptosystem

Caesar cryptosystem can be used to encrypt words in any alphabet. In order to encrypt words in English alphabet, we use the key space $K = \{0, 1, \ldots, 25\}$.

An encryption algorithm e_k substitutes any letter by the one occurring k positions ahead (cyclically) in the alphabet. A decryption algorithm d_k substitutes any letter by the one occurring k positions backwards (cyclically) in the alphabet.

4.3.2 Polybious Cryptosystem

Polybious cryptosystem is designed for encryption of words in the English alphabet without the letter J. The key space is formed by checkerboards of size 5×5 with English letters and with columns and rows labeled by symbols.

An encryption algorithm substitutes each character by the pair of symbols denoting the row and the column of the checkerboard in which the symbol is placed. The decryption algorithm substitutes every two symbols denoting the row and the column of checkerboard by the symbol that is found at these coordinates.

4.3.3 Hill Cryptosystem

In spite of the fact that Hill cryptosystem was probably never used, it played an important role in the history of modern cryptography.

The key space are matrices M of degree n with elements from the set $\{0, 1, \ldots, 25\}$ such that $M^{-1} \bmod 26$ exists. All the possible plaintexts and cryptotexts are words of length n. Any word w is represented by a column vector c_w of length n of the symbols of w. Encryption of column vector c_w is computed as $c_c = Mc_w \bmod 26$. Decryption of column vector c_c is computed as $c_w = M^{-1}c_c \bmod 26$.

4.3.4 Affine Cryptosystem

The Affine cipher is one of monoalphabetic substitution ciphers. The cryptosystem for an alphabet of size m is given by two integers a and b such that a and m are coprime and b is positive.

The encryption algorithm for letter x is $e(x) = (ax+b) \bmod m$, the decryption of received letter y is $d(y) = a^{-1}(y - b) \bmod m$, where a^{-1} is the inverse of a in the group $\mathbb{Z}_m$.

4.3.5 Playfair Cryptosystem

The playfair cipher was used in World War I by British Army. The key is a playfair square of size 5×5, or a word, or text in which repeated letters are removed and the remaining letters of alphabet (except J) are added and divided to form an array.

Encryption of a pair of letters x, y is done as follows:

- If x and y are in the same row (column), then they are replaced by the pair of symbols to the right (below) of them.
- If x and y are neither in the same row nor in the same column, then the smallest rectangle containing x and y is taken and symbols x and y are replaced by the pair of symbols in the remaining corners of the rectangle.

4.3.6 Vigenere and Autoclave Cryptosystems

Both Vigenere and Autoclave cryptosystem are polyalphabetic modifications of the Caesar cryptosystem. The key k of these cryptosystems is as long as the plaintext w. Using the Vigenere cryptosystem, a short key p is chosen and the key k is calculated as $k = prefix_{|w|}p^*$ (the prefix of length $|w|$ from the word p^*). It is a cyclic version of Caesar cryptosystem. Using the Autoclave cryptosystem, we also choose a short key p. The key k is then calculated as $k = prefix_{|w|}pw$.

4.3.7 One time pad Cryptosystem

One time pad is cryptosystem for encoding binary data using a binary key of the same length as the data. If w is binary plaintext, k binary key and c binary cryptotext, then the encryption algorithm e_k is $c = e_k(w) = w \oplus k$ and the decryption algorithm d_k is $w = d_k(c) = c \oplus k$.

4.4 Perfect Secret Cryptosystem

By Shannon, a cryptosystem is perfect if the knowledge of the cryptotext provides no information about the plaintext (with the exception of its length). It follows from Shannon's result that perfect secrecy is possible if and only if the key space is as large as the plaintext space. In addition, the key has to be as long as the plaintext and the same key should not be used twice.

A cryptosystem in which $|P| = |K| = |C|$ provides perfect secrecy if and only if every key is used with the same probability and for every plaintext $w \in P$ and cryptotext $c \in C$ there is a unique key $k \in K$ such that $e_k(w) = c$.

4.5 Exercises

Exercise 4.1

Decode the following cryptotexts:

1. `TEVSECMKOCKB`
2. `TSRLNCHHIAFCIEISIEEPR`
3. 
4. (Playfair cipher, password: `PLAYFAIR`)
   `BKLBPGQXKGFQTNQOKU`

Solution 4.1.1

1. This is a Caesar cryptosystem and every letter of the message is shifted by 10 positions ahead. To decode the message we need to shift every letter of the cryptotext 16 positions ahead or 10 positions backwards. Using this algorithm we get the message "Julius Caesar".
2. To decode this cryptotext is a bit harder. When we put the message into a table, in the columns we can see the hidden message. The message is "This is rail fence cipher" as you can see in the Table 4.1.

T	S	R	L	N	C	H
H	I	A	F	C	I	E
I	S	I	E	E	P	R

Table 4.1: Message hidden using the Rail Fence cipher

3. To decode this cryptotext we need the tables shown at the Figure 4.1.

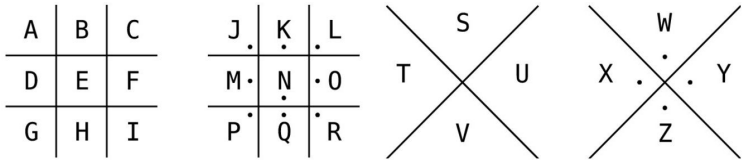


Figure 4.1: Tables for encoding and decoding messages using Pigpen cipher

And the hidden message is "Encrypted with Pigpen cipher".

4. Decode this cryptotext is very easy because we know that it is encoded with Playfair cipher and we also know the key. We just need to design the playfair square (see Table 4.2). The hidden message is "Charles Wheatstone x".

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

Table 4.2: Playfair square for the keyword `PLAYFAIR`

Exercise 4.2

Consider the following variation of the one time pad cryptosystem. Let $P = K = \{00, 01, 10\}^l$. Encryption and decryption work in the same way as in the one time pad. Decide whether this cipher is perfectly secure. Explain your reasoning.

Solution 4.2.1

A cipher is perfectly secure, if $|P| = |K| = |C|$. But we can see that $C = \{00, 01, 10, 11\}$ and therefore $|P| = |K| \neq |C|$. Now, it is obvious that this cipher is not perfectly secure.

Exercise 4.3

You have found an old cryptotext and you know that the plaintext is related to cryptography. You suppose Vigenere cryptosystem was used so you looked for repeated strings in the cryptotext. You found that the string `TICRMQUIRTJR` occurs twice in the cryptotext. The first occurrence begins at position 10 and the second one at position 241. You guess this cryptotext sequence is the encryption of the plaintext word `CRYPTOGRAPHY`. If you are right, what would be the key?

Solution 4.3.1

The key would be the word "CORRECT".

First, we need to guess the length of the keyword. We can see that the distance between this two occurrences is $241 - 10 = 231$. To guess the length of the keyword, we need to find the factors of 231 and we can see that $231 = 11 \cdot 21 = 11 \cdot 7 \cdot 3$.

Second, we get the letters of the keyword using the Vigenere encrypting table. And we get results that are shown in the table:

Position in the cryptotext	10	11	12	13	14	15	16	17	18	19	20	21
Cryptotext	T	I	C	R	M	Q	U	I	R	T	J	R
Message	C	R	Y	P	T	O	G	R	A	P	H	Y
Keyword	R	R	E	C	T	C	O	R	R	E	C	T

Now, we can see, that the length of the keyword is not 3 because there are more then three letters in the keyword. The length of the keyword is not 11 either because the 10th and 21st position of the keyword are different. Hence the length of the codeword must be 7 and we can see that the 10th and 17th position are equal and so on. Now we must find the beginning of the repeated keyword. Because its length is seven, it starts at 1st, 8th, 15th, 22nd ... position of the cryptotext – and here we get the keyword "Correct".

There is also another possibility – the length of the keyword is more then 11 symbols. In this case we are unable to say anything about the keyword.

Exercise 4.4

Alice used Vigenere cryptosystem for encryption but she has become afraid that it can be easily broken. She is now considering using double encryption, that means sender and receiver agree on two keywords key_1 and key_2 and sender encrypts message m by first encrypting it with Vigenere cipher using the key key_1 and then encrypting the resulting cryptotext with Vigenere cipher using the key key_2. Show that the proposed encryption has actually the same effect as a single Vigenere encryption using a keyword key_3 and describe how to find this keyword. What can you say about security of the double encryption?

Solution 4.4.1

Let e_k be the encryption algorithm and let d_k be the decryption algorithm described in the materials (lecture 4, site 6, 7). Let w be the plaintext and key_1 and key_2 be the two given keywords. We take $w[i]$ as the ith letter of the message w, $key_1[i']$ as the i'th character of keyword key_1 where $i' \equiv i \pmod{|key_1|}$.

The algorithm for encrypting the message w is following:

$$c[i] = e_{key_2[i']}(e_{key_1[i']}(w[i]))$$

where $i \in \{1, \ldots, |w|\}$. According to the encryption algorithm we get

$$\begin{aligned} c[i] &= e_{key_2}(w[i] + key_1[i']) \pmod{26} \\ &= (w[i] + key_1[i'] + key_2[i'']) \pmod{26} \\ &= (w[i] + key_3[i''']) \pmod{26} \end{aligned}$$

where $key_3[i'''] \equiv key_1[i'] + key_2[i''] \pmod{26}$ and $i \equiv i' \equiv i'' \equiv i''' \pmod{|key_3|}$.

Hence we can see that double Vigenere encryption using keywords key_1 and key_2 has the same effect as a single Vigenere encryption using the keyword key_3. The length n of the keyword key_3 is $\text{lcm}\{|key_1|, |key_2|\}$. And $key_3[i] = (key_1[i] + key_2[i]) \pmod{26}$ where $i \in \{1, \ldots, n\}$. Using double Vigenere encryption can be more secure than single Vigenere encryption only because of the length of key_3: $|key_3| \geq \max\{|key_1|, |key_2|\}$.

But encryption using key_3, which is dependent on keys key_1 and key_2 is less secure than encryption using a randomly chosen key key_r with the same length as key_3.

Exercise 4.5

Enigma was a family of portable electromechanical rotor machines used to encrypt and decrypt secret message during WorldWar II. Consider the following Enigma machine whose key consists of

- initial position of three different exchangeable rotors, each rotor has 26 different positions;
- plugboard setting allowing six swaps of two different characters from 26-letter alphabet.

1. How many different keys does this Enigma machine have?
2. What is the key length (in bits)?
3. What is the average complexity of an exhaustive key search?

Solution 4.5.1 by Lukáš Mojžíš

1. The number of rotor keys is $3! \cdot 26^3$. There is $3!$ because there is 6 different orders of rotors.

   The number of possible plugboard combinations according to number of cables is written in the Table 4.3. The sum of the right column of the table is our result.

Number of swaps	Number of combinations
0	1
1	$\binom{26}{2}$
2	$\frac{1}{2!}\binom{26}{2}\binom{24}{2}$
3	$\frac{1}{3!}\binom{26}{2}\binom{24}{2}\binom{22}{2}$
4	$\frac{1}{4!}\binom{26}{2}\binom{24}{2}\binom{22}{2}\binom{20}{2}$
5	$\frac{1}{5!}\binom{26}{2}\binom{24}{2}\binom{22}{2}\binom{20}{2}\binom{18}{2}$
6	$\frac{1}{6!}\binom{26}{2}\binom{24}{2}\binom{22}{2}\binom{20}{2}\binom{18}{2}\binom{16}{2}$

Table 4.3: The count of possible combinations of plugboard settings

Hence, the total number of keys is

$$\begin{aligned} &(3! \cdot 26^3) \cdot (\text{sum of right column of Table 4.3}) \\ &= (3! \cdot 26^3) \cdot 1.05578918576 \cdot 10^{11} \\ &= 1.1133930437350656 \cdot 10^{16} \end{aligned}$$

2. Key length is $\lceil \log_2(1.1133930437350656 \cdot 10^{16}) \rceil = 54$, i.e. 54 bits are necessary to encode key.

3. Using exhaustive key search we need to check $\approx \frac{1}{2} \cdot 1.1 \cdot 10^{16}$ keys on average. The complexity is therefore big.

Exercise 4.6

Find the key and decode the following ciphertext produced by Vigenere cryptosystem.

```
AjcrvqtvixmrgvlslkraykefqrlzsD4Mragocaskhym"Wuhtgmteoo",i
patvnihjwebijwjkgzucltthhrkpxlzs26cmxletusfsgdipdpcrijjwsjsf.
Tjkwwpqcchwdvjifwsunsjaugggfrfxijavqv,jwouqrytjgpsedjirv
wtkxafukpidevvijkrfer.LhgUgzjszjqsxycwhdotmhgnvqtgxhymIfiioe
esqyqrwapfaskqfvrwcvghlghympsmrrefwz;kwmfsvcpdlvvxvanvgv,lzs
ciqhcqxijsbuipdlkillticjwzafvstwfvusnef.Dikarvamlsjcrvabvaw,
atkotjgjvlshetcxagbrtwwcwtmlq:hymwagpcpgxtzkijnqnsfysipevtq
uiwlvvxpsipvipl,ojblwptkrlwfdqkztjczwtsvvmfsvcpdwrzvxze
ectlswe'agsbkpsxsgljqsrkpi,kghyixlhgumyfocasxfkeijvwublw
tarmfyoelowyjcrvdweofmtpgzwjurqrwdmpsodsuoigfuggjwhimgwixgh
hdozvxwxvkrxgfdixaop.

Crglvvzeucguwgjmniwlhgtieghvteeprcrwd.Wwblwmcelafsniwwqwkthwr
nqxzapgbljogirwl,vjiogcumruaugsxlvvMragocaskkzlijapfggmzu
axgrgvlwwlkzehapgp.Lzsimasscneehdrvidvgtwagbkpelcqwpvts
twrfeevivstkmvoatfw,tmhkpelrgsyajsu,ryktcuaalvkpiKcjtiatarf,
xzencqhhoempsnfnmyzhscptsvqfwjsdwzwd.Vjijwafbihapgpesrvqx
houumtdswwvspgtwgfhfzisdvjivwqigtlefvipl,kzblguvimnabxblw
orgvslciigueuuxgah.

Zv1944,xzeNwjloownianvtsvmqvlefezvvshzlofgatfwoahtp,gslnghlzs
Lpv(ulqeo).Lzsimasscnmllzvjsp,cqpxsabzvkssykxuzkzbl40
houkxagbj.Qxjerneuwrkpivehcydldcckk.Ahvijuceviutkpklzsgtyys,cu
hwlsiumfefkrlzsuimdymgckzsvb,xzeqrijshfzggunfxmjbkpikwkvgzab
fvigfvji40hggzbmgnu,geuzdfamliqpvwkicbmfgkpevatwmvwnv
esetweixaopqjhdixemjipi.Qgkhfnxzeugtdmutwrfeevmgfoim,yflkmi
lzsumjsunvtdmuj,vslpckv-oagv.
```

Solution 4.6.1

To decrypt the text we use the Kasiski method and frequency analysis. For example, the sequence `LZSIMASSCN` starts at 719th and at 1005th position; the sequence `MRAGOCASK` starts at 30th and at 679th position; the sequence `TWRFEEV` starts at 755th position and at 1250th position. The distances between their first and second occurrences are 286, 649 and 495. Because $\gcd(286, 649, 495) = 11$ the length of the keyword is most likely eleven. Now, it is easy to find that the keyword is "ACCESSORIES". And here is the hidden message:

```
A handy feature that was used on the M4 Enigma was the "Schreibmax", a
little printer which could print the 26 letters on a small paper ribbon.
This excluded the need for a second operator, reading the lamps and
writing the letters down. The Schreibmax was placed on top of the Enigma
machine and was connected to the lamp panel; to install the printer, the
lamp cover and all light bulbs had to be removed. Besides its handiness,
it improved operational security: the signal officer no longer had to
```

see the plaintext, as the printer might have been installed in the captain’s cabin of a submarine, so that the signals officer did the typing and keyhandling but never gained knowledge of secret received plaintext information.

Another accessory was the remote lamp panel. If the machine was equipped with an extra panel, the wooden case of the Enigma was wider and could store the extra panel. There was a lamp panel version that could be connected afterwards, but that required, just as with the Schreibmax, the lamp panel and light bulbs to be removed. The remote panel made it possible for a person to read the decrypted text, without giving the operator access to it.

In 1944, the Luftwaffe introduced an extra plugboard switch, called the uhr (clock). There was a little box, containing a switch with 40 positions. It replaced the default plugs. After connecting the plugs, as determined in the daily key sheet,the operator could turn the switch in one of the 40 positions, each position resulting in a different combination of plug wiring. Most of these plug connection are, unlike the default plugs, not pair-wise.

Chapter 5

Public Key Cryptography

The main disadvantage of the classical cryptography is the need to send a long key through a absolutely secure channel before sending the message itself. In secret key (symmetric key) cryptography both sender and receiver share the same secret key. In public key cryptography there are two different keys – a public (encryption) key and a secret (decryption) key. The basic idea is that if it is infeasible from the knowledge of encryption algorithm e_k to construct the decryption algorithm d_k then e_k can be made public.

5.1 Diffie-Hellman Key Exchange

The main problem of secret key cryptography is secure distribution of the key before transmission. The problem was solved in 1976 by Diffie and Hellman. They designed a protocol for secure key establishment over public channels.

If two parties, Alice and Bob, want to create a common secret key, then they first agree on a large prime p and a primitive root $q \pmod p$ and then they perform, through a public channel, the following activities:

- Alice chooses randomly a large integer $1 \leq x < p-1$ and computes $X = q^x \bmod p$.
- Bob also chooses randomly a large integer $1 \leq y < p-1$ and computes $Y = q^y \bmod p$.
- Alice and Bob exchange X and Y through a public channel and keep x and y secret.
- Alice computes $Y^x \bmod p$ and Bob computes $X^y \bmod p$. Then each of them has the key $K = Y^x \bmod p = X^y \bmod p = q^{xy} \bmod p$.

In order to determine x from X, p and q or y from Y, p and q, an eavesdropper seems to have the capability to compute the discrete logarithms, what is believed to be infeasible.

5.2 Blom's Key Predistribution Protocol

Blom's protocol allows trusted authority (Trent) to distribute secret keys to $\frac{1}{2}n(n-1)$ pairs of n users. Let a large prime $p > n$ be publically known. The protocol goes as follows:

- Each user U in the network is assigned by Trent a unique public number $r_U < p$.
- Trent chooses three random numbers a, b and c, smaller then p.
- For each user U Trent calculates two numbers $a_U = (a + br_U) \bmod p$ and $b_U = (b + cr_U) \bmod p$ and sends them via his secure channel to U.
- Each user U creates the polynomial $g_U(x) = a_U + b_U x$.

- If Alice (A) wants to send a message to Bob (B), then Alice computes her key $K_{AB} = g_A(r_B)$ and Bob computes his key $K_{BA} = g_B(r_A)$.
- It is easy to see that $K_{AB} = K_{BA}$ and therefore Alice and Bob can now use their keys to communicate using some secret key cryptosystem.

5.3 Cryptography and Computational Complexity

Modern cryptography uses such encryption methods that no enemy can have enough computational power and time to do decryption. Modern cryptography is based on positive and negative results of complexity theory – on the fact that for some algorithm problems no efficient algorithm seem to exist, and for some small modifications of these problems there is a simple, fast and good enough (randomized) algorithm.

Integer factorization Given $n = pq$, the task to find p, q is infeasible.

Prime recognition Is a given n prime? There is a fast randomized algorithm.

Discrete logarithm problem Given x, y, n, the task to compute a such that $y \equiv x^a \pmod{n}$ is infeasible.

Discrete square root problem Given y, n, the task to compute x such that $y \equiv x^2 \pmod{n}$ is infeasible in general, but easy if n is a prime.

5.4 RSA Cryptosystem

The most important public key cryptosystem is the RSA cryptosystem. It was invented in 1978 by Rivest, Shamir and Adleman. The basic idea is that prime multiplication is very easy but integer factorization seems to be infeasible. To design a RSA cryptosystem we need to choose two large primes p and q and compute $n = pq$. Then choose a large d such that $\gcd(d, \varphi(n)) = 1$, where φ is Euler's function, and compute $e = d^{-1} \pmod{\varphi(n)}$.

The public key is modulus n and encryption integer e. The private key is p, q and decryption integer d. The plaintext is first encoded as a word over alphabet $\{0, 1, \ldots, 9\}$, then divided into blocks of length $i - 1$ where $10^{i-1} < n < 10^i$. Each block is then taken as an integer and encrypted. The encryption of a plaintext w is the cryptotext $c = w^e \bmod n$. The decryption of a cryptotext c is the plaintext $w = c^d \bmod n$.

5.5 Rabin-Miller's Prime Recognition

One of the key problems for the development of RSA cryptosystem is the prime recognition. Rabin-Miller's prime recognition algorithm is based on the following result of number theory.

Let $n \in \mathbb{N}$, for $1 \leq x \leq n$, $C(x)$ denotes the following condition: "Either $x^{n-1} \neq 1 \pmod{n}$ or there is an $m = \frac{n-1}{2^i}$ for some i, such that $\gcd(n, x^m - 1) \neq 1$". If $C(x)$ holds for some $1 \leq x \leq n$, then n is not a prime. If n is not a prime, then $C(x)$ holds for at least half of x between 1 and n.

The algorithm goes as follows:

- Choose randomly integers $x_1, x_2, \ldots, x_m$ such that $1 \leq x_i \leq n$

- For each x_i determine whether $C(x)$ holds.
- If $C(x_i)$ holds for some x_i then n is not a prime for sure. Otherwise n is prime with the probability of error 2^{-m}.

5.6 Exercises

Exercise 5.1

1. Compute $7^{120007} \bmod 143$ by hand. (Use Chinese Remainder Theorem and Fermat's Little Theorem.)
2. Let n be any integer. Show that $n^5 - n$ is divisible by 30.

Solution 5.1.1

1. First, using Fermat's Little Theorem and modular exponentiation we get:

$$\begin{aligned} 7^{120007} \bmod 11 &= 7^7 7^{120000} \bmod 11 = 7^7 (7^{10} \bmod 11)^{12000} \bmod 11 \\ &= 7^7 1^{12000} \bmod 11 = 7^7 \bmod 11 = 6 \\ 7^{120007} \bmod 13 &= 7^7 7^{120000} \bmod 13 = 7^7 (7^{12} \bmod 13)^{10000} \bmod 13 \\ &= 7^7 1^{10000} \bmod 13 = 7^7 \bmod 13 = 6 \end{aligned}$$

Let x be a number such that $x \equiv 7^{120007} \pmod{143}$. Then we have two congruences:

$$\begin{aligned} x &\equiv 6 \pmod{11} \\ x &\equiv 6 \pmod{13} \end{aligned}$$

From Chinese Reminder Theorem we know that if $x \equiv y \pmod{a}$ and $x \equiv y \pmod{b}$ then $x \equiv y \pmod{ab}$. Hence we have $7^{120007} \equiv 6 \pmod{143}$.

2. We know, that $n^5 - n$ is divisible by 30 only if $n^5 - n$ is divisible by 2, 3 and 5 (its factors). But we have: $n^5 - n = n(n^4 - 1) = n(n^2 - 1)(n^2 + 1) = n(n-1)(n+1)(n^2+1)$. Now it is easy to see that

   $n^5 - n$ **is divisible by 2:** if $n \equiv 0 \pmod{2}$ then n is divisible by 2, else if $n \equiv 1 \pmod{2}$ then $n - 1$ is divisible by 2.

   $n^5 - n$ **is divisible by 3:** if $n \equiv 0 \pmod{3}$ then n is divisible by 3, else if $n \equiv 1 \pmod{3}$ then $n - 1$ is divisible by 3, else if $n \equiv 2 \pmod{3}$ then $n + 1$ is divisible by 3.

   $n^5 - n$ **is divisible by 5:** if $n \equiv 0 \pmod{5}$ then n is divisible by 5, else if $n \equiv 1 \pmod{5}$ then $n - 1$ is divisible by 5, else if $n \equiv 4 \pmod{5}$ then $n + 1$ is divisible by 5.

   Now, we need to get factors of $n^5 - n$ in $\mathbb{Z}_5[n]$ and these are: $n^5 - n = n(n-1)(n+1)(n-2)(n+2)$. And here we can see that if $n \equiv 2 \pmod{5}$ then $n - 2$ is divisible by 5 and if $n \equiv 3 \pmod{5}$ then $n + 2$ is divisible by 5.

   Hence $n^5 - n$ is divisible by 30.

Exercise 5.2

Alice and Bob computed a secret key k using Diffe-Hellman protocol with $p = 467$, $q = 4$, $x = 400$ and $y = 134$. Later they computed another secret key k' with the same p, q, y and with $x' = 167$. They became very surprised after finding that $k = k'$. Determine the value of both keys and explain why the keys are identical.

Solution 5.2.1

The secret key k is computed as $k = q^{xy} \bmod p$. For the values $p = 467$, $q = 4$, $x = 400$ and $y = 134$ we have $k = 161$. When we get the same value of secret key after choosing another x, say $x' = 167$, it means that $q^{xy} \bmod p = k = k' = q^{x'y} \bmod p$.

Euler's Totient Theorem says that $n^{\varphi(m)} \equiv 1 \pmod m$. It's corollary is that $n^{\varphi(m)+k} \equiv n^k \pmod m$. According to this theorem and corollary we have $k = q^{xy} \bmod p = q^{i\varphi(m)+k} \bmod p \equiv q^k \bmod p \equiv q^{j\varphi(m)+k} \bmod p = q^{x'y} \bmod p = k'$. Because p is a prime $\varphi(p) = p - 1$ and we can see that $xy \equiv x'y \pmod{p-1}$: $xy = 400 \cdot 134 \equiv 10 \pmod{466}$ and $x'y = 167 \cdot 134 \equiv 10 \pmod{466}$.

Now it is easy to see that $k = 4^{i\varphi(467)+10} \bmod 467 = 4^{10} \bmod 467 = 161 = k'$.

Exercise 5.3

To simplify the implementation of Diffe-Hellman protocol one replaces the multiplicative group $(\mathbb{Z}_p^*, \cdot)$ by the additive group $(\mathbb{Z}_p, +)$. How is security affected?

Solution 5.3.1

The Diffie-Hellman protocol is secure because it is infeasible to compute discrete logarithm. When we replace the multiplicative group $(\mathbb{Z}_p^*, \cdot)$ by the additive group $(\mathbb{Z}_p, +)$, we do not need to compute the discrete logarithm, we just need to compute the inverse elements in group $(\mathbb{Z}_p, +)$ which is easy. That means that this simplified Diffie-Hellman protocol is not secure.

In the Table 5.1 we can see the run of modified Diffie-Hellman protocol and also what Alice, Bob and Eve knows. At the end of the key exchange protocol Eve doesn't know the

Alice	Bob	Eve
prime p, base q	prime p, base q	prime p, base q
x	y	doesn't know x nor y
$qx \bmod p$	$qx \bmod p$	$qx \bmod p$
$qy \bmod p$	$qy \bmod p$	$qy \bmod p$
$k = qxy \bmod p$	$k = qxy \bmod p$	doesn't know secret key k

Table 5.1: The design of the run of modified Diffie-Hellmann key exchange protocol

secret key, but it is easy for her to compute $q^{-1} \bmod p$ such that $q^{-1}q \equiv 1 \pmod p$. She also knows that $qk = qxqy \bmod p$. Here she gets $k = q^{-1}qk = q^{-1}qxqy \bmod p = qxy \bmod p$.

Exercise 5.4

Consider RSA cryptosystem. Is it possible to decrypt a ciphertext by repeated encryption of the ciphertext?

Solution 5.4.1 **by Martin Vejnár**

The RSA cryptosystem is based on the fact that $m^{ed} \equiv m \pmod{n}$, where e, coprime of $\varphi(n)$, is a multiplicative inverse of d. Since e is a coprime of $\varphi(n)$, the Euler's theorem can be applied to get

$$e^{\varphi(\varphi(n))} = ee^{\varphi(\varphi(n))-1} \equiv 1 \pmod{\varphi(n)}.$$

Using the Fermat's little theorem we get

$$m^{ee^{\varphi(\varphi(n))-1}} = c^{e^{\varphi(\varphi(n))-1}} \equiv m \pmod{n}.$$

Now it is obvious that a cryptotext can be decoded by encrypting the cryptotext $\varphi(\varphi(n)) - 1$ times.

Exercise 5.5

Consider the following modification of RSA cryptosystem.
Public key is a pair $(n = pq, e)$ which is defined in the same way as in the standard RSA. Private key is a quintuple $(p, q, d_p, d_q, q_{inv})$ where $d_p = e^{-1} \bmod p - 1$, $d_q = e^{-1} \bmod q - 1$ and $q_{inv} = q^{-1} \bmod p$. Message m is encrypted by computing $c = m^e \bmod n$. Decryption is realized by computing numbers $m_p = c^{d_p} \bmod p$, $m_q = c^{d_q} \bmod q$ and $h = q_{inv}(m_p - m_q) \bmod p$ from which the original message $m = m_q + hq$ can be reconstructed. Show correctness of the described cryptosystem.

Solution 5.5.1

In plain RSA it holds that $m \equiv c^d \pmod{n}$. We can see that $d_p = d \bmod p - 1$ and $d_q = d \bmod q - 1$. Since

$$w \equiv c^d \equiv c^{d \bmod p-1} \equiv c^{d_p} \pmod{p} \text{ and}$$
$$w \equiv c^d \equiv c^{d \bmod q-1} \equiv c^{d_q} \pmod{q}$$

we can see that

$$m \equiv m^{ed} \equiv c^d \equiv c^{d_p} \equiv m_p \pmod{p} \text{ and}$$
$$m \equiv m^{ed} \equiv c^d \equiv c^{d_q} \equiv m_q \pmod{q}.$$

Now, we need to show that message $m' = m_q + hq$ is the original message. To do that we need to verify whether $m' \equiv m_p \pmod{p}$ and $m' \equiv m_q \pmod{q}$. We can easily see that

$$m' = m_q + hq \equiv m_q \pmod{q}.$$

To show that $m' \equiv m_p \pmod{p}$ we will start with $m_p - m_q \equiv m_p - m_q \pmod{p}$. Because $qq_{inv} \equiv 1 \pmod{p}$, we get

$$\underbrace{q_{inv}(m_p - m_q)}_{h} q \equiv m_p - m_q \pmod{p}$$
$$m' = m_q + hq \equiv m_p \pmod{p}.$$

Because $|m| < pq$, $m' \equiv m \pmod{p}$ and $m' \equiv m \pmod{q}$ it must hold that $m = m'$ and hence the modified cryptosystem is correct.

Exercise 5.6

We want to set up RSA cryptosystem in a network of n users.

1. How many prime numbers do we have to generate?
2. Now consider we want to reduce this number by generating a smaller pool of prime numbers and making combinations of two of these primes (for each user we pick up a new pair). How is security of RSA cryptosystem affected?

Solution 5.6.1

1. We need two primes for each user, so we need to generate $2n$ prime numbers.
2. When there is only k prime numbers, we can calculate $\gcd(n_i, n_j)$ where n_i, n_j are public keys of some users U_i and U_j of the network. When $\gcd(n_i, n_j) = x > 1$, we can compute such y and y' that $xy = n_i$ and $xy' = n_j$. Because e_i and e_j are known, we can also compute d_i and d_j.

   Now, we know the private keys of users U_i and U_j of the network and we can read messages addressed to them. This cryptosystem is not secure.

Chapter 6

Other Public Key Cryptosystems

6.1 Rabin Cryptosystem

Rabin cryptosystem is based on the discrete square root problem. To design Rabin cryptosystem, we need to find two primes p and q of the form $4k + 3$ (i.e. $p \equiv q \equiv 3 \pmod 4$). The public key is number $n = pq$, primes p and q are kept secret. We can see that n is a Blum integer what is important because of its nice properties.

The encryption of plaintext $w < n$ is the cryptotext $c = w^2 \bmod n$. The decryption of cryptotext c is done when all square roots of c modulo n are found. Because n is a Blum integer, we can see that $w \in \{c^{\frac{p+1}{4}} \bmod n, p - c^{\frac{p+1}{4}} \bmod n, c^{\frac{q+1}{4}} \bmod n, q - c^{\frac{q+1}{4}} \bmod n\}$. In case the plaintext w is a meaningful text, it should be easy to determine the value of w. However, if w is a random string, for example a secret key, it is impossible to determine the value of w.

6.2 ElGamal Cryptosystem

To design ElGamal cryptosystem, we need to choose a large prime p, a primitive element q of the group $\mathbb{Z}_p^*$ and a random integer x such that $1 \leq x < p$. Then we need to calculate $y = q^x \bmod p$. The public key consists of numbers p, q and y. Number x is kept secret for message decryption.

To encrypt a plaintext $w \in \mathbb{Z}_p^*$ we need to choose a random integer r and compute $a = q^r \bmod p$ and $b = y^r w \bmod p$. The cryptotext $c = (a, b)$. To decrypt a cryptotext, we need to calculate $\frac{b}{a^x} \bmod p = ba^{-x} \bmod p = w$. The security of ElGamal cryptosystem is based on the discrete logarithm problem. As we can see, the cryptosystem is not secure under a chosen cryptotext attack – for an encryption $c = (a, b)$ of message m we can easy construct an encryption $c' = (a, 2b)$ of message $2m$.

6.3 Exercises

Exercise 6.1

Show that with a chosen-ciphertext attack on RSA cryptosystem one can decrypt an arbitrary ciphertext with only one query.

Solution 6.1.1 by Libor Caha

A chosen-ciphertext attack is an attack in cryptoanalysis in which the cryptoanalyst chooses a ciphertext and let it decrypt. From some pairs of ciphertexts and decrypted ciphertexts the cryptoanalyst wants to know some information about the key or about the messages sent.

Our task is to decrypt an arbitrary ciphertext c using the chosen-ciphertext attack. Say the ciphertext c was sent to Alice, whose public key is (n, e).

We need to choose a random integer $r \in \mathbb{Z}_n^*$ and compute $c' = r^e c \bmod n$ where e is Alice's public key. We send c' to Alice and she sends back $d(c') = c'^d \bmod n$. Because $c'^d \bmod n = r^{ed} c^d \bmod n = rw \bmod n$ we have the message we are looking for multiplied by r. To get our message w, we need to calculate $r^{-1} c'^d \bmod n = r^{-1} rw \bmod n = w$.

Exercise 6.2

What is the probability that two students of IV054 have the same birthday (74 students attend IV054 course at this moment)?

Solution 6.2.1

Let $p(n)$ be the probability that two out of n students of IV054 have the same birthday.

Let $\overline{p}(n) = 1 - p(n)$ be the probability that no two students have the same birthday. Then the probability $p(n)$ is computed this as $p(n) = 1 - \overline{p}(n)$. The probability $\overline{p}(n)$ is computed as follows:

$$\overline{p}(n) = \frac{365(365-1)\cdots(365-n+1)}{365^n} = \frac{365!}{365^n(365-n)!}$$

For $n = 74$ we have $\overline{p}(74) = \frac{365!}{365^{74}(365-74)!} \approx 0,00035$ and $p(74) = 1 - \overline{p}(74) \approx 1 - 0,00035 \approx 0,99965$.

The probability that two students of IV054 have the same birthday is greater than 99,9%, although it cannot be 100% unless there are at least 365 people attending the course.

Exercise 6.3

Prove or disprove the following implication. Let g, h be generators of the group $(\mathbb{Z}_p^*, \cdot)$ where p is an odd prime. Suppose $g^{2u} \equiv h^{2v} \pmod{p}$. Then $g^u \equiv h^v \pmod{p}$.

Solution 6.3.1

The condition is not true. We can choose the prime $p = 7$, then we can see that the group $(\mathbb{Z}_7^*, \cdot)$ has two generators $g = 3$ and $h = 5$. When choosing $u = 1$ and $v = 2$ we get:

$$g^{2u} = 3^2 \equiv 2 \pmod{7} \qquad h^{2v} = 5^4 \equiv 2 \pmod{7}$$
$$g^u = 3^1 \equiv 3 \pmod{7} \qquad h^v = 5^2 \equiv 4 \pmod{7}$$

We can see that $g^{2u} \equiv h^{2v} \pmod{7}$ but $g^u \not\equiv h^v \pmod{7}$.

Exercise 6.4

Let p be a large prime, g a generator of the group $(\mathbb{Z}_p^*, \cdot)$ and $y = g^x \bmod p$. Show that it is possible to find the least significant bit of x by computing $y^{\frac{p-1}{2}} \bmod p$.

Solution 6.4.1 by Libor Caha

To find the least significant bit of x is equal to find the parity of x. Because g is a generator of group $(\mathbb{Z}_p^*, \cdot)$ then $\mathbb{Z}_p^* = g^i$ for $1 \leq i \leq p-1$. The expression $y^{\frac{p-1}{2}} \mod p$ can be rewrite as $g^{\frac{x\varphi(p)}{2}} \mod p$ (using the Euler's Totient Theorem). Now, we need to discuss two cases:

$x = 2k$**:** If x is even then the least significant bit is 0. We can see that $y^{\frac{p-1}{2}} = g^{\frac{x\varphi(p)}{2}} = g^{\frac{2k\varphi(p)}{2}} = g^{k\varphi(p)} \equiv 1 \pmod{p}$.

$x = 2k+1$**:** If x is odd then the least significant bit is 1. We can see that $y^{\frac{p-1}{2}} = g^{\frac{x\varphi(p)}{2}} = g^{\frac{(2k+1)\varphi(p)}{2}} = g^{k\varphi(p)} g^{\frac{\varphi(p)}{2}} \not\equiv 1 \pmod{p}$ because g is the generator of group $\mathbb{Z}_p^*$.

We can say that if $y^{\frac{p-1}{2}} \equiv 1 \pmod{p}$ then x is even.

Exercise 6.5

Show that Rabin cryptosystem is vulnerable to a chosen-ciphertext attack.

Solution 6.5.1 by Lukáš Mojžíš

Suppose that $\mathcal{D}$ is any algorithm that computes one plaintext (of the four possible plaintexts) corresponding to a valid cryptotext y. We choose a random $x \in \mathbb{Z}_n^*$ and compute $y = x^2 \mod n$. Now we need to compute $x' = \mathcal{D}(y)$ and we get $x^2 \equiv x'^2 \pmod{n}$. The probability that $x = \pm x' \pmod{n}$ is 0,5. In this case we have $x^2 - x'^2 = (x-x')(x+x') \equiv 0 \pmod{n}$ but neither factor is equal to 0 modulo n. Therefore, $\gcd((x-x'), n) = p$ or q and the factorization of n is obtained. After two attempts (on average) n is factored. Therefore any decryption algorithm can be used to factor n efficiently.

After factoring n we can decipher any ciphertext using normal decoding strategy of Rabin cryptosystem.

Exercise 6.6

Let p be a 1024-bit prime. Let g have order q in $(\mathbb{Z}_p^*, \cdot)$, where q is a 160-bit prime. Consider the following modification of ElGamal cryptosystem. Private key x is a randomly chosen element of $\{1, \ldots, q-1\}$. Public key is $y = g^x \mod p$. Message m is encrypted by computing pair

$$c = (g^r \mod p, y^r g^m \mod p),$$

where r is a randomly chosen integer.

1. Suppose you know factorization of $p-1$. Show a method of finding $g \in \{0, \ldots, p-1\}$ of order q.

2. How can the receiver compute $g^m \mod p$ from c?

3. Computing discrete logarithms is hard in $(\mathbb{Z}_p^*, \cdot)$. In general, the receiver is not able to recover m from $g^m \mod p$. Assume the sender only sends messages from the set $\{0, \ldots, 100\}$. Show that the receiver can recover m.

4. Suppose $c_1 = (g^{r_1} \bmod p, y^{r_1} g^{m_1} \bmod p)$ is a cryptotext for some unknown message m_1 and similarly $c_2 = (g^{r_2} \bmod p, y^{r_2} g^{m_2} \bmod p)$ is a cryptotext for some unknown message m_2. Find a cryptotext c for a message $m' = m_1 + m_2 \bmod q$ using c_1 and c_2.

5. Suppose the receiver is conducting an auction in which two bidders encrypt their bids using the scheme described above. Suppose also that both bidders can bid at most $100. The bidder who goes second can eavesdrop messages between the receiver and the first bidder. Show that he can almost always bid $1 more than the first bidder (even without knowing the value of his bid).

Solution 6.6.1

1. Because q is a prime and it is order of some element of $\mathbb{Z}_p^*$ then q must be a factor of $p-1$ because every order of some group element divides the size of the group. To get any element of order q we randomly choose a number x from 1 to $p-1$ and when it holds $x^q \equiv 1 \pmod p$ then x might be g. There is more elements of order q, not only g. Because q is a prime we don't need to check whether it has a smaller order – there are no factors of q.

2. Let $c = (a, b)$ where $a = g^r \bmod p$ and $b = y^r g^m \bmod p$. Then

$$\frac{b}{a^x} = \frac{y^r g^m}{g^{rx}} = \frac{g^{rx} g^m}{g^{rx}} = g^m$$

Because a is an element of a group, there must be some inverse a^{-1} in the group and this element and multiplication is used instead of a and division.

3. Because the set $M = \{1, \ldots, 100\}$ is finite, we can calculate the values of g^a for each $a \in M$ in a finite time. We save our results in a table as a couple (g^a, a). Then we can easily search the table for a g^m and find the appropriate m.

4. Let c' be the cryptotext for message $m' = m_1 + m_2 \bmod q$ and let c_i be the cryptotext for message m_i, then $c' = c_1 \cdot c_2$:

$$\begin{aligned} c' &= (g^s \bmod p, y^s g^{m_1+m_2 \bmod q} \bmod p) \\ &= (g^{r_1+r_2} \bmod p, y^{r_1+r_2} g^{m_1+m_2} \bmod p) = \\ &= (g^{r_1} g^{r_2} \bmod p, y^{r_1} g^{m_1} y^{r_2} g^{m_2} \bmod p) = \\ &= (a_1 \cdot a_2, b_1 \cdot b_2) = (a_1, b_1) \cdot (a_2, b_2) = \\ &= c_1 \cdot c_2. \end{aligned}$$

This is correct because r_1, r_2 are random numbers so as s and $g^{m_1+m_2} \equiv g^{m_1+m_2 \bmod q} \pmod p$ because $m_1 + m_2 = kq + l$ and $(m_1 + m_2) \bmod q = l$ we get $g^{kq+l} = g^{kq} g^l = g^l$ because q is the order of g.

5. The second bidder needs to calculate c_1 for message $m_1 = 1$. When he eavesdrops the encrypted bid b_i he just calculates $c_1 \cdot b_i$ and sends it to the auctioneer. He bids $1 more than the first bidder except for the case that the first bidder bids $100. In this case, the second bid is not valid.

Chapter 7

Digital Signature

Digital signatures are one of the most important applications of modern cryptography. Digital signatures are such that each user is able to verify signatures of other users, but that gives him no information about how to sign a message on behind of other users.

An important difference from handwritten signature is that digital signature of a message is intimately connected with the message and for different messages is different, whereas the handwritten signature is adjoint to the message and always looks the same.

Technically, a digital signature is performed by a signing algorithm and it is verified by a verification algorithm. It can be used any public key cryptosystem in which the plaintext space and cryptotext space are the same.

The *signature of message* w denoted as $sig(w)$ is $d_U(w)$ so as everyone can verify that the message was sent by user U. If the signature is important only for user V, then the signature is computed as $e_V(d_U(w))$. Now, only user V can verify, that the message was signed by user U.

7.1 Digital Signature Scheme

Digital signature allows anyone to verify signature of sender S without providing any information about generating signatures of S.

A *digital signature scheme* (M, S, K_s, K_v) is given by a set of messages to be signed (M), a set of possible signatures (S), a set of private keys for signing (K_s) and a set of public keys for verification (K_v).

It is required, that for each key k from K_s, there exists a single and easy to compute *signing mapping* $sig_k : \{0,1\}^* \times M \to S$, and for each key k from K_v, there exists a single and easy to compute *verification mapping* $ver_k : M \times S \to \{true, false\}$ such that the following conditions are satisfied:

Correctness: For a message $m \in M$ and public key $k \in K_v$, it holds $ver_k(m, s) = true$ if there is an $r \in \{0,1\}^*$ such that $s = sig_l(r, m)$ for a private key $l \in K_s$ corresponding to the public key k.

Security: For any $w \in M$ and $k \in K_v$, it is computationally infeasible, without the knowledge of the private key corresponding to k, to find a signature $s \in S$ such that $ver_k(w, s) = true$.

7.2 Attacks on Digital Signature

There are several types of attack on digital signature schemes:

Total break: The adversary manages to recover secret key from the public key.

Universal forgery: The adversary can derive from the public key an algorithm which allows him to forge signature of any message.

Selective forgery: The adversary can derive from the public key a method to forge signatures of selected messages (where the selection was made prior the knowledge of the public key).

Existential forgery: The adversary is able to create from the public key a valid signature of some message m (but has no control for which m).

7.3 RSA Signatures

Let us have an RSA cryptosystem with encryption and decryption exponents e and d. The signature of message w is a couple $s = (w, \sigma)$ where $\sigma = w^d \bmod n$. The signature s is valid if $\sigma^e = w \bmod n$.

There are some known attacks on this scheme. The forger can use some public key e to compute w^e, the signature of this message is $s = (w^e, w)$. Everybody, who verifies the signature, finds out that the signature is valid. The forger has no control over the content of the message w^e – it is an example of existential forgery.

Another attacker can produce some new valid signatures without the knowledge of secret key – when he obtains signatures $s_1 = (w_1, \sigma_1)$ and $s_2 = (w_2, \sigma_2)$, he can compute valid signatures of messages $w_1 w_2$ and w_1^{-1}. The signatures are $s_{12} = (w_1 w_2, \sigma_1 \sigma_2)$ and $s' = (w_1^{-1}, \sigma_1^{-1})$.

7.4 ElGamal Signatures

The public key for ElGamal signature scheme is $K = (p, q, y)$ where p is a prime, q is a primitive element of $\mathbb{Z}_p^*$ and $y = q^x \bmod p$. The integer $1 \leq x < p$ is secret key and is used for signing messages.

To create the signature s of a message m we need to choose a random integer $r \in \mathbb{Z}_{p-1}^*$. The signature $s = sig(m, r) = (a, b)$ where $a = q^r \bmod p$ and $b = (m - ax)r^{-1} \bmod p - 1$. The signature $s = (a, b)$ of message m is valid if $y^a a^b \equiv q^w \pmod p$.

There are ways of producing (using ElGamal signature scheme) valid forged signatures, but they do not allow the forger to create signature of message of his choice (see Exercise 7.1). There are also several ways of breaking the ElGamal signatures if these schemes are used not carefully enough. If the random integer r of some signature is known, the forger can compute the secret key x and then forge signatures at will. Another misuse of ElGamal signature scheme is to use the same r to sign two messages. In such a case the secret key x can be computed.

7.5 Digital Signature Algorithm

Digital Signature Algorithm (DSA) was accepted in 1994 as a standard.

The key for DSA is $K = (p, q, r, x, y)$, where p is a large prime, q is a prime dividing $p - 1$, $r > 1$ is a qth root of 1 in $\mathbb{Z}_p$ ($r = h^{\frac{p-1}{q}} \bmod p$ where h is a primitive element in $\mathbb{Z}_p$), x is a random integer such that $0 < x < q$ and $y = r^x \bmod p$. The components of public key are p, q and r; integers x and y are kept secret.

To sign a message w we need to choose a random integer k such that $0 < k < q$ and $\gcd(k, q) = 1$. The signature of message w is $s = sig(w, k) = (a, b)$, where $a = (r^k \bmod p) \bmod q$ and $b = k^{-1}(w + xa) \bmod q$, where $kk^{-1} \equiv 1 \pmod q$. The signature $s = (a, b)$ is valid if $(r^{u_1} y^{u_2} \bmod p) \bmod q = a$, where $u_1 = wz \bmod q$, $u_2 = az \bmod q$ and $z = b^{-1} \bmod q$.

7.6 Ong-Schnorr-Shamir Subliminal Channel Scheme

A subliminal channel is a covert communication channel among a set of users, such that everybody can see their common messages, without any secret information. The top secret message is hidden in the sent message and its signature.

To set up a subliminal channel, the users need to choose a large n and an integer k such that $\gcd(k, n) = 1$. The public key is a couple (n, h), where $h = k^{-2} \bmod n = (k^{-1})^2 \bmod n$. The secret key for all subliminal channel users is k.

When some user wants to send a secret message w, he needs to choose another harmless message w'. The messages have to be such that $\gcd(w, n) = 1$ and $\gcd(w', n) = 1$. The signature of those messages is $s = (S_1, S_2)$, where

$$S_1 = \frac{1}{2}\left(\frac{w'}{w} + w\right) \bmod n \text{ and}$$
$$S_2 = \frac{k}{2}\left(\frac{w'}{w} - w\right) \bmod n.$$

The signature s is for everybody just a signature of message w', for the subliminal channel users are the message w' and its signature s only numbers needed to get the hidden message w.

The signature of message w' is valid if $S_1^2 - hS_2^2 \bmod n = w'$. The hidden message w can be obtained by computing $w'(S_1 + k^{-1}S_2)^{-1} \bmod n$.

7.7 Lamport Signature Scheme

Lamport signature scheme shows how to construct a signature scheme for only one use from any one way function. This signature cannot be forged because we are unable to invert the one way function. On the other hand, Lamport signature scheme can be used to sign only one message.

Let k be a positive integer and let $P = \{0, 1\}^k$ be the set of messages. Let $f : Y \to Z$ be a one way function where Y is a set of partial signatures. $Y = \{y_{ij} | 1 \leq i \leq k, j = 0, 1\}$, where y_{ij} is chosen randomly and $Z = \{z_{ij} | z_{ij} = f(y_{ij})\}$. The key K consists of f, Y and $Z - Y$ is the secret key; f and Z are public.

The signature of message $x \in P$ is $s = sig(x_1 \dots x_k) = (y_{1x_1}, \dots, y_{kx_k})$. The signature s is denoted as $(a_1, \dots, a_k)$. The signature s of message x is valid if $f(a_i) = z_{ix_i}$ for each $i \in \{1, \dots, k\}$.

7.8 Exercises

Exercise 7.1

Consider the DSA signature scheme. Show that is possible to recover the secret key in the following situations.

1. A signer has precomputed one pair k, a with $a = (r^k \bmod q) \bmod p$ and always uses this pair to sign his messages.

2. A signer creates a signature $(a, 0)$ for some message w.

Solution 7.1.1

1. When we eavesdrop two messages w_1 and w_2 and their signatures (a, b_1) and (a, b_2) sent by the signer, we can calculate the value of k – one of the secret keys. Suppose that $b_1 > b_2$ and $kk^{-1} \equiv 1 \pmod q$, then we have:

$$\begin{aligned} b_1 &= k^{-1}(w_1 + xa) \bmod q \\ b_2 &= k^{-1}(w_2 + xa) \bmod q \\ b_1 - b_2 &= k^{-1}(w_1 + xa) - k^{-1}(w_2 + xa) \bmod q \\ k(b_1 - b_2) &\equiv w_1 + xa - w_2 - xa \equiv w_1 - w_2 \bmod q \\ k &= (w_1 - w_2)(b_1 - b_2)^{-1} \bmod q \end{aligned}$$

where $(b_1 - b_2)(b_1 - b_2)^{-1} \equiv 1 \pmod q$. Because q is a prime and $0 < b_1 - b_2 < b_1 < q$ we know that $\gcd(b_1 - b_2, q) = 1$. Hence we can calculate $(b_1 - b_2)^{-1}$ using the Extended Euclidean algorithm and the Bezout's identity.

With the knowledge of the value of k we can recover x:

$$\begin{aligned} b_1 &= k^{-1}(w_1 + xa) \bmod q \\ kb_1 &= w_1 + xa \bmod q \\ kb_1 - w_1 &\equiv xa \bmod q \\ x &= (kb_1 - w_1)a^{-1} \bmod q \end{aligned}$$

where $aa^{-1} \equiv 1 \pmod q$. The value of a^{-1} can be calculated the same way as described above. Now, we know the value of the secret key x and we can send messages pretending to be someone else.

2. When we sent a message w with signature $(a, 0)$, anybody can compute the value of our secret key x:

$$\begin{aligned} 0 = b &= k^{-1}(w + xa) \bmod q \\ k0 = 0 &= w + xa \bmod q \\ xa &\equiv -w \bmod q \\ x &= -wa^{-1} \bmod q \end{aligned}$$

where $aa^{-1} \equiv 1 \pmod q$. Whoever did this calculation, can now send messages with our signature.

Exercise 7.2

Consider the ElGamal digital signature scheme. A valid signature pair (a, b) for a random message w can be constructed as follows:

$$\begin{aligned} a &= q^i y^j \bmod p, \\ b &= -aj^{-1} \bmod (p-1), \\ w &= -aij^{-1} \bmod (p-1), \end{aligned}$$

for $0 \leq i, j \leq p-2$ and $\gcd(j, p-1) = 1$.

1. Let $p = 1367$, $q = 5$ be the generator of $\mathbb{Z}_p^*$ and $y = 307$ be the public key. Using the construction described above find a signature and the corresponding message for parameters i, j of your choice. Show derivation steps for equations for a, b and w. Verify this signature.

2. How can we prevent this attack on ElGamal signature scheme?

Solution 7.2.1

1. We can choose for example $i = 10$ and $j = 3$. First we need to calculate j^{-1} such that $jj^{-1} \equiv 1 \pmod{p-1}$. From Extended Euclidean algorithm we have $1366 = 3 \cdot 455 + 1$, that means that $1 = 1366 - 3 \cdot 455$. Here we get $1 \equiv 3 \cdot (-455) \pmod{1366}$, hence $j^{-1} \equiv -455 \equiv 911 \pmod{1366}$.

   Now, we can calculate a, b and w.

$$\begin{aligned} a &= q^i y^j \bmod p = 5^{10} 307^3 \bmod 1367 = 12 \\ b &= -aj^{-1} \bmod (p-1) = -12 \cdot 911 \bmod 1366 \\ &= -4(3 \cdot 911) \bmod 1366 = -4 \bmod 1366 = 1362 \\ w &= -aij^{-1} \bmod (p-1) = bi \bmod (p-1) = 1362 \cdot 10 \bmod 1366 \\ &= -4 \cdot 10 \bmod 1366 = -40 \bmod 1366 = 1326 \end{aligned}$$

   The signature (a, b) of message w is valid if $y^a a^b \equiv q^w \pmod{p}$. And we can see that

$$\begin{aligned} y^a a^b &= y^a a^{-aj^{-1} \bmod (p-1)} = y^a q^{-aij^{-1} \bmod (p-1)} y^{-ajj^{-1} \bmod (p-1)} \\ &= y^{a-a} q^{-aij^{-1} \bmod (p-1)} = q^w. \end{aligned}$$

   And we really get

$$\begin{aligned} y^a a^b \bmod p &= 307^{12} \cdot 12^{1362} \bmod 1367 = 1097 \\ q^w \bmod p &= 5^{1326} \bmod 1367 = 1097 \end{aligned}$$

2. This forgery attack can be prevented by signing only the hash of the message. The forger can determine the signature only for a random messages – he can never sign any message of his choice, provided the security wasn't broken.

Exercise 7.3

Consider the RSA signature scheme with the public key $(n = 9797, e = 131)$. Decide whether the following signatures are valid.

1. $w = 123$, $sig(w) = 6292$

2. $w = 4337$, $sig(w) = 4768$

3. $w = 4333$, $sig(w) = 1424$

Solution 7.3.1

The signature of message w using RSA signature scheme is $sig(w) = w^d \bmod n$. We can verify the signature by calculating $sig(w)^e \bmod n$. The signature is valid if $sig(w)^e \equiv w \pmod{n}$. The couple (n, e) is the public key.

1. Let $w = 123$ be the message and $sig(w) = 6292$ its signature.
   Because $6292^{131} \bmod 9797 = 123$ the signature is valid.

2. Let $w = 4337$ be the message and $sig(w) = 4768$ its signature.
   Because $4768^{131} \bmod 9797 = 9644$ the signature is not valid.

3. Let $w = 4333$ be the message and $sig(w) = 1424$ its signature.
   Because $1424^{131} \bmod 9797 = 4333$ the signature is valid.

Exercise 7.4

Consider the following signature scheme. Alice chooses two large secret primes p, q and computes their product n. She also chooses an element $g \in \{0, \ldots, n-1\}$ such that g generates a subgroup of order r in $(\mathbb{Z}_n^*, \cdot)$, where r is a large prime.

Alice's public key is a pair (n, g), her private key is a number r.

To sign a message m, Alice finds x such that $xm \equiv 1 \pmod{r}$. Then she computes the signature $s = g^x \pmod{n}$. Suppose Bob has received a pair (m, s) from Alice.

1. How is Bob able to verify her signature?

2. Show that r is a factor of at least one of numbers $p-1$, $q-1$.

3. Show that if r is a factor of exactly one of these numbers then one can factor n using only a public key.

Solution 7.4.1

1. When Bob receives couple (m, s) from Alice, he calculates

$$s^m = g^{xm} = g^{kr+1} = g^{kr} g \equiv g \pmod{n}.$$

   This is correct since $g^r \equiv 1 \pmod{n}$. Therefore, Alice's signature is valid if $s^m = g$.

2. The size of group $(\mathbb{Z}_n^*, \cdot)$ is $|\mathbb{Z}_n^*| = \varphi(n) = (p-1)(q-1)$. Let a be an element from $\mathbb{Z}_n^*$, let k be the order of a then k divides $|\mathbb{Z}_n^*|$ thus k divides $(p-1)(q-1)$. In our case $k = r$ and r is a prime. That means that r divides $(p-1)$ or r divides $(q-1)$. In other words r is factor of at least one of numbers $(p-1)$, $(q-1)$.

3. Suppose, r divides $q-1$ and does not divide $p-1$. Because $g^r \bmod n = 1$ it must also hold that $g^r \bmod p = 1$. Since r is a prime, we can see that the order of g in $\mathbb{Z}_p^*$ is r or 1. Because r does not divide $p-1$ (the size of group $\mathbb{Z}_p^*$), the order of g must be 1. Here we get $g \bmod p = 1$ and $g - 1 \bmod p = 0$. Now it is obvious that $g-1$ and n have a common divisor. Hence, to factor n we merely have to compute the greatest common divisor of numbers $g-1$ and n and we get $\gcd(g-1, n) = p$.

Exercise 7.5

Prove that the Ong-Schnorr-Shamir subliminal channel scheme is correct.

Solution 7.5.1

Ong-Schnorr-Shamir subliminal channel scheme is correct. Because w and w' are coprimes to n, we have $\gcd(n, w) = \gcd(n, w') = 1$. Therefore we can easily calculate $w^{-1} \bmod n$ and $w'^{-1} \bmod n$ using the Extended Euclidean algorithm and the Bezout's identity.

We can see that the verification is correct:

$$\begin{aligned} w' &= S_1^2 - hS_2^2 \bmod n \\ &= \frac{1}{4}\left(\frac{w'^2}{w^2} + 2w' + w^2\right) - k^{-2}\frac{k^2}{4}\left(\frac{w'^2}{w^2} - 2w' + w^2\right) \bmod n \\ &= \frac{1}{4}\left(w'^2 w^{-2} + 2w' + w^2 - w'^2 w^{-2} + 2w' - w^2\right) \bmod n \\ &= \frac{4w'}{4} \bmod n = w' \bmod n = w' \end{aligned}$$

The decryption of subliminal message is also correct:

$$\begin{aligned} S_1 + k^{-1}S_2 \bmod n &= \frac{1}{2}\left(\frac{w'}{w} + w\right) + k^{-1}\frac{k}{2}\left(\frac{w'}{w} - w\right) \bmod n \\ &= \frac{1}{2}\left(\frac{w'}{w} + w + \frac{w'}{w} - w\right) \bmod n \\ &= \frac{2(w'w^{-1})}{2} \bmod n = w'w^{-1} \bmod n \end{aligned}$$

And now, we can see that

$$\begin{aligned} w &= w'(S_1 + k^{-1}S_2)^{-1} \bmod n \\ &= w'(w'w^{-1})^{-1} \bmod n \\ &= w'w'^{-1}(w^{-1})^{-1} \bmod n \\ &= w \bmod n = w. \end{aligned}$$

Exercise 7.6

Assume that in the Lamport signature schemes two k-tuples, x and x', were signed by Bob. Let $f = d(x, x')$ be the Hamming distance of x and x'. How many new messages is an adversary able to sign in such a case?

Solution 7.6.1 by Lukáš Mojžíš

If $d(x, x') = f$ then messages x and x' differs in exactly f positions $p_{i_1}, p_{i_2}, \ldots, p_{i_f}$. At every position p_{i_q}, where $1 \leq q \leq f$, we can choose from values 0 and 1 because we know both $y_{i_q 0}$ and $y_{i_q 1}$. We are able to sign every message chosen this way. Therefore we can sign $(2^f - 2)$ new messages for $f \geq 1$. If $f = 0$ then $x = x'$ and the security was not broken – we are unable to sign any other message.

Chapter 8

Elliptic Curve Cryptography and Factorization

Cryptography based on manipulating with points of so called elliptic curves is getting monument and it has tendency to replace public key cryptography based on infeasibility of factorizing integers or of computing discrete logarithm.

The main advantage of elliptic curves cryptography is that to achieve a certain level of security shorter keys are required then in case of classical cryptography. Using shorter keys can result in savings in hardware implementations. The second advantage of elliptic curves cryptography is that many attacks available for cryptography based on factorization and discrete logarithm do not work for elliptic curves cryptography.

8.1 Elliptic Curve

An *elliptic curve* E is the graph of the equation

$$E : y^2 = x^3 + ax + b,$$

where a, b are (for our purposes) either rational number or integers $\pmod n$, we extend all the points of the graph by a *point of infinity*, denoted as $\mathcal{O}$, that can be regarded as sitting at the top and the bottom of y-axis at the same time. We consider only those elliptic curves that have no multiple roots (i.e. $4a^3 + 27b^2 \neq 0$).

8.2 Addition of Points

On elliptic curve, addition of points can be defined in such a way that they form an Abelian group. If the line through two different points P_1 and P_2 of an elliptic curve E intersects E in a point $Q = (x, y)$, then we define $P_1 + P_2 = P_3 = (x, -y)$. If the line through two different points P_1 and P_2 is parallel with y-axis, then we define $P_1 + P_2 = \mathcal{O}$. If $P_1 = P_2$ and the tangent to E in P_1 intersects E in a point $Q = (x, y)$, then we define $P_1 + P_1 = P_3 = (x, -y)$. Now, it is easy to verify that the addition of points forms Abelian group with $\mathcal{O}$ as the identity element.

Addition of points $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ of an elliptic curve $E : y^2 = x^3 + ax + b$ can be computed using the formula $P_1 + P_2 = P_3 = (x_3, y_3)$, where $x_3 = \lambda^2 - x_1 - x_2$ and $y_3 = \lambda(x_1 - x_3) - y_1$ and λ can be calculated as follows:

$$\lambda = \begin{cases} (y_2 - y_1)(x_2 - x_1)^{-1} & \text{if } P_1 \neq P_2 \\ (3x_1^2 + a)(2y_1)^{-1} & \text{if } P_1 = P_2 \end{cases}$$

If λ is not finite, then $P_3 = \mathcal{O}$

8.3 Elliptic Curves over a Finite Field

The points on an elliptic curve $E : y^2 = x^3 + ax + b \pmod n$ are such pairs $(x, y) \bmod n$ that satisfy the above equation, along with the point of infinity $\mathcal{O}$. The addition of points on an elliptic curve over a finite field is done the same way as described above. The number of points on an elliptic curve over a finite field is limited by the Hasse's theorem.

Hasse's theorem: If an elliptic curve $E \pmod n$ has N points, then $|N - n - 1| < 2\sqrt{n}$.

8.4 Discrete Logarithm Problem for Elliptic Curves

Let E be an elliptic curve and A, B its points such that $B = kA$ for some k. The task to find k is called the discrete logarithm problem for the elliptic curve E. No efficient algorithm to compute discrete logarithm problem for elliptic curves is known and also no good general attacks. Elliptic curves cryptography is based on these facts.

Every cryptosystem (protocol) based on discrete logarithm problem can be converted into a cryptosystem (protocol) based on elliptic curves. The conversion goes as follows:

- Assign to the message (plaintext) a point on an elliptic curve.
- Change in the cryptosystem modular multiplication to addition of points on an elliptic curve.
- Change in the cryptosystem exponentiation to multiplying a point on an elliptic curve by an integer.
- To the point of an elliptic curve that results from the modified cryptosystem assign a message (cryptotext).

8.5 Factorization

Factorization is a process, in which a composite number is decomposed into a product of factors that are prime numbers. For each number there is an unique decomposition and the product of factors is equal to the original integer.

When the number is very large, there is no efficient factorization algorithm known. The hardest problem is to find factors of $n = pq$, where p and q are distinct primes of the same size but with a great distance $|p - q|$.

8.5.1 Factorization with Elliptic Curves

To factorize an integer n we choose an elliptic curve E, a point P on $E \pmod n$ and compute either points iP for $i = 2, 3, 4, \ldots$ or points $2^j P$ for $j = 1, 2, 3, \ldots$. In doing that we need to compute $\gcd(x_a - x_b, n)$ for various points A, B (while computing λ). If one of these values is between 1 and n, we have a factor of n.

8.5.2 Pollard's Rho Method

This method is based on the Birthday paradox – when we keep choosing pseudorandom integers, there must be a pair a, b such that $a \equiv b \pmod p$, where p is a prime factor of n, the number we want to factor. First, we need to choose some pseudorandom function

$f : \mathbb{Z}_n \to \mathbb{Z}_n$ and an integer $x_0 \in \mathbb{Z}_n$. Then we keep computing $x_{i+1} = f(x_i)$ for $i = 0, 1, 2, \ldots$ and $\gcd(|x_j - x_k|, n)$, for each $k < j$. If $\gcd(|x_j - x_k|, n) \geq r > 1$ then we have found a factor r of n such that $x_j \equiv x_k \pmod{r}$.

There is several modification of this method. They differ in the frequency of calculating $\gcd(|x_j - x_k|, n)$. In the second Pollard's rho method, we calculate $\gcd(|x_j - x_k|, n)$ only for one $k < j$. If j is an $(h+1)$ bit integer ($2^h \leq j < 2^{h+1}$), then compute $\gcd(|x_j - x_{2^{h-1}}|, n)$.

8.6 Exercises

Exercise 8.1

Factorize the following numbers (Do not use just brute force; describe computation steps.)

1. $2^{32} - 1$
2. $2^{64} - 1$
3. $3^{32} - 1$

Solution 8.1.1 by Tomáš Laurinčík

Using the quadratic sieve method and properties of Fermat's numbers, we get:

1. $(2^{32} - 1) = (2^{16} + 1)(2^{16} - 1) = (2^{16} + 1)(2^8 + 1)(2^4 + 1)(2^2 + 1)(2 + 1)(2 - 1) = 1 \cdot 3 \cdot 5 \cdot 17 \cdot 257 \cdot 65\,537 = 3 \cdot F_1 \cdot F_2 \cdot F_3 \cdot F_4$, where F_i is ith Fermat's number. Because the first four Fermat's number are primes, the prime factors of $(2^{32} - 1)$ are 3, 5, 17, 257 and 65 537.
2. $(2^{64} - 1) = (2^{32} + 1)(2^{32} - 1) = 3 \cdot F_1 \cdot F_2 \cdot F_3 \cdot F_4 \cdot F_5$, where F_i is ith Fermat's number. The factors of F_5 are 641 and 6 700 417, hence the prime factors of $(2^{64} - 1)$ are 3, 5, 17, 257, 641, 65 537 and 6 700 417.
3. $(3^{32} - 1) = (3^{16} + 1)(3^{16} - 1) = (3^{16} + 1)(3^8 + 1)(3^4 + 1)(3^2 + 1)(3 + 1)(3 - 1) = 2 \cdot 4 \cdot 10 \cdot 82 \cdot 6\,562 \cdot 43\,046\,722$, where the prime factors are $4 = 2^2$, $10 = 2 \cdot 5$, $82 = 2 \cdot 41$, $6\,562 = 2 \cdot 17 \cdot 193$ and $43\,046\,722 = 2 \cdot 21\,523\,361$. It can be shown, that all of the listed factors are primes using the Fermat's test and the Lucas's test or any factorization tool. The prime factors of $(3^{32} - 1)$ are 2^7, 5, 17, 41, 193 and 21 523 361.

Exercise 8.2

Show that $2^{16} + 1$ is prime number.

Solution 8.2.1 by Dušan Katona

We can see that $2^{16} + 1 = 2^{2^4} + 1$ is fourth Fermat number. To test whether a Fermat number is prime, we can use the Pepin's test:

F_n is a prime if and only if $3^{\frac{F_n - 1}{2}} \equiv -1 \pmod{F_n}$

For F_4 we have $3^{\frac{F_4 - 1}{2}} = 3^{\frac{2^{16}+1-1}{2}} = 3^{2^{15}} \equiv -1 \pmod{2^{16} + 1}$ and therefore $2^{16} + 1$ is a prime.

Solution 8.2.2 by Martin Mec

To show that $2^{16}+1$ is prime we can use Lucas test:

> Let $n > 1$. If for every prime factor q of $n-1$ there is an integer a such that $a^{n-1} \equiv 1 \pmod{n}$ and $a^{\frac{n-1}{q}} \not\equiv 1 \pmod{n}$, then n is a prime.

In the first step of Lucas test we find out whether a and n are coprimes, in the second step we test the order of a. If its order is equal to $n-1$ then the size of the set $\mathbb{Z}_n^*$ is $n-1$ and therefore n is a prime.

The number 2^{16} has only one prime factor – 2. And we can see that $3^{2^{16}} \equiv 1 \pmod{2^{16}+1}$ and $3^{\frac{2^{16}}{2}} = 3^{2^{15}} \equiv -1 \pmod{2^{16}+1}$. The number $2^{16}+1$ passes the test and therefore it is a prime.

Exercise 8.3

Assume that $n = pq$, where p and q are distinct primes.

1. Compute $A = n + 1 - \varphi(n)$.
2. Compute roots of the equation $x^2 - Ax + n$ and give explicit expressions for computing p and q.
3. Find the factorization for $n = 15\,049$ and $\varphi(n) = 14\,800$.

Solution 8.3.1

1. $A = n + 1 - \varphi(n) = pq + 1 - (p-1)(q-1) = pq + 1 - (pq - p - q + 1) = p + q$
2. The roots of equation $x^2 - Ax + n$ are p and q:

$$\begin{aligned} x^2 - Ax + n &= 0 \\ x^2 - (p+q)x + pq &= 0 \\ (x-p)(x-q) &= 0 \end{aligned}$$

The roots can be calculated using the quadratic formula:

$$p = \frac{A + \sqrt{A^2 - 4n}}{2}, \quad q = \frac{A - \sqrt{A^2 - 4n}}{2}$$

3. When we know the values of n and $\varphi(n)$, we can find the factors of n. For $n = 15\,049$ and $\varphi(n) = 14\,800$ we get:

$$\begin{aligned} A &= n + 1 - \varphi(n) = 15049 + 1 - 14800 = 250 \\ p &= \frac{A + \sqrt{A^2 - 4n}}{2} = \frac{250 + \sqrt{2304}}{2} = 149 \\ q &= \frac{A - \sqrt{A^2 - 4n}}{2} = \frac{250 - \sqrt{2304}}{2} = 101 \end{aligned}$$

Exercise 8.4

Consider the finite field $K = GF(7) = \mathbb{Z}_7$. An elliptic curve $E_{a,b}$ over K is defined by

$$E_{a,b} = \{\mathcal{O}\} \cup \{(x, y) \in K^2 \,|\, y^2 = x^3 + ax + b\}.$$

1. Find all points of $E_{2,1}$.
2. Verify Hasse's Theorem.
3. For each point $P \in E_{2,1}$, compute $-P$ and check that it lies on the curve as well.
4. To which group is $E_{2,1}$ isomorphic to?

Solution 8.4.1

1. To find all point of the curve $E_{2,1}: \; y^2 = x^3 + 2x + 1$ we need to find the squares of all elements of $\mathbb{Z}_7$:

e	0	1	2	3	4	5	6
$e^2 \bmod 7$	0	1	4	2	2	4	1

For each $x \in \mathbb{Z}_7$ we should find y^2, if it is a square in $\mathbb{Z}_7$, we have found new points of the curve $E_{2,1}$:

x	0	1	2	3	4	5	6
$y^2 \bmod 7$	1	4	6	6	3	3	5

For $x = 0$ we get points $P_1 = (0, 1)$ and $P_2 = (0, 6)$, for $x = 1$ we get another two points $P_3 = (1, 2)$ and $P_4 = (1, 5)$. And there is another one point of $E_{2,1}$ – the point $P_0 = \mathcal{O}$: $E_{2,1} = \{\mathcal{O}, (0, 1), (0, 6), (1, 2), (1, 5)\}$. Because the curve has no multiple roots ($4a^3 + 27b^2 = 4 \cdot 2^3 + 27 = 3 > 0$), its points forms a group.

2. We should verify the Hasse's Theorem – and we can see that it holds:

$$|N - p - 1| < 2\sqrt{p}$$
$$|5 - 7 - 1| = 3 < 2 \cdot 2 < 2\sqrt{7}$$

3. For each point $P = (x, y) \in E_{2,1}$ we can find the point $-P = (x, -y \bmod 7)$:

P	$-P$
$P_0 = \mathcal{O}$	$-P_0 = \mathcal{O} = P_0$
$P_1 = (0, 1)$	$-P_1 = (0, -1 \bmod 7) = (0, 6) = P_2$
$P_2 = (0, 6)$	$-P_2 = (0, -6 \bmod 7) = (0, 1) = P_1$
$P_3 = (1, 2)$	$-P_3 = (1, -2 \bmod 7) = (1, 5) = P_4$
$P_4 = (1, 5)$	$-P_4 = (1, -5 \bmod 7) = (1, 2) = P_3$

And we can see that all of this points lies on the curve $E_{2,1}$.

4. The group $(E_{2,1}, +)$ is isomorphic to the group $(\mathbb{Z}_5, +)$. Let $f: E_{2,1} \to \mathbb{Z}_5$ be a function such that

$$\begin{aligned} f(\mathcal{O}) &\mapsto 0 \\ f((0,1)) &\mapsto 1 \\ f((1,5)) &\mapsto 2 \\ f((1,2)) &\mapsto 3 \\ f((0,6)) &\mapsto 4 \end{aligned}$$

We can see that f is a homomorphous mapping ($f(a +_{E_{2,1}} b) = f(a) +_{\mathbb{Z}_5} f(b)$) and we can easily find the inverse homomorphous mapping g:

$$\begin{aligned} g(0) &\mapsto \mathcal{O} \\ g(1) &\mapsto (0,1) \\ g(2) &\mapsto (1,5) \\ g(3) &\mapsto (1,2) \\ g(4) &\mapsto (0,6) \end{aligned}$$

Exercise 8.5

Use the rho method with $f(x) = x^2 - 1$ and $x_0 = 5$ to find a factor of $n = 7\,031$.

Solution 8.5.1

Using the Pollard's rho method with function $f(x) = x^2 - 1$, $x_0 = 5$ and $x_{i+1} = f(x_i)$ we get the following factors of $n = 7031$:

$$\begin{aligned} x_1 &= x_0^2 - 1 = 24 \\ &\quad \gcd(x_1 - x_0, n) = \gcd(19, 7031) = 1 \\ x_2 &= x_1^2 - 1 = 575 \\ &\quad \gcd(x_2 - x_0, n) = \gcd(570, 7031) = 1, \quad \gcd(x_2 - x_1, n) = \gcd(551, 7031) = 1 \\ x_3 &= x_2^2 - 1 = 167 \\ &\quad \gcd(x_3 - x_0, n) = \gcd(162, 7031) = 1, \quad \gcd(x_3 - x_1, n) = \gcd(143, 7031) = 1, \\ &\quad \gcd(x_3 - x_2, n) = \gcd(6623, 7031) = 1 \\ x_4 &= x_3^2 - 1 = 6795 \\ &\quad \gcd(x_4 - x_0, n) = \gcd(6790, 7031) = 1, \quad \gcd(x_4 - x_1, n) = \gcd(6771, 7031) = 1, \\ &\quad \gcd(x_4 - x_2, n) = \gcd(6220, 7031) = 1, \quad \gcd(x_4 - x_3, n) = \gcd(6628, 7031) = 1 \\ x_5 &= x_4^2 - 1 = 6478 \\ &\quad \gcd(x_5 - x_0, n) = \gcd(6473, 7031) = 1, \quad gcd(x_5 - x_1, n) = \gcd(6454, 7031) = 1, \\ &\quad \gcd(x_5 - x_2, n) = \gcd(5903, 7031) = 1, \quad \gcd(x_5 - x_3, n) = \gcd(6311, 7031) = 1, \\ &\quad \gcd(x_5 - x_4, n) = \gcd(6714, 7031) = 1 \end{aligned}$$

$$x_6 = x_5^2 - 1 = 3475$$
$$\gcd(x_6 - x_0, n) = \gcd(3470, 7031) = 1, \quad \gcd(x_6 - x_1, n) = \gcd(3450, 7031) = 1,$$
$$\gcd(x_6 - x_2, n) = \gcd(2900, 7031) = 1, \quad \gcd(x_6 - x_3, n) = \gcd(3308, 7031) = 1,$$
$$\gcd(x_6 - x_4, n) = \gcd(3711, 7031) = 1, \quad \gcd(x_6 - x_5, n) = \gcd(4028, 7031) = 1$$
$$x_7 = x_6^2 - 1 = 3397$$
$$\gcd(x_7 - x_0, n) = \gcd(3392, 7031) = 1, \quad \gcd(x_7 - x_1, n) = \gcd(3373, 7031) = 1,$$
$$\gcd(x_7 - x_2, n) = \gcd(2822, 7031) = 1, \quad \gcd(x_7 - x_3, n) = \gcd(3230, 7031) = 1,$$
$$\gcd(x_7 - x_4, n) = \gcd(3633, 7031) = 1, \quad \gcd(x_7 - x_5, n) = \gcd(3950, 7031) = \mathbf{79},$$
$$\gcd(x_7 - x_6, n) = \gcd(6953, 7031) = 1$$

And we can see that 79 is one factor of 7031: $7031/79 = 89$.

Solution 8.5.2 by Lukáš Mojžíš

To factor $n = 7031$ we can also use the second Pollard's rho method.

$$x_1 = x_0^2 - 1 = 24 \qquad \gcd(x_1 - x_0, n) = \gcd(19, 7031) = 1$$
$$x_2 = x_1^2 - 1 = 575 \qquad \gcd(x_2 - x_1, n) = \gcd(551, 7031) = 1$$
$$x_3 = x_2^2 - 1 = 167 \qquad \gcd(x_3 - x_1, n) = \gcd(143, 7031) = 1$$
$$x_4 = x_3^2 - 1 = 6795 \qquad \gcd(x_4 - x_3, n) = \gcd(6628, 7031) = 1$$
$$x_5 = x_4^2 - 1 = 6478 \qquad \gcd(x_5 - x_3, n) = \gcd(6311, 7031) = 1$$
$$x_6 = x_5^2 - 1 = 3475 \qquad \gcd(x_6 - x_3, n) = \gcd(3308, 7031) = 1$$
$$x_7 = x_6^2 - 1 = 3397 \qquad \gcd(x_7 - x_3, n) = \gcd(3230, 7031) = 1$$
$$x_8 = x_7^2 - 1 = 1737 \qquad \gcd(x_8 - x_7, n) = \gcd(5371, 7031) = 1$$
$$x_9 = x_8^2 - 1 = 869 \qquad \gcd(x_9 - x_7, n) = \gcd(4503, 7031) = \mathbf{79}$$

79 is prime and it is a prime factor of 7031, the other prime factor is $7031/79 = 89$.

Exercise 8.6

Let n be an odd integer.

1. Describe a method based on Fermat's theorem for testing whether n is composite.
2. Prove that for each odd composite number n there are always at least two numbers $a \in \mathbb{Z}_n^*$ such that $a^{n-1} \equiv 1 \pmod{n}$.
3. Are there any numbers n for which the test (from (1)) fails for any $a \in \mathbb{Z}_n^*$? Prove that such numbers do not exist or give an example of such number.

Solution 8.6.1 by Martin Mec

1. The method is called Fermat's primality test. It is based on the Fermat's little theorem:

   Let p be a prime, a any positive integer such that $\gcd(a, n) = 1$, then $a^{n-1} \equiv 1 \pmod{n}$.

It says that an odd positive integer n is composite if there exists a positive integer a such that $\gcd(a, n) = 1$ and $a^{n-1} \not\equiv 1 \pmod{n}$.

2. If $a = 1$ or $a = n - 1$, then $a^{n-1} \equiv 1 \pmod{n}$. If $a = 1$, then for any positive integer k it holds that $1^k = 1$. If $a = n - 1$, then a is even (because n is odd) and we have $(n-1)^{n-1} \equiv (-1)^{n-1} \equiv ((-1)^2)^{\frac{n-1}{2}} \equiv 1^{\frac{n-1}{2}} \equiv 1 \pmod{n}$.

3. The test fails for Carmichael numbers. Carmichael numbers are not primes, but they satisfy $a^{n-1} \equiv 1 \pmod{n}$ for all values of a such that $\gcd(a, n) = 1$. The smallest Carmichael number is 561.

Exercise 8.7

In 2002 three Indian scientists published the first deterministic polynomial algorithm deciding the primality problem. The method uses the following theorem.

> Let $n > 1$, a be integers such that $\gcd(a, n) = 1$. Then n is a prime if and only if $(x + a)^n = x^n + a$ in $\mathbb{Z}_n[x]$.

Prove this theorem.

Solution 8.7.1 by Tomáš Laurinčík

Firstly, we will show that if n is a prime then $(x + a)^n = x^n + a$ in $\mathbb{Z}_n[x]$. We have

$$(x+a)^n = \sum_{k=0}^{n} \binom{n}{k} x^k a^{n-k} = a^n + x^n + \sum_{k=1}^{n-1} \binom{n}{k} x^k a^{n-k}$$

Since n is a prime,

$$\binom{n}{k} = \frac{n!}{k!(n-k)!} = n\frac{(n-1)!}{k!(n-k)!}.$$

Because n is a prime, for all $0 < l < n$ it holds $\gcd(l, n) = 1$. Therefore $\gcd(k!(n-k)!, n) = 1$ and n divides $\binom{n}{k}$ for $0 < k < n$. Hence $(x + a)^n = x^n + a^n$ in $\mathbb{Z}_n[x]$. Using the Fermat's little theorem we get $a^n \equiv a \pmod{n}$ and therefore $(x + a)^n = x^n + a$ in $\mathbb{Z}_n[x]$.

Secondly, we will show that if n is a composite number, then $(x + a)^n \neq x^n + a$ in $\mathbb{Z}_n[x]$. Because n is composite we can write $n = p^e m$, where p is a prime such that p^{e+1} does not divide n. We have

$$\binom{n}{k} = \frac{n!}{k!(n-k)!} = \frac{(p^e m)!}{k!(p^e m - k)!},$$

for $k = p$ we get

$$\binom{n}{p} = \frac{(p^e m)!}{p!(p^e m - k)!} = \frac{p^{e-1} m (p^e m - 1)!}{(p-1)!(p^e m - k)!}.$$

It is obvious that p^e does not divide $\binom{n}{p}$. Since $\gcd(a, n) = 1$, it holds that $\gcd(a, p^e) = 1$ and also $\gcd(a^{n-p}, p^e) = 1$. There we get $\binom{n}{p} x^p a^{n-p} = cx^p$, where $c \neq 0$ in $\mathbb{Z}_n$. Hence $(x + a)^n \neq x^n + a$ in $\mathbb{Z}_n[x]$.

Chapter 9

User Identification, Message Authentication and Secret Sharing

Most applications of cryptography ask for authentic data rather then secret data. A practically very important problem is how to protect data and communication against an active attacker.

9.1 User Identification

User identification is a process at which one party (called a prover) convinces another party (called verifier) of prover's identity and that the prover has actually participated in the identification process. The purpose of any identification process is to preclude impersonation (pretending to be another person).

User identification has to satisfy following conditions:

- The verifier has to accept prover's identity if both parties are honest.
- The verifier cannot later, after succesful identification, pose as a prover and identify himself to another verifier (as the prover).
- A dishonest party that claims to be the other party has only negligible chance to identify himself successfully.

Every user identification protocol has to satisfy two security conditions:

- If one party (verifier) gets a message from the other party (prover), then the verifier is able to verify that the sender is indeed the prover.
- There is no way to pretend for a party when communicating with Bob, that he is Alice, without Bob having a large chance to find out that.

Identification system can be based on any public key cryptosystem. The identification goes as follows: Alice chooses a random r and sends $e_B(r)$ to Bob (e_B is the encryption algorithm for Bob). Alice identifies a communicating person as Bob, if he can send her back r. Bob identifies a communicating person as Alice, if she can send him r.

Identification scheme can be also based on any one way function f and key k. Both Alice and Bob share a key k and a one way function f. The identification goes as follows: Bob sends Alice a random number or string r. Alice sends Bob $P = f(k, r)$. If Bob gets P, then he verifies whether $P = f(k, r)$. If yes, he starts to believe that the communicating person is Alice. The process can be repeated to increase the probability of correct identification.

9.2 Message Authentication

The goal of the *data authentication* protocols is to handle the case that data are sent through insecure channels. By creating so-called Message Authentication Code (MAC) and sending this MAC together with the message through an insecure channel, the receiver can verify whether data were not changed in the channel. The price to pay is that the communicating parties need to share a secret random key that needs to be transmitted through a very secure channel.

The basic difference between MACs and digital signatures is that MACs are symmetric. Anyone who is able to verify MAC of a message is also able to generate the same MAC and vice versa. A scheme (M, T, K) for data authentication is given by a set of possible messages (M), a set of possible MACs (T) and a set of possible keys (K). It is required that to each key k from K there is a single and easy to compute authentication mapping $auth_k : \{0,1\}^* \times M \to T$ and a single easy to compute verification mapping $ver_k : M \times T \to \{true, false\}$. An authentication scheme should also satisfy the condition of correctness: For each m from M and k from K it holds $ver_k(m, c) = true$ if there exists an r from $\{0,1\}^*$ such that $c = auth_k(r, m)$; and the condition of security: For any m from M and k from K it is computationally unfeasible (without the knowledge of k) to find c from T such that $ver_k(m, c) = true$.

9.3 Secret Sharing Scheme

Secret sharing schemes distribute a *secret* among several users in such a way that only predefined sets of users can recover the secret.

Let $t \leq n$ be positive integers. A (n, t)-threshold scheme is a method of sharing a secret S among a set P of n participants, $P = \{P_i \,|\, 1 \leq i \leq n\}$, in such a way that any t, or more, participants can compute the value S, but no group of $t-1$, or less, participants can compute S. Secret S is chosen by a dealer $D \notin P$. It is assumed that the dealer distributes the secret to participants secretly and in such a way that no participant knows shares of other participants.

9.3.1 Shamir's (n, t)-secret sharing scheme

Initiation phase: Dealer D chooses a prime $p > n$, n distinct x_i, $1 \leq i \leq n$ and D gives the value x_i to the user P_i. The values x_i are public.

Share distribution phase: Suppose D wants to share secret $S \in \mathbb{Z}_p$ among the users. D randomly chooses $t-1$ elements from $\mathbb{Z}_p$, $a_1, \ldots a_{t-1}$. For $1 \leq i \leq n$ D computes the shares $y_i = f(x_i)$, where $f(x) = S + \sum_{j=1}^{t-1} a_j x^j \bmod p$. D gives the computed share y_i to the participant P_i.

Secret cumulation phase: Let participants $P_{i_1}, \ldots, P_{i_t}$ want to determine secret S. Since $f(x)$ has degree $t-1$, $f(x)$ has the form $f(x) = a_0 + a_1 x + \cdots + a_{t-1} x^{t-1}$, and coefficients a_m can be determined from t equations $f(x_{i_j}) = y_{i_j}$, where all arithmetics is done modulo p. It can be shown that equations obtained this way are linearly independent and the system has only one solution. In such a case we get $S = a_0$.

9.4 Exercises

Exercise 9.1

Consider the following identification scheme (based on RDSA signatures).

Let n be a large integer of size s, γ an element of $\mathbb{Z}_n^*$ and q a prime of size t (e.g. $s = 1\,024$ and $t = 160$ bits). Values n, γ and q are public parameters.

Let $a \in \{2, \ldots, q-1\}$ be a private key and $\alpha = \gamma^a \bmod n$ be the corresponding public key.

Identification goes as follows:

1. Prover chooses randomly $k \in \{0, \ldots, q-1\}$, computes $\mu = \gamma^k \bmod n$ and sends μ to Verifier.
2. Verifier chooses challenge $e \in \{0, \ldots, q-1\}$ and sends it to Prover.
3. Prover checks whether $e \in \{0, \ldots, q-1\}$ and computes $x = k - ae$. Prover computes r, l such that $x = lq + r$. Prover computes $\lambda = \gamma^l \bmod n$ and sends λ, r to Verifier.
4. Verifier checks whether $r \in \{0, \ldots, q-1\}$ and $\mu = \gamma^r \alpha^e \lambda^q \bmod n$.

Answer the following questions:

1. Show that the verification is correct if both Prover and Verifier follow the instructions.
2. What happens if Verifier chooses $e = 0$? Compute l, r and λ for this case. Does Verifier learns something about Prover's private key?
3. Show that Verifier, who sends $e = 1$ as his challenge, can learn (with high probability) one bit of Prover's private key after a few runs of the protocol. Compute l, r and λ for this case.

Solution 9.1.1

1. Let $\mu_P = \gamma^k \bmod n$ is μ computed by Prover.

   Verifier calculates $\mu_V = \gamma^r \alpha^e \lambda^q \bmod n = \gamma^r (\gamma^a)^e (\gamma^l)^q \bmod n = \gamma^{r+ae+lq} \bmod n = \gamma^{x+ae} \bmod n = \gamma^{(k-ae)+ae} \bmod n = \gamma^k \bmod n$.

   If both Prover and Verifier follow the protocol then $\mu_P = \mu_V$.

2. If Verifier chooses $e = 0$ then $x = k$. Because $0 \leq k < q$ we have $x = 0q + r$ and we can see that $r = k$, $l = 0$ and $\lambda = \gamma^0 \bmod n = 1$. Verifier then calculates $\mu = \gamma^r \alpha^e \lambda^q \bmod n = \gamma^r \cdot 1 \cdot 1 \bmod n = \gamma^r \bmod n$.

   If $\mu = \alpha$ we can see that $\gamma^r \equiv \gamma^a \pmod n$. Let o be the order of γ in $\mathbb{Z}_n^*$ then $r \equiv a \pmod o$. If $o \geq q$ then $r = a$. Because we don't know anything about the order of γ nor about the factorization of n, we cannot be sure that $r = a$, there is only the possibility that it holds.

3. If Verifier chooses $e = 1$ then $x = k - a$.

   - If $k \geq a$ then $0 \leq x < q$ and hence $x = 0q + (k - a)$, so $l = 0$, $r = k - a$ and $\lambda = 1$.

- If $k < a$ then $-q < x < 0$, hence $x = -q + (k - a + q)$, so $l = -1$, $r = k - a + q$ and $\lambda = \gamma^{-1} \underline{\mathrm{m}} odn$. If $\gamma \neq 1$ then $\lambda \neq 1$. If $\gamma = 1$ then this protocol makes no sense.

With each run of the protocol we calculate $dist$ and put the value into the the set $Dist$.

$$dist = \begin{cases} r & \text{if } \lambda = 1 \ (k - a \geq 0), \\ r - q & \text{if } \lambda \neq 1 \ (k - a < 0). \end{cases}$$

We can see that each $d \in Dist$ satisfies $d \geq -a$ (the least $d = r - q = k - a + q - q = k - a = -a$ for $k = 0$) and $d \leq q - 1 - a$ (the biggest $d = r = k - a = q - 1 - a$ for $k = q - 1$). So we have $-a \leq d \leq q - a - 1$ what is equal to $-d \leq a \leq q - 1 - d$.

Because this unequality is satisfied for each $d \in Dist$ it must hold also for $d_{min} = \min\{d \in Dist\}$ and $d_{max} = \max\{d \in Dist\}$. We can see that $-d_{max} \leq -d_{min}$ and $q-1-d_{max} \leq q-1-d_{min}$. Now we have $-d_{max} \leq -d_{min} \leq a \leq q-1-d_{max} \leq q-1-d_{min}$ and therefore $-d_{min} \leq a \leq q - 1 - d_{max}$.

Now, we know the interval where a lies. That means that we can learn several most significant bits of a.

Exercise 9.2

Consider Shamir's $(10, 3)$-secret sharing scheme over $\mathbb{Z}_p$ where p is a large prime. There is one cheating share holder. His goal is to give a bad share in the secret cumulation phase. The point is that nobody knows which share holder is the cheater.

1. Describe a method to reconstruct the secret given all 10 shares and explain why it works.

2. Determine the smallest number x of shares that are sufficient to reconstruct s. Explain.

3. Let us take any collection of fewer than x share holders. Can they obtain any information about the secret? Explain.

Solution 9.2.1 **by Mária Svoreňová**

1. We can compute secrets s_1, s_2, s_3 for three disjoint sets of shares. We obtain at least two same secrets $s_i = s_j = s$, because the cheater's bad share can be in only one set. If we have three disjoint sets of three shares and $s_1 = s_2 = s_3$, we know that the cheater is the one, not being in any set.

2. We know, that x must be greater then 3, otherwise we cannot reconstruct the secret.

   If $x = 4$, then we obtain up to $\binom{4}{3} = 4$ different secrets. If all four reconstructed secrets are the same, then there is no cheater in the group and we get the secret s. If the reconstructed secrets are all different or do not exist, there is the cheater in the group and only one of these secrets is our secret s – and we cannot find out which one it is.

   If $x = 5$, then we obtain $\binom{5}{3} = 10$ possible secrets. If there is the bad share, then the secret s appears $\binom{4}{3} = 4$ times. The other 6 secrets are different or do not exist. Therefore, the smallest number x of shares sufficient to reconstruct the secret s is 5.

3. If there are less then three share holders, they cannot obtain any information about the secret.

   Any three share holders can compute some secret, but they cannot be sure whether they have recover the secret s. The probability, that they have found it is $\frac{\binom{9}{3}}{\binom{10}{3}} = \frac{7}{10} = 70\,\%$, what is not bad.

   A group of four share holders can compute up to four possible secrets. If they are all equal, they reconstructed the secret s. Otherwise, if one of the share holders is the cheater, they know, that only one of the computed secrets is the secret s.

Exercise 9.3

Sender S broadcasts messages to n receivers $R_1, \ldots, R_n$. Privacy is not important but message authenticity is. Each of the receivers wants to be assured that the messages he has received were sent by S. The subjects decide to use a MAC.

1. Suppose all subjects share a secret key k. Sender S adds the MAC to every message he sends using k and each receiver verifies it. Explain why this scheme is insecure.

2. Suppose sender S has a set $A = \{k_1, \ldots, k_m\}$ of m secret keys. Each receiver R_i has some subset $A_i \subseteq A$ of the keys. Before sending a message, S computes MAC c_i of the message for each key k_i. Then S sends all MACs $c_1, \ldots, c_m$ with the message. When receiver R_i receives a message, he accepts it as authentic if and only if all MACs corresponding to keys in A_i are valid. Which property should sets $A_1, \ldots, A_n$ satisfy to be resistant to the attack from (1). Assume that the receivers cannot collude.

3. Suppose that $n = 6$. Show that it is sufficient for the sender to append 4 MACs to every message to satisfy the condition derived in (2). Describe sets $A_1, \ldots, A_6 \subseteq \{k_1, \ldots, k_4\}$.

Solution 9.3.1

1. When all receivers $R_1, \ldots, R_n$ share the same key k for verifying that the received message was sent by S, each of them can calculate the MAC using the key k for any message m of his choice. When this cheater broadcasts this message m and its MAC calculated using the key k of S, all receivers verify that the message was sent by S.

2. Let $1 \leq i, j \leq n$, $i \neq j$ then $A_i \nsubseteq A_j$. If the keyset A_i of receiver R_i is a subset of keyset A_j of receiver R_j then the receiver R_j can send a message m with MACs $c_1, \ldots, c_m$ where c_l is computed using the key $k_l \in A_j$, c_x such that $k_x \notin A_j$ is chosen randomly. Now the receiver R_i accepts the message (and thinks that the sender was S) because for each $k_w \in A_i$ is the MAC c_w valid.

   If there are no such key sets $A_i \subseteq A_j$ then the scheme is secure.

3. This scheme is secure because we can make 6 different sets A_i such that $|A_i| = 2$ and $A_i \subset \{k_1, k_2, k_3, k_4\}$. When we have elements k_1, k_2, k_3 and k_4 we can get six different pairs of them because

$$\binom{4}{2} = \frac{4!}{2!2!} = \frac{4 \cdot 3}{2} = 6.$$

Now, there is no receiver R_i who can make another receiver think that the sender was someone else.

Exercise 9.4

Alice wishes to prove to Bob that she really does know the private key d corresponding to her RSA public key (n, e). They decide to use the following protocol:

- Bob chooses a random r and sends its encryption $s = r^e \pmod{n}$ to Alice.
- Alice decrypts s by computing $s^d \pmod{n}$ and returns the result to Bob.
- Bob accepts if and only if the returned message is r.

1. Prove that the protocol is zero knowledge under the assumption that Bob is honest.
2. What kind of information can dishonest Bob learn?

Solution 9.4.1 **by Tomáš Laurinčík**

1. Assume that Bob is honest. Bob chooses a random number r and encrypts it with Alice's public key. Since the number r is random, it gives no information about Alice's private key. Alice uses her private key to decrypt the number r and sends it back to Bob. Bob cannot get any information about Alice's private key, because everything he gets is the random number r, that he already knows. If the parameters for RSA are properly chosen, then this protocol is zero knowledge.

2. A dishonest Bob can use this protocol to decrypt messages sent to Alice by someone else. He only sends the message he wants to decrypt (with some salt, so that Alice does not recognize, that it is message sent to her) to Alice, Alice decrypts it and sends it back to Bob.

   Bob can also misuse this protocol. He can send a message m to Alice (not encrypted with her public key), Alice sends him back $m^e \bmod n$, what is Alice's signature of message m. Then, Bob can send any message m pretending that the sender is Alice.

Chapter 10

Bit Commitment Protocols and Zero Knowledge Proofs

A protocol is an algorithm two (or more) parties have to follow to perform a communication. A cryptographical protocol is a protocol to achieve secure communication during some goal oriented cooperation.

10.1 Bit Commitment Protocols

In a *bit commitment protocol* Alice chooses a bit b and gets committed to b, in the sense, that Bob has no way of knowing which commitment Alice had made, and Alice has no way of changing her commitment once she has made it (after Bob announces his guess as to what Alice has chosen).

The basis of bit commitment protocols are *bit commitment schemes*. A bit commitment scheme is a mapping $f : \{0,1\} \times X \to Y$, where X and Y are finite sets. A commitment to bit $b \in \{0,1\}$ is any value $f(b,x)$ for $x \in X$. Each bit commitment protocol has two phases – the commitment phase and the opening phase. In the commitment phase, the sender sends a bit b he wants to commit to (in an encrypted form) to the receiver. In the opening phase, the sender sends to the receiver information that enables the receiver to get the bit b.

Each bit commitment scheme should have three properties:

Hiding(privacy): For no $b \in \{0,1\}$ and $x \in X$, it is feasible for Bob to determine b from $B = f(b,x)$.

Binding: Alice can open her commitment B by revealing x and b such that $B = f(b,x)$, but she should not be able to open a commitment B with both 0, 1.

Correctness: If both, the sender and the receiver, follow the protocol, then the receiver will always learn the commitment b.

10.2 Oblivious Transfer Problem

The *oblivious transfer* problem: Design a protocol for sending messages from Alice to Bob in such a way that Bob receives the message with probability $\frac{1}{2}$ and garbage otherwise. Moreover, Bob knows whether he got the message or garbage, but Alice has no idea which one he got.

The *1-out-of-2 oblivious transfer* problem: Alice sends two messages to Bob in such a way that Bob can choose which of the messages he receives (but he cannot choose both of them), but Alice cannot learn Bob's decision.

10.3 Zero Knowledge Proof Protocols

One of the most important, and at the same time very counterintuitive, primitives for cryptographic protocols are so called *zero knowledge proof protocols*. We can say that zero knowledge proof protocol allows one party, usually called prover, to convince another party, called verifier, that prover knows some facts without revealing to the verifier any information about his knowledge. Zero knowledge proof protocols are a special type of so called interactive proof systems.

An interactive proof system has the property of being zero knowledge if verifier, who interacts with the honest prover, learns nothing from the interaction beyond the validity of the statement being proved. There are several variants of zero knowledge, that differs in the way how "learning nothing" is specified.

In an interactive proof system, there are two parties: a prover, often called Peggy (a randomized algorithm using a private random number generator), and a verifier, often called Vic (a polynomial time randomized algorithm using a private random number generator). Prover knows some secret, or knowledge, or a fact about a specific object, and wishes to convince the verifier, through a communication with him, that he has this knowledge.

The interactive proof system consists of several rounds. In each round prover and verifier alternatively do the following: receive a message from the other party, perform a private computation and send a message to the other party. The communication starts usually by a challenge of verifier and a response of prover. At the end, verifier either accepts or rejects prover's attempts to convince him.

A zero knowledge proof of a theorem T is an interactive two party protocol, in which prover is able to convince verifier who follows the same protocol, by the overwhelming statistical evidence, that T is true, if T is really true, but no prover is able to convince verifier, that T is true, if T is not true. In addition, during the interaction, the prover does not reveal to verifier any other information, except whether T is true or not. Therefore, after verifier gets convinced, he can only believe that T is true.

10.4 3-Colorability of Graphs

With the following protocol Peggy can convince Vic that a particular graph G, known to both of them, is 3-colorable and that Peggy knows such a coloring, without revealing to Vic any information how such coloring looks.

Peggy colors the graph $G = (V, E)$ with three colors and then she perform with Vic $|E|^2$ times the following interaction, where $v_1, \ldots, v_n$ are vertices of V.

1. Peggy chooses a random permutation of colors, recolors G and encrypts, for $i = 1, \ldots, n$, the color c_i of vertex v_i by an encryption procedure e_i (different for each i).

   Peggy then removes colors from vertices, labels the ith vertex of G with cryptotext $y_i = e_i(c_i)$ and designs for her a table containing the color and cryptotext of each vertex. Then Peggy shows Vic the graph with vertices labeled by cryptotexts.

2. Vic chooses an edge and ask Peggy to show him coloring of the adjacent vertices.

3. Peggy shows Vic the colors and encryption procedures corresponding to the selected vertices.

4. Vic performs encryption to verify that vertices really have colors as shown.

10.5 Exercises

Exercise 10.1

There is a cryptographic conference in Monaco. The best student of a cryptographic course will be allowed to participate. Keiko and Hiroki are students with the maximum number of points from exercises. Unfortunately, only one of them is allowed to participate so they have to decide which one. Hiroki is now abroad, therefore Keiko suggest the following protocol that allows them to remotely flip a coin.

- Keiko chooses either $x =$ "HEAD" or $x =$ "TAIL" and picks a random number k. She encrypts x with DES cipher using the key k. She obtains $y = \text{DES}_k(x)$.
- Keiko sends y to Hiroki.
- Hiroki flips a coin and tells Keiko which face is up.
- Keiko reveals k.
- Hiroki decrypts y with DES using the key k and obtains the guess of Keiko. If Keiko's guess is correct, she travels to Monte Carlo.

Is Keiko able to cheat?

Solution 10.1.1

Keiko would be able to cheat only if she knew such keys k_1, k_2 that $DES_{k_1}(HEAD) = y = DES_{k_2}(TAIL)$. To find such keys, she can built two lists $DES_{k_1}(HEAD), k_1)$ and $(DES_{k_2}(TAIL), k_2)$. Both lists are sorted according to the first field of each entry. Keiko then looks for collisions between the two lists and obtains keys k_1, k_2, such that $DES_{k_1}(HEAD) = DES_{k_2}(TAIL)$.

Then when she sends y to Hiroki she learns what face of coin is up. Then she can send back to Hiroki such k_i that $DES_{k_i}(x) = y$ where x is the face.

Because of the computational complexity, it is not easy to find such k_1 and k_2; by the Birthday paradox, we need to perform about 2^{32} DES evaluations for getting one collision. Therefore, we can say that Keiko is not able to cheat.

Exercise 10.2

Let p be a large prime. Let g be a generator of the group $(\mathbb{Z}_p^*, \cdot)$. Discuss the security of the following commitment scheme.

- To commit to $m \in \{0, 1, \ldots, p-1\}$, Alice randomly picks $r \in \{0, 1, \ldots, p-1\}$ and sends $c = g^r m \pmod p$ to Bob.
- To open her commitment, Alice sends r and m to Bob.

1. Is this protocol hiding?
2. Is this protocol binding?

Solution 10.2.1 **by Lukáš Mojžíš**

1. The protocol is hiding if $m > 0$ because when Bob gets c, he knows that $c = g^k \bmod p$. But he doesn't know the two elements l_1 and l_2 such that $l_1 + l_2 = k$. So he is not able to learn anything about m.

   There is only one exception: if $m = 0$ then $c = 0$ irrespective of r that Alice chooses (g is a generator of group $\mathbb{Z}_p^*, \cdot$ and there is no s such that $g^s = 0$ because $0 \notin \mathbb{Z}_p^*$). And because $m = 0$ is allowed, the protocol is not hiding.

2. The protocol is not binding because Alice can choose two distinct $r_1, r_2 \in \{0, \ldots, p-1\}$ and commit to $m = g^{r_2}$. Then she sends to Bob $c = g^{r_1} m = g^{r_1} g^{r_2} = m' g^{r_2}$ and that means that Alice can open her commitment with m and r_1 or with m' and r_2 and it is up to her which of the two pairs she sends to Bob.

Exercise 10.3

Consider the following implementation of 1-out-of-2 oblivious transfer which uses standard oblivious transfer as the underlying primitive:

- Let $m = 3n$ where n is a security parameter. Alice randomly chooses a bit string $r = r_1 r_2 \ldots r_m$. Using standard oblivious transfer m times, she transfers it to Bob, one bit at a time. Bob learns approximately one half of the bits of r. Let $I \subseteq \{1, \ldots, m\}$ be a set of indices for which Bob learns r_i.

- Bob wants to learn Alice's bit b_s. He randomly chooses subset $I_s \subseteq I$ of size n and $I_{1-s} \subseteq \{1, \ldots, m\} \setminus I$ also of size n. He sends I_0, I_1 in this order to Alice.

- Alice checks that I_0 and I_1 are of the correct form. She computes $c_i = b_i \oplus \bigoplus_{j \in I_i} r_j$, where $i \in \{0, 1\}$ and sends c_0, c_1 (in this order) to Bob.

- Bob computes $b_s = c_s \oplus \bigoplus_{j \in I_s} r_j$.

Answer the following questions:

1. This protocol can fail sometimes. Explain why.

2. Explain how can Bob learn the desired value b_s.

3. Can cheating Bob obtain any information about b_{1-s}?

4. Explain why Alice learns nothing about s.

5. Can cheating Bob learn both b_0 and b_1?

6. Why does Alice need to check correctness of I_0 and I_1 in the third step?

7. Could the number m be defined to be $2n$ instead of $3n$? Could it be defined to be $5n$? Explain.

Solution 10.3.1

1. The protocol fails if Bob learns less then n bits of R because then he cannot construct the set I_s. If Bob learns more then $2n$ bits of R then the protocol also fails because Bob is not able to construct the correct set I_{1-s}.

2. Bob knows I_s and he learns c_s from Alice. Then he computes:

$$c_s \oplus \bigoplus_{j \in I_s} r_j = b_s \oplus \bigoplus_{j \in I_s} r_j \bigoplus_{j \in I_s} r_j = b_s$$

because the operation $\oplus$ is commutative and it holds that $x = x \oplus a \oplus a$ for each a.

3. Cheating Bob can learn both values b_s and b_{1-s} only if he knows more then or equal to $2n$ bits of R. When he knows less then $2n$ bits of R then he cannot learn anything about b_{1-s}.

4. She cannot learn anything about s because the only information she gets from Bob is the two distinct sets I_0 and I_1. The information about the set I is hidden to her and so she doesn't know which set I_s is the subset of I.

5. When Bob knows more then or equal to $2n$ bits of R. Then he construct such sets I_0 and I_1 that $I_0, I_1 \subseteq I$ and $I_0 \cap I_1 = \emptyset$. Because he knows all the values r_i where $i \in I_0 \cup I_1 \subseteq I$, he can compute both values b_0 and b_1.

6. Alice need to check the correctness of I_0 and I_1 because if $|I_s| < n$ then the probability that $I_0 \cup I_1 \subseteq I$ increases so as insecurity. If $I_0 \cap I_1 = A \neq \emptyset$ then the probability that $I_0 \cup I_1 \subseteq I$ also increases.

7. If $m = 2n$ than the security of the protocol increases but the probability that Bob learns at least n bits of R decreases. However, if Bob is lucky and learns more then n of R, he cannot follow the protocol correctly, because he is unable to construct correct sets I_0 and I_1.

   If $m = 5n$ than Bob learns approximately $\frac{m}{2} = 2,5n$ bits of R and so he can learn both b_0 and b_1 with higher probability and hence the protocol is less secure.

Exercise 10.4

Consider the zero knowledge proof protocol for 3-colorability of graphs that was described in the section 10.4.

1. Suppose Peggy does not know 3-coloring of a 3-colorable graph $G = (V, E)$, where $|V| = n$ and $|E| = m$. What is the maximal probability that Peggy makes Vic accept her proof in single iteration of the protocol? Explain.

2. Suppose Peggy is honest but her random number generator is faulty. The identity permutation is chosen with probability $\frac{1}{2}$ and each of the other permutations is chosen with probability $\frac{1}{10}$. Explain how cheating Vic can discover 3-coloring of G with high probability after sufficiently many iterations of the protocol.

Solution 10.4.1 **by Martin Vejnár**

1. Peggy does not know the 3-coloring of graph G, but according to the protocol, she must commit to a permutation of her coloring to Vic. Once committed, Peggy cannot change the coloring. Vic then chooses a random edge and asks Peggy to reveal the coloring of its adjacent vertices. Peggy cannot lie and Vic can check, whether the vertices have different color.

   Since Peggy does not know the coloring, she must color the graph randomly. The probability, that both vertices have the same color, for any given pair of vertices, is $\frac{1}{3}$.

   Hence, after one iteration of the protocol, the probability of Vic accepting the proof is $\frac{2}{3}$. After k iteration of the protocol, the probability would be $(\frac{2}{3})^k$.

2. With the broken generator, Vic will be able to determine the coloring of an arbitrary pair of adjacent vertices. In every iteration of the protocol, he just has to choose the same edge (the one connecting the aforementioned vertices), until a sufficient number of colorings is retrieved. Then, statistically, about half of the colorings will be the same. Such a dominant coloring is the Peggy's original coloring. Thus, Vic retrieves a coloring for the two vertices.

   Now, Vic merely has to use this procedure repeatedly, until the coloring of all vertices is revealed.

Bibliography

[1] Baignères, Thomas, et al.: *A classical introduction to cryptography exercise book*. New York : Springer, 2006. ISBN 0387279342.

[2] Kahn, David A.: *The codebreakers : the comprehensive history of secret communication from ancient times to the Internet : the story of secret writing*. New York : Scribner, 1996. ISBN 0684831309.

[3] Rothe, Jörg : *Complexity theory and cryptology : an introduction to cryptocomplexity*. New York : Springer, 2005. ISBN 3540221476.

[4] Stinson, Dougles R.: *Cryptography : theory and practice*. Boca Raton : CRC Press, 2002. ISBN 1584882069.

[5] *Introduction to ECC* [Online]. Certicom Inc., 2007 [cited 2007 May 12]. Available from <http://www.certicom.com/index.php?action=ecc,about_ecc>.

[6] *QuickMath : Automatic math solutions* [Online]. 1999–2007 [cited 2007 May 12]. Available from <http://www.quickmath.com/>.

[7] *Cryptography A-Z : Cryptography De-Mystified* [Online]. SSH Communications Security, 2007 [cited 2007 May 12]. Available from <http://www.ssh.com/support/cryptography/>.

[8] Wikipedia contributors: *Linear code* [Online]. Wikipedia, The Free Encyclopedia; last revision 19 April 2007 19:39 UTC [cited 2007 May 12]. Available from <http://en.wikipedia.org/w/index.php?title=Linear_code&oldid=124164759>.

[9] Wikipedia contributors: *Hamming code* [Online]. Wikipedia, The Free Encyclopedia; last revision 12 May 2007 16:46 UTC [cited 2007 May 12]. Available from <http://en.wikipedia.org/w/index.php?title=Hamming_code&oldid=130352788>.

[10] Wikipedia contributors: *Public-key cryptography* [Online]. Wikipedia, The Free Encyclopedia; last revision 8 May 2007 21:13 UTC [cited 2007 May 12]. Available from <http://en.wikipedia.org/w/index.php?title=Public-key_cryptography&oldid=129350176>.

[11] Wikipedia contributors: *Digital signature* [Online]. Wikipedia, The Free Encyclopedia; last revision 9 May 2007 01:12 UTC [cited 2007 May 12]. Available from <http://en.wikipedia.org/w/index.php?title=Digital_signature&oldid=129400670>.

[12] Wikipedia contributors: *Secret sharing* [Online]. Wikipedia, The Free Encyclopedia; last revision 3 May 2007 08:20 UTC [cited 2007 May 12]. Available from <http://en.wikipedia.org/w/index.php?title=Secret_sharing&oldid=127905516>.

[13] Wikipedia contributors: *Zero-knowledge proof* [Online]. Wikipedia, The Free Encyclopedia; last revision 27 April 2007 19:25 UTC [cited 2007 May 12]. Available from <http://en.wikipedia.org/w/index.php?title=Zero-knowledge_proof&oldid=126457594>.