IV054: Coding, Cryptography and Cryptographic Protocols

Exercise Book

Preliminary Version

Jozef Gruska, Lukáš Boháč, Luděk Matyska, Matej Pivoluska

Faculty of Informatics, Masaryk University, Brno

December 2015

INTRODUCTION

The main part of these lecture notes contain typical exercises and their solutions, which were chosen mainly from exercises that were given to the students, during years 2001-2014, as to-be graded homeworks of the lecture "IV054: Coding theory, cryptography and cryptographic protocols" at Faculty of Informatics, Masaryk University.

The goal of these lecture notes is to help the future students of this course to handle successfully homeworks that are given for the first 10 lectures of the course, by learning the way how the solutions of such exercises are done and presented. In some cases even more than one solution to an exercise is presented – in the cases some new ideas were used.

The lecture notes contain 10 chapters, each with a 2-3 pages of an introductory summary of some concepts and results of the corresponding lecture and about 12-20 exercises and their solutions as well as an appendix of some relevant concepts from (especially discrete) mathematics and theoretical informatics.

Exercises for the lectures used to be created and graded always by a collective consisting also from several very successful (in solving given exercises) students of the lecture in former years. The collective was supervised by L. Boháč and later also by M. Pivoluska.

In spite of the fact that both exercises and their solutions were carefully checked, the current version of the lecture notes are seen as preliminary ones and the final ones will be made in 1-2 years taking into account comments and corrections coming from students using them.

These lecture notes are the product of Masaryk University supported by the project
MUNI/FR/1740/2014 and this support is to be acknowledged and thanked.

The authors, December 2015

Contents

1 Basics of Coding Theory **1**
- 1.1 Introduction . . . 1
  - 1.1.1 Noiseless coding . . . 1
  - 1.1.2 Error correcting codes . . . 2
- 1.2 Exercises . . . 3
- 1.3 Solutions . . . 5

2 Linear Codes **11**
- 2.1 Introduction . . . 11
  - 2.1.1 Basic properties . . . 11
  - 2.1.2 Encoding and decoding with linear codes . . . 12
  - 2.1.3 Hamming codes . . . 12
- 2.2 Exercises . . . 13
- 2.3 Solutions . . . 15

3 Cyclic Codes **20**
- 3.1 Introduction . . . 20
  - 3.1.1 Polynomials over $GF(q)$ . . . 20
  - 3.1.2 Rings of polynomials . . . 20
  - 3.1.3 Algebraic characterization of cyclic codes . . . 21
  - 3.1.4 Check polynomials and parity check matrices for cyclic codes . . . 21
- 3.2 Exercises . . . 22
- 3.3 Solutions . . . 24

4 Secret Key Cryptosystems **30**
- 4.1 Introduction . . . 30
  - 4.1.1 Basics of the design and analysis of cryptosystems . . . 30
  - 4.1.2 Basic classical secret-key cryptosystems . . . 31
  - 4.1.3 Product cryptosystems . . . 32
  - 4.1.4 Perfect secrecy . . . 32
- 4.2 Exercises . . . 33
- 4.3 Solutions . . . 35

5 Public-Key Cryptography, I. **42**
- 5.1 Introduction . . . 42
  - 5.1.1 Diffie-Hellman protocol . . . 42
  - 5.1.2 Knapsack cryptosystem . . . 42
  - 5.1.3 McEliece cryptosystem . . . 43
  - 5.1.4 RSA cryptosystem . . . 43
  - 5.1.5 Rabin-Miller's prime recognition . . . 43
- 5.2 Exercises . . . 44
- 5.3 Solutions . . . 45

6 Public-Key Cryptography, II. 51
6.1 Introduction . . . 51
6.1.1 Rabin Cryptosystem . . . 51
6.1.2 ElGamal cryptosystem . . . 51
6.1.3 Shanks' algorithm for discrete logarithm . . . 51
6.1.4 Perfect security of cryptosystems . . . 52
6.1.5 Blum-Goldwasser cryptosystem . . . 52
6.1.6 Hash functions . . . 52
6.2 Exercises . . . 53
6.3 Solutions . . . 54

7 Digital Signatures 61
7.1 Introduction . . . 61
7.1.1 Signature schemes – basic ideas and goals . . . 61
7.1.2 Digital signature scheme – definition . . . 61
7.1.3 Attacks . . . 62
7.1.4 Examples . . . 62
7.2 Exercises . . . 63
7.3 Solutions . . . 65

8 Elliptic Curve Cryptography 69
8.1 Introduction . . . 69
8.1.1 Elliptic curves . . . 69
8.1.2 Group structure and addition law . . . 69
8.1.3 Elliptic curve cryptography . . . 70
8.1.4 Factorization . . . 70
8.2 Exercises . . . 71
8.3 Solutions . . . 73

9 Identification, Authentication and Secret Sharing 79
9.1 Introduction . . . 79
9.1.1 User identification . . . 79
9.1.2 Message authentication . . . 80
9.1.3 Secret sharing . . . 81
9.2 Exercises . . . 82
9.3 Solutions . . . 85

10 Coin Tossing, Bit commitment, Oblivious Transfer, Zero-knowledge Proofs and Other Crypto-protocols 90
10.1 Introduction . . . 90
10.1.1 Coin-flipping protocols . . . 90
10.1.2 Bit commitment protocols . . . 90
10.1.3 Oblivious transfers . . . 91
10.1.4 Interactive and zero-knowledge proofs . . . 91
10.2 Exercises . . . 92
10.3 Solutions . . . 94

A 99
A.1 Introduction . . . 99
A.2 Notation . . . 99
A.3 Central concepts and principles of modern cryptography . . . 99
A.4 Groups . . . 100
A.4.1 Groups $\mathbb{Z}_n$ and $\mathbb{Z}_n^*$ . . . 100

A.4.2 Order of the group 100
A.4.3 Properties of the group $\mathbb{Z}_n^*$ 100
A.5 Rings and fields 101
A.5.1 Finite fields 101
A.6 Arithmetics 101
A.6.1 Ceiling and floor functions 101
A.6.2 Modulo operations 102
A.6.3 Exponentiation 102
A.6.4 Euclid algorithm for GCD - I. 102
A.6.5 Extended Euclid algorithm 103
A.7 Basics of the number theory 103
A.7.1 Primes 103
A.7.2 Chinese remainder theorem 103
A.7.3 Euler totient function 104
A.7.4 Euler and Fermat theorems 104
A.7.5 Discrete logarithms and square roots 104
A.7.6 Quadratic residues and non-residues 105
A.7.7 Blum integers 106

Chapter 1

Basics of Coding Theory

1.1 Introduction

Coding theory aims to develop systems and methods to transmit information through communication channels efficiently and reliably.

Formally, a communication channel is described by a triple (Σ, Ω, p), where Σ is an input alphabet; Ω is an output alphabet; p is a probability distribution on $\Sigma \times \Omega$ and for $i \in \Sigma, o \in \Omega, p(i, o)$ is the probability that the output of the channel is o if the input is i.

Some examples of important channels are

- **Binary symmetric channel** maps with fixed probability p_0 each binary input into the opposite one, *i.e.* $\Sigma = \Omega = \{0, 1\}$. and $p(0, 1) = p(1, 0) = p_0$ and $p(0, 0) = p(1, 1) = 1 - p_0$
- **Binary erasure channel** maps, with fixed probability p_0, each binary input without an error and with probability $1 - p_0$ an erasure occurs, *i.e.* $\Sigma = \{0, 1\}, \Omega = \{0, 1, e\}$, where e is called *erasure symbol* and $p(0, 0) = p(1, 1) = p_0, p(0, e) = p(1, e) = 1 - p_0$.
- **A noiseless channel** maps inputs into outputs without an error, *i.e.* $\Sigma = \Omega$ and $\forall\ i \in \Sigma, p(i, i) = 1$.

A *Code C* over alphabet Σ is a subset of Σ^* (set of all strings over alphabet Σ). A *q-nary* code is a code over alphabet of q symbols and a *binary* code is a code over the alphabet $\{0, 1\}$.

There is a distinction in the goals of the codes for *noisy* and *noiseless* channels.

1.1.1 Noiseless coding

The main goal of noiseless coding is to send information through a noiseless channel as effectively as possible. Here we model information to be send by a random variable X taking values $x \in \mathcal{X}$ with probability $p(x)$. Information content (in bits) of X can be expressed as Shannon entropy:

$$S(X) = -\sum_{x \in \mathcal{X}} p(x) \log_2 p(x). \tag{1.1}$$

Shannon's noiseless coding theorem says that in order to transmit n values of X we need at least $nS(X)$ bits. An example of an optimal code for noiseless coding is a Huffman code.

Huffman coding. Given is a source S_n as a sequence of n objects, $x_1, \ldots, x_n$ with probabilities $p_1 \geq \cdots \geq p_n$. The Huffman code is designed as follows:

1. replace x_{n-1}, x_n with a new object y_{n-1} (thus creating source S_{n-1}) with probability $p_{n-1} + p_n$ and rearrange sequence so one has again non-increasing probabilities. Keep repeating the above step until there are only two objects left.

2. Optimal prefix code for two objects encodes the first object as 0 and the second object as 1. We can construct the code for more objects as follows. Repeatedly apply the following procedure. If $C = \{c_1, \dots, c_r\}$ is a prefix optimal code for a source S_r, then $C' = \{c'_1, \dots c'_{r+1}\}$ is an optimal code for S_{r+1}, where $c'_i = c_i$ for $1 \leq i \leq r-1$ and $c'_r = c_r 1, c'_{r+1} = c_r 0$.

1.1.2 Error correcting codes

The goal of error correcting codes is to encode the information over a noisy channel in such a way that errors can be *corrected*, or at least *detected*.

An example of an error correcting code is the *International Standard Book Number (ISBN)* code, which is a 10 digit number $x_1, \dots x_{10}$ such that the first 9 digits encode the language, publisher and the serial number of the book, while the last digit is used as a checksum, so that:

$$\sum_{i=1}^{10} (11-i)x_i \equiv 0 \mod 11.$$

The checksum $x_{10} = X$ if x_{10} is to have value 10. This code can correct one single digit error and also one transposition error.

In this chapter we deal only with *block codes* – all the codewords have the same length and we want to transmit a message through a binary symmetric channel.

An important concept is the *closeness* of two words x, y is formalized through *Hamming distance* $h(x, y)$, which is equal to the number of symbols in which words x and y differ.

An important parameter of codes is their *minimal distance* defined as:

$$h(C) = \min\{h(x,y) | x, y \in C x \neq y\}. \tag{1.2}$$

For decoding we use so called *nearest neighbor decoding strategy*, which says that the receiver should decode a received word w' as the codeword w that is the closest one to w'. With this error correcting strategy can formulate the basic error correcting theorem.

- A code C can *detect* up to s errors if $h(C) \geq s+1$ and
- A 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;
- d is the minimum distance in C.

A good (n, M, d)-code has small n, large M and large d. Let us denote $A_q(n, d)$ the largest M such that there is a q-nary (n, M, d)-code. An important upper bound on $A_q(n, d)$ is called a *Sphere packing bound*: If C is a q-nary $(n, M, 2t+1)$- code, then

$$M\left[\binom{n}{0} + \binom{n}{1}(q-1) + \cdots + \binom{n}{t}(q-1)^t\right] \leq q^n. \tag{1.3}$$

If a code obtains equality for the sphere packing bound, it is called a *perfect code.*

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

- a permutation of the positions of the code;
- a permutation of symbols appearing in a fixed position.

1.2 Exercises

1.1. Which of the following codes is a Huffman code for some probability distribution?

(a) $\{0, 10, 11\}$.

(b) $\{00, 01, 10, 110\}$.

(c) $\{01, 10\}$.

1.2. Assume a source X sends messages `A`,`B`,`C`,`D` with the following probabilities

symbol	probability
`A`	0.8
`B`	0.1
`C`	0.05
`D`	0.05

(a) Calculate the entropy of the source X.

(b) Design the Huffman code for the source X. Determine the average number of bits used per symbol.

(c) Assume that the source sends sequences of thousands of messages. Assume that the probability of each symbol occurring is independent of the symbol that have been sent previously. Find a way to modify the design of Huffman code so that the average number of bits used per source symbol decreases to a value no greater than 110% of the source entropy. Design a code using this modification and determine the average number of bits per symbols achieved.

* **1.3.**

(a) Prove for any binary Huffman code that if the most probable message symbol has the probability $p_1 > 2/5$, then that symbol must be assigned a codeword of length 1.

(b) Prove for any binary Huffman code that if the most probable message symbol has probability $p_1 < 1/3$, then that symbol must be assigned a codeword of length ≥ 2.

1.4.

(a) Consider the ISBN number $0486x00973$. Determine x. Which book has this ISBN code?

(b) Consider the code $C = \{x \in \mathbb{Z}_{10}^9 | \sum_{i=1}^{9} ix_i \equiv 0 \mod 10\}$. Show that this version of the ISBN code is not able to detect all transposition errors.

1.5.

(a) Find a binary $(10, 6, 6)$-code.

(b) Find a binary $(5, 4, 3)$-code.

1.6.

(a) Find the minimal distance of the code $C = \{10001, 11010, 01101, 00110\}$.

(b) Decode, for the code C, the strings $11110, 01101, 10111, 00111$ using the nearest neighbor decoding strategy.

1.7. Let us have an error correction code over the 5-ary alphabet

$$C = \{0 \mapsto 01234, 1 \mapsto 12340, 2 \mapsto 23401, 3 \mapsto 34012, 4 \mapsto 40123\}.$$

We want to use a channel X over the 5-ary alphabet. For p being a probability of error and x a character from $\{0,1,2,3,4\}$, the channel acts as follows:

$$\begin{aligned}
&x \mapsto x && \text{with probability } 1-p\\
&x \mapsto x+1 \mod 5 && \text{with probability } p/4\\
&x \mapsto x+2 \mod 5 && \text{with probability } p/4\\
&x \mapsto x+3 \mod 5 && \text{with probability } p/4\\
&x \mapsto x+4 \mod 5 && \text{with probability } p/4.
\end{aligned}$$

We have received the following message as the output of the channel

$$0120000110012301223022401444233333340023.$$

Decode the message.

1.8. Let $C = \{111111, 110000, 001100, 000011\}$. Suppose that the codewords are transmitted using a binary symmetric channel with an error probability $p < \frac{1}{2}$. Determine the probability that the receiver does not notice that a codeword has been corrupted during the transfer.

1.9. Let $q > 0$. What are relations $(\leq, =, \geq)$ between:

(a) $A_2(n, 2d-1)$ and $A_2(n+1, 2d)$;

(b) $A_q(n, d)$ and $A_q(n+2, 2d)$;

(c) $A_q(n, d)$ and $A_q(n+1, d)$.

1.10. Show that the following codes are perfect:

(a) Codes containing all words of given alphabet;

(b) Codes consisting of exactly one codeword;

(c) Binary repetition codes of odd length;

(d) Binary codes of odd length consisting of a vector c and the vector c' with zeros and ones interchanged.

* **1.11.** Consider a perfect binary $(n, M, 5)$-code. Find two lowest values of n for which such a code exists.

1.12. Consider an erasure channel with the erasure probability p.

(a) Suppose we use an error correcting code for this erasure channel with codewords of the length n. How many n-symbol strings can appear as the channel output?

(b) Derive an upper bound for the erasure channel analogous to the sphere packing bound.

* **1.13.** A (v, b, k, r, λ)-block-design D is a partition of v elements $e_1, e_2, \dots, e_v$ into b blocks $s_1, s_2, \dots, s_b$, each of cardinality k, such that each of the objects appears exactly in r blocks and each pair of them appears exactly in λ blocks. An incidence matrix of (v, b, k, r, λ) block design is a $v \times b$ binary matrix M such that for any $(i, j) \in \{1, 2, \dots, v\} \times \{1, 2, \dots, b\}$, $m_{i,j} = 1$, if $v_i \in s_j$ and $m_{i,j} = 0$ otherwise.

Let D be a (v, b, k, r, λ)-block-design. Consider a code C whose codewords are rows of the incidence matrix of D:

(a) Show that each codeword of C has the same weight;

(b) Find the minimal distance of C;

(c) How many errors is C able to correct and detect?

* **1.14.** For each of the following pairs of binary codes, prove their equivalence or prove that they are not equivalent.

(a)

$$A = \left\{\begin{matrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \end{matrix}\right\}, \qquad B = \left\{\begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \end{matrix}\right\},$$

(b)

$$A = \left\{\begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \end{matrix}\right\}, \qquad B = \left\{\begin{matrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 \end{matrix}\right\},$$

(c)

$$A_0 = \{0\}, \qquad A_i = \left\{\begin{array}{c|c} A_{i-1} & \begin{matrix} 1 \\ 0 \\ 1 \\ 0 \\ \vdots \end{matrix} \\ \hline 1\ \ 0\ \ 1\ \ 0\ \ \dots & \left(1 + (-1)^i\right)/2 \end{array}\right\},$$

$$B_0 = \{1\}, \qquad B_i == \left\{\begin{array}{c|c} B_{i-1} & \begin{matrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{matrix} \\ \hline 0\ \ 1\ \ 1\ \ \dots\ \ 1 & 1 \end{array}\right\},$$

1.3 Solutions

1.1.

(a) Yes. An example is a random variable with probabilities $1/2, 1/4, 1/4$.

(b) No. This code is not a Huffman code for any distribution. Huffman code has two longest codewords of the same length.

(c) No. This code is not minimal. A code $\{0, 1\}$ is shorter and in fact this is always the code for variable with two outcomes only.

1.2.

(a)

$$\begin{aligned} S(X) &= -\left(p(A)\log_2 p(A) + p(B)\log_2 p(B) + p(C)\log_2 p(C) + p(D)\log_2 p(D)\right) \\ &= -\left(p(0.8)\log_2 p(0.8) + p(0.1)\log_2 p(0.1) + p(0.05)\log_2 p(0.05) + p(0.05)\log_2 p(0.05)\right) \\ &= 1.022 \end{aligned}$$

(b) The code is given in the following table:

symbol	probability	code
A	0.8	0
B	0.1	10
C	0.05	110
D	0.05	111

Average code length $L(C)$ is

$$L(C) = \sum_x \mathit{length}(C(x)) \Pr(X = x) = 0.8 \cdot 1 + 0.1 \cdot 2 + 0.05 \cdot 3 + 0.05 \cdot 3 = 1.3$$

(c) The solution is to divide the stream of symbols into pairs and then find Huffman coding for the pairs. We know that if we divide the stream of symbols into substrings of length n and use Huffman encoding with these strings as items, the average length approaches entropy. In our case $n = 2$ is sufficient. The solution is:

symbols	code	symbols	code
AA	1	BA	001
AB	010	DA	0001
AC	0110	CA	01110
AD	011111	BB	000001
BD	0000100	BC	0000101
DB	0000110	CB	0000111
CD	00000000	CC	00000001
DD	00000010	DC	00000011

With this encoding the average number of code bits per symbol is 1.06.

1.3.

(a) Let us order the probabilities of n outcomes of the random variable X as $p_1 \geq p_2 \geq p_3 \cdots \geq p_n$ In case of a random variable with three outcomes, there is always one codeword of length 1. In order to have the optimal code, it is assigned to the most probable outcome. In case of four possible outcomes, let us take a look at the Huffman algorithm. In order to have the shortest codeword of length 2, the sum of probabilities of the least two probable outcomes has to be greater than p_1. Then we have: $p_1 > 2/5$, $p_3 + p_4 > p_1$, then $p_2 < 1/5$. Hence $p_3, p_4 < 1/5$, which is a contradiction. This argument can be extended to random variable of arbitrary number of outcomes.

(b) Note that if $p_1 < 1/3$, then there are at least four outcomes. Hence, Huffman algorithm continues to a stage where there are only three last items left. In order to assign a codeword of length 2 to the most probable element we need $p_1 \geq p_2' + p_3'$, but since $p_1 < 1/3$, we also have $p_2' + p_3' > 2/3$, which is a contradiction.

1.4.

(a) We have to solve the following equation for x:

$$\begin{aligned} 1\cdot 0 + 2\cdot 4 + 3\cdot 8 + 4\cdot 6 + 5\cdot x + 6\cdot 0 + 7\cdot 0 + 8\cdot 9 + 9\cdot 7 + 10\cdot 3 &\equiv 0 \mod 11 \\ 221 + 5\cdot x &\equiv 0 \mod 11 \\ x &= 2. \end{aligned}$$

The full ISBN is 0486200973 and belongs to the book Cryptanalysis by Helen F. Gaines.

(b) The checksum of the 9 digit code is $\sum_{i=1}^{9} ix_i \equiv 0 \mod 10$. Let us exchange two positions x_j and x_k in order to create a transposition error. The resulting checksum can be written as:

$$\begin{aligned} \sum_{i=1}^{9} ix_i + (j-k)x_k + (k-j)x_j &\equiv 0 \mod 10 \\ \sum_{i=1}^{9} ix_i + (k-j)(x_j - x_k) &\equiv 0 \mod 10 \end{aligned}$$

.

Since we know that $\sum_{i=1}^{9} ix_i \equiv 0 \mod 10$, in order to find a transposition error which cannot be corrected it suffices to find j, k, x_j, x_k, such that $(k-j)(x_j - x_k) \equiv 0 \mod 10$. An example of such a solution is a code 005000013 and $j = 3, k = 8$. After the transposition we have a code 01000053, which also has a checksum 0.

1.5.

(a) For example $C_a = \{0000001111, 00011110001, 11100000001, 0110110110, 1011011010, 1101101100\}$ is a $(10, 6, 6)$-code.

(b) For example $C_b = \{11111, 00100, 10010, 01001\}$ is a $(5, 4, 3)$-code.

1.6.

(a) $h(C) = h(10001, 11010) = 3$.

(b) Nearest neighbor decoding is given by the following table:

$$\begin{aligned} 11110 &\mapsto 11010 \\ 01101 &\mapsto 01101 \\ 10111 &\mapsto 10001 \text{ or } 00110 \\ 00111 &\mapsto 00110 \end{aligned}$$

Note, that the decoding 10111 is not unique.

1.7.
The decoded message is 040124?4. The symbol "?" is used to inform that the given symbol cannot be decoded uniquely.

1.8.
The distance between any two codewords in the code C is $d = 4$. If the error happens on 4 particular bits such that the resulting word is another codeword, the receiver would not notice it. Let X be the event of 4 particular errors happening. $P(X) = p^4(1-p)^2$. There are three codewords the codeword can change into, therefore the overall probability is $3p^4(1-p)^2$.

1.9.

(a) $A_2(n, 2d-1) = A_2(n+1, d)$. Substitute $m = n+1$ and $f = 2d$ to obtain $A_2(m-1, f-1) = A_2(m, f)$. This holds for even f as shown in the lecture.

(b) There is no given relation. As an example consider $A_2(2,1) = 4 < A_2(4,2) = 8$, but $A_2(4,2) = 8 > A_2(6,4) = 4$.

(c) $A_q(n,d) \leq A_q(n+1,d)$. Appending a zero to every codeword preserves M and d but raises the length of the codeword. We can only improve d by appending in a more sophisticated way.

1.10.

(a) Such codes are $(n, q^n, 1)$-codes, therefore $t = 0$ and the perfect code condition states:

$$q^n \left[\binom{n}{0}\right] = q^n \cdot 1 = q^n.$$

(b) Such codes have $M = 1$ and can correct up to n errors (every error can be detected and corrected, since there is only one valid codeword). The perfect code condition states:

$$1 \cdot \left[\binom{n}{0} + \binom{n}{1}(q-1) + \cdots + \binom{n}{n}(q-1)^n\right] = (1 + (q-1))^n = q^n,$$

where the first equality follows from the binomial theorem.

(c) Binary repetition code off the odd length is a $(2k+1, 2, 2k+1)$-code. The perfect code condition then states:

$$2 \cdot \left[\binom{2k+1}{0} + \binom{2k+1}{1}(2-1) + \cdots + \binom{2k+1}{k}(2-1)^k\right] = 2 \cdot \frac{1}{2}\left[\sum_{i=0}^{2k+1}\binom{2k+1}{i}\right] = 2^{2k+1},$$

where we used the equality $\binom{n}{k} = \binom{n}{n-k}$.

(d) The distance of two codewords is $2k+1$, therefore it is a $(2k+1, 2, 2k+1)$-code and we can use the argumentation from the previous case.

1.11. First we will give necessary conditions for values n. Obvious condition is $n \geq 5$ (since $d = 5$). Another condition follows from definition of perfect code:

$$M \cdot \left[\binom{n}{0} + \binom{n}{1} + \binom{n}{2}\right] = 2^n.$$

We know that $M \in \mathbb{N}$ and from that we get

$$M = \frac{2^{n+1}}{n^2 + n + 2}$$

which means that $n^2 + n + 2$ is some power of 2. This is very strict restriction as we will see. We want to solve diophantine equation (for $n \geq 5$):

$$\begin{aligned} n^2 + n + 2 &= 2^k \\ (n + \frac{1}{2})^2 + \frac{7}{4} &= 2^k \\ (2n+1)^2 + 7 &= 2^{k+2} \\ x^2 + 7 &= 2^y \end{aligned}$$

where $x = 2n + 1$ and $y = k + 2$. Last form of our diophantine equation is actually *Ramanujan-Nagell equation.* This equation was conjectured in 1913 by Srinivasa Ramanujan and solved in 1948 by Trygve Nagell and the result is that the only solutions for $x \in \mathbb{N}$ are 1, 3, 5, 11 and 181. Our first condition $n \geq 5$ gives $x \geq 11$ and from this we obtain two solutions: $n = 5$ and $n = 90$.

It is quite easy to see that for $n = 5$ we have repetition code $C_1 = \{00000, 11111\}$ which is perfect. For $n = 90$ we would like to obtain $(90, 2^{78}, 5)$-code (we do not know if such a code even exists because we obtained only necessary conditions for n but not sufficient).

Assume that such code exists and without loss of generality that it contains word $00\ldots0$. Then we consider words with three nonzero digits and with 1 in the first two places:

$$11100\ldots0, \quad 11010\ldots0, \quad 110010\ldots0, \quad \ldots \quad 110\ldots01$$

There are 88 of those words. If any word of weight 5 is at distance 2 from any of these words then it is at distance 2 from exactly 3 of these words (note that the mentioned word of weight 5 has to begin with 11). For example

$$111110\ldots0 \text{ is at distance 2 from } 111000\ldots0,\ 110100\ldots0,\ 110010\ldots0$$

Since the code is perfect, each word is at distance at most 2 from some code word. We have 88 words and they are divided into groups of 3 which is impossible. We have a contradiction with the only assumption that $(90, 2^{78}, 5)$-code exists.

There exists only one value of n which meets required conditions and that is $n = 5$.

1.12.

(a) At each position three different symbols can appear, therefore there are 3^n possible channel outputs.

(b) Let t be the maximum number of erasures that we want C to be able to correct. Erasure of exactly k symbols of an n-bit codeword can result in $\binom{n}{k}$ possible channel outputs. Erasure of t or less symbols can result in $\sum_{i=0}^{t} \binom{n}{i}$ different channel outputs. The upper bound on maximum number of codewords M is therefore

$$M \left[\sum_{i=0}^{t} \binom{n}{i} \right] \leq 3^n,$$

where 3^n is the number of possible outputs.

1.13.

(a) We know that $m_{i,j} = 1$, iff v_i belongs to block s_j and because every v_i appears exactly in r blocks. Row i represents the incidence of the element v_i, so the weight of each row is r.

(b) Each two elements v_i and v_j are together contained in exactly λ blocks, therefore, every two rows have exactly λ columns, in which they have both value 1. On the other hand, each row has exactly r values 1 and therefore there are exactly $r-\lambda$ columns k in which $m_{i,k}=0$ and $m_{j,k}=1$ and exactly $r-\lambda$ columns k', in which $m_{i,k'}=1$ and $m_{j,k'}=0$. In all the other columns both rows have value 0. Summing it up, each pair of rows differs in exactly $2(r-\lambda)$ positions.

(c) If $h(C)=s+1=2t+1$, the code can detect up to s errors and correct up to t errors. The code C is therefore able to detect up to $2(r-\lambda)-1$ errors and repair $\left\lfloor \frac{2(r-\lambda)-1}{2} \right\rfloor$.

1.14.

(a) In order to obtain B from A do the following: (a) In the third column of A permute symbols. (b) Exchange second and fourth columns. The resulting matrix is the matrix of the code B.

(b) Code A contains an all zero codeword. All other codewords contain three values 1 and 2 values zero. We exchange 1s and 0s in columns to obtain an equivalent code. We will use this rule to transform codewords in code B into all zero code words. After we do this for all 5 codewords we can see that the resulting 5 codes are not equivalent to code A, as their non-zero codewords do not all contain three symbols 1.

(c) A_0 is equivalent to A_1 as both codes contain only one codeword. In order to show equivalence also for $i>0$, we present an algorithm to transform the matrix A_i to the matrix B_i:

   (a) Sort all codewords except the first one according to their weight – the codeword with the highest weight first. The topmost codeword stays on the top.

   (b) Permute columns except the first one, so that all symbols 1 are on the right side of the matrix.

   (c) Permute all symbols.

   The resulting matrix is a matrix of the code B_i.

Chapter 2

Linear Codes

2.1 Introduction

Linear codes are a very important class of codes, because they have a concise description, easy encoding and easy to describe (and often efficient) decoding.

Linear codes are special sets of words of a fixed length over an alphabet $\Sigma_q = \{0, \dots, q-1\}$, where q is a power of prime. The reason for this restriction is that with a suitable operations Σ_q constitutes a *Galois field* (also called *finite field*) $GF(q)$. In case of q being a prime, the suitable operations are sum and product modulo q. We will denote $\mathbb{F}_q^n$ a vector space of all n-tuples over the Galois field $GF(q)$.

Definition 2.1.1. *A subset $C \subseteq \mathbb{F}_q^n$ is a linear code, if*

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

It follows from the definition that $C \subseteq \mathbb{F}_q^n$ is a linear code, iff C is a subspace of $\mathbb{F}_q^n$. Moreover for the case $q = 2$, $C \subseteq \mathbb{F}_q^n$ is a linear code, if sum of any two codewords is a codeword as well.

If C is a k-dimensional subspace of $\mathbb{F}_q^n$, then it is called $[n, k]$-code and it has q^k codewords. If the minimal distance of C is d, then it is called a $[n, k, d]$-code.

2.1.1 Basic properties

The minimal distance of the code $w(C)$ is equal to the smallest of the weights of non-zero codewords. If C is a linear $[n, k]$-code, then it has a basis Γ consisting of k linearly independent codewords and each codeword of C is a linear combination of the codewords from Γ.

Definition 2.1.2. *A $k \times n$ matrix with rows forming a basis of a linear $[n, k]$-code C is called* a generator matrix *of C.*

Two $k \times n$ matrices generate equivalent linear $[n, k]$-codes over $\mathbb{F}_q^n$ if one matrix can be obtained from the other by a sequence of the following operations: (a) permutation of the rows; (b) multiplication of a row by a non-zero scalar; (c) addition of one row to another; (d) permutation of columns; (e) multiplication of a column by a non-zero scalar.

With the use of operations (a)–(c) it is possible to transform every generator matrix G into a form $[\mathbb{I}_k | A]$, where $\mathbb{I}_k$ is a $k \times k$ identity matrix.

2.1.2 Encoding and decoding with linear codes

Encoding

Encoding of a dataword $u = (u_1, \dots, u_k)$ using a generator matrix G of an $[n, k]$-code C is

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

where $r_1, \dots, r_k$ are rows of G.

Nearest neighbor decoding

If a codeword $x = (x_1, \dots, x_n)$ is sent through a channel and a word $y = (y_1, \dots, y_n)$ is received, then $e = x - y = (e_1, \dots, e_n)$ is called the *error vector.*

Definition 2.1.3. *Suppose C is an $[n, k]$-code over $\mathbb{F}_q^n$ and $u \in \mathbb{F}_q^n$. Then the set $u + C = \{u + x | x \in C\}$ is called a* coset *of C in $\mathbb{F}_q^n$.*

Suppose C is a linear $[n, k]$-code. It holds that (a) every vector of $\mathbb{F}_q^n$ is in some coset of C; (b) every coset contains exactly q^k elements; (c) two cosets are either identical or disjoint. Each vector having minimum weight in a coset is called a *coset leader.*

Let $C = \{c_0, \dots c_{2^k-1}\}$, with c_0 being all zero codeword (thus being a coset leader of C). Also let us denote $l_i, i \in \{1, \dots, q^{n-k} - 1\}$ the coset leader of the i^{th} coset $C_i = C + \text{bin}(i)$, where $\text{bin}(i)$ is the n-bit binary representation of i.

The standard array for an $[n, k]$-code C is a $q^{n-k} \times q^n$ array of the form

c_0	c_1	$\dots$	c_{2^k-1}
l_1	$c_1 + l_1$	$\dots$	$c_{2^k-1} + l_1$
$\vdots$	$\vdots$		$\vdots$
$l_{q^{n-k}-1}$	$c_1 + l_{q^{n-k}-1}$	$\dots$	$c_{2^k-1} + l_{q^{n-k}-1}$

A received word y is decoded as the codeword in the first row of the columns in which y occurs. Error vectors which are corrected are precisely the coset leaders l_i.

Syndrome decoding

Inner product of two vectors $u = (u_1, \dots, u_n), v = (v_1, \dots, v_n)$ in $\mathbb{F}_q^n$ is defined as $u \cdot v = u_1 v_1 + \dots + u_n v_n$. If $u \cdot v = 0$, then u and v are *orthogonal.* The *dual code* $C^\perp$ of a linear $[n, k]$-code C is defined as $C^\perp = \{v \in \mathbb{F}_q^n | v \cdot u = 0, \forall u \in C\}$. The dual code $C^\perp$ is a linear $[n, n-k]$-code.

A *parity check matrix* H for an $[n, k]$-code C is any generator matrix of $C^\perp$. If $G = [\mathbb{I}_k | A]$ is the standard form generator matrix of an $[n, k]$-code C, then $H = [-A^\top | \mathbf{I}_{n-k}]$, where $A^\top$ is the transpose of A, is it's parity check matrix. A *syndrome* $S(y)$ of the received word y is calculated as $S(y) = yH^\top$.

Two words have the same syndrome, iff they are in the same coset. Therefore it is sufficient to store only two columns – one for syndromes z and one for their corresponding coset leaders $l(z)$. The decoding procedure is as follows: (a) Given y, compute $S(y)$; (b) Locate $z = S(y)$ in the syndrome column; (c) Decode y as $y - l(z)$.

2.1.3 Hamming codes

An important family of simple linear codes that are easy to encode and decode are called *Hamming codes.*

Definition 2.1.4. *Let r be an integer and H be an $r \times 2^r - 1$ matrix with columns being all non-zero distinct words from $\mathbb{F}_2^r$. The code having H as it's parity check matrix is called a binary Hamming code and denoted by* Ham(3, 2)*.*

The code Ham$(r, 2)$ is a $[2^r - 1, 2^r - 1 - r, 3]$-code, therefore it can repair exactly one error. If the columns of H are arranged in the order of increasing binary numbers the columns represent, than the syndrome $S(y)$ gives, in the binary representation, the position of the error.

2.2 Exercises

2.1. Find the standard form of generator matrix for codes that are linear.

(a) Binary code $C_1 = \{00000, 00110, 00101, 10111, 10010, 10001, 10100, 00011\}$.

(b) 5-ary code $C_2 = \{000, 224, 132, 444, 312\}$.

2.2. How many codewords does the smallest ternary linear code containing keywords 100, 010 and 210 have?

2.3. Consider the 5-ary code C such that

$$x_1x_2x_3x_4 \in C \Leftrightarrow x_1 + 2x_2 + 3x_3 + 4x_4 = 0 \mod 5.$$

Show that C is a linear code and find it's generator matrix in standard form.

2.4. Consider a ternary linear code C. What fraction of codewords can have the digit 2 as the last digit?

2.5.

(a) Show that in a linear binary code, either all the codewords begin with 0, or exactly half begin with 0 and half with 1.

(b) Show that in a linear binary code, either all the codewords have even weight, or exactly half have even weight and half have odd weight.

2.6. A binary code C is called weakly self dual if $C \subset C^{\perp}$. Prove the following.

(a) If C is binary weakly self dual code, every codeword is of even weight.

(b) If each row of the generator matrix G of weakly self-dual code C has weight divisible by 4, then so does every codeword.

2.7. Consider a binary linear code C generated by the matrix

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

(a) Construct a standard array for C.

(b) Find an example of a received word with two errors which is not decoded correctly using the coset decoding method.

2.8. Consider a binary $[n, k]$-code C with a parity check matrix

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

(a) Find $n, k, h(C)$ and $|C|$.

(b) Find the standard form generator matrix for C.

(c) Prove that $C^{\perp} \subset C$.

(d) Find coset leaders and the corresponding syndromes

2.9. Let C be a binary code of length n. Consider a binary code C' of length $n+1$ such that $C' = \{x_1 \dots x_n x_{n+1} | x_1 \dots x_n \in C, x_{n+1} = \sum_{i=1}^{n} x_i\}$, where the addition modulo 2. Show that:

(a) if C is a linear code, then C' is also a linear code;

(b) if H is a parity check matrix of C, then matrix

$$G = \begin{pmatrix} H & r^{\top} \\ s & 1 \end{pmatrix},$$

where r is vector of all 0s and s vectors of all 1s, is a parity check matrix of code C'.

2.10. Let C be a binary linear $[4, 2]$-code such that $C = C^{\perp}$. Show that C contains at least two words of weight 2.

* **2.11.** How many different binary linear $[6, 3]$-codes C fulfill $C = C^{\perp}$?

2.12. Let C_1 and C_2 be linear codes of the same length. Decide whether the following statements hold

(a) If $C_1 \subseteq C_2$, then $C_2^{\perp} \subseteq C_q^{\perp}$;

(b) $(C_1 \cap C_2)^{\perp} = C_1^{\perp} \cup C_2^{\perp}$.

* **2.13.** Show that there exists a $[2k, k]$ self dual code over $\mathbb{F}_q$, if and only if there is a $k \times k$ matrix P with entries from $\mathbb{F}_q$ such that $PP^{\top} = -\mathbb{I}_k$.

2.14. Let M_i be the family of all binary linear codes with weight equal to m_i, where m_i is the i^{th} Mersenne prime (a prime of the form $2^j - 1$ for some $j \in \mathbb{N}$). For all $\mathbb{N}$, decide whether there exists a self-dual code in M_i.

2.15. Let G_1, G_2 be generator matrices of $[n_1, k, d_1]$-linear code and $[n_2, k, d_2]$-linear code, respectively. Find values n, k, d of codes with generator matrices:

(a)

$$G_3 = [G_1 | G_2]$$

(b)

$$G_4 = \begin{pmatrix} G_1 & 0 \\ 0 & G_2 \end{pmatrix}$$

* **2.16.** Let G_{24} be the extended Golay code with the following generator matrix:

row	∞	0	1	2	3	4	5	6	7	8	9	10	∞	0	1	2	3	4	5	6	7	8	9	10
0	1	1												1	1		1	1	1				1	
1	1		1												1	1		1	1	1				1
2	1			1										1		1	1		1	1	1			
3	1				1										1		1	1		1	1	1		
4	1					1										1		1	1		1	1	1	
5	1						1										1		1	1		1	1	1
6	1							1						1				1		1	1		1	1
7	1								1					1	1				1		1	1		1
8	1									1				1	1	1				1		1	1	
9	1										1				1	1	1				1		1	1
10	1											1		1		1	1	1				1		1
11													1	1	1	1	1	1	1	1	1	1	1	1

Show that:

(a) $G_{24} = G_{24}^{\perp}$.

(b) Every codeword of G_{24} code has weight divisible by 4.

(c) G_{24} contains word with all ones.

(d) If G_{24} contains codeword $|L|R|$ with

$$L = a_\infty a_0 a_1 \dots a_{10}, R = b_\infty b_0 b_1 \dots b_{10},$$

it also contains codeword $|L'|R'|$ with

$$L' = b_\infty b_0 b_{10} b_9 \dots b_1, R' = a_\infty a_0 a_{10} a_9 \dots a_1.$$

2.3 Solutions

2.1.

(a) The code C_1 is linear, because $\forall c_1, c_2 \in C_1, c_1 + c_2 \in C_1$, which is a sufficient condition for binary codes. Since C_1 contains 8 codewords, it is a subspace of $\mathbb{F}_2^5$ of dimension 3. Therefore the generator matrix has $\log_2(8) = 3$ rows. Without the loss of generality we can choose any three non-zero and linearly independent vectors as the base of C_1 and write them down in a matrix form:

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

In order to obtain a standard form of generator matrix we need two steps: (a) Swap second and fifth column; (b) Add second row to the first one and third row to a second one:

$$\begin{pmatrix} 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix} \mapsto_{(a)} \begin{pmatrix} 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix} \mapsto_{(b)} \begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix}.$$

(b) This is not a linear code, since $444 + 444 = 333 \notin C_2$.

2.2. Since $210 = 2 \cdot 100 + 010$, there are only two linearly independent codewords in the assignment. The smallest bases of the code containing them is of the size 2 and contains only these two codewords. Such code contains $3^2 = 9$ codewords.

2.3. Let $u = u_1u_2u_3u_4$ and $v = v_1v_2v_3v_4$ be two codewords from C. Then we have that $u + v = (u_1+v_1)(u_2+v_2)(u_3+v_3)(u_4+v_4)$ and $(u_1+v_1)+2(u_2+v_2)+3(u_3+v_3)+4(u_4+v_4) = 0 \mod 5$. Also we have that for any scalar w and corresponding word wu it holds that $w(u_1 + 2u_2 + 3u_3 + 4u_4) = 0 \mod 0$.

Note that C is defined by an orthogonality relation $(u_1, u_2, u_3, u_4) \cdot (1, 2, 3, 4) = 0$. Since we are working with words in $\mathbb{F}_q^4$, there are 3 linearly independent vectors orthogonal to vector $(1, 2, 3, 4)$, which also constitute a basis of code C. To construct a generator matrix of C we first construct it's parity check matrix $H = (1, 2, 3, 4)$. It's normal form is $4H = (4, 3, 2, 1)$. If $H = [-A^\top|\mathbb{I}]$, then $G = [\mathbb{I}|A]$, *i.e.*

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

2.4. There are linear codes with all codewords ending in 0. However, this case is pathological, since these codes can be shortened by leaving out the last symbol, retaining all the parameters.

Therefore, let us consider only codes, which have at least one basis codeword w not ending in 0. without the loss of generality assume that w ends in 1. We now can create a basis, in which all other codewords $w' \neq w$ end in 0, by subtracting in the generator matrix from each row the correct multiple of w. Each codeword ending in 2 can be decomposed as a linear combination of vectors in this new basis, *i.e.* $2 \cdot w + u$, where u is some linear combination of basis vectors ending in 0. If the dimension of the code is d, there are 3^{d-1} different linear combinations u. Together with the fact that the number of codewords in C is 3^d, we have that exactly $\frac{1}{3}$ of codewords ends in digit 2.

2.5.

(a) Consider a basis of the code C. If all the basis codewords begin with 0, also all their linear combinations start with 0 and we are done.

Suppose l out of k words in the basis start with 1. All codewords starting with 1, can be written as a linear combination of basis words b_i, out of which odd number start with 1. It remains to calculate the number of such linear combinations.

$$\sum_{i=2j+1, j\in\mathbb{N}}^{l} \binom{k}{i} 2^{k-l} = 2^{l-1}2^{k-l} = 2^{k-1},$$

which is exactly a half of all the codewords in C.

(b) Let us label a codeword with even number of symbols 1 as even and a word with odd number of 1s as odd. Note that a sum of two even words results in an even word. If the basis of the code contains only even words, we are done. Additionally sum of an even word and an odd word is odd and sum of two odd words is even. Now we can argue similarly to the previous question. If l out of k basis words are odd, then linear combination containing odd number of odd words result in an odd codeword. As we have shown previously there are exactly 2^{k-1} such linear combinations.

2.6.

(a) Since C is a binary weakly self dual code we have that $\forall c \in Cc \cdot c$. This means that every code c must have even weight.

(b) We have established that each row has even weight. Let u and v be two rows of the generator matrix G of C. In order to have $u \cdot v = 0$, u and v must both have symbol 1 in even number of positions. This implies that in even number of positions u has symbol 1 and v has symbol 0 and vice versa. These are exactly the positions in which word $u + v$ will have symbol 1. Sum of two even numbers is always divisible by 4, therefore the weight of $u + v$ is divisible by 4. This fact can be be proven for any linear combination of rows of G by induction.

2.7.

(a) The standard array for C:

00000	10110	01011	11101
10000	00110	11011	01101
01000	11110	00011	10101
00100	10010	01111	11001
00010	10100	01001	11111
00001	10111	01010	11100
10001	00111	11010	01100
00101	10011	01110	11000

(b) An example of such codeword is 01100, which can be obtained by two errors on both codewords 00000 and 11101. This is however not surprising, since $h(C) = 3$, and therefore the code can reliably correct only a single error.

2.8.

(a) H is a $n - k \times k$ matrix, therefore $n = 7$ and $k = 4$. $|C| = q^k = 2^4 = 16$. $h(C) = w(C) = 3$ (see (c) below).

(b) A normal form of H is

$$H_{\text{norm}} = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}.$$

It is of the form $[-A^\top | \mathbb{I}_{n-k}]$, which implies that standard form of $G = [\mathbb{I}_k | A]$, *i.e.*

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

(c) We have that $C = \{u \cdot G | u \in \mathbb{F}_2^4\}$and $C^\perp = \{u \cdot H | u \in \mathbb{F}_2^3\}$. In our case this yields:

$$\begin{aligned} C^\perp &= \{0000000, 1101001, 1110010, 0111100, 1001110, 0011011, 1010101, 0100111\} \\ C &= \{0000000, 1000011, 0100111, 0010110, 0001101, 1100100, 1010101, 1001110, \\ &\quad 0110001, 0101010, 0011011, 1110010, 1101001, 1011000, 0111100, 1111111\} \end{aligned}$$

(d)

$l(z)$	z
0000000	000
1000000	101
0100000	110
0010000	010
0001000	111
0000100	011
0000010	001
0000001	100

2.9.

(a) Since we are working with a binary code C, we need only to prove that $\forall x, y \in C$, $xx_{n+1} + yy_{n+1} \in C'$. We have that:

$$xx_{n+1} + yy_{n+1} = (x+y)\left(\sum_{i=1}^{n} x_i + \sum_{i=1}^{n} y_i\right) = (x+y)\left(\sum_{i=1}^{n} x_i + y_i\right).$$

(b) Let us denote g_i the i^{th} row of G and h_i the i^{th} row of H. We need to prove that $x \cdot g_i = 0$, where $\cdot$ is the scalar product. Since H is the parity check matrix of C, we have that

$$\forall x \in C, \forall i \in \{1, \dots, n\} : x \cdot h_i = 0.$$

This implies that

$$\forall xx_{n+1} \in C', \forall i \in \{1, \dots, n\} : xx_{n+1} \cdot g_i = x \cdot h_i + x_{n+1} \cdot 0 = 0.$$

It now suffices to investigate the scalar product of codewords from C' and the last row of G.

$$\forall xx_{n+1} \in C', x \cdot (11\dots11) = \sum_{i=1}^{n} x_i + x_{n+1} = x_{n+1} + x_{n+1} = 0.$$

Since all codewords of C' are orthogonal to all rows of G, G is the parity check matrix of C'.

2.10. Since C is a $[4,2]$-code and $C = C^{\perp}$, we have that $\forall x \in C : x \cdot x = 0$, therefore $x_1^2 + x_2^2 + x_3^2 + x_4^2 = x_1 + x_2 + x_3 + x_4 = 0$. In other words each x contains an even number of 1s. Since C is a linear code, it has a generator matrix $\left(\begin{smallmatrix} x \\ y \end{smallmatrix}\right)$, with $x, y \in$C$, x \neq y \neq 0000$. There are two possibilities: (a) both x and y contain exactly two 1s and we are done; (b) x contains all 1s and y contains two 1s. Then the codewords we are looking for are $x + y$ and y.

2.11. For every $x, y \in C, x \cdot y = 0$, especially $x \cdot x = 0$, therefore each codeword is of even weight and since $\forall x, y \in C, x + y \in C, h(x, y)$ is even. Now let us take a look at couple of facts this implies:

- C contains at most three codewords of weight 2. This is implied by the fact that each pair of such words has to have symbols 1 in different positions.
- C contains at most three codewords of weight 4. This is implied by the fact that each pair of such words has to share two positions with symbols 1 and be different in all the other positions, *i.e.* these words have 0s in different positions.
- The previous two facts imply that C contains all 1s codeword. Together with all 0s codeword we now know weights of all the codewords.

Since the codewords of weight 2 have symbols 1 in all different positions, they constitute a basis of C. Therefore the task is reduced to a question of how many choices for these three words we have. We can choose the first word in $\binom{6}{2}$ ways, the second word in $\binom{4}{2}$ ways and the last one is determined by the first two. The order in which we pick the words is not important, therefore we divide by 3! to obtain the solution:

$$\frac{\binom{6}{2}\binom{4}{2}}{3!} = 15.$$

2.12.

(a) Let $C_1 = \{c_1, dots, c_n\}, C_2 = \{c_1, \dots, c_n, \dots, c_m\}$, with $n, m \in \mathbb{N}, m \geq n$. We have that $C_1^{\perp} = \{v | v \cdot c_1 = 0 \wedge \dots \wedge v \cdot c_n = 0\}$ and $C_2^{\perp} = \{v | v \cdot c_1 = 0 \wedge \dots \wedge v \cdot c_n = 0 \wedge \dots \wedge v \cdot c_m = 0\}$. Since for all $v \in C_2^{\perp}$ and $c_i \in C_2$ it holds that $v \cdot c_i = 0$, it holds particularly for all $c_i \in C_1$, (*i.e.* for all c_i with $i \leq n$), therefore we have that $C_2^{\perp} \subseteq C_1^{\perp}$.

(b) Let $C_1 = \{00, 01\}$ and $C_2 = \{00, 10\}$. Then $C_1^\perp = \{00, 10\}$ and $C_2^\perp = \{00, 01\}$. We have $(C_1 \cap C_2)^\perp = \{00, 10, 01, 11\}$ and $C_1^\perp \cup C_2^\perp = \{00, 10, 01\}$, hence $(C_1 \cap C_2)^\perp \neq C_1^\perp \cup C_2^\perp$.

2.13. We will first prove the "if" part of the statement. Let P be a $k \times k$ matrix such that $PP^\top = -\mathbb{I}_k$. Let $G = [\mathbb{I}_k|P]$. Rows of G are independent, therefore G is a generator matrix of some $[2k, k]$ linear code C. Also

$$GG^\top = [\mathbb{I}_k|P]\left[\begin{smallmatrix}\mathbb{I}_k\\ P\end{smallmatrix}\right] = \mathbb{I}_k - PP^\top = 0,$$

which means G is also a parity check matrix for code C and therefore C is self dual.

The "only if" part can be proven as follows. Let C be the self-dual $[2k, k]$-code. As C is a linear code it has a generator matrix G of the form $G = [\mathbb{I}_k|A]$, where $A$is a $k \times k$ matrix. Since C is self-dual, we have that $GG^\top = 0$. However

$$GG^\top = [\mathbb{I}_k|P]\left[\begin{smallmatrix}\mathbb{I}_k\\ P\end{smallmatrix}\right] = \mathbb{I}_k + AA^\top.$$

We obtained $\mathbb{I}_k + AA^\top = 0$, therefore $AA^\top = -\mathbb{I}_k$ and we can take P = A.

2.14. There is no self-dual code in M_i for any $i \in \mathbb{N}$. The reason for this is the fact that each Marsenne prime m_i is odd. We know that the weight of the code is equivalent to the weight of the word with the lowest weight among the non-zero codewords. Therefore code of weight m_i contains at least one word of weight m_i. On the other hand, for every word c of a self-dual code C holds that $c \cdot c = 0$, which doesn't hold for codewords of odd weight.

2.15.

(a)
- $n = n_1 + n_2$,
- $k_3 = k$ because G_3 has the same dimension as G_1 and G_2,
- $d_3 = d_1 + d_2$, as the code C_3 contains a codeword composed of concatenation of c_1 and c_2, where c_1 and c_2 are the codewords of minimum weight from C_1 and C_2.

(b)
- $n = n_1 + n_2$,
- $k_4 = 2k$ because G_4 contains all rows from G_1 and G_2 supplemented by zeros,
- $d_3 = \min(d_1, d_2)$, because C_4 contains the same codewords as C_1 and C_2 only extended by 0's so the codeword of the minimal weight stays the same as for one of the original codes.

2.16.

(a) It is easy to check that for every pair u, v of rows of G, we have $u \cdot v = 0$, therefore $G_{24} \subseteq G_{24}^\perp$. However, we also know, that G_{24} has dimension 12. For each $[n, k]$-code C the dimension of $C^\perp$ is $n - k$, therefore $G_{24}^\perp$ also has dimension 12 and $G_{24} = G_{24}^\perp$.

(b) We have established that G_{24} is weakly self-dual and we can see that each row of G_{24} is divisible by 4. Then by exercise 6 it follows that each word of G_{24} has weight divisible by 4.

(c) It is easy to verify that the sum of all rows gives the all 1 codeword, as the number of 1s in each column is odd.

(d) Let us compute the inner product of $L|R$ and $L'|R'$ for arbitrary $L|R \in G_{24}$:

$$\begin{aligned} L|R \cdot L'|R' &= a_\infty b_\infty + a_0 b_0 + \sum_{i=1}^{10} a_i b_{11-i} b_\infty a_\infty + b_0 a_0 + \sum_{i=1}^{10} b_i a_{11-i} \\ &= 2\left(a_\infty b_\infty + a_0 b_0 + \sum_{i=1}^{10} a_i b_{11-i}\right) = 0. \end{aligned}$$

Since G_{24} is self dual, it mus contain all the words orthogonal to $L|R$, so in particular it contains $L'|R'$.

Chapter 3

Cyclic Codes

3.1 Introduction

Cyclic codes are very special linear codes. They are of large interest and importance for several reasons:

- They posses a rich algebraic structure that can be utilized in variety of ways.
- They have extremely concise specifications.
- Their encodings can be efficiently implemented using shift registers.

Definition 3.1.1. *A code C is cyclic, if:*

(i) C is a linear code;

(ii) any cyclic shift of a codeword is also a codeword i.e. whenever $a_0 \dots a_{n-1} \in C$, then also $a_{n-1}a_1 \dots a_{n-2} \in C$.

3.1.1 Polynomials over $GF(q)$

Let us denote a codeword of a cyclic code as $a_0a_1 \dots a_{n-1}$. With each codeword we will associate a codeword polynomial $a_0 + a_1x + a_2x^2 + \dots + a_{n-1}x^{n-1}$. $\mathbb{F}_q[x]$ will denote the set of all polynomials $f(x)$ over GF_q. Also let us define the degree $\deg(f(x))$ of a polynomial $f(x)$ as the largest m, such that x^m has non-zero coefficient in $f(x)$. Some basic properties of polynomials follow:

- **Multiplication.**If $f(x), g(x) \in \mathbb{F}_q[x]$, then $\deg(f(x)g(x)) = \deg(f(x)) + \deg(g(x))$.
- **Division.** For every pair of polynomials $a(x), b(x) \neq 0$ in $\mathbb{F}_q[x]$ there exists a unique pair of polynomials $q(x), r(x) \in \mathbb{F}_q[x]$, such that $a(x) = q(x)b(x) + r(x), \deg(r(x)) < \deg(b(x))$.
- **Congruence.** Let $f(x)$ be a fixed polynomial in $\mathbb{F}_q[x]$. Two polynomials $g(x), h(x)$ are said to be *congruent modulo* $f(x)$ (notation $g(x) \equiv h(x) \mod f(x)$), if $g(x) - h(x)$ is divisible by $f(x)$.

A cyclic code C with codewords of length n can be seen as a set of polynomials of degree (at most) $n - 1$.

3.1.2 Rings of polynomials

For any polynomial $f(x)$, the set of all polynomials $\mathbb{F}_q[x]$ of degrees less than $\deg(f(x))$, with addition and multiplication modulo $f(x)$ forms a *ring* denoted $\mathbb{F}_q[x]/f(x)$. A polynomial $f(x)$ in $\mathbb{F}_q[x]$ is said to be *reducible* if $f(x) = a(x)b(x)$, where $a(x), b(x) \in \mathbb{F}_q[x]$ and $\deg(a(x)) < \deg(f(x)), \deg(b(x)) <$

$\deg(f(x))$. If $f(x)$ is not reducible, then it is said to be *irreducible* in $\mathbb{F}_q[x]$. The ring $\mathbb{F}_q[x]/f(x)$ is a field, if $f(x)$ is irreducible in $\mathbb{F}_q[x]$.

An important factor ring in the context of cyclic codes is the ring $R_n = \mathbb{F}_q[x]/(x^n - 1)$. The reason for this is that a cyclic shift is easy to represent in R_n. Since $x^n \equiv 1 \mod (x^n - 1)$, we have that

$$x(a_0 + a_1x + \dots + a_{n-1}x^{n-1}) = a_{n-1} + a_0x + a_1x2 + \dots + a_{n-2}x^n - 1.$$

3.1.3 Algebraic characterization of cyclic codes

A binary code C of words of length n is cyclic if and only if it satisfies two conditions:

(i) $a(x), b(x) \in C \Rightarrow a(x) + b(x) \in C$,

(ii) $a(x) \in C, r(x) \in R_n \Rightarrow r(x)a(x) \in C$.

For any $f(x) \in R_n$ we can define $\langle f(x)\rangle = \{r(x)f(x)|r(x) \in R_n\}$ with multiplication modulo $x^n - 1$ to be a set of polynomials. Now we are ready to formulate the main characterization theorem for cyclic codes.

For any $f(x) \in R_n$, the set $\langle f(x)\rangle$ is a cyclic code (generated by $f(x)$). Conversely, for each non-zero cyclic code C with codeword length n, there exists a unique monic polynomial $g(x) \in \mathbb{F}_q[x]$ such that $C = \langle g(x)\rangle$ and $g(x)$ is a factor of $x^n - 1$. The polynomial $g(x)$ is called *generator polynomial* of C.

Suppose C is a cyclic code of codewords of length n with generator polynomial

$$g(x) = g_0 + g_1x + \dots + g_rx^r.$$

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

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

Encoding using a cyclic code can be done by multiplication of polynomials. Let C be a cyclic $[n,k]$-code with generator polynomial $g(x)$ of degree $n-k-1$. Each message m can be represented by a polynomial $m(x)$ of degree at most k. If message is encoded by a standard generator matrix introduced above, then we have $c = mG$ and for $m(x)$ and $c(x)$ it holds that $c(x) = m(x)g(x)$.

3.1.4 Check polynomials and parity check matrices for cyclic codes

Let C be a cyclic $[n,k]$-code with the generator polynomial $g(x)$ of degree $n-k$. Since $g(x)$ is a factor of $x^n - 1$, we have that $x^n - 1 = g(x)h(x)$ for some $h(x)$ of degree k. Polynomial $h(x)$ is called the *check polynomial* of C.

Let C be a cyclic code in R_n with a generator polynomial $g(x)$ and a check polynomial $h(x)$. Then $c(x) \in R_n$ is a codeword of C if and only if $c(x)h(x) \equiv 0 \mod x^n - 1$. Also, if $h(x) = h_0 + h_1x + \dots + h_kx^k$, then

(i) a parity-check matrix for C is

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

(ii) $C^{\perp}$ is the cyclic code generated by the polynomial

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

i. e. by the reciprocal polynomial of $h(x)$.

3.2 Exercises

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

(a) The ternary code $C_1 = \{0000, 1212, 2121\}$

(b) The ternary code $C_2 = \{x | x \in \mathbb{Z}_3^5 \wedge w(x) \equiv 0 \mod 3\}$

(c) The ternary code $C_3 = \{x | x \in \mathbb{Z}_3^5 \wedge x_1 + x_2 + \cdots + x_5 \equiv 0 \mod 3\}$

(d) The 7-ary code $C_4 = \{x | x \in \mathbb{Z}_7^5 \wedge \sum_{i=1}^{5} i x_i \equiv 0 \mod 7\}$

3.2. Which of the following polynomials are generator polynomials of a binary cyclic code of length 7?

(a) $x^3 + x^2 + x + 1$

(b) $x^3 + x^2 + 1$

(c) $x^2 + x + 1$

3.3. Consider the following binary linear $[8, 5]$-code C generated with

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

(a) Prove that C is a cyclic code.

(b) Find the generator polynomial of C.

3.4. Let C be a binary cyclic code of length 15 and dimension 11 such that each word from $C^{\perp}$ has even weight and $011111111110000 \in C$. Find a generator polynomial $g(x)$, generator matrix G and a minimum distance of C.

3.5. Consider the binary cyclic code C of length 7 with the generator polynomial $g(x) = x^3+x+1$.

(a) Find the generating matrix G and the parity check matrix H.

(b) Decide, whether the code C is perfect or not.

(c) Encode the message 1001.

3.6. Find a generator polynomial for the smallest binary cyclic code containing codewords 00101000 and 01001000.

3.7. Let C be the smallest binary cyclic code which contains the word 011011.

(a) List the codewords of C.

(b) Determine the the generator polynomial $g(x)$ of C.

(c) Use $g(x)$ to encode a message 11.

3.8.

(a) Factorize $x^6 - 1 \in \mathbb{F}_3[x]$ into irreducible polynomials.

(b) Let n_k be the number of ternary cyclic codes of length 6 and dimension k. Determine n_k for $k \in \{0, 1, 2, 3, 4, 5, 6\}$.

(c) For each cyclic code of dimension 5, find the check polynomial and parity check matrix and determine whether it contains the word 120210.

* **3.9.**

(a) Describe all binary cyclic codes of length 19.

(b) How many different binary cyclic $[65, 36]$ codes are there?

(c) Is it possible to find a binary cyclic $[65, 20]$ code?

* **3.10.** Let C_1, C_2 be q-ary cyclic codes of length n with generator polynomials $g_1(x)$ and $g_2(x)$ respectively. Show that $C_3 = C_1 \cap C_2$ is also cyclic. Find it's generator polynomial.

* **3.11.** Let C_1, C_2 be q-ary cyclic codes of length n with generator polynomials $g_1(x)$ and $g_2(x)$ respectively. Show that $C_3 = \{c_1 + c_2 | c_1 \in C_1, c_2 \in C_2\}$, where the addition is in $\mathbb{F}_q$, is also cyclic. Find it's generator polynomial.

* **3.12.** Let C be a q-ary cyclic code of length n and let $g(x)$ be its generator polynomial. Show that all the codewords $c_0c_1 \ldots c_{n-1} \in C$ satisfy $c_0 + c_1 + \cdots + c_{n-1} = 0$, where addition is in $\mathbb{F}_q$, if and only if the polynomial $x - 1$ is a factor of $g(x)$ in $\mathbb{F}_q[x]$.

3.13. Let $g(x) = g_k x^k + \cdots + g_1 x^1 + g_0 \neq 0$ be a generator polynomial of some cyclic code C. Show that $g_0 \neq 0$.

* **3.14.** Consider a binary cyclic code with a generator polynomial $g(x)$. Show that $g(1) = 0$ if and only if weight of each word in C is even.

3.15. Let C be a binary cyclic code whose codewords have length n. Let c_i denote the number of codewords of weight i in C. Show that ic_i is a multiple of n

3.16. Let C be a binary cyclic code of odd length. Show that C contains a codeword of odd weight if and only if it contains the all 1s word $111\ldots11$.

* **3.17.** Let C be a cyclic code over $\mathbb{F}_q$ of length 7, such that 1110000 is a codeword of C. Show that C is a trivial code (*i.e.* $\mathbb{F}_q^n$ or $\{0^n\}$), of q is not a power of 3.

3.3 Solutions

3.1.

(a) C_1 is cyclic, because every sum of two codewords from C_1 is in C_1 and every cyclic shift of any codeword from C_1 is also in C_1.

(b) C_2 is not a cyclic code. Counterexample: $00111 \in C_2$ and $00112 \in C_2$, but $00111 + 00112 = 00220 \notin C_2$.

(c) C_3 is a cyclic code. Let $x = x_1x_2x_3x_4x_5 \in C_3, y = y_1y_2y_3y_4y_5 \in C_3$. Then

- $x + y = (x_1 + y_1)(x_2 + y_2)(x_3 + y_3)(x_4 + y_4)(x_5 + y_5) = z_1z_2z_3z_4z_5 \in C_3$, because $z_1 + z_2 + z_3 + z_4 + z_5 = (x_1 + y_1) + (x_2 + y_2) + (x_3 + y_3) + (x_4 + y_4) + (x_5 + y_5) = (x_1+x_2+x_3+x_4+x_5)+(y_1+y_2+y_3+y_4+y_5) \equiv 0 \mod 3$, since $x_1+x_2+x_3+x_4+x_5 \equiv 0 \mod 3$ and $y_1 + y_2 + y_3 + y_4 + y_5 \equiv 0 \mod 3$.
- if $x \in C_3$, then its cyclic shift is also in C_3, because the sum of positions is commutative.

(d) C_4 is not a cyclic code. Counterexample: $12341 \in C_4$, but its cyclic shift $11234 \notin C_4$, because $1 \cdot 1 + 2 \cdot 1 + 3 \cdot 2 + 4 \cdot 3 + 5 \cdot 4 = 41 \equiv 6 \mod 7$.

3.2. The polynomial needs to divide $x^7 - 1$ in $\mathbb{F}_2$. We have:

(a) $x^7 - 1 = (x^4 + x)(x^3 + x^2 + x + 1) + x^3 + 1$;

(b) $x^7 - 1 = (x^4 + x^3 + x^2 + 1))(x^3 + x^2 + 1)$;

(c) $x^7 - 1 = (x^5 + x^4 + x^2 + x)(x^2 + x + 1) + x + 1$;

Therefore only the second polynomial generates a binary cyclic code of length 7.

3.3. If C is cyclic, it has to be generated by a divisor of $x^8 - 1$, with degree $8 - 5 = 3$. Since in $\mathbb{F}_2$, $x^8 - 1 = (x + 1)^8$, only one of it's divisors is of degree 3 – $f(x) = (x + 1)^3 = x^3 + x^2 + x + 1$. Using the algorithm from the introduction, the generator matrix of a code with generator polynomial $f(x)$ is

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

We can obtain G from F by the following transformations:

1. The first rows are identical.
2. Add the first row of F to it's second row to obtain the second row of G.
3. Add the second row of F to it's third row to obtain the third row of G.
4. Add the third row of F to it's fourth row to obtain the fourth row of G.
5. The last row of G can be obtained by adding the first, fourth and fifth row's of F.

We have therefore established that G and F generate the same code, which is a cyclic code generated by $f(x)$.

3.4. It follows from the codeword length that $g(x)$ divides $x^{15}-1$. It must also hold that the polynomial

$$w(x) = x + x^2 + x^3 + x^4 + x^5 + x^6 + x^7 + x^8 + x^9 + x^{10}$$

associated with the codeword 011111111110000 is a multiple of $g(x)$. The factors of $w(x)$ and $x^{15}-1$ are the following:

$$\begin{aligned} x^{15}-1 &= (x+1)(x^2+x+1)(x^4+x^3+x^2+x+1)(x^8+x^7+x^5+x^4+x^3+x+1) \\ w(x) &= x(x+1)(x^4+x^3+x^2+x+1)(x^4+x^3+x^2+x+1). \end{aligned}$$

Candidates for $g(x)$ are therefore $(x+1), (x^4+x^3+x^2+x+1)$ and their product $(x+1)(x^4+x^3+x^2+x+1) = x^5+1$. However, we also need to take into account that $C^\perp$ contains only codewords of even weight. Since both C and $C^\perp$ are linear, we also know from the previous chapter that all basis codewords of $C^\perp$ need to have even weight, which in turn, together with the construction of generator matrix of $C^\perp$ implies, that generator polynomial $\bar{h}(x)$ of $C^\perp$ needs to have even weight.

We also know that $x^{15}-1 = g(x)h(x)$. Therefore it suffices to divide $x^{15}-1$ by all the candidates for $g(x)$ and see which one has a check polynomial $h(x)$ of even weight:

$$\begin{aligned} & x^{15}-1/(x+1) = \sum_{i=0}^{14} x^i; \\ & x^{15}-1/(x^4+x^3+x^2+x+1) = x^{11}+x^{10}+x^6+x^5+x+1; \\ & x^{15}-1/(x^5+1) = x^{10}+x^5+1. \end{aligned}$$

The only check polynomial of even weight is the second one, therefore $g(x) = (x^4+x^3+x^2+x+1)$.

The generator matrix is therefore:

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

Minimum distance of a linear code is equal to the minimum weight of non-zero codewords. In this case the minimum weight is 2. An example is the codeword we get by adding first and second row of G. Codeword with weight 1 doesn't belong to the code C, because otherwise all its cyclic shifts would be in C as well and then C would be a trivial code containing all words of length 15.

3.5.

(a) The generating matrix is:

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

In order to find the parity check matrix H, we need to find the check polynomial. In this case it is $h(x) = 1 + x + x^2 + x^4$, because $g(x)h(x) = x^7 - 1$. The reciprocal polynomial of $h(x)$ is $\bar{h}(x) = 1 + x^2 + x^3 + x^4$, therefore the parity check matrix is:

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

(b) After examining H, we can see that this is in fact the Hamming $(3, 2)$ code, which is perfect.

(c) With the message we can associate a polynomial $x^3 + 1$. Multiplying with the generator polynomial we get $1 + x + x^4 + x^6$, and therefore the codeword is 1100101.

3.6. Since we are dealing with codewords of length 8, we know that all candidate cyclic codes are generated by a polynomial dividing $x^8 - 1$. Since $x^8 - 1 = (x - 1)^8$, we need to find $0 \leq k \leq 8$, such that a cyclic code C_k generated by $g_k(x) = (x - 1)^k$ contains both codewords.

In order to check whether the codewords belong to a code generated by $g_k(x)$, we will consider the check polynomials of the form $h_k(x) = (x - 1)^{8-k}$. Both codewords belong to C_k, if the product of their polynomial representation with $h_k(x)$ is equal to zero modulo $x^8 - 1$. Checking all the polynomials $h_k(x)$ reveals that the smallest code containing both codewords is generated by $x + 1$.

3.7.

(a) The smallest code has to be linear and contain 011011, all its cyclic shifts and the zero codeword. Examining this set $\{000000, 011011, 101101, 110110\}$ of codewords reveals that it indeed is a linear subset of $\mathbb{F}_2^6$, and therefore it is the smallest cyclic code containing 011011.

(b) The generator polynomial can be obtained as the non-zero polynomial with the smallest degree in C. Since we have a list of all the codewords, it is easy to see that such polynomial corresponds to codeword 110110. Therefore, $g(x) = 1 + x + x^3 + x^4$.

(c) The message 11 is equivalent to the polynomial $1 + x$. Then $(1 + x)(1 + x + x^3 + x^4) = 1 + x^2 + x^3 + x^5$. Finally, the codeword corresponding to message 11 is 101101.

3.8.

(a) $x^6 - 1 = (x + 1)^3(x + 2)^3$.

(b) n_k is equal to the number of factors of $x^6 - 1$ with degree $6 - k$ in $\mathbb{F}_q$ – $n_0 = 1, n_1 = 2n_2 = 3, n_3 = 4, n_4 = 3, n_5 = 2, n_6 = 1$.

(c) We now know there are 2 non-equivalent codes of dimension 5:

(i) $g(x) = (x+1)$. Then the check polynomial is $(x+1)^2(x+3)^3 = x^5+2x^4+x^3+2x^2+x+2$ and the parity check matrix is

$$H = \begin{pmatrix} 1 & 2 & 1 & 2 & 1 & 2 \end{pmatrix}.$$

Since $(120210) \cdot H^\top = 1$, the code does not contain the word 120210.

(ii) $g(x) = (x + 2)$. Then the check polynomial is $(x+1)^3(x+3)^2 = x^5 + x^4 + x^3 + x^2 + x + 1$ and the parity check matrix is

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

Since $(120210) \cdot H^\top = 0$, the code contains the word 120210.

3.9.

(a) Factorizing $x^{19}-1$ yields two factors:

$$f_1(x) = x+1$$
$$f_2(x) = \sum_{i=0}^{18} x^i.$$

Therefore there are $2^2 = 4$ cyclic codes in $\mathbb{F}_2^{19}$. These codes are described by their generator polynomials $1, f_1(x), f_2(x), f_1(x)f_2(x) = x^{19}-1$. Codes therefore are $C_1 = \langle 1\rangle, C_2 = \langle f_1(x)\rangle, C_3 = \langle f_2(x)\rangle, C_4 = \langle f_1(x)f_2(x)\rangle$.

(b) The dimension of an $[n,k]$ code is $n-k$, therefore the $[65,36]$ codes we are searching for have dimension 29. After factoring $x^{65}-1$ (*e. g.* using a suitable software), we know there are 5 factors with degree 12, one factor with degree 4 and one factor with degree 1. The only way how to get a polynomial of degree 29 by multiplying these factors is $12+12+4+1$. We can do this in $\binom{5}{2} = 10$ ways. Thus there are 10 distinct $[65,36]$ binary cyclic codes.

(c) A polynomial that is both product of the previously listed factors and has degree $65-20=45$ doesn't exist. Therefore, a $[65,20]$ binary cyclic code does not exist.

3.10. Let us first prove that C_3 is a linear code. We know that C_1 and C_2 are linear codes, therefore:

$$u,v\in C_3 \Rightarrow u,v\in C_1\cap C_2 \Rightarrow u,v\in C_1\wedge u,v\in C_2 \Rightarrow u+v\in C_1\wedge u+v\in C_2 \Rightarrow u+v\in C_1\cap C_2 \Rightarrow u+v\in C_3.$$

Also,

$$\forall a\in\mathbb{F}_q, u\in C_3 \Rightarrow u\in C_1\cap C_2 \Rightarrow \in C_1\wedge u\in C_2 \Rightarrow au\in C_1\wedge au\in C_2 \Rightarrow au\in C_1\cap C_2 \Rightarrow au\in C_3.$$

We will continue by showing that C_3 is a cyclic code. We know that C_1 and C_2 are cyclic codes, therefore:

$$u\in C_3 \Rightarrow u\in C_1\cap C_2 \Rightarrow u\in C_1\wedge u\in C_2 \Rightarrow u_s\in C_1\wedge u_s\in C_2 \rightarrow u_s\in C_1\cap C_2 \Rightarrow u_s\in C_3,$$

where u_s is any cyclic shift of u.

The generator polynomial of C_3 is the least common multiple of $g_1(x)$ and $g_2(x)$ $(\text{lcm}(g_1(x), g_2(x)))$. We can prove this fact in the following way.

$$u(x)\in C_3 \Rightarrow u(x)\in C_1\cap C_2 \Rightarrow u(x)\in C_1\wedge u(x)\in C_2 \Rightarrow g_1(x)|u(x)\wedge g_2(x)|u(x).$$

If $u(x)\neq 0$, then $\deg(u(x)) \geq \deg(\text{lcm}(g_1(x), g_2(x)))$. This implies that $\text{lcm}(g_1(x), g_2(x))$ has the smallest degree of all non-zero polynomials in C_3, hence it is a generator polynomial of C_3.

3.11. Let us first prove that C_3 is linear. If $u,v\in C_3$, then $u = u_1+u_2$ and $v = v_1+v_2$, where $u_1,v_1\in C_1$ and $u_2,v_2\in C_2$. Since C_1 and C_2 are both linear, we have that $(u_1+v_1)\in C_1$ and $(u_2+v_2)\in C_2$. Therefore, $(u_1+v_1)+(u_2+v_2) = (u_1+u_2)+(v_1+v_2) = u+v\in C_3$.

Next let us prove that C_3 is also cyclic. A word $w\in C_3$ has the form

$$w = (u_0+v_0)(u_1+v_1)\dots(u_{n-1}+v_{n-1}),$$

where $u = u_0\dots u_{n-1}\in C_1$ and $v = v_0\dots v_{n-1}\in C_2$. Since C_1 and C_2 are cyclic codes, they also contain arbitrary cyclic shifts u_s, v_s of codewords u, v. Therefore, their corresponding cyclic shift w_s of codeword w belongs to the code C_3.

The generator polynomial of C is $g(x) = \gcd(g_1(x), g_2(x))$. We will now show that $C_3 = \langle g(x)\rangle$.

- $C_3 \subseteq \langle g(x)\rangle$: Each codeword $w \in C_3$ has the form $w = a(x) + b(x)$, where $a(x) \in C_1$ and $b(x) \in C_2$. We can write $a(x) = r(x)g_1(x)$ and $b(x) = s(x)g_2(x)$ for some $r(x), s(x) \in R_n$. Therefore we have
$$w = r(x)g_1(x) + s(x)g_2(x) = g(x)\left(r(x)\frac{g_1(x)}{g(x)} + s(x)\frac{g_2(x)}{g(x)}\right).$$
Since $g(x)$ divides both $g_1(x)$ and $g_2(x)$, we have that $\left(r(x)\frac{g_1(x)}{g(x)} + s(x)\frac{g_2(x)}{g(x)}\right) \in R_n$ and therefore $w \in \langle g(x)\rangle$, hence $C \subseteq \langle g(x)\rangle$.

- $\langle g(x)\rangle \subseteq C_3$: C_3 contains words of the form $r(x)g_1(x) + s(x)g_2(x)$ for some $r(x), s(x) \in R_n$. According to the Bézout's identity there exist some $r'(x), s'(x) \in R_n$ such that $r'(x)g_1(x) + s'(x)g_2(x) = \gcd(g_1(x), g_2(x))$, hence $g(x) \in C_3$. Any word in $\langle g(x)\rangle$ has the form $t(x)g(x)$, where $t(x) \in R_n$. Since C is a cyclic code, it contains also any cyclic shifts of $g(x)$ and their linear combinations. These can be expressed as $t(x)g(x)$ for any $t(x) \in R_n$. Therefore $\langle g(x)\rangle$.

3.12. Since $a_0 \dots a_{n-1}$ corresponds to a polynomial $a_0 + a_1x + \cdots + a_{n-1}x^{n-1}$, we will use both notations interchangeably.

- "$\Rightarrow$". Suppose $(x-1)$ is a factor of $g(x)$. Let $f(x) = g(x)/(x-1)$. Let D be the code with generator polynomial $f(x)$. Note that $C = \{d(x)(x-1) | d(x) \in D\}$. Therefore, for each $d = d_0 \dots d_{n-1} \in D$ there is a word
$$\begin{aligned} d(x)(x-1) &= d(x)x - d(x) = (d_{n-1}d_0 \dots d_{n-2}) - (d_0 \dots d_{n-1}) \\ &= (d_{n-1} - d_0)(d_0 - d_1)\dots(d_{n-2} - d_{n-1}) \in C. \end{aligned}$$
The sum of characters of the word in C is therefore $d_{n-1} - d_0 + d_0 - d_1 + \cdots + d_{n-2} - d_{n-1} = 0$

- "$\Leftarrow$". Assume that $c_0 + \dots c_{n-1} = 0$ for any $c_0c_1 \dots c_{n-1} \in C$. Every $c \in C$ has the form $g(x)r(x)$ for some $r(x) \in R_n$, hence $g(x) \cdot 1 \in C$. Then $g(x) = a_0 + a_1x + \cdots + a_{n-1}x^{n-1}$, and $a_0 + \cdots + a_{n-1} = 0$. This implies that 1 is a root of $g(x)$: $g(1) = a_0 + \cdots + a_{n-1} = 0$. Finally, because 1 is a root of $g(x)$, $g(x) = (x-1)f(x)$ for some $f(x) \in R_n$.

3.13. The fact that $g_0 = 0$ implies that $g(x) = x(g_kx^{k-1} + \cdots + g_1)$. Therefore, x is a factor of $x^n - 1$. This is clearly false as $x^n - 1$ divided by x always yields a remainder -1.

3.14. We begin with an observation that in $\mathbb{F}_q$ the sum of even number of 1s is equal to 0 and sum of odd number of 1s is equal to 1. This implies that $\forall f(x) \in C : 2|w(f(x)) \Leftrightarrow f(1) = 0$, where $w(f(x))$ is the number of nonzero coefficients of polynomial $f(x)$.

- "$\Rightarrow$". Suppose $g(1) = 0$. Since $\forall f(x) \in C$ we have that $f(x) \in \langle g(x)\rangle$, which is equivalent to $f(x) = g(x)a(x)$ for some $a(x) \in R_n$, we also have $f(1) = g(1)a(1) = 0a(1) = 0$. Hence, each word in C is of even weight.

- "$\Leftarrow$". Converse is trivial. weight of every codeword of C is even, so is the weight of codeword corresponding to $g(x)$ and therefore $g(1) = 0$.

3.15. Let us work with the subset $C_i \in C$ containing all codewords of weight i. Without loss of generality suppose C_i contains k words starting with symbol 1. After applying a cyclic shift to all the words in C_i, we get exactly k words ending with 1 with weight i contained in C.

We can argue similarly for all the positions of the code. Hence we have that at each position of words from C_i there are exactly k symbols 1. All in all we have that $ic_i = nk$.

3.16.

- "$\Leftarrow$" If code of odd length contains the all 1s word, it trivially contains a word of odd weight.
- "$\Rightarrow$" Let C contain an odd-weight word $a = a_1 \dots a_n$, with $n = 2k+1$ for some $k \in \mathbb{N}$. Clearly $a_0 + a_1 + \cdots + a_{n-1} \equiv 1 \mod 2$. Because C is cyclic, it also contains all cyclic shifts of a and their sum w. Every position of w can be expressed as $a_0 + a_1 + \cdots + a_{n-1}$, which as we already argued is equal to 1 in $\mathbb{F}_2$

3.17. We need to show that if C is not a trivial code, q is a power of 3. Suppose C is not trivial and it's generator polynomial is $g(x)$. We have that $g(x)|(x^7 - 1)$ and also $1110000 \in C$. Therefore, $1 + x + x^2 = g(x)a(x)$ for some $a(x) \in R_7$. Together with non-triviality of $g(x)$ this implies that $1 \leq \deg(g(x)) \leq 2$. Regardless of q we can write

$$x^7 - 1 = (x-1)(x^6 + x^5 + x^4 + x^3 + x^2 + x + 1).$$

Since $(x^6 + x^5 + x^4 + x^3 + x^2 + x + 1)$ has no divisors of degree 1 or 2 in any basis, we have that $g(x) = x - 1$. As $1110000 \in C$, we have that $(x-1)|(1 + x + x^2)$. After the division in $\mathbb{R}$ we get a remainder 3, which is congruent to 0 only in fields of characteristic equal to three, implying that q is a power of 3.

Chapter 4

Secret Key Cryptosystems

4.1 Introduction

Cryptosystems (called also as *ciphers*) serve to provide secret communication between two parties. They specify how to encrypt a message (usually called *plaintext*) by a sender, to get the encypted message (usually called *cryptotext*), and then how the receiver can decrypt the received cryptotext to get the original plaintext.

Security of *secret key cryptosystems* is based on the assumption that both the sender and the receiver posses the same secret key (and that is why secret key cryptosystems are called also *symmetric cryptosystems*) that is an important parameter of both encoding and decoding processes.

Some secret key cryptosystems are very old. Many of them are easy to break using the power of modern computers. They were dealt with in Chapter 4 of the lecture for the following reasons: (1) They are often simple enough to illustrate basic approaches to the design of cryptosystems and to methods of cryptanalysis; (2) They can be used even nowadays in combination with more modern cryptosystems; (3) Historically, many of them used to play an important role for centuries.

In the Chapter 4 of the lecture basic ideas and examples of such main secret key cryptosystems are presented, analysed and illustrated.

4.1.1 Basics of the design and analysis of cryptosystems

Every cryptosystem S works with a *plaintexts space* P (a set of plaintexts over an alphabet Σ), a *cryptotexts space* C (a set of cryptotexts over an alphabet Δ) and a *keyspace* K (a set of possible keys).

In any cryptosystem S, each key $k \in K$ determines an *encryption algorithm* e_k and a *decryption algorithm* d_k such that for any plaintext $w \in P$, $e_k(w)$ is the corresponding cryptotext and it holds

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

where the last relation is to be valid if the encryption is done by a *randomized algorithm.*

A good cryptosystem should have the following properties: (1) Both encryption and decryption should be easy once the key is known; (2) Cryptotexts should be not much larger than corresponding plaintexts and the set of keys should be very large; (3) Encryption should not be closed under composition and should create so called *avalanche effect* – small changes in plaintexts should lead to big changes in cryptotexts; (4) Decryption should be unfeasible when the used key is not known.

Two basic types of cryptosystems are: (a) *Block cryptosystems* – to encrypt plaintexts of a fixed size; (b) *Stream cryptosystems* – to encrypt (arbitrarily long) streams of input data.

Another basic classification of secret-key cryptosystems divides cryptosystems into: (1) *substitution* based cryptosystems (characters of plaintexts are replaces by another ones); (2) *transposition* based cryptosystems (characters of plaintexts are permuted). Different cryptosystems use different, but usually easy to work with, substitutions and permutations.

If in a substitution based cryptosystem the substitution used is fixed (that is any character is always replaced in the same way) we talk about a mono-alphabetic cryptosystem; otherwise a cryptosystem is called poly-alphabetic.

Some of the main types of cryptoanalytics attacks are:

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

2. **Known-plaintexts attack.** The cryptanalysts know some pairs $w_i, e_k(w_i), 1 \leq i \leq n$, and try to infer k, or at least w_{n+1} for a new cryptotext $e_k(w_{n+1})$.

For mono-alphabetic cryptosystems the basic cryptanalytic attack is based on performing frequency analysis of symbols of the plaintext and comparing it to the publicly known frequency analysis tables for a given language. Highly frequent symbols in the cryptotext are then likely substitutes for highly probably symbols in such publishd tables. Cryptanalytic attacks for POLYALPHABETIC cryptosystems can be much more complicated, but in some important cases can be, after some effort, reduced to those for monoalphabetic cryptosystems.

4.1.2 Basic classical secret-key cryptosystems

Caesar cryptosystem can be used to encrypt texts in any alphabet. For English the key space $K = \{0, 1, 2, \dots, 25\}$ is usually used. For any $k \in K$ the encryption algorithm e_k substitutes any letter by the one occurring k positions ahead (cyclically) in the alphabet; the decryption algorithm d_k substitutes any letter by the one occurring k position backwards (cyclically) in the alphabet.

Polybious cryptosystem was originally designed for encrypting messages in the old English alphabet without the letter "J". The key space consists of checkerboards of the size 5×5 of all remaining 25 letters of English and with rows and also columns labelled by different letters of English. At encryptions, each letter is replaced by a pair of letters denoting the row and the column the letter is in the checkerboard. At the decryption consecutive pairs XY of letters are replaced by the letter in the X-row and the Y-column of the checkerboard used.

When the **Hill cryptosystem** is used to encrypt n-ary vectors v of integers from the set $\{0, 1, 2, \dots, 25\}$, the key space consists of $n \times n$ matrices M_n of integers from the set $\{0, 1, 2, \dots 25\}$ such that $M_n^{-1} \bmod 26$ exists. Encryption is done by multiplication $c = M_n p$ and decryption by another multiplication $p = M^{-1}c$. In order to encrypt text-messages, the alphabet symbols are at first replaced by their ordering numbers in the alphabet and the reverse process is used at decryptions.

Playfair cryptosystem uses the same key space as the Polybious cryptosystems. Encryption is done by replacing subsequent pairs of input symbols (x, y) as follows: (1) If x and y are in the same row (column) they are replace by the pair of symbols to the right of them; (2) If x and y are in different rows and columns, then they are replaced by the symbols in the opposite corners of the rectangle crated by x and y.

Affine cryptosystem has as the key space K the set of all pairs of integers $\{(a, b) \mid 0 \leq a, b < 26; \gcd(a, 26) = 1\}$. Encryption of an integer w from the set $\{0, 1, 2, \dots, 25\}$ is done by computation $c = aw + b \bmod 26$; decryption by the computation $w = a^{-1}(c - b) \bmod 26\}$, where a^{-1} is computed modulo 26.

The key space of **(Keyword) Vigenere cryptosystem** space consists of all words (strings of symbols) in the English alphabet. To encrypt a message m of length n using a key k_0 another key k is created as the prefix of the word k_0^∞ (infinite concatenation of k_0) of the length n and a matrix T of size 26×26 is used the first row of which consists of all symbols of the English alphabet and each column consists again of all symbols of the English alphabet, in their natural order but starting

with the symbol in the first row and cyclically extended. At any encryption the i-th symbol s of the plaintext is replaced by the symbol in the ith row and in the column with i-th symbol of k in the first row.
In **Autoclave cryptosystem**, a key k_0 is concatenated with the plaintext message m to form the key k.

Keyword Caesar cryptosystem has as the key space the set of all pairs (k_0, i) where k_0 is an English word with all letters different and $1 \leq i \leq 26$. To make the encryption of a plaintext w using a key (k_0, i), k_0 is at first extended by adding in the usual order all letters not in k_0 to get a word k that specify a substitution in which i-th letter of the alphabet is replaced by the i-th symbol of k.

A **homophonic cryptosystem** is specified by describing for each letter of the input alphabet a set of potential substitutes (symbols or words). The number of substitutes for any letter X is usually proportional to the frequency of X in usual texts in the input alphabet. Encryption is done by replacing each letter X of the plantext by one, randomly chosen, of the substitutes of X.

In binary **One-time paid cryptosystem** the key space consists of all binary strings. To encrypt a binary plaintext w using a binary string k of the same length, as the key, the cryptotext $c = w \oplus k$ where the operation $\oplus$ is a bit-wise XORing; decryption is then done by computation $w = c \oplus k$.

4.1.3 Product cryptosystems

A cryptosystem $S = (P, K, C, e, d)$ with the sets of plaintexts P, keys K and cryptotexts C and encryption (decryption) algorithms $e(d)$ is called endomorphic if $P = C$. If $S_1 = (P,\ K_1,\ P,\ e^{(1)},\ d^{(1)})$ and $S_2 = (P,\ K_2,\ P,\ e^{(2)},\ d^{(2)})$ are endomorphic cryptosystems, then the product cryptosystem is

$$S_1 \times S_2 = (P,\ K_1 \times K_2,\ P,\ e,\ d),$$

where encryption is performed by the procedure $e(k_1, k_2)(w) = e_{k_2}(e_{k_1}(w))$ and decryption by the procedure $d(k_1, k_2)(c) = d_{k_1}(d_{k_2}(c))$.

4.1.4 Perfect secrecy

Let P, K and C be sets of plaintexts, keys and cryptotexts. Let $Pr[K = k]$ be the probability that the key k is chosen from K and let $Pr[P = w]$ be the probability that the plaintext w is chosen from P. If for a key $k \in K$, $C(k) = \{e_k(w) | w \in P\}$, then the probability that c is the cryptotext that is transmitted is

$$Pr[C = c] = \sum_{\{k | c \in C(k)\}} Pr[K = k] Pr[P = d_k(c)].$$

For the conditional probability $Pr[C = c | P = w]$ that c is the cryptotext if w is the plaintext it holds

$$Pr[C = c | P = w] = \sum_{\{k | w = d_k(c)\}} Pr[K = k].$$

Using Bayes' conditional probability formula $Pr[y]Pr[x|y] = Pr[x]Pr[y|x]$ we get for probability $Pr[P = w | C = c]$ that w is the plaintext if c is the cryptotext the following expression

$$Pr[P = w | C = c] = \frac{Pr[P = w] \sum_{\{k | w = d_k(c)\}} Pr[K = k]}{\sum_{\{k | c \in C(k)\}} Pr[K = k] Pr[P = d_k(c)]}.$$

A cryptosystem has perfect secrecy if $Pr[P = w | C = c] = Pr[P = w]$ for all $w \in P$ and $c \in C$. That is, the *a posteriori* probability that the plaintext is w, given that the cryptotext is c is obtained, is the same as *a priori* probability that the plaintext is w.

4.2 Exercises

4.1. During the siege of a castle one of the attackers tried to choke you to death by a leather belt. You were lucky and managed to escape and steal the belt. Afterwards, you noticed that on the belt there is written the following text. Try to decrypt it.

```
AERTLTTEHAVGCEAKNTANETO
```

4.2. Encrypt the word "*cryptology*" with

(a) the Polybius square cryptosystem;

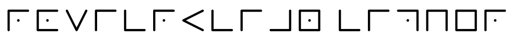
(b) the Hill cryptosystem with $M = \begin{pmatrix} 6 & 7 \\ 3 & 11 \end{pmatrix}$;

(c) the keyword Caesar cryptosystem with $k = 6$ and the keyword "SHIFT";

(d) the Autoclave cryptosystem with the keyword "KEY".

4.3. What is the relation between the permutation cipher and the Hill cipher.

4.4. Decrypt the following cryptotexts:

(a) 

(b) `CJ CI CF EI AG BI DJ DH DH DF DJ AF DG AJ`

(c) `AQ HP NT NQ UN`

Hint: The password used to encrypt this message is the name of the cryptosystem used.

(d)
```
GEOGRAPHY ANTS
MARKETING WAR
```

4.5. If possible, decode the following ciphertext obtained with the one-time pad knowing that the key used to encrypt a message starts with `GSC`:

```
GLUYM YIFGH EJPCR OFLSM DOFML QSFCDF MZHLL VDJLE
TTYNM XDKEC ALIOP DHTFN ECRKF GKDVRJ DJVMR WICKF
```

4.6. On the basis of frequency analysis it has been guessed that the most frequent cryptotext letter, Z, corresponds to the plaintext letter *o* and the second most frequent cryptotext letter, I, corresponds to *t*. Knowing that the Affine cryptosystem is used, determine its coefficients *a* and *b*.

* **4.7.** Decrypt the following cryptotext:

```
XQFXMGAFFDSCHFZGYFZRSHEGHXQZXMFQRSPEGHXQKPZNKZGHGNGHX
QDEEFDSZHQGAFVDJDZNGSDDGFIGBSHGGFQHQGAF4GARFQGNSPDYKP
GAFKSDAJHQZRAXCDSUDGZPDPDQDKNGKDZFYXQJDQNZRSHEGZYDGHQ
TKDRVGXGAF4GARFQGNSPKRGAFVDJDZNGSDSFRXJJFQYZGADGBXJFQ
ZAXNCYZGNYP64DSGZHQRCNYHQTRXXVHQTYSFZZHQTJDZZDTFDQYGA
FESFEDSDGHXQXMEFSMNJFZGAFCHZGDCZXHQRCNYFZZXJFCFZZXKUH
XNZDSGZHQRCNYHQTRXQONSHQTRAFZZKXXVKHQYHQTDQYRDSEFQGSP
QNJKFS45XQGAFCHZGHZJCFRRAHGDUHVDCEDGAFDSGXMZFRSFGBSHG
HQTDYUXRDGFYHQXSYFSGXAFCEBXJFQRXQRFDCGAFYFGDHCZXMGAFH
SCHDHZXQZXQFXMGAFSFRXJJFQYFYGFRAQHLNFZHQUXCUFZSDQYXJC
PEDHSHQTCFGGFSZXMGAFDCEADKFGDQYGAFQZNKZGHGNGHQTFDRACF
GGFSHQGAFXSHTHQDCJFZZDTFBHGAHGZEDSGQFS
```

* **4.8.** A simple substitution cipher $f : \mathbb{Z}_{26} \to \mathbb{Z}_{26}$ (a bijective function which maps plaintext letters x to cryptotext letters $f(x)$) is called self-inverting if $f(x) = f^{-1}(x)$, *i.e.* $f(f(x)) = x$.

(a) How many different self-inverting substitution ciphers are there?

(b) What is the proportion of self-inverting substitution ciphers to all simple substitution ciphers?

(c) How many self-inverting substitution ciphers which do not map any letter onto itself are there?

4.9. For the following cryptosystems, describe a chosen plaintext attack which enables the adversary to determine the key using only a single message. This message should be as short as possible.

(a) the Caesar cryptosystem;

(b) monoalphabetic substitution cryptosystem;

(c) transposition cryptosystem with a block length not greater than the number of symbols in the alphabet

(d) the Affine cryptosystem;

(e) the Vigenère cryptosystem with a key of known length.

4.10. Assume that the Affine cryptosystem is implemented in $\mathbb{Z}_{126}$.

(a) Determine the number of possible keys.

(b) For the encryption function $e(x) = 23x + 7 \pmod{126}$ find the corresponding decryption function.

* **4.11.** Decrypt the following cryptotext obtained with the Vigenère cryptosystem:

```
DLCCC QDIKW YQDFC ZVYMX GMEJV CGPRM DQYDL CWERS GYVPW SRBOG GZLCB EZVI
SXKEW RXSRL IPOUS SVCNX MLIQO GPOXY XHGDQ SCXZO EZVIR YJYVP GXXMD LCREL NWMPX
FOILO QWGMR RSSDM LMSLF ILSIL MI
SXQUI WWYQD FCMSK WYLSG YLPCK RBBIR KMLKF JOAGD LMEXR RIFOP NYJUB MRDIL XSROW
YXHAR ELQIY LPCYV KYHGP MYLPC KXRRI USPJY JRRIA YVPOW NYRBO RRC
SXKEW RLIYZ TJSGY LPCDS ROPCQ VYZLG MGMBV CCTMX HCXGC
SXKEW RLINY VRKFJ OELNM RCYQK KCKRB PYLMX GYRKE WRXSR BIOEM POXFO GMXGM EVQOS
DCITO VYVTC YTJO
PMLKP JIMRS WLOGC CWYBC ESZCX XFOGG BGSWW RKRAO WRRER MSKWE LNMRC ENZPG MERSS
LDLYD XFOWW CXCWF COEQI XMEWC BIOEM PSREX IGDLC BQCXX YVWRB EGXRM BXFOO LYAJO
HEOSD KPMXK QOVGO WMPVS VIQDS MLWCB ZC
```

4.12. Let S be an endomorphic cryptosystem. Let $S^2 = S \times S$. Which of the following simple cryptosystems – the Caesar, Vigenère, Hill or Affine cryptosystem – would you prefer to use as S^2. Explain your reasoning.

4.13. You have two devices – a machine E, which encrypts the given text (suppose that it provides perfect encryption) and a machine H, which performs the Huffmann compression of a given message. Decide the order in which you should use these machines to get the shorter encrypted message. Justify your answer.

* **4.14.** Let $\mathcal{C} = (P, C, K, e, d)$ be a cryptosystem.

(a) Suppose that $\mathcal{C}$ is perfectly secure. Show that for any $m \in P$ and $c \in C$ it holds that $Pr[C = c|P = m] = Pr[C = c]$.

(b) Suppose that for any m, $m' \in P$ and $c \in C$ it holds that

$$Pr[P = m|C = c] = Pr[P = m'|C = c].$$

Show that $\mathcal{C}$ is perfectly secure.

4.15. Let $\mathcal{C}$ be a cryptosystem with the plaintext space $P = \{x, y, z\}$, the key space $K = \{k_1, k_2, k_3\}$ and the cryptotext space $C = \{a, b, c\}$. Let the probability distributions $Pr[P = w]$ and $Pr[K = k]$ be defined as $Pr[P = x] = \frac{3}{8}$, $Pr[P = y] = \frac{1}{8}$, $Pr[P = z] = \frac{1}{2}$ and $Pr[K = k_1] = \frac{1}{3}$, $Pr[K = k_2] = \frac{1}{6}$, $Pr[K = k_3] = \frac{1}{2}$, respectively. Let the encryption function be given by the following table:

	x	y	z
k_1	a	b	c
k_2	b	c	a
k_3	c	a	b

(a) Determine the probability distribution $Pr[C = c]$.

(b) Is the proposed cryptosystem perfectly secure? Provide your reasoning.

* **4.16.** A cryptosystem is said to be 2-perfectly secure if two cryptotexts encrypted using the same key provide no information about the corresponding plaintexts.

(a) Propose a formal definition of a 2-perfectly secure cryptosystem.

(b) Show that one-time pad is not 2-perfectly secure.

* **4.17.** Let p be a prime. Consider the Hill system with alphabet of order p and matrices of degree 2. What is the number of keys?

* **4.18.** Write a short interesting text in English or Czech/Slovak concerning cryptography or informatics without using the letter "E".

4.3 Solutions

4.1. A scytale was used to encrypt the message. One can read the message if the belt is wrapped around a cylinder with the right diameter. Scytale used to encrypt this message allowed one to write three letters around in a circle. We can write the cryptotext in columns of length 3:

```
ATTACKAT
ELEVENNO
RTHGATE
```

Finally, we can read the secret message as "*attack at eleven north gate*".

4.2.

(a) Let the key be the table:

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

Then $c \to$ AH, $r \to$ DG, and so on until we get the resulting cryptotext "`AHDGEICKDICICFCIBGEI`".

(b) We have to convert the word *cryptology* to a sequence of vectors over $\mathbb{Z}_{26}$ of length 2. Each vector will then be multiplied by the matrix M and the resulting vectors transformed back to a string.

$$cr \to v_1 = \begin{pmatrix} 2 \\ 17 \end{pmatrix},\ yp \to v_2 = \begin{pmatrix} 24 \\ 15 \end{pmatrix},\ to \to v_3 = \begin{pmatrix} 9 \\ 14 \end{pmatrix},$$

$$lo \to v_4 = \begin{pmatrix} 11 \\ 14 \end{pmatrix},\ gy \to v_5 = \begin{pmatrix} 6 \\ 24 \end{pmatrix}$$

$$Mv_1 = \begin{pmatrix} 1 \\ 11 \end{pmatrix} \to BL,\ Mv_2 = \begin{pmatrix} 15 \\ 3 \end{pmatrix} \to PD,\ Mv_3 = \begin{pmatrix} 4 \\ 3 \end{pmatrix} \to ED,$$

$$Mv_4 = \begin{pmatrix} 3 \\ 5 \end{pmatrix} \to IF,\ Mv_5 = \begin{pmatrix} 22 \\ 22 \end{pmatrix} \to WW.$$

The cryptotext is therefore "`BLPDEDIFWW`".

(c) For the given key we get this permutation table:

```
abcdefghijklmnopqrstuvwxyz
UVWXYZSHIFTABCDEGJKLMNOPQR
```

The letter c maps to W, r to J and so on, until we get the cryptotext `WJQELDADSQ`.

(d) The autoclave key is formed with the keyword "KEY" concatenated with the plaintext, to obtain the ciphertext we perform the following operation:

```
  cryptology 02 17 24 15 19 14 11 14 06 24
+ KEYCRYPTOL 10 04 24 02 17 24 15 19 14 11
= MVWRKMAHUJ 12 21 22 17 10 12 00 07 20 09
```

The cryptotext is "`MVWRKMAHUJ`".

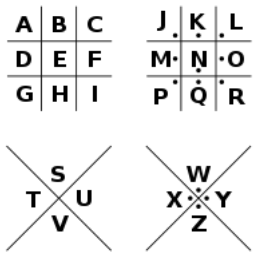
4.3. The permutation cipher is a special case of the Hill cipher.
Let π be a permutation of the set $\{1, \ldots, n\}$. We can define an $n \times n$ permutation matrix $M_\pi = (m_{ij})$ as follows:

$$m_{ij} = \begin{cases} 1 & \text{if } i = \pi(j) \\ 0 & \text{otherwise} \end{cases}$$

Using the matrix M_π in the Hill cryptosystem is equivalent to perform encryption using the permutation π. Decryption in the Hill cryptosystem is done with the matrix $M_\pi^T = M_\pi^{-1} = M_{\pi^{-1}}$.

4.4.

(a) The given message is encrypted with the Pigpen cipher and can be easily decoded with the following tables:



The hidden message is "*rosicrucian cipher*".

(b) One can notice that in the given cryptotext similar characters appear on even positions and similar on odd positions. One can guess that message was encrypted with the Polybius cryptosystem and can be decoded to "*polybius square*".

(c) This is the Playfair cipher with the key "PLAYFAIR". The Playfair square is as follows:

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

and the hidden message reads "*wheatstone*" (Charles Wheatstone was an inventor of the Playfair cipher).

(d) These are anagrams for "*steganography*" and "*watermarking*".

4.5. Assuming the one-time pad was used correctly, that is, no other message was encrypted with the same key and the key is truly random, we are unable to decrypt this message even if we know that key starts with `GSC`. We can only say that the first three characters of the decrypted message are *ats*. Without knowledge of the whole key we could not decrypt the whole plaintext message.

4.6. We have $e_{(a,b)}(o) = \mathrm{Z}$ and $e_{(a,b)}(T) = \mathrm{I}$ which can be rewritten to the following system of linear equation:

$$\begin{aligned} a \cdot 14 + b &= 25 \\ a \cdot 19 + b &= 8 \end{aligned}$$

Solving the system modulo 26 gives $a = 7$ and $b = 5$.

4.7. Frequency analysis shows that F and G are the most frequent letters and that GAF is with 16 occurences the most frequent trigram, followed up with the trigram HQT with 10 occurences. We can guess that GAF corresponds to the word *the* and HQT corresponds to *and*. We can test whether the Affine cryptosystem was used by determining coefficients from 2 equations corresponding to proposed letter transformations and testing if these coefficients are valid for the whole text. From equations $t \to \mathrm{G}$ and $h \to \mathrm{A}$ we get $a = 7$ and $b = 3$. These coefficients are valid for transformation $e \to \mathrm{F}$ (decrypting HQT actually yields *ing*). The plaintext finally reads:

One of the earliest descriptions of encryption by substitution appears in the Kama-sutra, a text written in the 4th century AD by the Brahmin scholar Vatsyayana, but based on manuscripts dating back to the 4th century BC. The Kama-sutra recommends that women should study 64 arts, including cooking, dressing, massage and the preparation of perfumes. The list also includes some less obvious arts, including conjuring, chess, bookbinding and carpentry. Number 45 on the list is mlecchita-vikalpa, the art of secret writing, advocated in order to help women conceal the details of their *liaisons. One of the recommended techniques involves randomly pairing letters of the alphabet, and then substituting each letter in the original message with its partner.*

4.8.

(a) Let f be a self-inverting permutation over $\mathbb{Z}_n$.

Suppose n pairs of letters map to each other and remaining $26 - 2n$ letters stay fixed. These n pairs can be chosen in the following way. First two letters are chosen in $\frac{26\times 25}{2}$ ways. Next pair can be chosen from the remaining 24 in $\frac{24\times 23}{2}$ ways and n-th pair can be chosen in $\frac{(28-2n)\times(27-2n)}{2}$ ways. Multiplying together and cancelling the ordering of n pairs we obtain

$$P(n) = \frac{26!}{(26-2n)! \times 2^n \times n!}.$$

The total number of self-inverting functions is then

$$\sum_{i=0}^{13} \frac{26!}{(26-2n)! \times 2^n \times n!} = 532985208200576$$

Another approach could be to derive a recursive function $\phi(n)$ that returns the number of self-inverting permutations over a set of cardinality n. There is only one permutation over a singleton set, the identity, which is self-inverting: $\phi(1) = 1$.
There are two permutations over a two-element set, the identity and the transposition. Both are self-inverting: $\phi(2) = 2$.
Now, if there are n elements, either $f(0) = 0$ and then there are $\phi(n-1)$ possibilities how to permute the rest, or $f(0) = i$, $i \in \{1, \ldots, n-1\}$, implying $f(i) = 0$, and then there are $\phi(n-2)$ ways one can permute the rest. Together, we obtain the following recurrent formula:

$$\phi(n) = \phi(n-1) + (n-1)\phi(n-2)$$

For $n = 26$, $\phi(26) = 532985208200576$.

(b) The number of substitution ciphers equals the number of permutations. For a set of 26 elements it is 26!. The proportion is:

$$\frac{\phi(26)}{26!} = 1.32 \cdot 10^{-12}$$

(c) From the solution from part (a) one can easily see that picking 13 pairs can be done in

$$P(13) = \frac{26!}{2^{13} \times 13!} = 25 \cdot 23 \cdots 5 \cdot 3 \cdot 1 = \Pi_{k=1}^{13}(2k-1) = 7905853580625$$

ways.

4.9.

(a) A single letter plaintext.

(b) For an alphabet of n symbols, it suffices to send a message of length $n-1$ which contains distinct symbols.

(c) Message of the block length which contains each symbol at most once.

(d) Plaintext consisting of two distinct letters. We can easily find the key by solving the system of linear equations.

(e) Plaintext of the key length.

4.10.

(a) The keyspace is the set $\{(a, b) \in \mathbb{Z}_{126} \times \mathbb{Z}_{126} \,|\, \gcd(a, 126) = 1\}$. There are 126 possibilities for b and $\varphi(126)$ possibilities for a (where φ is the Euler totient function).

$$\varphi(126) = \varphi(2) \cdot \varphi(3^2) \cdot \varphi(7) = 1 \cdot 3 \cdot 2 \cdot 6 = 36$$

The number of possible keys is $36 \cdot 126 = 4536$.

(b) The corresponding decryption function is

$$d(y) = 23^{-1}(y - 7) \pmod{126}.$$

To find the inverse of 23 we use the Euclidean algorithm:

$$126 = 5 \cdot 23 + 11$$

$$23 = 2 \cdot 11 + 1$$

$1 = 23 - 2 \cdot 11 = 23 - 2(126 - 5 \cdot 23) = 11 \cdot 23 - 2 \cdot 126$. Because $-2 \cdot 126 \equiv 0 \pmod{126}$, we get $23^{-1} = 11 \pmod{126}$.

$$d(y) = 11(y - 7) \pmod{126}$$
$$d(e(x)) = 11((23x + 7) - 7) = 11(23x) = x \pmod{126}$$

4.11. We can determine the keylength using the Friedman method:

$$l = \sum_{i=1}^{26} \frac{n_i(n_i - 1)}{n(n-1)} = 0.475883826064,$$

$$L = \frac{0.027n}{(n-1)l - 0.038n + 0.065} = 2.80672431976 = 3.$$

We can assume that the length of the key is 3, Frequency analysis for each position modulo the key length gives the following results:

Position 0	Position 1	Position 2
O (29)	I (23)	C (26)
D (16)	X (20)	R (26)
S (15)	W (16)	L (17)
...	...	...

The most frequent character in the English text is e, so we will try to substitute it for the most frequent symbols in the cryptotext.With substitutions O $\rightarrow e$, I $\rightarrow e$, C $\rightarrow e$ (which correspond to the key "KEY") we get the following plaintext (which is actually the Kerckhoff's principle):

The system must be practically if not mathematically indecipherable.
It must not be required to be secret and it must be able to fall into the hands of the enemy without inconvenience.
Its key must be communicable and retainable without the help of written notes and changeable or modifiable at the will of the correspondents.
It must be applicable to telegraphic correspondence.
It must be portable and its usage and function must not require the concourse of several people.
Finally it is necessary given the circumstances that command its application that the system be easy to use requiring neither mental strain nor the knowledge of along series of rules to observe.

4.12. Each of the given cryptosystem is an *idempotent cryptosystem, i.e.* $S^2 = S$.

(a) **Caesar**: Composition of two shifts is again a shift. If k_1, k_2 are the keys then it is equivalent to use a single shift with the key $k = k_1 + k_2 \pmod{|P|}$.

(b) **Vigenère**: $P = C = (\mathbb{Z}_{26})^n$ and $K = (\mathbb{Z}_{26})^m$. Let $k_1 = (k_{1_1}, \ldots, k_{1_m})$, $k_2 = (k_{2_1}, \ldots, k_{2_m})$ be keys of length m. Then $k = (k_{1_1} + k_{2_1} \pmod{n}, \ldots, k_{1_m} + k_{2_m} \pmod{n}) \in K$.

(c) **Hill**: K is a set of invertible matrices of a degree n. If $k_1 = M_1$, $k_2 = M_2$ then $k = M_1 {\cdot} M_2 \in K$. Invertible matrices with the multiplication operation form a group $GL(n)$.

(d) **Affine:** $K = \mathbb{Z}_n^* \times \mathbb{Z}_n$. If $k_1 = (a_1, b_1)$, $k_2 = (a_2, b_2)$ then $k = (a_1a_2, a_1b_2 + b_1) \in K$,

4.13. One should first compress and then encrypt. If we first encrypt then the resulting cryptotext is indistinguishable from random text and compression algorithm fails because it cannot find compressible patterns to reduce size.

4.14.

(a)
$$Pr[C = c|P = m]Pr[P = m] = Pr[P = m|C = c]Pr[C = c].$$

We have (Bayes' theorem)

$$Pr[C = c|P = m] = \frac{Pr[P = m|C = c]Pr[C = c]}{Pr[P = m]}.$$

Perfect secrecy gives

$$Pr[C = c|P = m] = \frac{Pr[P = m]Pr[C = c]}{Pr[P = m]} = Pr[C = c].$$

(b) Let m, $m' \in P$ be arbitrary plaintexts and let c, $c' \in C$ be any cryptotexts. From the assumption one can derive the following:

$$Pr[P = m] = \sum_{c \in C} Pr[P = m|C = c]Pr[C = c] =$$

$$= \sum_{c \in C} Pr[P = m'|C = c]Pr[C = c] = Pr[P = m'].$$

Therefore

$$Pr[C = c|P = m] = \frac{Pr[P = m|C = c]Pr[C = c]}{Pr[P = m]} =$$

$$= \frac{Pr[P = m'|C = c]Pr[C = c]}{Pr[P = m']} = Pr[C = c|P = m'].$$

From both equalities one can derive the following relation:

$$Pr[C = c] = \frac{1}{|P|}|P|Pr[C = c|P = m] = Pr[C = c|P = m]$$

from which the equality characterizing perfect secrecy can be obtained.

4.15.

(a) For $c \in C$, $Pr[C = c] = \sum_{\{k|c\in C(k)\}} Pr[K = k]Pr[P = d_k(c)]$.

$$Pr[C = a] = Pr[K = k_1]Pr[P = d_{k_1}(a)]+Pr[K = k_2]Pr[P = d_{k_2}(a)]+Pr[K = k_3]Pr[P = d_{k_3}(a)]$$

$$Pr[C = a] = Pr[K = k_1]Pr[P = x] + Pr[K = k_2]Pr[P = z] + Pr[K = k_3]Pr[P = y]$$

$$Pr[C = a] = \frac{13}{48}$$

Similarly,

$$Pr[C = b] = \frac{17}{48} \text{ and } Pr[C = c] = \frac{18}{48}.$$

(b) We have

$$Pr[C = a|P = x] = \sum_{\{k|x\in d_k(c)\}} Pr[K = k] = Pr[K = k_1] = \frac{1}{3}$$

$$Pr[C = a|P = x] \neq Pr[C = a]$$

Therefore, C is not perfectly secure (see previous exercise and equivalent definition of perfect secrecy).

4.16.

(a) A cryptosystem is 2-perfectly secure if for any m, $m' \in P$ and any c, $c' \in C$

$$Pr[(P_1, P_2) = (m, m')|(C_1, C_2) = (c, c')] = Pr](P_1, P_2) = (m, m')].$$

(b) Let $|P| = |C| = |K| = n$. We have $Pr[(P_1, P_2) = (m, m')] = \frac{1}{n^2}$. Since cryptotexts c, c' were produced with the same key, there are only n pairs of plaintexts which can be encrypted to cryptotexts c and c'. For these plaintext pairs we have $Pr[(P_1, P_2) = (m, m')|(C_1, C_2) = (c, c')] = \frac{1}{n}$. Therefore one-time pad is not 2-perfect secure.

4.17. One has to found the number of matrices of degree 2 invertible over $\mathbb{Z}_p$ field. We use the fact that, over a field , a square matrix is invertible if and only if its columns are linearly independent. We have to describe how 2 column vectors over $\mathbb{Z}_p$ can be chosen such that they will form an invertible matrix. The only restriction on the first vector is that it be nonzero, because this would destroy the linear independence of the columns. Therefore, there are $p^2 - 1$ possibilities for the first column. The second column can be chosen in $p^2 - p$ ways to avoid linear combinations of the first column. Thus, there are $(p^2 - 1)(p^2 - p)$ possible keys.

4.18. *(by L. Pekárková)*
To tak kdysi šli Bob a Bobík spolu na procházku. Do rytmu si zpívali písničky a skákali do kaluží. Hodili pucku do okna a to bylo rozbito. Koukali na to, avšak pro jistotu zdrhli pryč. Jak tak utíkali, potkali Barboru — kamarádku. Šťastná to dívka Barbora vyzvídala, proč oba utíkají. Kupodivu s nimi začala taky utíkat, aniž by jí Bob či Bobík vyklopili příhodu s puckou. Tato partička doutíkala až k obchodu s počítači. Tam si chvíli oddychla, aby nabrala síly. Tady si všimli dvou vystavovaných kousků. Zalíbily si ty kousky natolik, aby si další ráno došli do obchodu a koupili si každý svou mašinku (za maminčiny prachy :)). Nyní už mohli hrát svou milou hru i po síti. Pařba jim imponovala natolik, aby si podali (zas všichni tři) přihlášku na fakultu informatiky, aby si tam ujasnili znalosti, nabyvší za dlouhou hráčskou dráhu. Po strastiplných zkouškových obdobích si prokousali sic úzkou uličku k titulu a začali podnikat v oboru. Z počátku jim obchody vázly, pak však došlo k zlomu. Vymysleli si vlastní šifrovací programy, což jiní považovali za hloupost. Avšak programy si našly na trhu „kamarády“, a tak firma tří informatiků vzrůstala. Jak vidno i malá pucka by mohla ovlivnit život i nás ostatních :).

Chapter 5

Public-Key Cryptography, I.

5.1 Introduction

Symmetric cryptography is based on the use of secret keys. This means that massive communication using symmetric cryptography would require the distribution of large amount of data in some other, very secure, way, unless we want to compromise the secrecy by reusing the secret key data.

The answer to this problem is the *public key* cryptography, also known as asymmetric cryptography. Public key cryptography uses different keys for encryption and decryption. Every party has her secret decryption key, while the encryption key is publicly known. This allows anyone to encrypt messages, while only the one in the possession of decryption key can decrypt them.

The main difficulty of the public-key cryptosystems is to make the decryption impossible for those without the secret key. To achieve this goal so called *trapdoor one-way* functions are used for the encryption. These functions have easy to compute their inverse, but only if we posses a certain trapdoor information.

5.1.1 Diffie-Hellman protocol

This is the first asymmetric protocol designed to solve the secret-key distribution problem. Using this protocol two parties, say Alice and Bob, generate and share a pretty random and pretty secret key. The protocol uses modular exponentiation as the one-way function.

The parties first agree on large primes p and a $q < p$ of large order in $\mathbb{Z}_p^*$. The protocol then proceeds as follows

- Alice randomly chooses a large $x < p - 1$ and computes $X = q^x \mod p$.
- Bob also chooses a large $y < p - 1$ and computes $Y = q^y \mod p$.
- The two parties then exchange X and Y, while keeping x and y secret.
- Alice computes her key $Y^x \mod p$ and Bob computes $X^y \mod p$ and this way they both have the same key $k = q^{xy} \mod p$.

5.1.2 Knapsack cryptosystem

Knapsack cryptosystem is based on the infeasibility of solving the general *kanpsack problem*:

Knapsack problem: Given a vector of integers $X = (x_1, \dots, x_n)$ and an integer c. Determine a binary vector $B = (b_1, \dots b_n)$ such that $XB^\top = c$.

The knapsack problem is easy if the vector is superincreasing, that is for all $i > 1$ it holds $x_i > \sum_{j=1}^{i-1} x_j$. The knapsack cryptosystem first uses a secret data to change a knapsack problem with a superincreasing vector to a general, in principle infeasible, knapsack problem.

- Choose a superincreasing vector $X = (x_1, \dots, x_n)$.

- Choose m and u such that $m > 2x_n$, $\gcd(m, u) = 1$.
- Compute $X' = (x'_1, \dots, x'_n)$, $x'_i = ux_i \mod m$.

X' is then the public key while X, u, m is the secret trapdoor information.

Encryption of a word w is done by computing $c = X'w^\top$.

To decrypt ciphertext c we compute $u^{-1} \mod m$, $c' = u^{-1}c \mod m$ and solve the knapsack problem with X and c'.

5.1.3 McEliece cryptosystem

Just like in the knapsack cryptosystem, McEliece cryptosytem is based on tranforming an easy to break cryptosystem into one that is hard to break. The cryptosystem is based on an easy to decode linear code, that is then transformed to a generally infeasible linear code. The class of starting linear codes is that of the *Goppa codes*, $[2^m, n - mt, 2t + 1]$-codes, where $n = 2^m$.

- Let G be a generating matrix of an $[n, k, d]$ Goppa code C.
- Let S be a $k \times k$ binary matrix invertible over $\mathbb{Z}_2$.
- Let P be an $n \times n$ permutation matrix.
- $G' = SGP$.

Public encryption key is G' and the secret is (G, S, P).

Encryption of a plaintext $w \in (\mathbb{Z}_2)^k$ is done by computing $e_K(w, e) = wG' + e$, where e is each time a new random binary vector of length n and weight t.

Decryption of a ciphertext $c = wG' + e \in (\mathbb{Z}_2)^n$ is done by first computing $c_1 = cP^{-1}$. Then decoding c_1 to get $w_1 = wS$, and finally computing the plaintext $w = w_1S^{-1}$.

5.1.4 RSA cryptosystem

RSA is a very important public-key cryptosystem based on the fact that prime multiplication is very easy while integer factorization seems to be unfeasible.

To set up the cryptosystem we first choose two large (about 1024 bits long) primes p, q and compute

$$n = pq, \quad \phi(n) = (p - 1)(q - 1).$$

Then we choose a large d such that $\gcd(d, \phi(n)) = 1$ and compute $e = d^{-1} \mod \phi(n)$. The public key is then the modulus n and the encryption exponent e. The trapdoor information is the primes p, q and the decryption exponent d.

Encryption of a plaintext w is done by computing $c = w^e \mod n$.

Decryption of a ciphertext c is done by computing $w = c^d \mod n$.

5.1.5 Rabin-Miller's prime recognition

Rabin-Miller's prime recognition is a simple randomized Monte Carlo algorithm that decides whether a given integer n is a prime. It is based on the following lemma.

Lemma 5.1.1. *Let $n \in \mathbb{N}$, $n = 2^s d + 1$, d is odd. Denote, for $1 \le x < n$, by $C(x)$ the condition:*

$$x^d \not\equiv 1 \pmod n \text{ and } x^{2^r d} \not\equiv -1 \pmod n \text{ for all } 1 < r < s.$$

If $C(x)$ holds for some $1 \le x < n$, 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 then chooses random integers $x_1, \dots, x_m$ such that $1 < x_j < n$ and evaluates $C(x_j)$ for every one of them. If $C(x_j)$ holds for some x_j then n is not a prime. If it does not hold for any of the chosen integers n is a prime with probability of error 2^{-m}.

5.2 Exercises

5.1.

(a) Consider the Diffie-Hellman protocol with $q = 3$ and $p = 353$. Alice chooses $x = 97$ and Bob chooses $y = 233$. Compute X, Y and the key.

(b) Design an extension of the Diffie-Hellman protocol that allows three parties Alice, Bob and Charlie to generate a common secret key.

5.2. Alice and Bob computed a secret key k using Diffie-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 were very surprised when they found that $k = k'$. Determine the value of both keys and explain why the keys are identical.

5.3. To simplify the implementation of Diffie-Hellman protocol one can replace the multiplicative group $(\mathbb{Z}_p, \cdot)$ by the additive group $(\mathbb{Z}_p, +)$. How is security affected?

5.4. Consider the Bloom's key pre-distribution protocol (see the lecture slides) with two parties Alice and Bob and a trusted authority Trent. Let $p = 44887$ be the publicly know prime and $r_A = 4099$ and $r_B = 31458$ be the unique public numbers. Let $a = 556$, $b = 13359$ and $c = 3398$ be the secret random numbers. Use Bloom's protocol to distribute a secret key between Alice and Bob.

5.5. Bob sets up the Knapsack cryptosystem with $X = (2, 5, 8, 17, 35, 70)$, $m = 191$, $u = 34$ so that Alice can send him messages.

(a) Find Bob's public key X'.

(b) Encode the messages 101010 and 100010.

(c) Perform in details Bob's decryption of $c_1 = 370$ and $c_2 = 383$.

5.6. Consider the McEliece cryptosystem with

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}, \quad S = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \end{pmatrix}, \quad P = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

(a) Compute the public key G'.

(b) Using the error vector $e = 0010000$ encode the message $w = 1001$.

(c) Decode cryptotext $c = 0110110$.

5.7. Consider the RSA cryptosystem with $p = 43$, $q = 59$ and $d = 937$.

(a) Determine the encryption exponent e.

(b) Encrypt the plaintext and 13487947504.

(c) Decrypt the ciphertext 175807260375 that was sent in subwords of size 4.

5.8. Consider the RSA cryptosystem with $n = 1363$. It has been revealed that $\phi(n) = 1288$. Use this information to factor n.

5.9. Consider the RSA cryptosystem with a public key (n, e). An integer m, $m \leq n-1$, is called a fixed point if $m^e \equiv m \pmod{n}$. Show that if m is a fixed point then $n - m$ is also a fixed point.

5.10. Suppose that Eve receives a cryptotext $c = m^e \mod n$ encrypted using the RSA cryptosystem. Suppose further that she is permitted to request a decryption of a single cryptotext $c' \neq c$. Show how she can find the plaintext m.

5.11. Design of parameters for the RSA cryptosystem starts with choosing two large primes. Because these primes are part of the private key, they have to be chosen very carefully. More precisely, they need to be chosen at random by a cryptographically secure random number generator. Failing to do so can lead to problems. Indeed, consider the following set of RSA moduli, chosen by an imperfect random number generator, biased towards some numbers (some numbers appear with larger probability than others). Determine which of these moduli are secure:

$$\{8844679, 11316499, 13490941, 18761893, 21799573, 22862761, 48456493, 43831027, 58354333\}.$$

Do not use brute force factorization.

5.12. Use the Rabin-Miller's Monte Carlo algorithm for prime recognition to decide whether the number $n = 5417$ is a prime and state the accuracy of your outcome. Use the numbers $x_1 = 58$, $x_2 = 864$ and $x_3 = 3312$ as the random integers in the algorithm.

* **5.13.** Both Alice and Bob use the RSA cryptosystem with the same modulus n and encryption exponents e_A and e_B such that $\gcd(e_A, e_B) = 1$. Let a third user Charlie send the same message m to both Alice and Bob using their individual encryption exponents. Eve intercepts the encrypted messages $c_A = m^{e_A} \pmod{n}$ and $c_B = m^{e_B} \pmod{n}$. She then computes $x_1 = e_A^{-1} \pmod{e_B}$ and $x_2 = (x_1 e_A - 1)/(e_B)$.

(a) How can Eve compute m using c_A, c_B, x_1 and x_2?

(b) Use the proposed method to compute m if $n = 18721$, $e_A = 43$, $e_B = 7717$, $c_A = 12677$ and $c_B = 14702$.

* **5.14.** Alice, Bob and Eve use the RSA cryptosystem with $n = 99443$. Let $e_A = 7883$, $e_B = 5399$ and $e_E = 1483$ be the corresponding public key exponents. Let messages be written in ASCII, divided into subwords of length 5, each subword being encrypted separately. Imagine that you are Eve and you have captured the following message intended for Bob which was sent by Alice:

$$162784902043554002 79.$$

You know your $d_E = 3931$. Decrypt the cryptotext (do not use brute force or factorization).

5.3 Solutions

5.1.

(a) Following the Diffie-Hellman protocol we compute $X = q^x \mod p = 3^{97} \mod 353 = 40$ and $Y = q^y \mod p = 3^{233} \mod 353 = 248$. The secret key k is then

$$k = q^{xy} \mod p = 3^{97 \cdot 233} \mod 353 = 3^{73} \mod 353 = 160.$$

Note: Euler's totient theorem was used to simplify the last computation.

(b) As in the two-party case, the three parties first agree on large primes p and $q < p$ of large order in $\mathbb{Z}_p^*$. Alice, Bob and Charlie then choose large secret random integers x, y and z respectively, such that $1 \leq x, y, z < p-1$.

Alice then computes $X_1 = q^x \mod p$ and sends it to Bob, who computes $X_2 = X_1^y \mod p$ and sends X_2 to Charlie. Charlie can now get his key $k_C = X_2^z \mod p$.

Bob then continues by computing $Y_1 = q^y \mod p$ and sends it to Charlie, who computes $Y_2 = Y_1^z \mod p$ and sends it to Alice. Alice then computes $k_A = Y_2^x \mod p$.

This procedure is repeated the third time with Charlie computing $Z_1 = q^z \mod p$ and sending it to Alice, Alice computing $Z_2 = Z_1^x \mod p$ and sending it to Bob and Bob computing $k_B = Z_2^y \mod p$.

It can easily be seen that $k = k_A = k_B = k_C = q^{xyz} \mod p$ is the secret key now shared between all three parties. The public information in this protocol is $p, q, q^x, q^y, q^z, q^{xy}, q^{yz}, q^{xz}$. Computing q^{xyz} from this public information is at least as hard as breaking the original protocol.

5.2. First we compute the secret keys k and k' individually:

$$k \equiv q^{xy} \equiv 4^{400 \cdot 134} \equiv 2^{230 \cdot 466} \cdot 2^{20} \equiv 2^{\phi(p) \cdot 230} \cdot 2^{20} \equiv 20^{20} \equiv 161 \pmod{467}.$$

$$k' \equiv q^{x'y} \equiv 4^{167 \cdot 134} \equiv 2^{96 \cdot 466} \cdot 2^{20} \equiv 2^{\phi(p) \cdot 96} \cdot 2^{20} \equiv 20^{20} \equiv 161 \pmod{467}.$$

Note: Euler's totient theorem and the fact that $\phi(p) = p - 1 = 466$ was used to simplify the calculation. This also shows the reason why $k = k'$. We can see that $2x \equiv 2x' \pmod{p-1}$, indeed $2 \cdot 400 \equiv 2 \cdot 167 \equiv 334 \pmod{467}$, and so $k \equiv 2^{2xy} \equiv 2^{2x'y} \equiv k' \pmod{p}$ due to the Euler's totient theorem.

5.3. By replacing the multiplicative group by the additive group we change the calculation of X and Y to

$$X = qx \mod p$$
$$Y = qy \mod p.$$

But now from X and Y we can easily obtain the secrets x and y by first computing the multiplicative inverse q^{-1}

$$q^{-1} \equiv q^{p-1} q^{-1} \equiv q^{p-2} \pmod{p}.$$

Using this we can then obtain x

$$x \equiv q^{-1} qx \equiv q^{-1} X \pmod{p}$$

and also y in the same way. But if we know both secrets we can now easily compute the secret key $k = xyq \mod p$. This simplification is therefore not secure as the inverse to multiplication by known number is easy to compute.

5.4. First Trent calculates a_A, a_B, b_A and b_B:

$$a_A = 556 + 13359 \cdot 4099 \mod 44887 = 41844, \quad a_B = 556 + 13359 \cdot 31458 \mod 44887 = 15884,$$

$$b_A = 13359 + 3398 \cdot 4099 \mod 44887 = 26791, \quad b_B = 13359 + 3398 \cdot 31458 \mod 44887 = 31696.$$

He then sends a_A, b_A to Alice and a_B, b_B to Bob. Alice now computes her key

$$K_{AB} = a_A + b_A r_B \mod p = 41844 + 26791 \cdot 31458 \mod 44887 = 34810.$$

And Bob does the same

$$K_{BA} = a_B + b_B r_A \mod p = 15884 + 31696 \cdot 4099 \mod 44887 = 34810,$$

and as expected we have $K_{AB} = K_{BA}$.

5.5.

(a) Following the Knapsack protocol we have $x'_i = ux_i \mod m$ and therefore

$$X' = 34 \cdot (2, 5, 8, 17, 35, 70) \mod 191 = (68, 170, 81, 5, 44, 88).$$

(b) Alice encrypts a message w by computing $X'w^\top$ so the encryption is

$$(68, 170, 81, 5, 44, 88)(1, 0, 1, 0, 1, 0)^\top = 193$$

$$(68, 170, 81, 5, 44, 88)(1, 0, 0, 0, 1, 0)^\top = 112.$$

(c) Bob first computes $u^{-1} \mod m = 118$. He then computes the cryptotext for the original Knapsack problem $c'_1 = u^{-1}c_1 \mod m = 118 \cdot 370 \mod 191 = 112$ and $c'_2 = u^{-1}c_2 \mod m = 118 \cdot 383 \mod 191 = 118$.

Bob now has to solve the knapsack problem with the superincreasing vector X and the cryptotexts c'_1 and c'_2. First we solve the problem for $c'_1 = 112$:

$$112 > 70 = x_6$$

$$x_6 = 70 > 112 - 70 = 42 > 35 = x_5$$

$$x_3 = 8 > 42 - 35 = 7 > 5 = x_2$$

$$7 - 5 = 2 = x_1.$$

We have the sixth, fifth, second and first bit equal to 1. Therefore $w_1 = 110011$. For the second cryptotext $c'_2 = 118$ we get:

$$118 > 70 = x_6$$

$$x_6 = 70 > 118 - 70 = 48 > 35 = x_5$$

$$x_4 = 17 > 48 - 35 = 13 > 8 = x_3$$

$$13 - 8 = 5 = x_2.$$

So we have the sixth, fifth, third and second bit equal to 1. Therefore $w_2 = 011011$.

5.6.

(a) According to the protocol G' is given by $G' = SGP$ so

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

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

(b) Encryption of a word w using an error vector e is done by computing

$$e_K(w, e) = wG' + e = (1001) \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \end{pmatrix} + (0010000) = (1000110)$$

(c) Following the protocol we start the decryption of c by computing $c_1 = cP^{-1}$, but because P is an orthogonal matrix its inverse is equal to its transpose so

$$c_1 = (0110110)\begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} = (1000111)$$

Now we decode c_1. The syndrome of c_1 is (001) with coset leader (0000001), so the error is only on the 7th bit. Without the error the code word is 1000110. But that is the first row of the generating matrix G, so the decoded word is $w_1 = wS = 1000$. Now we just find the inverse matrix S^{-1} and compute $w_1 S^{-1}$ to obtain the word w.

$$S^{-1} = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix}$$

and $w = w_1 S^{-1} = (1101)$.

5.7.

(a) First we compute the modulus $n = pq = 43 \cdot 59 = 2537$ and its Euler's totient function $\phi(n) = (p-1)(q-1) = 42 \cdot 58 = 2436$. Then we find the encryption exponent $e = d^{-1} \mod \phi(n)$. To find the inverse we can use the extended Euclidean Algorithm and Bezout's identity for d and $\phi(n)$:

$$\begin{aligned} 2436 &= 3 \cdot 937 - 375 \\ 937 &= 2 \cdot 375 + 187 \\ 375 &= 2 \cdot 187 + 1 \end{aligned}$$

The Bezout's identity is then $1 = 375 - 2 \cdot 187$. Going backwards we obtain

$$\begin{aligned} 1 &= 375 - 2 \cdot (937 - 2 \cdot 375) \\ &= -2 \cdot 937 + 5 \cdot 375 \\ &= -2 \cdot 937 + 5 \cdot (-2436 + 3 \cdot 937) \\ &= -5 \cdot 2436 + 13 \cdot 937, \end{aligned}$$

which gives us $13 \cdot 937 \equiv 1 \pmod{\phi(n)}$ so $e = 13$.

(b) Because we can only send messages smaller than n we split the plaintext into four subwords of size 3: 134 879 475 204. The encryption of a subword w is then done by computing $w^e \mod n$:

$$\begin{aligned} 134^{13} &\mod 2537 = 248 \\ 879^{13} &\mod 2537 = 579 \\ 475^{13} &\mod 2537 = 1441 \\ 204^{13} &\mod 2537 = 2232 \end{aligned}$$

So the whole encrypted message is 0248057914412232.

(c) Splitting the message into subwords of size 4 we get: 1758 0726 0375. Decryption of a subword c is done by computing $c^d \mod n$:

$$\begin{aligned} 1758^{937} &\mod 2537 = 397 \\ 726^{937} &\mod 2537 = 569 \\ 375^{937} &\mod 2537 = 169 \end{aligned}$$

So the whole decrypted message is 397569169.

5.8. We need to solve the following system of two equations with two variables:

$$\begin{aligned} 1288 &= (p-1)(q-1) \\ 1363 &= pq \end{aligned}$$

We can do this by expressing p from the first equation $p = pq - 1288 - q = 1363 - 1288 - q = 76 - q$. Plugging this into to the second equation we get the quadratic equation $q^2 - 76 + 1363 = 0$. The two possible solutions $q_1 = 29$ and $q_2 = 47$ correspond to the fact that p and q are interchangeable and so we obtain the unique factorization $1363 = 29 \cdot 47$.

5.9. We first show that e has to be odd. That is because at least one of p, q is odd, therefore $\phi(n) = (p-1)(q-1)$ is even and because e is coprime to $\phi(n)$, it must be odd. Then we notice that $n - m \equiv -m \pmod{n}$. Now because e is odd we have $(-m)^e \equiv -m^e \pmod{n}$. But m is a fixed point so $-m^e \equiv -m \pmod{n}$. That means we have

$$n - m \equiv -m \equiv (n-m)^e \pmod{n},$$

in other words $n - m$ is a fixed point.

5.10. Let Eve choose $c' = r^e c \mod n$ for some r such that $c \neq c'$ and assume she is also provided with the decryption $m' = (r^e c)^d \mod n$. All she now needs to do is compute $m'r^{-1} \mod n$. Indeed we can see that

$$m'r^{-1} \equiv (r^e c)^d r^{-1} \equiv rc^d r^{-1} \equiv c^d \equiv m \pmod{n}.$$

5.11. If some primes are more likely to be generated than others, then there is a higher probability that two of the generated moduli have a common factor and this factor is the greatest common divisor of these moduli. So for every tuple of the generated moduli we can compute the greatest common divisor. A generated modulus is then secure if it has no non trivial common divisor with every other modulus. We don't need to evaluate all pairs, we only need one nontrivial divisor for every modulus to factorize it as every modulus has only two prime divisors.

The moduli 13490941 and 48456493 have no nontrivial common divisors with any other modulus and thus are the only secure of the given set. For every other modulus we have
$\gcd(8844679, 11316499) = 3169$ which gives us a divisor of 88446979 and 11316499.
$\gcd(18761893, 21799573) = 4219$ which gives us a divisor of 18761893 and 21799573.
$\gcd(22862761, 18761893) = 4219$. which gives us a divisor of 22862761.
$\gcd(43831027, 58354333) = 7057$ which gives us a divisor of 43831027 and 58354333.

5.12. First we express $n = 2^3 \cdot 677 + 1$, so $s = 3$ and $d = 677$. Now we determine whether the conditions $C(x_1)$, $C(x_2)$ and $C(x_3)$ hold. We begin by computing the first part of the condition $x^d \mod n$ for all the integers:

$$58^{677} \mod 5417 = 368, \quad 864^{677} \mod 5417 = 4438, \quad 3312^{677} \mod 5417 = 1.$$

Already we can see that $C(x_3)$ does not hold. The possible r for the second part of the condition $x^{2^r d} \mod n$ are $r_1 = 1$ and $r_2 = 2$ so we compute

$$58^{2^1 \cdot 677} \mod 5417 = 5416, \quad 864^{2^1 \cdot 677} \mod 5417 = 5049,$$

$$58^{2^2\cdot 677} \mod 5417 = 1, \quad 864^{2^2\cdot 677} \mod 5417 = 5416.$$

But $5416 \equiv -1 \pmod{5417}$ so neither $C(x_1)$ nor $C(x_2)$ holds. Neither of the three conditions hold, that means 5417 is a prime with probability of error $\frac{1}{8}$.

5.13.

(a) Because the two encryption exponents are coprime the Bezout's identity for e_A and e_B has the form

$$e_A x + e_B y = 1$$

for some integers x and y. If we know x and y we can compute m using to the following identity:

$$m \equiv m^{e_A x + e_B y} \equiv m^{e_A x} \cdot m^{e_B y} \equiv c_A^x \cdot c_B^y \pmod{n}.$$

We can find the values of x and y using the extended Euclidean algorithm. Because the encryption exponents are coprimes, x is the multiplicative inverse of $e_A \pmod{e_B}$, that is x_1. From the Bezout's identity itself we have $y = (1 - e_A x)/e_B$, but that is $-x_2$. So we get

$$m = c_A^{x_1} \cdot c_B^{-x_2} \mod n.$$

(b) First we compute the coefficients $x_1 = e_A^{-1} \pmod{e_B} = 2692$ and $x_2 = (x_1 e_A - 1)/(e_B) = 15$. With this we use the above formula to obtain m:

$$m = c_A^{x_1} \cdot c_B^{-x_2} \mod n = 7717^{2692} \cdot 12677^{-15} \mod 18721 = 18022.$$

5.14. Consider the following attack:

We compute $f = \gcd(e_E d_E - 1, e_B)$ and $m = \frac{e_E d_E - 1}{f}$. Since $\gcd(e_B, \phi(n)) = 1$ we have $gcd(f, \phi(n)) = 1$ and so m is a multiple of $\phi(n)$. The Bezout's identity for m and e_B has the form

$$mx + e_B y = 1,$$

for some integers x, y. Using this identity we have $e_B y \equiv 1 - mx \equiv 1 \pmod{\phi(n)}$ and since $e_B d_B \equiv 1 \pmod{\phi(n)}$ we can see that $(y - d_B)e_B \equiv 0 \pmod{\phi(n)}$. But because e_B is coprime to $\phi(n)$ we have $y \equiv d_B \pmod{\phi(n)}$. This means we can use y instead of Bob's decryption exponent. So to decrypt our message we compute

$$f = \gcd(c_E d_E - 1, e_B) = 1$$

$$m = (c_E d_E - 1) = 5829672.$$

Now we can find y using the extended Euclidean algorithm. We obtain $y = 2807399$ and decipher the captured message in subwords of 5:

$$\begin{aligned} 16278^{2807399} &\mod 99443 = 73327 \\ 49020^{2807399} &\mod 99443 = 67986 \\ 43554^{2807399} &\mod 99443 = 69328 \\ 279^{2807399} &\mod 99443 = 97985 \end{aligned}$$

The sent message is 73327679866932897985 which using the ASCII table translates to *I LOVE YOU.*

Chapter 6

Public-Key Cryptography, II.

6.1 Introduction

In this chapter we continue with public-key cryptosystems. This time we focus on systems whose security depends on the fact that the computation of square roots and discrete logarithms is in general infeasible in certain groups.

6.1.1 Rabin Cryptosystem

The first such cryptosystem is the Rabin cryptosystem. Its secret key are the Blum primes p, q and the public key is the modulus $n = pq$.

Encryption of a plaintext $w < n$ is done by computing $c = w^2 \mod n$.

Decryption of a ciphertext c is done by finding the square roots of c modulo n. There are in total four square roots of c so the decryption process is not deterministic. This does not pose a problem when decrypting a meaningful text, but in the case of random strings it is impossible to determine uniquely the right square root.

To calculate square roots modulo n we use its factors p and q and the following result.

Theorem 6.1.1 (Chinese remainder theorem)**.** *Let* $m_1, \dots, m_t$ *be integers,* $\gcd(m_i, m_j) = 1$ *if* $i \neq j$, *and* $a_1, \dots, a_t$ *be integers such that* $0 < a_i < m_i$, $1 \leq i \leq t$. *Then the system of congruences*

$$x \equiv a_i \pmod{m_i}, 1 \leq i \leq t$$

has the solution $x = \sum_{i=1}^{t} a_i M_i N_i$, *where* $M = \prod_{i=1}^{t} m_i$, $M_i = \frac{M}{m_i}$, $N_i = M_i^{-1} \mod m_i$ *and the solution is unique up to the congruence modulo* M.

6.1.2 ElGamal cryptosystem

The ElGamal cryptosystem relies on the infeasibility of computing a discrete logarithm $\log_q y$ in $\mathbb{Z}_p^*$. It has nondeterministic encryption due to using randomness.

Let p be a large prime and q, x two random integers such that $1 \leq q, x \leq p$ and q is a primitive element of $\mathbb{Z}_p^*$. Let $y = q^x \mod p$. Then p, q, y is the public key of the cryptosystem and x is its trapdoor information.

Encryption of a plaintext w consists of choosing a random r and computing $a = q^r \mod p$ and $b = y^r w \mod p$. The ciphertext is then $c = (a, b)$.

To decrypt a ciphertext $c = (a, b)$ we use x to calculate $w = \frac{b}{a^x} \mod p = ba^{-x} \mod p$.

6.1.3 Shanks' algorithm for discrete logarithm

Shanks' algorithm, called also Baby-step giant-step, is an algorithm for computing the discrete logarithm $\log_q y$ in $\mathbb{Z}_p^*$, which provides an improvement over the naive brute force method.

Let $m = \lceil \sqrt{p-1} \rceil$. The algorithm proceeds as follows

- Compute $q^{mj} \mod p$, for all $0 \leq j \leq m-1$.
- Create the list L_1 of m pairs $(j, q^{mj} \mod p)$, $0 \leq j \leq m-1$, sorted by the second item.
- Compute $yq^{-i} \mod p$, for all $0 \leq i \leq m-1$.
- Create the list L_2 of m pairs $(i, yq^{-i} \mod p)$, $0 \leq j \leq m-1$, sorted by the second item.
- Find two pairs, one $(j, z) \in L_1$ and $(i, z) \in L_2$ with the identical second element.

If such a search is successful, then $q^{mj+i} \equiv y \pmod p$, so the solution to the discrete logarithm is $\log_q y = mj + i$.

6.1.4 Perfect security of cryptosystems

When designing secure cryptosystems we not only require that it is impossible to obtain the corresponding plaintext from a ciphertext. We require that absolutely no information about the plaintext, such as some of its bits or parity, can be obtained.

One of the conditions for perfectly secure cryptosystems is the use of randomized encryptions. Otherwise possible attacker can guess the plaintext, compute its deterministic encryption and compare it to intercepted ciphertext.

To properly define perfect security we need the concept of *a negligible function*

Definition 6.1.2. *A function $f : \mathbb{N} \to \mathbb{R}$ is a negligible function if for any polynomial $p(n)$ and for almost all n it holds $f(n) \leq \frac{1}{p(n)}$.*

6.1.5 Blum-Goldwasser cryptosystem

Blum-Goldwasser cryptosystem is a public-key cryptosystem with randomized encryptions. Its security relies solely on the infeasibility of integer factorization. The private key are two Blum primes p and q, such that $p \equiv q \equiv 3 \pmod 4$, and the public key is the modulus $n = pq$.

Encryption of a plaintext $x \in \{0,1\}^m$

- Randomly choose $s_0 \in \{0, 1, \ldots, n\}$
- For $i = 1, 2, \ldots, m+1$ compute $s_i = s_{i-1}^2 \mod n$ and σ_i, the least significant bit of s_i.

The ciphertext is then (s_{m+1}, y), where $y = x \oplus \sigma_1 \sigma_2 \ldots \sigma_m$.

Decryption of the ciphertext (r, y)

- Compute $r_p = r^{((p+1)/4)^m} \mod p$ and $r_q = r^{((q+1)/4)^m} \mod q$
- Let $s_1 = q(q^{-1} \mod p)r_p + p(p^{-1} \mod q)r_q \mod n$.
- For $i = 1, \ldots, m$, compute σ_i and $s_{i+1} = s_i^2 \mod n$.

The plaintext is then $y \oplus \sigma_1 \sigma_2 \ldots \sigma_m$.

6.1.6 Hash functions

Hash functions are functions that map arbitrarily huge data to small fixed size output. Required properties for good cryptographic hash function f are

Definition 6.1.3 (Pre-image resistance)**.** *Given hash h it should be infeasible to find message m such that $h = f(m)$. In such a case we say that f has one-wayness property.*

Definition 6.1.4 (Second pre-image resistance)**.** *Given a message m_1 it should be infeasible to find another message m_2 such that $f(m_1) = f(m_2)$. In such a case we say f is weakly collision resistant.*

Definition 6.1.5 (Collision resistance)**.** *It should be infeasible to find two messages m_1 and m_2 such that $f(m_1) = f(m_2)$. In such a case we say that f is strongly collision resistant.*

6.2 Exercises

6.1. Let $c = 56$ and $n = 143$. Using the Chinese remainder theorem, determine in detail all square roots of c modulo n.

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

6.3. Consider the ElGamal cryptosystems with a public key (p, q, y) and a private key x.

(a) Encrypt the message $w = 15131$ using parameters $p = 199999$, $q = 23793$, $x = 894$ and $r = 723$.

(b) Decrypt the ciphertext $c = (299, 457)$ using parameters $p = 503$, $q = 2$, $x = 42$.

6.4. Consider the ElGamal cryptosystem with a public key (p, q, y) and a private key x.

(a) Let $c = (a, b)$ be a ciphertext. Suppose that Eve can obtain the decryption of any chosen cryptotext $c' \neq c$. Show that this enables her to decrypt c.

(b) Let $c_1 = (a_1, b_1)$, $c_2 = (a_2, b_2)$ be the two ciphertexts of the messages m_1 and m_2, $m_1 \neq m_2$, respectively, using the same public key. Encrypt some other message m'.

6.5. Consider the congruence

$$5^x \equiv 112 \pmod{131}.$$

Calculate x using Shanks' algorithm. Show all steps of the calculation.

6.6. Consider the subset of all negligible functions defined as follows:

$$G = \{\rho \mid \rho \text{ is a negligile function with } \operatorname{Im}(\rho) \subseteq \mathbb{N}\}.$$

Which of the following is $(G, \circ)$, where $\circ$ is the operation of function composition, if any:

- semigroup,
- monoid,
- group,
- Abelian group.

How would the previous answer change if the previous definition of G is modified as follows:

(a) $G = \{\rho \mid \rho \text{ is a negligile function with } \operatorname{Im}(\rho) \subseteq \mathbb{N}, \rho \text{ is a strictly increasing function}\}$,

(b) $G = \{\rho \mid \rho \text{ is a negligile function with } \operatorname{Im}(\rho) \subseteq \mathbb{N}, \rho \text{ is a strictly decreasing function}\}$.

6.7. Consider the Blum-Goldwasser cryptosystem with parameters $p = 43$ and $q = 59$.

(a) Encode the message $x = 0110$ with $s_0 = 1337$.

(b) Decode the message $(2218, 1001)$.

6.8. Suppose $h : \{0, 1\}^{2m} \to \{0, 1\}^m$ is a strongly collision-free hash function. Let $h' : \{0, 1\}^{4m} \to \{0, 1\}^m$ be defined as

$$h'(x) = h(h(x_1) \| h(x_2)),$$

where x_1 is the first half of x and x_2 is t he second half of x. Prove that h' is strongly collision-free hash function.

* **6.9.** Consider a large prime p and a primitive root a modulo p. Suppose that an encryption scheme encodes an integer $x \in \mathbb{Z}_p^*$ as follows

$$c = a^x \mod p.$$

Show that given c, an enemy can find (in polynomial time) the value of the least significant bit of x.

6.10. Consider the uniform distribution of birthdays in a 365-day year. What is the probability that two people in a group have a birthday on the same day if the group consists of

(a) 2 people,

(b) 23 people,

(c) 97 people.

* **6.11.** Consider the following modification of the ElGamal cryptosystem. Let G be a group with q elements, where q is a large prime, g is its generator and $h = g^x$ for some $x < q$. Suppose that multiplication and exponentiation in G can be done efficiently while computing discrete logarithms is hard. Let message $m \in \{0, 1, \dots, b\}$, where b is small, be encrypted using a public key (G, g, h) as $c = (g^r, g^m h^r)$, where r is chosen uniformly and randomly from $\{0, 1, \dots, q-1\}$.

(a) Describe how to recover the message m.

(b) Let m_1, m_2 and c_1, c_2 be two messages and their corresponding cryptotexts, respectively. Show that there is an algorithm which from a public key (G, g, h) and the cryptotexts c_1, c_2 determines a cryptotext c encrypting the message $m_1 + m_2$. Cryptotext c should be randomly distributed as if it was encrypted with a fresh randomly chosen r. The algorithm is expected to have no information about neither m_1 nor m_2.

(c) Suppose that $n > 2$ people would like to compute their average salary but they would be very displeased if any information about their individual salaries was revealed. Let x_i be a salary of the person i, where $i \in \{1, 2, \dots, n\}$. Suppose that the sum of all the salaries is at most b. Show how to compute the average $a = \frac{x_1+x_2+\cdots+x_n}{n}$ without revealing any other information about any x_i. Observe that a might not be an integer.

* **6.12.** Which of the following functions $f : \mathbb{N} \to \mathbb{N}$ are negligible? Prove your answer.

(a) $2^{-\sqrt{\log n}}$

(b) $n^{-\log \log n}$

6.3 Solutions

6.1. Factors of $n = 143$ are $m_1 = 11$ and $m_2 = 13$. Therefore we can express c as the tuple

$$(c \mod m_1, c \mod m_2) = (1, 4).$$

Since for all square roots s of c modulo n it must hold $s^2 \equiv 1 \pmod{11}$ and $s^2 \equiv 4 \pmod{13}$. That gives us four conditions for the square root s:

$$s \equiv \pm 1 \pmod{11}, \quad s \equiv \pm 2 \pmod{13}$$

Using the Chinese remainder theorem we can now compute all four solutions for s. The solutions have the form

$$s = a_1M_1N_1 + a_2M_2N_2 \mod n,$$

where

$$M_1 = \frac{n}{m_1} = 13, \quad M_2 = \frac{n}{m_2} = 11,$$

$$N_1 = M_1^{-1} \mod m_1 = 6, \quad N_2 = M_2^{-1} \mod m_2 = 6,$$

$$a_1 = s \mod 11, \quad a_2 = s \mod 13.$$

The last two equations give us $a_1 \in \{1, 10\}$ and $a_2 \in \{2, 11\}$ and we can see why the number of square roots is four. All the possible combinations of a_1 and a_2 and the resulting square root s can now be seen in the table bellow

a_1	a_2	s
1	2	67
10	2	54
1	11	89
10	11	76

So the square roots of 56 modulo 143 are 67, 54, 89 and 76.

6.2. Let $n = pq$ be the public modulus of the Rabin cryptosystem, where p and q are primes. We will show how we can find the factorization of n using the chosen-ciphertext attack. This would allow us to decrypt any encrypted message.

Let us chose a random x and ask for the decryption of $c = x^2 \mod n$. We receive one of the four square roots y of c. With probability $\frac{1}{2}$ it holds $y \neq \pm x$. In such case it holds

$$0 \neq (x - y)(x + y) = cn = cpq$$

because $x^2 - y^2 \equiv 0 \pmod{n}$. So if we now compute $\gcd(x + y, n)$ and $\gcd(x - y, n)$ we can obtain p or q. The probability of success can be amplified to $(1 - \frac{1}{2}^k)$ by asking for k plaintexts.

6.3.

(a) Following the protocol of ElGamal we first compute

$$y = q^x \mod p = 23793^{723} \mod 199999 = 137565.$$

Then we compute the two components of the ciphertext, namely a and b

$$a = q^r \mod p = 23793^{723} \mod 199999 = 89804$$

$$b = y^r w \mod p = 137565^{723} \cdot 15131 \mod 199999 = 7512.$$

The encrypted message is then $c = (a, b) = (89804, 7512)$.

(b) Let $(299, 457) = (a, b)$, the decryption is done by computing $w = b(a^x)^{-1} \mod p$ so

$$w = 457 \cdot (299^{42})^{-1} \mod 503 = 457 \cdot 393 \mod 503 = 30.$$

The plaintext is therefore $w = 30$.

6.4.

(a) We know that $c = (a, b) = (q^r, y^r m)$, where r is some random number and m is the given original message. Consider the cryptotext $c' = (a, 2b) = (q^r, 2y^r m)$. Clearly $c \neq c'$. The decryption of c' is then

$$2ba^{-x} = 2y^r mq^{-xr} = 2q^{xr} mq^{-xr} = 2m.$$

Now because 2 is coprime to p it is invertible modulo p. That means we can just divide the obtained plaintext by 2 and obtain the original message m.

(b) Let $c_1 = (a_1, b_1) = (q^{r_1}, y^{r_1} m_1)$ and $c_2 = (a_2, b_2) = (q^{r_2}, y^{r_2} m_2)$, for some random r_1 and r_2. Consider the cryptotext $c' = (a_1 a_2, nb_1 b_2) = (q^{r_1} q^{r_2}, ny^{r_1} y^{r_2} m_1 m_2)$, $n \in \mathbb{Z}_p^*$. Decrypting c' we get

$$nb_1 b_2 (a_1 a_2)^{-x} = ny^{r_1} y^{r_2} m_1 m_2 (q^{r_1} q^{r_2})^{-x} = nq^{x(r_1+r_2)} m_1 m_2 q^{-x(r_1+r_2)} = nm_1 m_2.$$

So c' is the correct encryption of the message $nm_1 m_2$ for any $n \in \mathbb{Z}_p^*$.

6.5. We use the Shanks' algorithm to calculate the discrete logarithm $\log_q y$ in $\mathbb{Z}_p^*$. In our case we have $q = 5$, $p = 131$ and $y = 112$.

First we determine the parameter $m = \lceil \sqrt{p-1} \rceil = \lceil \sqrt{130} \rceil = 12$. Now we create the list L_1 of m tuples $(j, q^{mj} \mod p)$, for $0 \leq j \leq m-1$. The list L_1 is shown in the following table.

j	0	1	2	3	4	5	6	7	8	9	10	11
$q^{mj} \mod p$	1	117	65	7	33	62	49	100	41	81	45	25

Next we create the list L_2 of m tuples $(i, yq^{-i} \mod p)$, for $0 \leq i \leq m-1$. The list L_2 is shown in the table below.

i	0	1	2	3	4	5	6	7	8	9	10	11
$yq^{-i} \mod p$	112	101	125	25	5	1	105	21	109	48	62	91

We can now see that there are three pairs of tuples with equal second item

$$(5, 62) \quad (10, 62),$$

$$(11, 25) \quad (3, 25),$$

$$(0, 1) \quad (5, 1).$$

This gives us three solutions to the discrete logarithm in the form $\log_q y = mj + i$. The first pair gives us

$$\log_5 112 = (12 \cdot 5 + 10) \mod 130 = 70.$$

The second

$$\log_5 112 = (12 \cdot 11 + 3) \mod 130 = 5.$$

And the third pair gives the same solution as the second one

$$\log_5 112 = (12 \cdot 0 + 5) \mod 130 = 5.$$

So using Shanks' algorithm we have found two solutions to the original congruence, namely $x_1 = 70$ and $x_2 = 5$. Indeed we can check that

$$5^{70} \equiv 112 \pmod{131}, \quad 5^5 \equiv 112 \pmod{131}.$$

6.6. Because for every $\rho, \rho' \in G$ it holds $\operatorname{Im}\rho$, $\operatorname{Im}\rho \subseteq \mathbb{N}$ it also holds $\operatorname{Im}(\rho \circ \rho') \subseteq \mathbb{N}$. At the same time there is $n_{\rho\rho'}$ such that for all $n > n_{\rho\rho'}$ it holds $\rho(\rho'(n)) \leq \rho'(n)$ so the composition is still negligible. That means the composition operation $\circ$ is well defined on G.

Now we look at the associative property of the composition. Let $f, g, h \in G$, then we have for every $n \in \mathbb{N}$

$$((f \circ g) \circ h)(n) = (f \circ g)(h(n)) = f(g(h(n)))$$
$$(f \circ (g \circ h))(n) = f((g \circ h)(n)) = f(g(h(n)))$$

We can see that the composition is indeed associative and therefore $(G, \circ)$ is a semigroup. The identity element for the composition $\circ$ is the identity function. But the identity function is clearly not negligible as $n \leq \frac{1}{n}$ holds for only one $n \in \mathbb{N}$. So G is neither monoid nor group nor Abelian group.

Now let us look at the modified G. We will show by contradiction that both sets are empty and so they are only semigroups.

(a) Let $\rho \in G$. ρ is strictly increasing and so $\rho(n) \geq n$ for all $n \in \mathbb{N}$. That means ρ is greater or equal to the identity function and we already know the identity is not negligible, so $\rho \notin G$, a contradiction.

(b) Let $\rho \in G$ and let $x, y \in \mathbb{N}$ such that $\rho(x) = y$. Then from the strictly decreasing property we get $\rho(x + y) \leq 0$. But that means $\rho(x + y + 1) < 0$ which is a contradiction as $\operatorname{Im}\rho \subseteq \mathbb{N}$.

6.7.

(a) The public key is $n = pq = 43 \cdot 59 = 2537$. We need to calculate s_i for $i \in \{1, 2, \ldots, 5\}$ using the formula $s_i = s_{i-1}^2 \mod n$.

$$\begin{aligned} s_1 &= 1337^2 \mod 2537 = 1521 \\ s_2 &= 1521^2 \mod 2537 = 2234 \\ s_3 &= 2234^2 \mod 2537 = 477 \\ s_4 &= 477^2 \mod 2537 = 1736 \\ s_5 &= 1736^2 \mod 2537 = 2277. \end{aligned}$$

Now we look at σ_i, the least significant bit of s_i, $i \in \{1, 2, 3, 4\}$. We have $\sigma_1\sigma_2\sigma_3\sigma_4 = 1010$. So the ciphertext of x is

$$c = (s_5, x \oplus \sigma_1\sigma_2\sigma_3\sigma_4) = (2277, 1100).$$

(b) Let $(r, y) = (2218, 1001)$. We have $m = 4$ and we compute

$$\begin{aligned} r_p &= r^{((p+1)/4)^m} \mod p = 2218^{(11^4)} \mod 43 = 13 \\ r_q &= r^{((q+1)/4)^m} \mod q = 2218^{(15^4)} \mod 59 = 46. \end{aligned}$$

Now we obtain s_1 by computing

$$s_1 = q(q^{-1} \mod p)r_p + p(p^{-1} \mod q)r_q \mod n = 59 \cdot 35 \cdot 43 + 43 \cdot 11 \cdot 21 \mod 2537 = 400.$$

We continue by calculating $s_{i+1} = s_i^2$ for $i \in \{1, 2, 3\}$.

$$\begin{aligned} s_2 &= 400^2 \mod 2537 = 169 \\ s_3 &= 169^2 \mod 2537 = 654 \\ s_4 &= 654^2 \mod 2537 = 1500 \end{aligned}$$

Finally, we look at the least significant bits of s_i. They are $\sigma_1\sigma_2\sigma_3\sigma_4 = 0100$. This is all we need to obtain the plaintext as

$$m = y \oplus \sigma_1\sigma_2\sigma_3\sigma_4 = 1001 \oplus 0100 = 1101.$$

6.8. We prove the strongly collision-free property by a contradiction. Suppose that h is a strongly collision-free hash function, but that h' is not.

Because h' is not strongly collision-free, it is feasible to find words w, w' such that $h'(w) = h'(w')$ and $w \neq w'$. Let w_1 and w_2 be the first and second half of w respectively. And let w_1' and w_2' be the first and second half of w', respectively. Then

$$h(h(w_1)\|h(w_2)) = h(h(w_1')\|h(w_2')).$$

But h is strongly collision-free so it has to hold

$$h(w_1)\|h(w_2) = h(w_1')\|h(w_2'),$$

otherwise the two words $h(w_1)\|h(w_2)$ and $h(w_1')\|h(w_2')$ would break the strongly collision-free property. That means $h(w_1) = h(w_1')$ and $h(w_2) = h(w_2')$. But using the strongly collision-free property of h again we get

$$w_1 = w_1'$$
$$w_2 = w_2',$$

which is a contradiction as it gives us $w = w'$. So h' is a strongly collision-free hash function.

6.9. We will use Euler's criterion to find the least significant bit of x. First we calculate $c^{\frac{p-1}{2}}$ mod p and distinguish two following cases

(a) $c^{\frac{p-1}{2}} \equiv 1 \pmod p$

As c is clearly coprime to p this means that c is a quadratic residue modulo p. So we can write $c = a^{2k} \mod p$ for some k, because a is a primitive root modulo p. This allows us to write

$$a^x \equiv a^{2k} \equiv c \pmod p.$$

So we have that x and $2k$ have the same parity and because $2k$ is even so is x. Therefore the least significant bit of x is 0.

(b) $c^{\frac{p-1}{2}} \not\equiv 1 \pmod p$

In this case c is not a quadratic residue modulo p and we can write $c = a^{2k+1}$ for some k. And again we have

$$a^x \equiv a^{2k+1} \equiv c \pmod p.$$

So in this case we get that x has to be even, therefore the least significant bit of x is 1.

So all we need to do to get the least significant bit of x is to compute $c^{\frac{p-1}{2}}$, which we can do in polynomial time.

6.10. The probability $p(n)$ that all $n \leq 365$ people in a room have their birthday in different days is

$$p(n) = \frac{365!}{365^n(365-n)!}.$$

The probability that there are two people in a group of size n that have a birthday on the same day is $1 - p(n)$. So we plug in the numbers for the three groups

(a) If $n = 2$, then the probability is $1 - p(2) = 1 - \frac{365!}{365^2(365-2)!} \approx 3 \cdot 10^{-3}$

(b) If $n = 23$,then the probability is $1 - p(23) = 1 - \frac{365!}{365^{23}(365-23)!} \approx 0.507$

(c) If $n = 97$, then the probability is $1 - p(97) = 1 - \frac{365!}{365^{97}(365-97)!} \approx 0.9999992$

We can see that while it is very improbable for two people to have birthday on the same day, for only 23 people we have over 50% chance that there are two people with birthday on the same day. For 97 people it is almost certain there will be such a pair.

6.11.

(a) Let $c = (y, z)$ be the ciphertext. First we compute $y^x = g^{rx} = h^r$ and $z(y^x)^{-1} = g^m h^r h^{-r} = g^m$. Because b is small we can just try every possible $m' \in \{0, 1, \dots, b\}$ and compare $q^m = q^{m'}$ to obtain m.

(b) Let $c_1 = (g^{r_1}, g^{m_1}h^{r_1})$ and $c_2 = (g^{r_2}, g^{m_2}h^{r_2})$. Let $c_3 = (y, z)$, where

$$y = g^{r_1}g^{r_2}g^{r_3} = g^{r_1+r_2+r_3}$$

$$z = g^{m_1}h^{r_1}g^{m_2}h^{r_2}h^{r_3} = g^{m_1+m_2}h^{r_1+r_2+r_3},$$

for a randomly and uniformly chosen r_3. It can be easily seen that c_3 is now the encryption of $m_1 + m_2$ using $r = r_1 + r_2 + r_3$. Because r_3 is chosen uniformly and independently of both r_1 and r_2, the sum $r_1 + r_2 + r_3$ is also distributed uniformly.

(c) Let the first person, $i = 1$, set up the modified ElGamal cryptosystem with public key (G, g, h) and decryption exponent x. He then sends the ciphertext $c_1 = (g_1^r, g^{x_1}h^{r_1})$ to the second person, $i = 2$. Now we define the protocol inductively: when a person i receives the ciphertext $c_{i-1} = (g_{i-1}^r, g^{m_{i-1}}h^{r_{i-1}})$ he sends, over a secure channel (using another encryption), the ciphertext $c_i = (g^{r_{i-1}+r'_i}, h^{r_{i-1}+r'_i}g^{m_{i-1}+x_i})$ to the person $i + 1 \mod n$, where r'_i is chosen randomly and uniformly.

Generalizing the reasoning of (b), it can be easily seen that the ciphertext c_{i-1} is the proper ciphertext for the sum of the first $i - 1$ salaries that only the fist person can decrypt. But because the communication is done over secure channels he gets only the first and last ciphertext. He gains no information from the first and gets only the whole sum from the last. He can now compute the average salary and announce it to everyone or the procedure can be repeated n-times with new person setting up the cryptosystem every time, making him the one who obtains the whole sum.

6.12.

(a) Consider the polynomial $p(n) = n^2$. Let's now try to solve the inequality $f(n) > \frac{1}{p(n)}$:

$$\frac{1}{2^{\sqrt{\log n}}} > \frac{1}{n^2}$$
$$n^2 > 2^{\sqrt{\log n}}$$
$$2\sqrt{\log n} > 1$$

It is easy to see that the inequality holds for any $n > 2$. This means it certainly cannot hold that $f(n) \leq \frac{1}{p(n)}$ for almost all n and therefore the function $2^{-\sqrt{\log n}}$ is not negligible.

(b) Consider any polynomial $p(m) = a_m n^m + \dots + a_0$, $a_m, \dots, a_0, m \in \mathbb{N}$. Now consider the function $r_p : \mathbb{N} \to \mathbb{N}$, $r_p(n) = (\sum_{i=0}^{m} |a_i|)\, n^m$. It is easy to see that $r_p(n) \geq p(n)$ for all $n \in \mathbb{N}$. Therefore if $f(n) \leq \frac{1}{r_p(n)}$ it holds $f(n) \leq \frac{1}{p(n)}$.

Let us now try to find the solutions to the inequality $n^{-\log\log n} \leq \frac{1}{r_p(n)}$:

$$\begin{aligned}
\frac{1}{n^{\log\log n}} &\leq \frac{1}{r_p(n)} \\
r_p(n) &\leq n^{\log\log n} \\
\left(\sum_{i=0}^{m} |a_i|\right) n^m &\leq n^{\log\log n}
\end{aligned}$$

$$\begin{aligned}
\log\left[\left(\sum_{i=0}^{m} |a_i|\right) n^m\right] &\leq \log(n^{\log\log n}) \\
\log\left(\sum_{i=0}^{m} |a_i|\right) + m\log n &\leq (\log\log n)\log n \\
m &\leq \log\log n - \frac{\log\left(\sum_{i=0}^{m} |a_i|\right)}{\log n}
\end{aligned}$$

We now take the limit of the right hand side

$$\lim_{n\to\infty} \log\log n - \frac{\log\left(\sum_{i=0}^{m} |a_i|\right)}{\log n} = \infty.$$

From the definition of the limit we know that for every $k \in \mathbb{N}$, there is an $n_k \in \mathbb{N}$ such that for all $n > n_k$ it holds that the right hand side is greater than k. And since this holds for every k it must also hold for m. Therefore the inequality $n^{-\log\log n} \leq \frac{1}{r_p(n)}$ must hold for all $n > n_m$ for some $n_m \in \mathbb{N}$. In short it holds for almost all n and thus even the inequality $f(n) \leq \frac{1}{p(n)}$ holds for almost all n. But this means the function $n^{-\log\log n}$ is negligible.

Chapter 7

Digital Signatures

7.1 Introduction

Digital signatures are electronic analogues to handwritten signatures. Signature is an integral part of the person's identity. It is intended to be unique for any individual and serves as a way to identify and authorize. When electronic information is considered, the concept of signature has to be adapted, because it cannot be something independent of the message signed. A digital signature is something, *e.g.* a number, which depends both on the secret known only by the signer and on the message to be signed. It should be verifiable without an access to signer's secrets. Only the hash of the original message is usually signed.

7.1.1 Signature schemes – basic ideas and goals

A *signature scheme* consists of two algorithms: a *signing algorithm* and *a verification algorithm.* A message m is signed by Alice using the signing algorithm that produces a signature $sig(m)$. If the public verification algorithm is applied to a signature, then it returns *true* if and only if $sig(m)$ is a signature created by Alice for the message m, *i.e.* if the signature is authentic.
These algorithms work with two keys: the secret key used for signing and the public key which serves for verification. The sets of potential messages, signatures and keys are finite.

Security requirements and objectives, which should be satisfied by any signature scheme, are the following ones.

- It should be infeasible to forge Alice's signature for any message. Only Alice should be able to produce her valid signatures – this is so-called *authenticity* requirement.
- Because the messages and their signatures have to be inseparably tied, any change either in the created signature or in the message to be signed should result in the rejection of the signature – this is so-called *data integrity* requirement.
- Another basic requirement is the impossibility of revocation of a valid signature in some later time – that is so-called *non-repudiation* requirement.
- Finally, from a technical point of view, signing and verification algorithms should be sufficiently fast.

7.1.2 Digital signature scheme – definition

A digital signature allows any signer S to sign any message m in such a way that no one can forge S's signature but anyone familiar with the system can verify that a signature claimed to be from S is indeed from S and has not been changed during transmission.
A *digital signature scheme* (M, S, K_s, K_v) is given by (1) by a set M of messages that may need to be signed, a set S of potential signatures, a set K_s of so-called private keys (used at signings) and

a set K_v of so-called public/verification keys used for verification of signatures.
(2) for each key $k \in K_s$ is given a single and easy to perform signing mapping $sig_k : \{0,1\}^* \times M \to S$, and for each key $k_v \in K_v$, there exists a single and easy to compute verification mapping $ver_{k_v} : M \times S \to \{true, false\}$ such that the following two conditions are satisfied:

Correctness: For any message $m \in M$ and any public key $k \in K_v$, and $s \in S$ it holds that $ver_k(m,s) = true$ 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 $m \in M$ and $k_v \in K_v$, it is computationally infeasible, without the knowledge of the private key corresponding to k, to find a signature $s \in S$ such that $ver_{k_v}(m,s) = true$.

7.1.3 Attacks

The following describes the basic attack models and levels of breaking a digital signature scheme.

Key-only attack: The attacker is only given the public verification key.
Known signatures attack: The attacker is given valid signatures for several messages not chosen by her.
Chosen signatures attack: The attacker is given valid signatures for several messages of her choice.
Adaptive chosen signatures attack: The attacker is given valid signatures for several messages chosen by the attacker where messages chosen may depend on previous signatures given for chosen messages.

Total break: The adversary manages to recover secret key.
Universal forgery: The adversary can derive an algorithm which allows him to forge signature of any message.
Selective forgery: The adversary can derive 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 a valid signature of some message m (but has no control for which m).

The strongest notion of security is security against existential forgery under an adaptive chosen signatures attack.

7.1.4 Examples

RSA signatures
Consider the RSA cryptosystem with encryption and decryption exponents e and d and modulus n. The signature of a message m is $s = sig(m) = m^d \mod n$. The signature s for the message m is valid if $s^e \mod n = m$.

ElGamal signature scheme
The public key for the ElGamal signature scheme is $K = (p, q, y)$, where p is a prime, q is a primitive element of $\mathbb{Z}_p^*$ and $y = q^x \mod p$, where $1 < x < p$ is the secret key.
To create the signature s of a message m we need to choose a random integer $r \in Z_{p-1}^*$ and calculate $s = sig(m,r) = (a,b)$, where $a = q^r \bmod p$ and $b = (m - ax)r^{-1} \bmod p-1$. The signature $s = (a,b)$ for the message m is valid if $y^a a^b \equiv q^m \mod p$.

DSA signature scheme
A variant of the ElGamal system later adopted as a standard is the Digital Signature Algorithm (DSA). The key for the 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 q^{th} root of 1 in $\mathbb{Z}_p$, ($r = h^{\frac{p-1}{q}} \mod p$, where h is a primitive element in $\mathbb{Z}_p$), x is a random integer such that $0 < x < q$ and $y = r^x \mod p$. The values p, q, y and r are made public, x is kept secret.

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

Lamport one-time signatures
A Lamport signature scheme is method for constructing a digital signature from a one-way function. Only one message can be signed with such a scheme, therefore the term "one-time".
Suppose that k-bit messages are to be signed. Let $f : Y \to Z$ be a one-way function. For randomly chosen $y_{i,j} \in Y$ calculate $z_{i,j} = f(y_{i,j})$, $i \in \{1,\dots,k\}$, $j \in \{0,1\}$.
The parameter i refers to the position of the bit in a message being signed and the parameter j refers to its value. The z's are made public together with f, y's are kept secret.
To sign a k-bit message $x = (x_1,\dots,x_k)$, the corresponding y's are made public: $sig(x_1,\dots,x_k) = (y_{1,x_1},\dots,y_{k,x_k})$
To verify a signature $(s_1,\dots,s_k)$ for the message x, one needs to verify that $f(s_i) = z_{i,x_i}$ for all $i \in \{1,\dots,k\}$.

Fiat-Shamir signature scheme
An example of an identification scheme that can be converted into a signature scheme is the Fiat-Shamir scheme. Choose primes p, q, compute $n = pq$ and choose: as a public key integers $v_1,\dots,v_k$ and compute, as a secret key, $s_1,\dots,s_k$, $s_i = \sqrt{v_i^{-1}} \mod n$. To sign a message w, Alice first chooses as a security parameter an integer t, random integers $1 \leq r_1,\dots,r_t < n$, and computes $x_i = r_i^2 \mod n$, for $1 \leq i \leq t$. Alice uses a publicly known hash function h to compute $H = h(wx_1x_2\dots x_t)$ and then uses the first kt bits of H, denoted as b_{ij}, $1 \leq i \leq t$, $1 \leq j \leq k$ as follows. Alice computes $y_i = r_i \prod_{j=1}^{k} s_j^{b_{ij}} \mod n$ and sends w, all b_{ij}, all y_i and h to Bob. Bob finally computes $z_1,\dots,z_k$, where $z_i = y_i^2 \prod_{j=1}^{k} v^{b_{ij}} \mod n = x_i$ and verifies that the first kt bits of $h(wx_1x_2\dots x_t)$ are the b_{ij} values that Alice has sent to him.

Blind signatures
A blind signature is a form of signature in which the content of a message is blinded before it is signed. Applications are in electronic voting and digital cash systems.
RSA blinding signature scheme with keys (n,e,d) works as follows. In order to make Bob to produce, blindly, $sig_B(m)$ of the message m, Alice chooses a random r and asks Bob to sign $m' = mr^e \mod n$ and to send her back his signature $sig_B(m')$. Alice then gets easily $sig_B(m) = r^{-1}sig_B(m')$.

Subliminal channels
A subliminal channel is a covert channel that allows to communicate secretly in a normally looking communication. Several signature schemes contain subliminal channels, for example the Ong-Schnorr-Shamir channel.

7.2 Exercises

7.1. Alice and Bob are using the RSA signature scheme. Alice's public key is $(n,e) = (899,17)$. Malicious Eve captured two signed messages which Alice sent $(m_1, \text{sig}(m_1)) = (13,644)$ and $(m_2, \text{sig}(m_2)) = (15,213)$. Is Eve able to forge signatures of messages $m_a = 195$ and $m_b = 627$ without using brute-force? Try to calculate this signatures and verify.

7.2. Let m be a message which the adversary Eve intends to sign using the RSA signature scheme with a public key (n,e) and a private key d. Suppose that Eve can obtain a signature of any message $m' \neq m$. Show that this enables her to sign m.

7.3. Consider the following version of the Rabin signature scheme:
Let $n = pq$ where p and q are primes. n is made public, p and q are kept secret. To sign a message m, solve the equation $s^2 \equiv m \pmod{n}$ (assume that $m \in QR(n)$). The value s is the signature for the message m. To verify a given message-signature pair (m, s), check whether $s^2 \equiv m \pmod{n}$.

(a) Show that the proposed scheme is vulnerable to an existential forgery with a key-only attack.

(b) Show that the scheme can be totally broken with chosen signatures attack.

7.4. Show that you can totally break the DSA signature scheme if you are given a signature $(a, 0)$ for a message w.

7.5. Consider the following modification of the ElGamal signature scheme in which $x \in Z^*_{p-1}$ and b is computed as

$$b = (w - ra)x^{-1} \pmod{p-1}.$$

Describe how verification of a signature (a, b) on a message x would proceed.

7.6. A lazy signer who uses the ElGamal signature algorithm has precomputed one pair (k, a) satisfying $a = q^k \mod p$ which he always uses for generating a signature. Given two message-signature pairs, recover his secret key.

* **7.7.** How would the security of the ElGamal signature scheme be affected if one would accept signatures (a, b) with a larger than p?

7.8. Consider the Fiat-Shamir signature scheme. What happens if an adversary is able to find the random integers $r_1, \ldots, r_t$?

7.9. Let us consider Chaum's blind signatures RSA scheme with $n = pq, p = 101, q = 97, e = 59$. Only Bob knows d and uses it to sign documents. Alice wants him to sign message $m = 4242$ without him knowing m. Perform steps of the protocol in detail. Random k is $k = 142$.

7.10. Let f be a one-way permutation. Let us define a one-way signature scheme in the following way. The public key is an l-tuple $(y_1, y_2, \ldots, y_l)$. The secret key is an l-tuple $(x_1, x_2, \ldots, x_l)$ such that $f(x_i) = y_i$ for each $i \in \{1, 2, \ldots, l\}$. The signature of a message $m_1 m_2 \cdots m_l \in \{0, 1\}^l$ consists of all x_i such that $m_i = 1$.

(a) Show that this one-time signature scheme is not secure.

(b) Describe all the message pairs (m_1, m_2) such that $m_1 \neq m_2$ and the adversary can produce a valid signature for m_2 using a valid signature for m_1.

7.11. Consider the Lamport one-time signature scheme. A signer has signed m $(2 \leq m \leq 2^k - 1)$ different k-bit messages (instead of only one message he was allowed to sign). How many new messages an adversary would be able to forge?

7.12. Consider the following one-time signature scheme used for signing of N-bit messages. Let H be a cryptographically secure hash function. Let $H^k(x)$ denote k successive applications of the hash function H to x – so-called *hash chain, e.g.* $H^4(x) = H(H(H(H(x))))$.

- (Initial phase) Alice chooses two random numbers x_1 and x_2 and computes $y_1 = H^M(x_1)$ and $y_2 = H^M(x_2)$ where $M = 2^N$. Alice publishes y_1 and y_2.
- (Signing) Alice computes $s_1 = H^n(x_1)$ and $s_2 = H^{M-n-1}(x_2)$, where $0 \leq n \leq 2^N - 1$ is the value of an N-bit message to be signed.

- (Verification) To verify a signature, Bob checks validity of the following equations: $y_1 = H^{M-n}(s_1)$ and $y_2 = H^{n+1}(s_2)$.

(a) Demonstrate usage of the proposed scheme on signing of 2-bit message 11.

(b) Explain why it is insufficient to compute only a value of s_1.

* **7.13.** Propose a subliminal channel in the DSA signature scheme. Assume that the receiver of a subliminal message shares the secret key x with the sender.

* **7.14.** Consider the following signature scheme for a group of two users. Each of the users i has his own RSA signature scheme with public key (n_i, e_i) and secret key d_i, $i \in \{1, 2\}$. Their respective trapdoor one way permutation is $f_i(x) = x^{e_i} \pmod{n_i}$.

For the smallest b such that $2^b > \max(n_1, n_2)$ we define new trapdoor one way permutations g_i in the following way. For b-bit input x define integers q_i and r_i such that $x = q_i n_i + r_i$, $0 \le r_i < n_i$ (we know r_i, q_i are unique). Then

$$g_i(x) = \begin{cases} q_i n_i + f_i(r_i) & \text{if } (q_i + 1)n_i \le 2^b \\ x & \text{otherwise} \end{cases}$$

Let h be a public collision-resistant hash function that maps messages to b-bit strings.
Now the user i signs any message m as follows:

1. Computes the key $k = h(m)$
2. Chooses random b-bit x_j, $j \neq i$.
3. Computes $y_j = g_j(x_j)$ using n_j, e_j.
4. Finds y_i such that $y_i \oplus y_j = k$.
5. Using d_i finds $x_i = g^{-1}(y_i)$.
6. Outputs the signature $((e_1, n_1), (e_2, n_2), x_1, x_2)$.

(a) Find the verification of the signature.

(b) Given a message and its signature, can you discover which user signed the message?

7.3 Solutions

7.1. We know that for the RSA signature scheme it holds: if s_1 and s_2 are signatures of messages m_1 and m_2, we can easily compute the signature of the message $m = m_1 m_2 \bmod n$ as $s = s_1 s_2 \bmod n$. This is the first case and the signature of $m_a = 195 = 13 \times 15$ is $s_a = \text{sig}(m_1) \times \text{sig}(m_2) \bmod n = 524$. Verification: $(s_a)^e \bmod n = 524^{17} \bmod 899 = 195$ which is equal to m_a.
The message m_b is equal to m_a^{-1} modulo n (which is easy to verify: $m_a m_b \equiv 1 \pmod{n}$.) We know that for the RSA signature scheme the following holds: if s is a signature of a message m then $s^{-1} \bmod n$ is the signature of the message $m^{-1} \bmod n$. Using the Extended Euclidean Algorithm we get $1 = -362 \times 524 + 211 \times 899$. Hence $s_b = -362 \bmod n = 537$. Verification: $(s_b)^e \bmod n = 537^{17} \bmod 899 = 627 = m_b$.

7.2. Eve first chooses r that has inverse modulo n and later she asks for the signature for the message $m' = m \cdot r^e$ which is

$$s' \equiv (m')^d \equiv m^d \cdot r^{ed} \equiv m^d \cdot r \pmod{n}.$$

Now she can multiply s' with r^{-1} to obtain the valid signature for the message m.

7.3.

(a) Choose a signature s and square it to produce a corresponding message m. This yields a valid message-signature pair.

(b) If we can find two distinct square roots of a message, we can consequently factor the modulus n. We can again choose a value s and compute $m = s^2$. The next step is submitting m to the black box. There is a one in two chance that it will produce the same signature s. In such case, repeat this process. If not, we have both square roots of m and can recover the factors of n.

7.4. Since the signature is $(a, 0)$, we have $0 = b \equiv k^{-1}(w + xa) \pmod{q}$. We know $k^{-1} \neq 0$ therefore $w + xa \equiv 0 \pmod{q}$ and since q is prime a^{-1} exists. Together $x \equiv -wa^{-1} \pmod{q}$.
Values w and a are known, so one can easily calculate the secret key x and hence totally break the DSA scheme.

7.5. A signature (a, b) is valid if

$$a^a y^b \equiv q^w \pmod{p}.$$

Recall that $y = q^x \mod p$ and $a = q^r \mod p$. Now we prove correctness of the proposed verification method:

$$a^a y^b \equiv y^{ar} y^b \equiv y^{ar+w-ar+k(p-1)} \equiv q^w \pmod{p}.$$

7.6. If the signer signs two messages w_1 and w_2 he obtains signatures (a, b_1) and (a, b_2) where $b_1 = k^{-1}(w_1 - xa) \mod p-1$ and $b_2 = k^{-1}(w_2 - xa) \mod p-1$. We have $xa = w_1 - kb_1 = w_2 - kb_2$ and $k = (w_1 - w_2)(b_1 - b_2)^{-1} \mod p-1$. Now we can calculate the secret key as $x = (w_1 - kb_1)a^{-1} \mod p-1$. There are $d = \gcd(a, p-1)$ solutions for x. The forger can compute q^x for all the x's found until she finds y and therefore the proper x.

7.7. If one would accept signatures (a, b) with $a > p$, an adversary would be able to forge the valid signature for any message w_2 after having intercepted the valid signature (a_1, b_1) of w_1:
The adversary computes $c = w_2 w_1^{-1} \mod p-1$. It holds that

$$q^{w_2} \equiv q^{cw_1} \equiv y^{a_1 c} a_1^{b_1 c} \pmod{p}$$

Since we have $x \equiv y \pmod{p-1} \Rightarrow z^x \equiv z^y \pmod{p}$, the adversary can find such a_2, b_2 that the following equations hold:

$$b_2 \equiv b_1 c \pmod{p-1}$$

$$a_2 \equiv a_1 c \pmod{p-1}$$

$$a_2 \equiv a_1 \pmod{p}$$

To determine b_2 the adversary computes $b_2 = b_1 c \mod p-1$.
To determine a_2 the adversary uses the Chinese remainder theorem. The result will be modulo $p(p-1)$, that is why $a_2 > p$. Now it is easy to see that

$$q^{w_2} \equiv y^{a_2} a^{b_2} \pmod{p}$$

and (a_2, b_2) is the valid signature of w_2.

7.8. If an adversary finds the random integers $r_1, \dots, r_t$ and intercepts the message w together with b_{ij} and y_i, she can construct the following system of congruences:

$$y_1 \equiv r_1 s_1^{b_{1,1}} s_2^{b_{1,2}} \dots s_k^{b_{1,k}} \pmod{n}$$
$$y_2 \equiv r_2 s_1^{b_{2,1}} s_2^{b_{2,2}} \dots s_k^{b_{2,k}} \pmod{n}$$
$$\dots$$
$$y_t \equiv r_t s_1^{b_{t,1}} s_2^{b_{t,2}} \dots s_k^{b_{t,k}} \pmod{n}$$

Now it depends on k, t and $b_{i,j}$ what the adversary could do. For example if $k \leq t$ and $b_{i,j} = 1$ for all $1 \leq i \leq t$, $1 \leq j \leq k$, then she is able to compute all s_j and thus the private key.
In other cases, she might be able to compute only some of the s_j, which would enable her to sign some messages (those with $b_{i,j} = 0$ for $1 \leq i \leq t$ and j such that s_j is unknown).
In other cases, this would not allow the adversary to find any part of the private key.

7.9. To perform computation, we need to find d. It holds $d = e^{-1} \bmod \phi(n)$, thus $d = 59^{-1} \bmod 9600 = 1139$.

- Alice computes $m' = mk^e \bmod n = 4242 \cdot 142^{59} \bmod 9797 = 808$ and sends it to Bob.
- Bob computes $s' = (m')^d \bmod n = 808^{1139} \bmod 9797 = 6565$ and sends it to Alice.
- Alice now computes signature s of message m, $s = k^{-1}s' \bmod n = 69 \cdot 6565 \bmod 9797 = 2323$.

Message can be verified as $m = s^e \bmod n$.

7.10. Only bits with the value 1 are signed, 0-valued bits are not included therefore one can forge signatures for any message which has 0 where the original message had 0 and which has 0 or 1 on the remaining positions. For example given the signed message 1^l an adversary would be able to sign any message of length l.

7.11. Let us assume that the signer has signed two messages that are bit inverses of each other. This way the signer made public his whole private key and the adversary can forge any k-bit message.

We analyze the case in which the signer has signed $2 \leq m \leq 2^k - 1$ different messages in such a way that he made public the least possible number of y's.
Signing the first message a half of y's is made public. A second message must differ in at least one position, say i_1, so the $y_{i_1 j}$ are known for both $j = 0, 1$.Three or four messages must differ in at least 2 positions, allowing the adversary to sign 2 new messages. Finally, m messages have to differ in at least l positions where $2^{l-1} + 1 \leq m \leq 2^l$, so $l = \lceil \log_2 m \rceil$.The adversary is therefore able to forge at least $2^{\lceil \log_2 m \rceil} - m$ new messages.

7.12.

(a) (Initial phase)

$$N = 2,\ M = 2^N = 4,$$
$$y_1 = H^M(x_1) = H^4(x_1),$$
$$y_2 = H^M(x_2) = H^4(x_2)$$

(Signing)

$$n = 3,$$
$$s_1 = H^n(x_1) = H^3(x_1),$$
$$s_2 = H^{M-n-1}(x_2) = H^0(x_2) = x_2$$

(Verification)

$$y_1 = H^{M-n}(s_1) = H(s_1) = H(H^3(x_1)) = H^4(x_1),$$
$$y_2 = H^{n+1}(s_2) = H^4(s_2) = H^4(x_2)$$

(b) Given a value of $s_1 = H^n(x_1)$, one could easily sign messages m where $2^N - 1 \geq m \geq n$.

7.13. Knowing the private key x, the only unknown value is k.

$$s = k^{-1}(h - xr) \mod q$$

$$sk = h - xr \mod q$$

$$k = s^{-1}(h - xr) \mod q$$

To embed a subliminal message, it suffices to set k to a specific value.

7.14.

(a) To verify the signature we first find $k = h(m)$, then we calculate both $y_i = g_i(x_i)$ using the (n_i, e_i) and verify that $y_1 \oplus y_2 = k$. Indeed, if both x_i were created by the signature procedure it must hold that $y_1 \oplus y_2 = k$.

(b) If x_1 is chosen randomly and $x_2 = a$ is computed by the signature procedure we get the same tuple (x_1, x_2) as if $x_2 = a$ was chosen randomly and x_1 was computed. Indeed, in both cases it must hold $x_1 = g_1^{-1}(g_2(x_2) \oplus k)$ and as both g_i are permutations we can obtain any x_i from some x_j. So for the same x_2 we get the same x_1 (and vice versa). So given the signature $((e_1, n_1), (e_2, n_2), x_1, x_2)$ we cannot find which x_i was chosen randomly and which computed as both cases might have happened and thus we cannot discover who signed the message.

Chapter 8

Elliptic Curve Cryptography

8.1 Introduction

Elliptic curve cryptography (ECC) is an approach to public-key cryptography based on the algebraic structure of elliptic curves over finite fields. There are several advantages over the classical public-key cryptography – shorter keys can be used which results in savings in hardware implementation and attacks that are based on factorization or finding discrete logarithms so far do not work for ECC.
Elliptic curves are also employed in Lenstra's algorithm for integer factorization.

8.1.1 Elliptic curves

Elliptic curve E is a plane curve defined by the equation

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

where a and b are real numbers such that the curve has no self-intersections or isolated points. It means the curve has no multiple roots,*i.e.* it is non-singular. The curve is non-singular if and only if $4a^3 - 27b^2 \neq 0$.

8.1.2 Group structure and addition law

On an elliptic curve, it is possible to define addition of points in such a way that the points of the elliptic curve together with the addition operation form an Abelian group. This requires the curve to be extended with the so-called "point at infinity", denoted with $\mathcal{O}$, which serves as the neutral element of the group.
Addition of points $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ is calculated as $P_3 = P_1 + P_2 = (x_3, y_3)$, where $x_3 = \lambda^2 - x_1 - x_2$ and $y_3 = \lambda(x_1 - x_3) - y_1$ and

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

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

Elliptic curves over a finite field and Hasse's theorem
The addition of points on an elliptic curve over a finite field is done the same way as described above, with division understood as the multiplication with an inverse element. The number of points on an elliptic curve over a finite field with q elements is limited by the Hasse's theorem:

If an elliptic curve $E \pmod{n}$ has N points, then $|N - (q + 1)| \leq 2\sqrt{q}$.

8.1.3 Elliptic curve cryptography

Elliptic curve discrete logarithm problem
Elliptic curve cryptosystems are based on the intractability of computing discrete logarithms. Discrete logarithm problem for elliptic curves is stated as follows. Let E be an elliptic curve and P, Q be points on the elliptic curve such that $Q = kP$ for some integer k, where

$$kP = \overbrace{P + P + \cdots + P}^{k-\text{times}}.$$

The task is to calculate k given P, Q and E.

Converting classical systems into elliptic curve counterparts
Cryptosystems or protocols based on discrete logarithm problem can be converted easily into cryptosystems or protocols based on elliptic curves. The conversion goes as follows:

- Assign a point on an elliptic curve to the plaintext message.
- Change modular multiplications into additions of points and exponentiations into multiplying a point on an elliptic curve by an integer.
- Assign a cryptotext message to the point of an elliptic curve that results from the modified cryptosystem.

8.1.4 Factorization

Factorization is a process, in which a composite number is decomposed into a product of its prime factors. For each number there is a unique decomposition. 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.

Pollard ρ-method
This method is based on the birthday paradox – when we keep choosing pseudorandom integers, there must be a pair a, b such that $a \equiv b \pmod{p}$ and $a \not\equiv b \pmod{n}$, where p is a prime factor of n, the number we want to factor. First, choose some pseudorandom function $f : \mathbb{Z}_n \to \mathbb{Z}_n$ and an integer $x_0 \in \mathbb{Z}_n$. Then keep computing $x_{i+1} = f(x_i)$ for $i = 0, 1, 2, \dots$ and $\gcd(|x_j - x_k|, n)$, for each $k < j$. If $\gcd(|x_j - x_k|, n) = d > 1$ then we have found a factor d of n.There are several modifications that differ in frequency of calculation of greatest common divisors.

Pollard $(p - 1)$-method
This method is based on Fermat's Little Theorem and discovers a prime factor p of an integer n whenever $p - 1$ has only small prime factors. To factor n, first fix an integer B, compute $M = \prod_{\text{primes } q \leq B} q^{\lfloor log_q B \rfloor}$. Choose an integer a and compute $d = \gcd(a^M - 1, n)$. If $1 < d < n$, we have a factor d of n.

Factorization with elliptic curves
To factorize an integer n we choose in many elliptic curve E over $\mathbb{Z}_n$ points $P \in E$ and compute either points iP for $i = 2, 3, 4, \dots$ or points $2jP$ for $j = 1, 2, 3, \dots$. When calculating these points we need to evaluate $\gcd(x_A - x_B, n)$ for various points A, B when computing λ. If one of these values is between 1 and n, we have a factor of n.

8.2 Exercises

8.1. Consider the elliptic curve $E : y^2 = x^3 + 568x + 1350 \pmod{1723}$ and the point $P = (524, 1413)$. Compute the point $144P$ with as few point operations as possible.

8.2. Let $E : y^2 = x^3 + 9x + 17$ be the elliptic curve over the field $\mathbb{F}_{23}$. What is the discrete logarithm k of $Q = (4, 5)$ to the base $P = (16, 5)$?

8.3. Consider the elliptic curve $E : y^2 = x^3 + 6x^2 + 14x + 16$ over $\mathbb{Z}_{29}$. Transform E to the form $y^2 = x^3 + ax + b$ where $a, b \in \mathbb{Z}_{29}$.

8.4. Decide whether the points of the following elliptic curves define a group over $\mathbb{Z}_p$ where p is a prime? If yes, find an isomorphism between points of the elliptic curve and the additive group of integers $(\mathbb{Z}_p, +)$.

(a) $E : y^2 = x^3 + 10x + 5 \pmod{17}$

(b) $E : y^2 = x^3 + 4x + 1 \pmod{7}$

8.5. Is there a (non-singular) elliptic curve E defined over $\mathbb{Z}_5$ such that

(a) E contains exactly 11 points (including the point at infinity $\mathcal{O}$);

(b) E contains exactly 10 points (including the point at infinity $\mathcal{O}$)?

If the answer is positive, find such a curve and list all of its points, If it is negative, prove it.

8.6. Let $E : y^2 = x^3 + ax + b$ with $a, b \in \mathbb{Q}$ be an elliptic curve. Show that there is another equation $Y^2 = X^3 + AX + B$ with $A, B \in \mathbb{Z}$ whose solutions are in bijection with the solutions to $y^2 = x^3 + ax + b$.

* **8.7.** We call a nonzero rational number n a congruent number if $\pm n$ is the area of a right-angled triangle with rational side lengths or, equivalently, n is a congruent number if the system of two equations:

$$a^2 + b^2 = c^2$$
$$\frac{1}{2}ab = n$$

has a solution with $a, b, c \in \mathbb{Q}$.

The relation between congruent numbers and elliptic curves is described with the following theorem:
Let n be a rational number. There is a bijection between $A = \{(a, b, c) \in \mathbb{Q}^3 \mid \frac{1}{2}ab = n, a^2 + b^2 = c^2\}$ and $B = \{(x, y) \in \mathbb{Q}^2 \mid y^2 = x^3 - n^2x \text{ with } y \neq 0\}$ given by the map $g : B \to A$ defined as

$$g(x, y) = \left(\frac{n^2 - x^2}{y}, -\frac{2xn}{y}, \frac{n^2 + x^2}{y}\right).$$

Using the previous theorem show that the numbers 5 and 6 are congruent numbers and give the corresponding values of a, b and c.

* **8.8.**

(a) How many points P such that $2P = \mathcal{O}$ can be found on non-singular elliptic curves? Does there always exist at least one? Consider curves over $\mathbb{R}$ and over $\mathbb{Z}_p$ where p is a prime.

(b) Prove that on a non-singular elliptic curve over $\mathbb{Z}_p$, where p is a prime, for any two different points P_1, P_2, there exists exactly one point P_3 such that $P_1 + P_2 + P_3 = \mathcal{O}$ (You are not allowed to use addition formulas).

(c) Prove or disprove that for P_3 as described in (b): $P_1 \neq P_3 \wedge P_2 \neq P_3$.

8.9. Consider the elliptic curve variant of the Diffie-Hellman key exchange protocol. Let $E : x^3 + 4x + 20 \pmod{29}$ and $P = (1, 5)$. Suppose Alice chooses her secret exponent $n_a = 11$ and Bob chooses his secret exponent $n_b = 7$.

(a) Show in detail all steps of the key exchange protocol.

(b) Suggest a more efficient way to exchange the computed points.

8.10. Consider the following elliptic curve cryptosystem.
Let E be an elliptic curve over the field $\mathbb{Z}_p$ and let $G \in E$ be a generator point of order n. E and G are public.
Each user U chooses a private key, an integer $s_U < n$, and computes the corresponding public key, $P_U = s_U G$.
To encrypt a message M for U, one chooses a random integer k and computes the ciphertext $C = [(kG), (M + kP_U)]$.

(a) Show how the user U decrypts C to obtain M.

(b) Let $E : y^2 = x^3 + x + 6 \pmod{11}$, $G = (2, 7)$ and $s_A = 7$. Recover the plaintext M from $C = [(8, 3), (10, 2)]$.

8.11. Decide whether $n^3 + (n+1)^3 + (n+2)^3 \equiv 0 \pmod{9}$ for any nonnegative integer.

8.12. Suppose $n = pq$ where p, q are primes. Let integers i, j, k and L with $k \neq 0$ satisfy

$$L = i(p-1), \quad L = j(q-1) + k \text{ and } a^k \not\equiv 1 \pmod{q}.$$

Let a be a randomly chosen integer satisfying $p \nmid a$ and $q \nmid a$. Prove that

$$\gcd(a^L - 1, n) = p.$$

8.13. Prove that all Carmichael numbers are odd.
(A Carmichael number is a composite number n such that $b^{n-1} \equiv 1 \pmod{n}$ for all integers $1 \leq b < n$ which are relatively prime to n.)

8.14. Prove or disprove the following claims:

(a) If n is even and $n > 2$, then $2^n - 1$ is composite.

(b) If $3 \mid n$ and $n > 3$, then $2^n - 1$ is composite.

(c) If $2^n - 1$ is prime, then n is prime.

8.15. Prove or disprove the following claim: An integer $n > 1$ is a prime if and only if n divides $2^n - 2$.

* **8.16.**

(a) Let n be an integer and p be the smallest prime factor of n. Prove that $\binom{n}{p}$ is not divisible by n.

(b) Let $n > 0$ be an integer. Show that n is a prime if and only if $\binom{n}{k}$ is divisible by n for all $k \in \{1, 2, \dots, n-1\}$.

(c) Let $n > 1$ be an integer. Let a be an integer coprime to n. Then n is prime if and only if

$$(x + a)^n = x^n + a \pmod{n}$$

in polynomial ring $Z_n[x]$.

8.17.

(a) Using the Pollard's ρ-method with $f(x) = x^2 + 1$ and $x_0 = 2$ find a factor of $2^{29} - 1$.

(b) Using the Pollard's $(p-1)$-method with $B = 5$ and $a = 2$ find a factor of 23729.

8.18. Using the elliptic curve $E : y^2 = x^3 + 4x + 4$ and its point $P = (0, 2)$ try to factorize the number 551.

8.19. Consider the Pollard's ρ-method with a pseudo-random function $f(x) = x^2 + c \mod n$ with a randomly chosen c, $0 \le c < n$. Why should be the values $c = 0$ and $c = n - 2$ avoided?

* **8.20.** For a modulus n, an exponent e is called a universal exponent if $x^e \equiv 1 \pmod{n}$ for all x with $\gcd(x, n) = 1$.
Universal Exponent Factorization Method
Let e be a universal exponent for n and set $e = 2^b m$ where $b \ge 0$ and m is odd. Execute the following steps.

(i) Choose a random a such that $1 < a < n - 1$. If $\gcd(a, n) > 1$, then we have a factor of n, and we terminate the algorithm. Otherwise go to step (ii).

(ii) Let $x_0 \equiv a^m \pmod{n}$. If $x_0 \equiv 1 \pmod{n}$, then go to step (i). Otherwise, compute $x_j \equiv x_{j-1}^2 \pmod{n}$ for all $j = 1, \dots, b$.

- If $x_j \equiv -1 \pmod{n}$, then go to step (i).
- If $x_j \equiv 1 \pmod{n}$, but $x_{j-1} \not\equiv 1 \pmod{n}$, then $\gcd(x_{j-1} - 1, n)$ is a nontrivial factor of n so we can terminate the algorithm.

(a) Use the algorithm above to factor $n = 76859539$ with the universal exponent $e = 12807000$.

(b) Find a universal exponent for $n = 2^{a+2}$. Justify your answer.

8.3 Solutions

8.1. Since $144 = 128 + 16 = 2^7 + 2^4$, we can calculate $144P$ in 8 additions as follows. We compute points $2P$, $4P = 2P + 2P$, $8P = 4P + 4P$, $16P = 8P + 8P$, $32P = 16P + 16P$, $64P = 32P + 32P$, $128P = 64P + 64P$, $144P = 128P + 16P = (1694, 125)$.

8.2. We are looking for k such that $Q = kP$. We compute kP, $k > 1$, until we find Q.

k	kP
2	$(20, 20)$
3	$(14, 14)$
4	$(19, 20)$
5	$(13, 10)$
6	$(7, 3)$
7	$(8, 7)$
8	$(12, 17)$
9	$(4, 5)$

Therefore, $k = 9$.

8.3. We know that elliptic curve given in the following form $y^2 + uxy + vy = x^3 + ax^2 + b^x + c$ can be transformed into the form $y^2 = x^3 + dx^2 + ex + f$ (if $p \neq 2$) and consequently into the form $y^2 = x^3 + gx + h$ (if $p \neq 3$). In this case we can immediately apply the second transformation using the substitution:

$$x \to x - \frac{d}{3}$$

For E we have $x \to x - \frac{6}{3} = x - 2$ ($3^{-1} \equiv 10 \mod 29$) and $y^2 = (x-2)^3 + 6(x-2)^2 + 14(x-1) + 16$ yields

$$y^2 = x^3 + 2x + 4$$

8.4.

(a) The curve is singular: $4a^3 - 27b^2 \equiv 0 \pmod{17}$, therefore it has multiple roots: $y^2 = x^3 + 10x + 5 = (x+5)^2(x+7) \pmod{17}$. Therefore the points of the curve does not form a group.

(b) Four points together with the point at infinity form a group. The addition table is explicitly given as follows:

	$\mathcal{O}$	(0, 1)	(4, 5)	(4, 2)	(0, 6)
$\mathcal{O}$	$\mathcal{O}$	(0, 1)	(4, 5)	(4, 2)	(0, 6)
(0, 1)	(0, 1)	(4, 5)	(4, 2)	(0, 6)	$\mathcal{O}$
(4, 5)	(4, 5)	(4, 2)	(0, 6)	$\mathcal{O}$	(0, 1)
(4, 2)	(4, 2)	(0, 6)	$\mathcal{O}$	(0, 1)	(4, 5)
(0, 6)	(0, 6)	$\mathcal{O}$	(0, 1)	(4, 5)	(4, 2)

The isomorphism between $(E, +)$ and $(\mathbb{Z}_5, +)$ is given as $\mathcal{O} \leftrightarrow 0$, $(0, 1) \leftrightarrow 1$, $(4, 5) \leftrightarrow 2$, $(4, 2) \leftrightarrow 3$ and $(0, 6) \leftrightarrow 4$.

8.5.

(a) Let $|E|$ denote the order of the group of points on E. By Hasse's theorem, we have $||E| - 6| < 2\sqrt{5}$, which implies $|E| < 6 + 2\sqrt{5} < 11$. Thus such an elliptic curve does not exist.

(b) Yes, for example the elliptic curve E given by the equation $y^2 = x^3 + 3x$ (whose discriminant $-16(4 \cdot 3^3 + 27 \cdot 0^2) = -2^6 \cdot 3^3$ is nonzero) contains ten points: $(0,0)$, $(1,2)$, $(1,3)$, $(2,2)$, $(2,3)$, $(3,1)$, $(3,4)$, $(4,1)$, $(4,4)$ and $\mathcal{O}$.

8.6. Let $a = \frac{p}{q}$ and $b = \frac{r}{s}$, where a, b are rational numbers and p, q, r, s are integers. Multiplying the equation given by E with q^6s^6 yields

$$q^6s^6y^2 = q^6s^6x^3 + pq^5s^6x + rq^6s^5$$

Let $X = q^2s^2x$ and $Y = q^3s^3y$. Now the equation can be rewritten as

$$Y^2 = X^3 + pq^3s^4X + rq^6s^5,$$

thus $A = pq^3s^4$ and $B = rq^6s^5$.

8.7. Using the previous theorem it suffices to find a point on $E : y^2 = x^3 - 25x$ such that its y-coordinate is nonzero. The point $(-4, -6)$ lies on the elliptic curve E. Using the transformation g we obtain the triple

$$g(x, y) = g(-4, -6) = \left(-\frac{3}{2}, -\frac{20}{3}, -\frac{41}{6}\right).$$

We can multiple the triple with -1 to obtain sizes of the right-angled triangle whose area is equal to 5.
Similarly, we can prove that 6 is the congruent number using the elliptic curve $E : y^2 = x^3 - 36x$ and its point $(-2, 8)$ yielding the sizes 3, 4 and 5.

8.8.

(a) Obviously, $2\mathcal{O} = \mathcal{O}$, since $\mathcal{O}$ is the identity element. Let $P \neq \mathcal{O}$. Rearranging the equation $2P = \mathcal{O}$, we have $P = -P$. Following the definition of addition of points of an elliptic curve, the inverse of P corresponds to the pair $(x, -y)$, where $P = (x, y)$. Thus, $y = -y$ and therefore $y = 0$.
The collection of points such that $y = 0$ is characterized by the solutions of the equation

$$x^3 + ax + b = y^2 = 0,$$

which depends on how many roots the cubic polynomial has. Certainly, the cubic has at least one real root. The other two roots are either two real numbers or complex conjugates. Therefore, curves over $\mathbb{R}$ have either two such points (one being a root of the cubic equation plus $\mathcal{O}$) or four such points (three roots plus $\mathcal{O}$). Note that there can be no multiple roots as the elliptic curve is assumed to be non-singular. Considering solutions over $\mathbb{Z}_p$ where p is a prime, the cubic modular equation has either 0, 1 or 3 roots.Therefore, there can be 1, 2, or 4 points such that $2P = \mathcal{O}$ over $\mathbb{Z}_p$.

(b) Let P_1 and P_2 be points of a non-singular elliptic curve E over $\mathbb{Z}_p$ where p is a prime such that $P_1 \neq P_2$. Then $P_1 + P_2$ must be a necessarily a point of E, since points of E with the addition operation form a group. Let $Q = P_1 + P_2$. Then Q has necessarily a unique inverse $-Q$, since once again $(E, +)$ is a group. Setting $P_3 = -Q$ gives the desired result as $P_1 + P_2 + P_3 = (P_1 + P_2) + P_3 = Q + (-Q) = \mathcal{O}$, and P_3 must be exactly one.

(c) The proposition $P_1 \neq P_3 \wedge P_2 \neq P_3$ does not hold in general. A counterexample can be given when the elliptic curve has at least two points P such that $2P = \mathcal{O}$, *i.e.* $P = -P$. Let $Q \neq \mathcal{O}$ be one such point.
Now observe that if $P_1 = \mathcal{O}$ and $P_2 = Q$, then $P_1 + P_2 = P_2 = Q$ but the only possibility for P_3 to make the equation $P_1 + P_2 + P_3 = \mathcal{O}$ hold, is $P_3 = -Q = Q = P_2$, which disproves the proposition.

8.9.

(a) (1) Alice computes $n_aP = (10, 25)$.
(2) Bob computes $n_bP = (24, 22)$.
(3) Alice and Bob interchange values n_aP and n_bP.
(4) Alice computes $n_a(n_bP) = (20, 3)$.
(5) Bob computes $n_b(n_aP) = (20, 3)$.

(b) We need not send the y-coordinate of a point (x, y). The value of $\pm y$ can be determined by computing the square root of x. Therefore, to send a point (x, y) it suffices to send x together with one bit to determine the sign of y. This idea is known as point compression.

8.10.

(a) The user U computes $(M + kP_U) - s_U(kG) = M + kP_U - k(s_UG) = M + kP_U - kP_U = M$.

(b) $s_A(kG) = 7(8,3) = (3,5)$, $M = (M + kP_A) - s_A(kG) = (10,2) - (3,5) = (10,2) + (3,-5) = (10,9)$.

8.11. We have $n^3 + (n+1)^3 + (n+2)^3 = 3n^3 + 9n^2 + 15n + 9 \equiv 3n^3 + 6n \pmod 9$. Now we prove by induction that $3n^3 + 6n \equiv 0 \pmod 9$. For $n = 0$, the equation holds. Assume the equation holds n. For $n+1$, we have $3(n+1)^3 + 6(n+1) = 3n^3 + 9n^2 + 9n + 3 + 6n + 6 = (3n^3 + 6n) + 9(n^2 + n + 1) \equiv 0 \pmod 9$.

8.12. Applying Fermat's little theorem we have

$$a^L - 1 \equiv a^{i(p-1)} - 1 \equiv (a^{p-1})^i - 1 \equiv 0 \pmod p,$$

$$a^L - 1 \equiv a^{j(q-1)+k} - 1 \equiv (a^{q-1})^j a^k - 1 \equiv a^k - 1 \pmod q.$$

Therefore $a^L - 1 = mp$ for some integer m and since $q \nmid a^L - 1$ and p, q are primes, it holds that

$$\gcd(a^L - 1, n) = \gcd(mp, pq) = p.$$

8.13. A Carmichael number is a composite number n which satisfies the equation $b^{n-1} \equiv 1 \pmod n$ for all $1 \le b < n$ with $\gcd(b,n) = 1$. Consider n being an even number, then $(n-1)^{n-1} \equiv (-1)^{n-1} \equiv -1 \pmod n$ which contradicts the definition of Carmichael numbers.

8.14.

(a) If $n = 2k$, then $2^n - 1 = 2^{2k} - 1 = (2^k - 1)(2^k + 1)$ is composite, since $n > 2 \Rightarrow k > 1 \Rightarrow 2^k - 1 > 1$.

(b) If $n = 3k$, then $2^n - 1 = 2^{3k} - 1 = (2^k - 1)(2^{2k} + 2^k + 1)$ is composite, since $n > 3 \Rightarrow k > 1 \Rightarrow 2^k - 1 > 1$.

(c) We prove the contraposition of (c): If n is composite, then $2^n - 1$ is composite. Assume $n = kl$ for some k, $l > 1$. Then, $2^{kl} - 1 = (2^k - 1)(2^{l(k-1)} + 2^{l(k-2)} + \ldots + 1)$. Therefore, $2^n - 1$ is composite.

8.15. The condition n divides $2^n - 2$ can be rewritten as $2^n \equiv 2 \pmod n$.
$\Rightarrow$:
This is true: Directly from Fermat's little theorem because n is prime.
$\Leftarrow$:
This is not true: $2^n - 2 \equiv 2 \pmod n$ for composite numbers $n = 341$ or $n = 561$ and others.

8.16.

(a) Let $n = p^k q$, where q is not divisible by p and $k \ge 1$. We have $\binom{n}{p} = \frac{n!}{(n-p)!p!} = \frac{n(n-1)(n-2)\ldots(n-p+1)}{p!} = \frac{p^k q(n-1)(n-2)\ldots(n-p+1)}{p!} = \frac{p^{k-1}q(n-1)(n-2)\ldots(n-p+1)}{(p-1)!}$. Since p is a prime and p divides n, p does not divide any number between n and $n - p$ and it does not divide the product of these numbers. Therefore $(n-1)(n-2)\ldots(n-p+1)$ is not divisible by p, q is not divisible by p and $\binom{n}{p} = \frac{p^{k-1}q(n-1)(n-2)\ldots(n-p+1)}{(p-1)!}$ is not divisible by p^k which gives that $\binom{n}{p}$ is not divisible by n.

(b) $\Rightarrow$:
We have $\binom{n}{k} = \frac{n!}{(n-k)!k!} = \frac{n(n-1)\ldots(n-k+1)}{k!}$ and since n is a prime and $k < n$, $k, k-1, \ldots, 1$ are coprime to n and therefore $\gcd(k!, n) = 1$. Since $\binom{n}{k}$ is an integer, $k! \mid (n-1)\ldots(n-k+1)$. Let $\frac{(n-1)\ldots(n-k+1)}{k!} = m$. Then $\binom{n}{k} = mn$ and $n \mid \binom{n}{k}$.
$\Leftarrow$:
We use obversion: If n is not a prime, then there exists $k \in \{1, 2, \ldots, n-1\}$ such that $n \nmid \frac{n}{k}$. Let p be the smallest prime factor of n. Obviously $1 < p < n-1$ and using the theorem proved in (a), $\binom{n}{p}$ is not divisible by n.

(c) $\Rightarrow$:
Given equation can be rewritten using binomial coefficients as

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

Let n be a prime. Using the theorem proved in (b), $\binom{n}{k}$ is divisible by n for all $1 \leq k \leq n-1$. Terms $\binom{n}{1} = \binom{n}{n-1} = n$ are divisible by n as well. Together, $\sum_{i=1}^{n-1} \binom{n}{i} x^{n-i} a^i \equiv 0 \pmod{n}$. From Fermat's little theorem, $a^n \equiv a \pmod{n}$ because $\gcd(a, n) = 1$. Therefore $(x+a)^n = x^n + a \pmod{n}$.
$\Leftarrow$:
Let n be a composite number. Let p be the smallest prime factor of n. Using the theorem proved in (a), $\binom{n}{p}$ is not divisible by n. Since $\gcd(a, n) = 1$, a^p is coprime to n as well. Therefore $\binom{n}{p} a^p$ is not divisible by n which means the coefficient corresponding to x^{n-p} is not congruent to zero modulo n and $(x+a)^n \not\equiv x^n + a \pmod{n}$.

8.17.

(a) This algorithm proceeds with evaluating $x_0, x_1, \ldots$ and for every x_i it computes $\gcd(x_i - x_0, n), \gcd(x_i - x_1, n), \ldots \gcd(x_i - x_{i-1}, n)$ until some gcd is greater than 1.
In this case, computing $\gcd(x_{14} - x_{12}, 2^{29} - 1) = \gcd(334654399 - 210447727, 2^{29} - 1) = 233$ yields a factor of n.

(b) We have $M = 2^2 \cdot 3 \cdot 5$ and $\gcd(a^M - 1, n) = \gcd(2^{60} - 1, 23729) = \gcd(16226, 23729) = 61$.

8.18. We subsequently calculate points kP for $k > 2$.

$kP = (x_k, y_k)$
$P = (0, 2)$
$2P = (1, 548)$
$3P = (24, 118)$
$4P = (382, 172)$
$5P = (136, 275)$
$6P = (507, 297)$
$7P = (260, 539)$
$8P = (516, 314)$
$9P = (477, 94)$
$10P = (214, 547)$
$11P = (495, 326)$
$12P = (171, 397)$

When computing $13P$ we try to compute $\lambda = (y_2 - y_1)/(x_2 - x_1)$, but fail because $(x_{12} - x_1) = 171$ and $\gcd(171, 551) = 19$.

8.19. If $c = 0$, it can happen that the sequence x_0, $x_1 = f(x_0)$, $x_2 = f(x_1) \dots$ contains some $x_k \equiv 1 \pmod{n}$. In such case $x_l \equiv 1 \pmod{n}$ for all $l > k$.
Similarly, if $c = n - 2$ then $c \equiv -2 \pmod{n}$ and if $x_k \equiv -1 \pmod{n}$, then $x_l \equiv (-1)^2 - 2 \equiv -1 \pmod{n}$. In both cases the sequence contains a cycle which makes the algorithm useless.

8.20.

(a) We have that $e = 2^3 \cdot 1600875$ and we select $a = 2$ as the base. All congruences are modulo n in the following. Since $x_0 \equiv 2^{1600875} \equiv 76859538 \equiv -1$, we have to choose a new base. Let $a = 3$, then $x_0 \equiv 3^{1600875} \equiv 44940756$, $x_1 \equiv x_0^2 \equiv 9649071$, $x_2 \equiv x_1^2 \equiv 1$. Since $x_1 \not\equiv 1$, $\gcd(x_1 - 1, n) = 8539$ and $n = 8539 \cdot 9001$.

(b) We will show that 2^a is the universal exponent for $n = 2^{a+2}$. We use induction on a to prove that $x^{2^a} \equiv 1 \pmod{2^{a+2}}$ for odd x. If $a = 1$ then it is easy to see that $x^2 \equiv 1 \pmod{8}$. Now assume that $x^{2^{a-1}} \equiv 1 \pmod{2^{a+1}}$. Therefore, $x^{2^a} = (1 + 2^{a+1}t)^2$ for some $t \in \mathbb{Z}$, thus $x^{2^a} \equiv 1 \pmod{2^{a+2}}$.

Chapter 9

Identification, Authentication and Secret Sharing

9.1 Introduction

More applications of cryptography ask for identification of communicating parties and for data integrity and authentication rather than for secret data. A practically very important problem is how to protect data and communication against an active attacker.

9.1.1 User identification

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

- The verifier has to accept prover's identity if both parties are honest.
- The verifier cannot later, after successful identification, pose as a prover and identify himself (as the prover) to another verifier.
- 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 the following 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 an adversary when communicating with Bob that he is Alice, without Bob having a large chance to find that out.

Static means like passwords or fingerprints can be used, or identification can be implemented by dynamic means, with challenge and response protocols. In a challenge-response identification protocol Alice proves her identity to Bob by demonstrating knowledge of a secret known to be associated with Alice only, without revealing the secret itself to Bob. Structure of challenge-response protocols is as follows:

1. Alice sends a commitment (some random element) to Bob.
2. Bob sends a challenge to Alice.
3. Alice sends the response that depends on the challenge received and her commitment to Bob.
4. Bob verifies the response.

Simplified Fiat-Shamir identification scheme

Let p,q be large random primes, $n = pq$, $v \in QR(n)$ be a quadratic residue and s such that $s^2 \equiv v \pmod{n}$. Alice is given as her private key s, while v is made public.

1. Alice chooses a random $r < n$, computes $x = r^2 \mod n$ and sends x to Bob.
2. Bob sends to Alice a random bit b as a challenge.
3. Alice sends Bob as the response $y = rs^b \mod n$.
4. Bob identifies the sender as Alice if and only if $y^2 = xv^b \mod n$.

Alice and Bob repeat this protocol several times, until Bob is convinced that Alice knows s. If Alice does not know s, she can choose r such that she can give the correct response to Bob if he sends her the challenge 0, or she can choose r such that she can give the correct response to Bob if he sends her the challenge 1, but she cannot do both. The probability of her giving the correct response to Bob t times is $\frac{1}{2^t}$.

Schnorr identification scheme

Setup phase: Trusted authority (TA) involved chooses a large prime p, a large prime q dividing $p-1$, an $\alpha \in \mathbb{Z}_p^*$ of order q and a security parameter t such that $2^t < q$. These parameters p, q, α, t are made public. TA also establishes a secure digital signature scheme with a secret signing algorithm sig_{TA} and a public verification algorithm ver_{TA}.
Certificate issuing: TA verifies Alice's identity by conventional means and forms a string $ID(Alice)$ which contains her identification information.
Alice chooses a secret random $1 \leq a \leq q-1$ and computes $v = \alpha^{-a} \pmod{p}$ and sends v to the TA. TA generates signature $s = sig_{TA}(ID(Alice), v)$ and sends $C(Alice) = (ID(Alice), v, s)$ back to Alice as her certificate.
Identification protocol:

1. Alice chooses a random commitment $0 \leq k < q$ and computes $\gamma = \alpha^k \pmod{p}$.
2. Alice sends to Bob γ and her certificate $C(Alice) = (ID(Alice), v, s)$.
3. Bob verifies the signature of TA.
4. Bob chooses a random challenge $1 \leq r \leq 2^t$ and sends it to Alice.
5. Alice computes the response $y = (k + ar) \pmod{q}$ and sends it to Bob.
6. Bob verifies that $\gamma \equiv \alpha^y v^r \pmod{p}$.

9.1.2 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 $k \in K$ there is a single and easy to compute authentication mapping $auth_k : \{0,1\}^* \times M \to T$ and a single and 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 $m \in M$ and $k \in K$ 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 $m \in M$ and $k \in K$ it is computationally unfeasible (without the knowledge of k) to find c from T such that $ver_k(m, c) = true$.

9.1.3 Secret sharing

A secret sharing scheme is a method to distribute a secret among several users in such a way that only predefined sets of users (so called access structure) can recover the secret. In particular, an (n, t)-threshold scheme, where n and $t < n$ are integers, is a method of sharing a secret S among a set P of n participants, $P = \{P_i \,|\, 1 \le i \le n\}$, in such a way that any t, or more, participants can compute the value S, but no group of $t-1$, or less, 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.

Shamir's (n, t)-threshold scheme

The essential idea of Shamir's threshold scheme is that t points define a polynomial of degree $t-1$ (2 points are sufficient to define a line, 3 points are sufficient to define a parabola, and so on).
Initiation phase: Dealer D chooses a prime $p > n$, n distinct x_i, $1 \le i \le n$, and D gives the value x_i to the user P_i. The values x_i are made public.
Share distribution phase: Suppose D wants to share a secret $S \in \mathbb{Z}_p$ among the users. D randomly chooses and keeps secret $t-1$ elements from $\mathbb{Z}_p$: $a_1, \dots a_{t-1}$. For $1 \le i \le n$, D computes the shares $y_i = f(x_i)$, where

$$f(x) = S + \sum_{j=1}^{t-1} a_j x^j \pmod{p}.$$

D gives the computed share y_i to the participant P_i.
Secret cumulation phase: Suppose participants $P_{i_1}, \dots, P_{i_t}$ want to determine s shared secret S. Since $f(x)$ has degree $t-1$, $f(x)$ has the form $f(x) = f_0 + a_1 x + \cdots + a_{t-1}x^{t-1}$, and coefficients 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 $S = a_0$.
More technically, using so called Lagrange formula, any group J of t or more parties can reconstruct the secret S using the following equalities:

$$S = a_0 = f(0) = \sum_{i \in J} f_i \prod_{j \in J, j \ne i} \frac{j}{j-i}.$$

Access structures

A more general structure of participants that are allowed to reconstruct a secret may be required. An authorized set of parties $A \subseteq P$ is a set of parties who should be able, when cooperating, to construct the secret. An unauthorized set of parties $U \subseteq P$ is a set of parties who alone should not be able to learn anything about the secret. Let P be a set of parties. The access structure $\Gamma \subseteq 2^P$ is a set such that $A \in \Gamma$ for all authorized sets A and $U \subseteq 2^P - \Gamma$ for all unauthorized sets U.

Orthogonal arrays

An orthogonal array $OA(n, k, \lambda)$ is a $\lambda n^2 \times k$ array of n symbols, such that in any two columns of the array every one of the possible n^2 pairs of symbols occurs in exactly λ rows. The following holds for orthogonal arrays:

- If p is a prime, then $OA(p, p, 1)$ exists.

- If p is a prime and $d \geq 2$ is an integer then there exists an orthogonal array OA$(p, \frac{p^d-1}{p-1}, p^{d-2})$.

$$\text{Example of } OA(3,3,1): \begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 2 & 2 & 2 \\ 0 & 1 & 2 \\ 1 & 2 & 0 \\ 2 & 0 & 1 \\ 0 & 2 & 1 \\ 1 & 0 & 2 \\ 2 & 1 & 0 \end{pmatrix}$$

There is the following relation between orthogonal arrays and authentication matrices. If there is an orthogonal array $OA(n, k, \lambda)$, then there is an authentication code with $|M| = k$, $|T| = n$, $|K| = \lambda n^2$ and $P_I = P_S = \frac{1}{n}$ where P_I (P_S) is the probability of impersonation (substitution). Rows of an orthogonal array are used as authentication rules with each row chosen with equal probability $\frac{1}{\lambda n^2}$, columns correspond to messages and symbols correspond to authentication tags.

9.2 Exercises

9.1. Alice and Bob share a bit string k. Alice identifies herself to Bob using the following protocol:

(a) Bob randomly chooses a bit string r and sends it to Alice.

(b) Alice computes $r \oplus k$ and sends it to Bob.

(c) Bob accepts if and only if $k = r \oplus c$ where c is the received bit string.

Is this protocol secure?

9.2. Consider the following identification protocol. Let K_{XY} denote a secret key shared between parties X and Y. Let N_X denote a random integer generated by X and $\{m\}_K$ denote a message m encrypted (using a given encryption system) with the key K. To authenticate herself to Bob (denoted as B), Alice (denoted as A) performs, with the help of an authentication server (denoted as S), the following steps:

(i) $A \to B$: “Alice”

(ii) $B \to A : N_B$

(iii) $A \to B : \{N_B\}_{K_{AS}}$

(iv) $B \to S$: “Bob”, $\{$“Alice”, $\{N_B\}_{K_{AS}}\}_{K_{BS}}$

(v) $S \to B : \{N_B\}_{K_{BS}}$

(vi) verify N_B

(a) Show that a malicious user Mallot can impersonate Alice to Bob.

(b) Propose a modification of the above protocol to prevent an attack from (a).

9.3. Let G be a cyclic group of the prime order p and g be its generator. Alice chooses as her private key $x \in \mathbb{Z}_p$, the corresponding public key will be $X = g^x \pmod p$. Consider the following user identification scheme:

(1) Alice randomly chooses $r \in \mathbb{Z}_p$, and sends $R = g^r \pmod p$ and $S = g^{x-r} \pmod p$ to Bob.

(2) Bob responds by sending a randomly chosen bit b.

(3) If $b = 0$, Alice sends $z = r$ to Bob, otherwise she sends $z = x - r$.

(a) Find and explain the acceptance condition.

(b) Show that the adversary Eve is able to impersonate Alice with probability $\frac{1}{2}$.

(c) Propose a change which makes the protocol more secure.

* **9.4.** Consider the Schnorr identification scheme.

(a) Why is it important that the steps 1, 2 and 4 in the scheme are in this order? Would it affect security of the protocol if Bob chooses and sends the r first?

(b) Let, when following the protocol, Bob realize that Alice is using the same γ that she used previously when she was identifying herself to him. Let Bob save logfiles of that communication with Alice. Can he abuse this?

9.5. Let Alice and Bob use the Fiat-Shamir identification scheme. Let, for some reason, Bob needs to convince Charles that he communicated with Alice recently. In order to do that, he shows Charles what he claims to be a transcript of a recent execution of the Fiat-Shamir scheme in which he accepted Alice's identity. Should Charles be convinced after seeing the transcript? Explain your reasoning.

9.6. Consider the following protocol for mutual identification between Alice (denoted as A) and Bob (denoted as B). Let K_{AB} denote a secret key shared between parties A and B. Let N_X denote a random integer generated by X and $\{m\}_K$ denote a message m encrypted (using a given encryption system) with the key K.

(i) $A \to B$: "Alice", N_A

(ii) $B \to A : \{N_A\}_{K_{AB}}, N_B$

(iii) $A \to B : \{N_B\}_{K_{AB}}$

Show that the proposed protocol is not secure and propose a secure variant.

9.7. Alice and Bob share a secret random key k and let they intend to use it to authenticate their two bit messages with single bit tags as follows. The protocol consists of picking one of the functions from H, where H is a set of hash functions, according to the key k. Alice's message is then $(m, h_k(m))$, where h_k is the hash function chosen according to the key k. Bob, after receiving (possibly modified) message (m', t') computes $h_k(m')$ and verifies if $t' = h_k(m')$.

(a) Consider H given by the following table, where rows are labeled by hash functions and columns by their arguments. Is the above protocol secure?

h \ m	00	01	10	11
h_1	1	1	0	0
h_2	0	0	1	1
h_3	1	0	0	1
h_4	0	1	1	0

(b) Can you find a set of hash functions that provides secure authentication?

9.8. Eleven scientists are working on a secret project. They wish to lock up the documents in a cabinet so that the cabinet can be opened if and only if six or more of the scientists are present.

(a) What is the smallest number of locks needed?

(b) What is the smallest number of keys to the locks each scientist must carry?

(c) Generalize the result for n scientists of which m have to be present to be able to open the cabinet.

9.9. Show how you could extend Shamir's (n, t)-threshold scheme to a $(n + 1, t)$-scheme that includes an additional user without changing the existing shares.

* **9.10.** Consider the modification of the Shamir's (n, t)-threshold scheme where calculations are over all integers and not over a finite field. Show that this modification is not perfectly secure, *i.e.* knowledge of less than t shares gives some information about the secret.

9.11. There are four people in a room and exactly one of them is a spy. The other three people share a secret using the Shamir's $(3, 2)$-threshold scheme over $\mathbb{Z}_{11}$. The foreign spy has randomly chosen his share. The four pairs are $P_1 = (1, 7)$, $P_2 = (3, 0)$, $P_3 = (5, 10)$ and $P_4 = (7, 9)$. Find out which pair was created by the foreign spy.

* **9.12.** Consider the following (n, t)-secret sharing scheme. Let $a_1, a_2, \dots a_n$ be an increasing sequence of pairwise co-prime integers such that the product of the smallest t of them is greater than the product of the $t - 1$ largest ones, *i.e.* $\prod_{i=1}^{t} a_i > \prod_{i=1}^{t-1} a_{n-i+1}$. Choose a secret s from the interval $(\prod_{i=1}^{t-1} a_{n-i+1}, \prod_{i=1}^{t} a_i)$ and compute the corresponding shares $s_i = s \pmod{a_i}$ for $1 \leq i \leq n$. Show that t participants can reconstruct the secret s.

9.13. Consider the Shamir's $(10, 3)$-threshold scheme over $\mathbb{Z}_p$, where p is a large prime. Suppose an adversary corrupts one of the share holders and this share holder intends to give a bad share in the secret cumulation phase. The problem is that nobody knows which share holder is corrupted.

(a) Describe a method how to reconstruct s given all 10 shares and explain why it works.

(b) Determine the smallest number x of shares that are sufficient to reconstruct s. Explain.

(c) Is it true that no collection of fewer than x share holders can obtain some information about s? Explain.

9.14. A military office consists of one General, two Colonels and five Majors. They have control of a powerful missile, but they do not want to launch it unless the General decides to launch it, or two Colonels decide to launch it, or five Majors decide to launch it, or one Colonel together with three Majors decide to launch it. How they would do that with a secret sharing scheme.

9.15. Secret sharing schemes for general access structures can be constructed by using several independent instances of a (n, t)-threshold scheme.

(a) Design a secret sharing scheme for five participants $\{A, B, C, D, E\}$ and access structure $\{\{A, B\}, \{B, C, D\}, \{A, D, E\}\}$ with the use of as few instances of a threshold scheme as possible.

(b) Which subset of participants can we add to the access structure given in (a) to make it implementable by a single threshold scheme?

9.16. Find an example of an orthogonal array $\mathrm{OA}(3, 4, 1)$ or at least prove that such exists.

9.3 Solutions

9.1. The protocol is not secure because an eavesdropper can intercept r and $r \oplus k$ and easily compute $k = r \oplus c$ for later impersonation.

9.2.

(a) Mallot (denoted as M) may proceed as follows:

(i) He establishes two connections with B simultaneously. In the first connection, M is acting as himself, in the second connection M is pretending to be A.

(ii) B sends him random integers N_{B_1} designated for M and N_{B_2} designated for A.

(iii) M throws away N_{B_1} and in both sessions he sends $\{N_{B_2}\}_{K_{MS}}$ to B.

(iv) B sends "Bob", $\{$"Alice", $\{N_{B_2}\}_{K_{MS}}\}_{K_{BS}}$ and "Bob", $\{$"Mallot", $\{N_{B_2}\}_{K_{MS}}\}_{K_{BS}}$ to S.

(v) S successfully restores N_{B_2} from $\{$"Mallot", $\{N_{B_2}\}_{K_{MS}}\}_{K_{BS}}$ and restores some garbage G from $\{$"Alice", $\{N_{B_2}\}_{K_{MS}}\}_{K_{BS}}$.

(vi) S sends $\{N_{B_2}\}_{K_{BS}}$ and $\{G\}_{K_{BS}}$ to B.

(vii) B successfully restores N_{B_2} (B is thinking that A authenticated herself successfully to B), and restores some garbage G (B is thinking that M did not authenticate himself to B).

(viii) M can now impersonate A.

(b) The protocol can be corrected by enforcing the server S to include in step (v) information about the user who wants to authenticate, *i.e.* that A wants to authenticate herself to B. The server S therefore sends $\{$"Alice", $N_B\}_{K_{BS}}$ to B.

9.3.

(a) Bob accepts if and only if $R \cdot S \equiv X \pmod p$ and either $b = 0$ and $R = g^z \pmod p$ or $b = 1$ and $S = g^z \pmod p$.

(b) Eve can always send $R = 0$, $S = X$ and $z = 0$. If $b = 0$, Bob will accept and Eve successfully impersonates Alice, In case $b = 1$, Bob will reject. Probability of impersonation is $\frac{1}{2}$.

(c) The presented scheme can be modified as follows (resulting eventually in the Schnorr identification scheme):

(i) Alice chooses a random $r \in \mathbb{Z}_p$ and sends $R = g^r \pmod p$ to Bob.

(ii) Bob sends a random challenge $b \in \mathbb{Z}_p$ to Alice.

(iii) Alice sends $z = r + bx \pmod p$ to Bob.

Bob accepts if and only if $R \cdot X^b \equiv g^z \pmod p$.

9.4.

(a) It is necessary that Alice makes commitment first, before Bob picks and sends his challenge r. Suppose to the contrary that Bob sends r first. In order to impersonate Alice, an adversary Eve can use Alice's public certificate, stored for example during his communication with Alice in the past. Eve can choose an arbitrary y and sends to Bob

$$\gamma \equiv \alpha^y v^r \pmod p.$$

In such case, Bob will successfully verify the received γ without Eve proving the knowledge of the secret key a.

(b) After receiving $\gamma_2 = \gamma_1$, Bob could realize that using of the same γ's implies that Alice used the same k's as well, because $0 \leq k < q$ and q is the order of α in $\mathbb{Z}_p^*$. In such a situation he avoids using the same r_1 as before, *i.e.* he sends r_2 such that $r_2 \neq r_1$, so that $y_2 \neq y_1$. After receiving y_2, he obtains the following two equations:

$$y_1 \equiv k_1 + ar_1 \pmod q \text{ and } y_2 \equiv k_2 + ar_2 \pmod q$$

Now, the equations can be combined to obtain:

$$y_1 - y_2 \equiv k_1 - k_2 + ar_1 - ar_2 \pmod q.$$

Since $k_1 \equiv k_2$ but $y_1 \not\equiv y_2$:

$$y_1 - y_2 \equiv ar_1 - ar_2 \pmod q$$
$$y_1 - y_2 \equiv a(r_1 - r_2) \pmod q$$

All of the values y_1, y_2, r_1, r_2 are known to Bob and so he is able to compute a and he can later easily impersonate Alice.

9.5. Charles should not get convinced because Bob can easily forge communication as follows: Bob randomly chooses his challenge b that he will use in the transcript.
For the challenge $b = 0$, Bob can choose a random r, compute $x = r^2 \pmod n$ and write to the transcript that he received x as the commitment from Alice. Then he pretends that he sent the challenge $b = 0$ to Alice and that he received $y = r$. According to the protocol, everything seems OK, because $y^2 \equiv xv^0 \pmod n \rightarrow r^2 \equiv r^2 \pmod n$.
For the challenge $b = 1$, Bob can choose a random r, compute $x = r^2v^{-1} \pmod n$ and write to the transcript that he received x as the commitment. He can pretend to have sent challenge $b = 1$ to Alice and received r. According to the protocol, everything seems OK, because $y^2 \equiv xv^1 \pmod n \rightarrow r^2 \equiv r^2v^{-1}v \pmod n \rightarrow r^2 \equiv r^2$.
He could have written to the transcript as many repetitions of the protocol as he wants (randomly choosing between $b = 0$ and $b = 1$) and this way there would be no difference between the forged transcript and the genuine one.

9.6. The proposed protocol is vulnerable to the so-called reflection attack (the adversary Mallot is denoted as M).

(i) $M \rightarrow B$: "Alice", N_E

(ii) $B \rightarrow M : \{N_M\}_{K_{AB}}, N_B$

(iii) $M \rightarrow B$: "Alice", N_B (Mallot initiates a new round)

(iv) $B \rightarrow M : \{N'_B\}_{K_{AB}}$

(v) $M \rightarrow B : \{N_B\}_{K_{AB}}$

To prevent this attack the protocol can be augmented as follows:

(i) $A \rightarrow B$: "Alice", N_A

(ii) $B \rightarrow A$: {"Alice", $N_A\}_{K_{AB}}, N_B$

(iii) $A \rightarrow B$: {"Bob", $N_B\}_{K_{AB}}$

With this modification there is still a problem: Bob encrypts a message chosen by Alice making the protocol vulnerable to a chosen-plaintext attack. This problem can be eliminated as follows

(i) $A \rightarrow B : A, N_A$

(ii) $B \to A : \{\text{“Alice”}, N_A, N_B\}_{K_{AB}}$

(iii) $A \to B : \{N_B, N_A\}_{K_{AB}}$

9.7.

(a) The messages 00 and 10 have opposite values of authentication tags, therefore whenever an adversary can see the message-tag pair $(00, 1)$ she knows that the message-tag pair $(10, 1)$ is valid as well.

(b) The set H can be given as follows:

h \ m	00	01	10	11
h_1	1	0	0	0
h_2	1	0	1	1
h_3	1	1	0	1
h_4	1	1	1	0
h_5	0	0	0	0
h_6	0	0	1	1
h_7	0	1	0	1
h_8	0	1	1	0

9.8.

(a) For each group of five scientists, there must be at least one lock for which they do not have the key, but for which every other scientist does have the key. There are $\binom{11}{5} = 462$ groups of five scientists, therefore there must be at least 462 locks.

(b) Similarly, each scientist must hold at least one key for every group of five scientists of which s(he) is not a member, and there are $\binom{10}{5} = 252$ such groups.

(c) We generalize the previous results. For each group of $m - 1$ scientists, there must be at least one lock for which they do not have the key, but for which every other scientist does have the key. There are $\binom{n}{m-1}$ such groups.
Each scientist must hold at least one key for every group of $m-1$ scientists of which he or she is not a member, and there are $\binom{n-1}{m-1}$ such groups.

9.9. When t is kept fixed, shares can be dynamically added, or deleted, without affecting the other shares. The dealer simply evaluates the secret polynomial f in a new point $x \in \mathbb{Z}_p$ and shares $(x, f(x))$ with the new user.

9.10. Let f be a linear polynomial with $f(0) = s$ and $f(1) = a_1$. We have

$$f(x) = (a_1 - s)x + s.$$

Assume you are given the share $f(2)$. We can see that $f(2) = 2a_1 - s$. Since a_1 and s are integers, given this single share $f(2)$, one can learn parity of s. Because $2a_1$ is always even and if $f(2)$ is even, the secret s has to be even. Similarly, when $f(2)$ is odd, then s has to be odd.

9.11. The given shares correspond to the following equations:

$$7 \equiv a + b \pmod{11}$$

$$0 \equiv 3a + b \pmod{11}$$
$$10 \equiv 5a + b \pmod{11}$$
$$9 \equiv 7a + b \pmod{11}$$

From the first two equations we have $a = 2$ and $b = 5$, but this solution is not valid for the third and fourth equation. Therefore, the foreign spy has to be either the person with P_1 or P_2. From the first and third equation we have $a = 9$ and $b = 9$. Since this solution is not valid for the fourth equation, the foreign spy is the person with the share P_1.

9.12. Given t distinct shares $I_{i_1}, \ldots, I_{i_t}$, the secret s is recovered using the Chinese Remainder Theorem, as the unique integer solution of the system of modular equations

$$x \equiv I_{i_1} \pmod{a_{i_1}}$$

$$\ldots$$

$$x \equiv I_{i_t} \pmod{a_{i_t}}.$$

Moreover, s lies in $\mathbb{Z}_{a_{i_1} \cdots a_{i_t}}$ because $s < \prod_{i=1}^{t} a_i$.
On the other hand, having only $t-1$ distinct shares $I_{i_1}, \ldots, I_{i_{t-1}}$, we obtain only that $s \equiv x_0 \pmod{a_{i_1} \cdots a_{i_{t-1}}}$, where x_0 is the unique solution modulo $a_{i_1} \cdot a_{i_2} \cdot \ldots \cdot a_{i_{t-1}}$ of the resulted system $(S > \prod_{i=1}^{t-1} a_{n-i+1} \geq a_{i_1} \cdot a_{i_2} \cdot \ldots \cdot a_{i_{t-1}} > x_0)$.

9.13.

(a) Given all 10 shares we can reconstruct s as follows. We divide 10 shares into 4 groups: three groups containing 3 shares and one group containing 1 share. The bad share could be in at most one group containing 3 shares, therefore the same secret will be computed from at least two groups containing 3 shares and that will be equal to s.

(b) We show that 4 shareholders are not enough to recover the secret reliably. In case there is a corrupted share among 4 shares, four different values will be recovered and we cannot tell which one is correct.
Let us consider the case of 5 shareholders with a corrupted share. We can compute secrets for all possible triples. There are $\binom{4}{3} = 4$ triples from which the correct secret s will be recovered and $\binom{4}{2} = 6$ triples resulting in incorrect secrets - these will be pairwise different or do not exist - therefore it is possible to reliably recover the secret s given 5 shares.

(c) This is not true. From (b) we can see that 4 shareholders compute in the worst case 4 different secrets and the correct one has to be between them.

9.14. Suppose the knowledge of a secret s is needed to launch the missile. We can realize the desired access structure with the $(20, 10)$-threshold scheme in which each Colonel is given five shares and each Major is given two shares. The General is given the secret s directly. Then, two Colonels or five Majors have together 10 shares and one Colonel with three Majors have 11 shares.

9.15.

(a) The simplest, but not optimal, solution is to have $(2, 2)$-threshold scheme for $\{A, B\}$ and other two $(3, 3)$-schemes for $\{B, C, D\}$ and $\{A, D, E\}$. The solution that is using two instances can be as follows: a $(3, 3)$ scheme for $\{A, D, E\}$ and a $(7, 5)$-scheme for $\{A, B, C, D\}$ in which A obtains two shares, B obtains three shares and both C and D obtain one share.

(b) We can add $\{B, D, E\}$ and use a $(15, 9)$-scheme in which A is given four shares, B is given five shares, C is given one share, D is given three shares and E is given two shares.

9.16. We know that if p is a prime and $d \leq 2$ is an integer then there exists an orthogonal array

$$\mathrm{OA}\left(p, \frac{p^d - 1}{p - 1}, p^{d-2}\right).$$

For $p = 3$ and $d = 2$ we have that $\mathrm{OA}(3, 4, 1)$ exists. To construct such array we can extend the $\mathrm{OA}(3, 3, 1)$ with the fourth column as follows:

$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 2 & 2 & 2 & 0 \\ 0 & 1 & 2 & 1 \\ 1 & 2 & 0 & 1 \\ 2 & 0 & 1 & 1 \\ 0 & 2 & 1 & 2 \\ 1 & 0 & 2 & 2 \\ 2 & 1 & 0 & 2 \end{pmatrix}$$

Chapter 10

Coin Tossing, Bit commitment, Oblivious Transfer, Zero-knowledge Proofs and Other Crypto-protocols

10.1 Introduction

Cryptographic protocols are specifications of how two parties, usually called Alice and Bob, should prepare themselves for their communication and how they should behave during their communication in order to achieve their goal and be protected against an adversary.
Cryptographic protocols can be very complex. However, they are often composed of several, very simple, though special, protocols. These protocols are called cryptographic primitives – coin-flipping protocols, commitment protocols or oblivious transfers.

10.1.1 Coin-flipping protocols

In *coin-flipping* (or *coin-tossing*) protocols, Alice and Bob can flip a coin over a distance in such a way that neither of them can determine the outcome of the flip, but both can agree on the outcome in spite of the fact that they do not trust each other. Both outcomes – head or tail – should have the same probability and both parties should influence the outcome.

10.1.2 Bit commitment protocols

In *bit commitment* protocols, Alice can choose a bit and get committed to its value such that Bob has no way of learning Alice's commitment (without Alice's help) and Alice has no way of changing her commitment.
A commit function is a mapping $commit : \{0,1\} \times X \rightarrow Y$ where X, Y are finite sets. Each bit commitment scheme consists of two phases:

Commitment phase: Alice sends a bit b she wants to commit to, in an encrypted form, to Bob.
Opening phase If required, Alice sends to Bob some information that enables him to recover b.

Each bit commitment scheme should satisfy the following properties:

Hiding (or privacy): Bob cannot determine the value of b, he cannot distinguish a commitment to 0 and a commitment to 1. More formally, for no $b \in \{0,1\}$ and no $x \in X$, it is feasible for Bob to determine the value b from the commitment $B = commit(b,x)$.
Binding: Alice cannot later, after the commitment phase, change her mind. Alice can open her commitment, by revealing b and x such that $B = commit(b,x)$, but she cannot open her commitment as both 0 and 1.
Correctness (or viability): If both Alice and Bob follow the protocol, Bob will always recover the committed value b.

Hiding can be
unconditional: A commitment to b reveals no information to a infinitely powerful Bob. Distributions of $commit(0, r)$ and $commit(1, r)$ are indistinguishable.
computational: Bob will not be able to tell efficiently which of the two given values is in a commitment, with probability larger than just guessing at random.

Binding can be
unconditional: Alice, even with infinite computing power, cannot change her mind after committing.
computational: Unless Alice has unrealistically large computing resources, her chances of being able to change her mind are very small.

10.1.3 Oblivious transfers

The *standard oblivious transfer* is a protocol in which Alice transmits a message to Bob in such a way that Bob receives the message with probability $\frac{1}{2}$ and some garbage otherwise. Moreover, Bob knows whether he has received the message or garbage. However, Alice will not know which one he has received.
In the 1-*out-of-*2 *oblivious transfer*, Alice transmits two messages to Bob. Bob can choose whether to receive the first or the second message, but he cannot receive both. Again, Alice has no idea which of them Bob has received (*1-out-of-k oblivious transfer* is a generalization to k messages).

10.1.4 Interactive and zero-knowledge proofs

In an *interactive proof* system, there are two parties: a prover, often called Peggy, and a verifier, often called Victor. The prover knows some secret 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 the prover and the 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 the verifier and a response of the prover. At the end, the verifier either accepts or rejects the prover's attempts to convince him.

A *zero-knowledge proof* of a theorem T is an interactive two party protocol, in which the prover is able to convince the verifier who follows the same protocol, by the overwhelming statistical evidence, that T is true, if T is indeed true (*completeness*), but no prover is able to convince the verifier that T is true, if this is not so (*soundness*). In addition, during interactions, the prover does not reveal during their communication to the verifier any other information, except whether T is true or not (*zero-knowledge*). Therefore, the verifier who got convinced about the correctness of the statement gets from their communication not enough knowledge to convince a third person about that.

Zero-knowledge proof of the graph isomorphism

Example of a zero-knowledge proof for proving that two graphs are isomorphic is as follows:

Given are: Peggy and Victor know two graphs G_1 and G_2 with a set of nodes $V = \{1, \ldots, \}$. The following steps are then repeated t times (where t is a chosen security parameter).

(1) Peggy chooses a random permutation π of $V = \{1, \ldots, n\}$ and computes H to be the image of G_1 under the permutation π, and sends H to Victor.

(2) Victor randomly chooses $i \in \{1, 2\}$ and sends it to Peggy. This way Victor asks for an isomorphism between H and G_i.

(3) Peggy creates a permutation ρ of $V = \{1, \ldots, n\}$ such that ρ specifies the isomorphism between H and G_i and sends ρ to Victor.
If $i = 1$, Peggy takes $\rho = \pi$; if $i = 2$, Peggy takes $\rho = \sigma \circ \pi$, where σ is a fixed isomorphic mapping of nodes of G_2 to G_1.

(4) Victor checks whether H provides the isomorphism between G_i and H. Victor accepts Peggy's proof if H is the image of G_i in each of the t rounds.

If G_1 and G_2 are isomorphic then Victor accepts with probability 1 (completeness). If graphs G_1 and G_2 are not isomorphic, then Peggy can deceive Victor only if she is able to guess in each round the value i that Victor has chosen and then she sends as H the graph G_i. However, the probability that this happens, in each of t rounds, is 2^{-t} (soundness).

10.2 Exercises

10.1. Suppose you can predict results of coin flips. At least how many coin flips would you need to prove this to your friend without revealing your secret so that he would be at least $n\%$ sure about it?

* **10.2.** Consider the following coin-flipping protocol:

(1) Alice generates a Blum integer, n, a random x relatively prime to n, $x_0 = x^2 \mod n$, and $x_1 = x_0^2 \mod n$. She sends n and x_1 to Bob.

(2) Bob guesses the parity of x_0.

(3) Alice sends x and x_0 to Bob.

(4) Bob checks that n is a Blum integer (Alice would have to give Bob the factors of n and proofs of their primality, or execute some zero-knowledge protocol to convince him that n is a Blum integer), and he verifies that $x_0 = x^2 \mod n$ and $x_1 = x_0^2 \mod n$. If all checks are alright, Bob wins the flip if he guessed correctly.

Would this protocol be secure if we omit the requirement that n be a Blum integer?

10.3. Let p be a large prime. Let $g < p$ be an integer such that g is a generator of the group $\mathbb{Z}_p^*$. Discuss security of the following commitment scheme for numbers from $\{0, 1, \dots, p-1\}$.

Commitment phase:
To commit to $m \in \{0, 1, \dots, p-1\}$, Alice randomly chooses $r \in \{0, 1, \dots, p-1\}$ and sends $c = g^r m \pmod p$ to Bob.
Opening phase:
To open her commitment, Alice sends r and m to Bob.

* **10.4.** Is it possible to build a bit commitment scheme which is both unconditionally hiding and binding (in case both party sees everything the other party sends)?

10.5. Show how to construct a bit commitment scheme from a cryptographically secure pseudo-random generator G. Discuss the binding and hiding properties of your resulting bit commitment scheme.

10.6. Let $n = pq$ be a modulus and let $y \in QNR(n)$. Consider the following bit commitment scheme with $commit(b, r) = y^b r^2 \pmod n$ where $r \in \mathbb{Z}_n^*$ and $b \in \{0, 1\}$. Is the proposed scheme

(1) binding (computationally or unconditionally)?

(2) hiding (computationally or unconditionally)?

10.7. Show how to utilize 1-out-of-2 oblivious transfer to implement a bit commitment protocol in which both parties can cheat only with probability lower than 2^{-64}.

10.8.

(a) Show how to implement the standard oblivious transfer using a 1-out-of-2 oblivious transfer.

(b) Show how to implement a 1-out-of-k oblivious transfer using multiple instances of a 1-out-of-2 oblivious transfer.

* **10.9.** Suppose Alice and Bob are separated and cannot communicate. Let them play the following game. Both of them receive a single bit input x and y respectively. Alice does not know Bob's input and Bob does not know Alice's input. Their goal is to produce single bit answers a and b respectively. They win the game if $a \oplus b = x \cdot y$.

(a) Show that if they use deterministic strategies (*i.e.* Alice chooses a based only on x and Bob chooses b based only on y), they cannot win the game with probability 1.

(b) *Random Access Code* is the following protocol. Let Alice own a random binary string $(a_1, a_2, \ldots, a_n)$, $a_i \in \{0, 1\}$ of length n. She is allowed to send to Bob a single bit message m. Bob randomly generates a number $j \in \{1, \ldots, n\}$. Then he applies a corresponding decoding function D_j to the received bit m. The protocol is successful if $D_j(m) = a_j$ for every $j \in \{1, \ldots, n\}$. Show that if Alice and Bob own a hypothetical device that allows them to win the game introduced above with probability 1, they can construct Random Access Code for $n = 2$.

10.10. Let Peggy and Victor play the following game. They have a very large paper full of small, randomly placed, letters digits and other symbols but there is only one digit 7. The goal is to find the number 7 sooner than the other player. After some time Peggy found 7 but Victor does not believe her. How can Peggy prove to Victor that she knows the position of the number 7 without revealing it. A non-cryptographic solution is acceptable.

* **10.11.** Let Peggy and Victor share an $n \times n$ Sudoku puzzle. How can Peggy prove to Victor that she has a solution to this puzzle while not giving away any information about the solution itself. A non-cryptographic solution is acceptable.

10.12. Does the 3-SAT problem have a zero-knowledge proof?

10.13. Let $n = pq$, where $p \equiv q \equiv 3 \pmod 4$ are large primes. Peggy needs to prove to Victor that she knows factors of n without revealing any information about the factors. She has developed the following protocol:

- Peggy and Victor perform the following actions 20 times.
  1. Victor randomly chooses an integer $x < n$, computes $y = x^2 \mod n$ and sends y to Peggy.
  2. Peggy computes all four square roots of $y \mod n$, randomly chooses one of them, let us denote it r, and sends r to Victor.
  3. Victor verifies whether $r^2 \equiv y \pmod n$.
- Victor accepts if and only if all verifications have been successful.

Find out whether the protocol is a zero-knowledge proof.

* **10.14.**

For given two non-isomorphic graphs G_1, G_2 of n vertices, Peggy tries to convince Victor that $G_1 \not\cong G_2$. Suppose she has an efficient way of distinguishing non-isomorphic graphs and she does not want to reveal him any other information beyond the fact that graphs are not isomorphic.

Is the following protocol zero-knowledge if both verifier and prover are honest - that is they fully follow the protocol? Does an unhonest verifier have a chance to get some additional knowledge? If he does, how to modify the protocol that this is not possible?

(a) Victor chooses randomly an integer $i \in \{1, 2\}$ and a permutation π of $\{1, \dots, n\}$. Victor then computes the image H of G_i under the permutation π and sends H to Peggy.

(b) Peggy determines the value j such that G_j is isomorphic to H, and sends j to Victor.

(c) Victor checks to see if $i = j$.

(d) The steps (a)–(c) are repeated until Victor is convinced.

* **10.15.** Suppose that a group of n participants wants anonymously, to find out whether they all agree with a given specific statement. If all participants agree, the result will be seen as "yes". If any participant disagrees, the result will be seen as "no". Consider the following protocol that solves the problem stated above.
Let G be a finite cyclic group of prime order q in which finding discrete logarithms is intractable. Let g be the generator of G. Let us have n participants, and they all agree on (G, g). Each participant P_i selects a random secret value $x_i \in \mathbb{Z}_q$.

(1) Each participant P_i broadcasts g^{x_i} and gives a zero-knowledge proof to all other participants for x_i (*i.e.* provides a zero-knowledge proof that P_i really knows the discrete logarithm of g^{x_i} modulo q).

(2) Each participant P_i computes

$$\prod_{j=1}^{i-1} g^{x_j} \bigg/ \prod_{j=i+1}^{n} g^{x_j} \ .$$

The above value is g_{y_i} for some y_i.

(3) Each participant P_i makes public $g^{c_i x_i}$ where $c_i = x_i$ if P_i wants to send 0 and c_i is a random value if P_i wants to send 1. P_i provides a zero-knowledge proof to all other participants for the exponent c_i.

(a) Prove that $\sum_i x_i y_i = 0$.

(b) Show how the result – "yes" or "no" – can be recovered.

(c) Show how the dining cryptographers problem can be solved with the above protocol.
(Three cryptographers gather around a table for dinner. The waiter informs them that the meal has been paid by someone, who could be one of the cryptographers or the National Security Agency (NSA). The cryptographers respect each other's right to make an anonymous payment, but want to find out whether the NSA paid dinner.)

10.3 Solutions

10.1. The probability of correctly predicting k coin flips is given as 2^{-k}. The probability of making a mistake when predicting is hence $p = 1 - 2^{-k}$. If someone wants to be $n\%$ sure about your predicting ability, you have to perform at least k flips so that $\frac{n}{100} = 1 - 2^{-k}$ holds. Expressing this equation in terms of k and performing ceiling (so that number of flips is integer) we get $k = \lceil -\log_2(1 - \frac{n}{100}) \rceil$.

10.2. First recall that a Blum integer is of form $n = pq$, where p and q are Blum primes (*i.e.* $p \equiv q \equiv 3 \mod 4$). Blum integers have a special property – if a is a quadratic residue modulo n (where n is a Blum integer), it has exactly four square roots, out of which exactly one is a quadratic residue modulo n and three are quadratic non-residues.

In this light it is easy to see that in the protocol the requirement of n being a Blum integer is crucial. Otherwise, x_1 might have two square roots which are also quadratic residues, resulting in two different numbers x and y, such that $x^4 \equiv y^4 \equiv x_1 \mod n$. If $y_0 \equiv y^2 \mod n$ has different parity than x_0, Alice could cheat.

10.3. The commitment scheme is not secure because it is not binding. Indeed, once Alice has committed to m, it is possible for her to change her choice by replacing the value m with some m' without being detected by Bob. Recall that g is a generator of the group $\mathbb{Z}_p^*$ and so $m' \equiv g^i \pmod p$, for some $i \in \{0, 1, \ldots, p-1\}$, and also

$$c \equiv g^r m \equiv g^j \equiv g^{r'} g^i \equiv g^{r'} m' \pmod p$$

for appropriate $j, r' \in \{0, 1, \ldots, p-1\}$, $j \equiv r' + i \pmod{p-1}$. When the opening of commitment is required, Alice can simply send r' and m' instead of her previously chosen r and m, which shows that the commitment scheme is not binding.

10.4. No, it is not possible. Suppose such a bit commitment scheme exists. Then, when Alice sends a commitment to 0 as $B = \mathit{commit}(0, x)$ for some $x \in X$, there must exist an x', such that $B = \mathit{commit}(1, x')$. If not, Bob could easily conclude that the committed value could not be 1, violating the unconditional hiding property. But then, if Alice has unlimited computing power, she can find x' and change her mind from 0 to 1, violating the unconditional binding property.

10.5. Suppose that G produces for any n bit pseudorandom seed a pseudorandom $3n$-bit long output. We can design the following bit commitment scheme in which Alice commits herself to a bit b:

(1) Bob sends to Alice a random binary vector v of length $3n$.

(2) Alice chooses a random binary vector u of length n and computes $G(u)$.

(3) If $b = 0$, Alice sends $G(u)$ to Bob. If $b = 1$, Alice sends $G(u) \oplus v$ to Bob.

(4) In the opening phase Alice sends u to Bob.

(5) Bob can then compute $G(u)$ and check whether he received $G(u)$ or $G(u) \oplus v$ during the commitment phase.

The protocol is statistically binding – Alice cannot cheat with probability higher than $\frac{1}{2^n}$ because in order to cheat she would have to find such u' that $G(u') = G(u) \oplus v$. However $G(u)$ and $G(u')$ each produces 2^n values (together 2^{2n}) but v is picked from 2^{3n} possible values which are not chosen by Alice. Therefore there is only $\frac{2^{2n}}{2^{3n}} = 2^{-n}$ probability that Alice finds u' satisfying the required relation.
Protocol is hiding as Bob is unable to distinguish between outcomes of G and true randomness as G is cryptographically secure.

10.6.

(a) The proposed scheme is unconditionally binding because y is a quadratic non-residue modulo n, therefore there does not exist r' such that $yr'^2 = r^2 \pmod n$.

(b) The proposed scheme is computationally hiding because in order to retrieve b from $\mathit{commit}(b, r)$ one would need to compute quadratic residues which is believed to be computationally infeasible. With unlimited computational power, it would be easy to check whether $\mathit{commit}(b, r)$ is a quadratic residue (then $b = 0$) or not (then $b = 1$). Therefore, the proposed scheme cannot be unconditionally binding.

10.7.
Commitment phase:

(i) Alice chooses her commitment bit b and 64 random bits $r_1, \dots, r_{64}$.

(ii) Bob chooses 64 random bits $c_1, \dots, c_{64}$.

(iii) For $i \in \{1, \dots, 64\}$ the following steps are done:

  (1) Alice gives $y_{i,0} = r_i$ and $y_{i,1} = r_i \oplus b$ as the inputs into the oblivious transfer.

  (2) Bob gives c_i as the input into the oblivious transfer and receives $x_i = y_{i,c_i}$.

Opening phase:
Alice sends b and $r_1, \dots, r_{64}$ to Bob.
Bob checks if $x_i = r_i \oplus bc_i$ for every $i \in \{1, \dots, 64\}$.

Would Alice want to change her commitment to $\neg b$, she would have to find such r'_i that $r'_i \oplus \neg bc_i = r_i \oplus bc_i$. This implies $r'_i = r_i \oplus c_i$. This means that r'_i cannot be computed without knowledge of c_i. In order to cheat successfully, Alice would have to guess c_i for every i, which can happen with probability $(\frac{1}{2})^{64} = 2^{-64}$.
Bob's only way to cheat is to reveal b prematurely, but he cannot do that without knowing $r_1, \dots, r_{64}$.

10.8.

(a) Given is a 1-out-of-2 oblivious transfer. Let x_0 and x_1 be Alice's input messages, c be Bob's input bit and x_c be Bob's output. The standard oblivious transfer can be implemented as follows. Let m be Alice's message and g a garbage message.

  (1) Alice randomly chooses a bit b.

  (2) If $b = 0$, Alice inputs $x_0 = m$ and $x_1 = g$. If $b = 1$, Alice inputs $x_0 = g$ and $x_1 = m$.

  (3) Bob chooses a bit c and uses it as his input.

  (4) Alice sends b to Bob.

  (5) Bob obtains m if $b = c$.

  If $b = c$, $x_c = x_b = m$, otherwise $x_c = x_{\neg b} = g$. Bob does not know which c he should use to get m, hence he obtains m with probability $\frac{1}{2}$. Alice has no idea whether Bob gets m or g.

(b) We can assume without loss of generality that $k = 2^n$ for some $n \in \mathbb{N}$ (if not, we can add garbage messages $x_{k+1}, x_{k+2}, \dots, x_{2^n}$ to the original messages $x_1, \dots, x_k$. Alice uses an instance of 1-out-of-2 oblivious transfer on each pair of messages $(x_1, x_2), (x_3, x_4), \dots, (x_{2^{n-1}}, x_{2^n})$, thus receiving 2^{n-1} messages $x_{1,1}, x_{2,1}, \dots, x_{2^{n-1},1}$. Then in every other step, she will use the 2^l current messages $(x_{1,n-l}, x_{2,n-l}, \dots, (x_{2^{l-1},n-l}, x_{2^l,n-l})$ as inputs for another 2^{l-1} instances of 1-out-2 oblivious transfer, thus receiving 2^{l-1} new messages. She repeats this process until $l = 0$ and she has only one message left. She sends this final message to Bob, who will receive exactly one of the messages $x_{1,n}, x_{2,n}$, but Alice will not know which one. But that message itself provides Bob with another choice of one message out of two, and so on, until he will finally receives one of the original messages x_j.

10.9.

(a) Let a_x and b_y be answers of Alice and Bob respectively, when the inputs are x and y. Then we require

$$\begin{aligned} a_0 \oplus b_0 &= 0 \\ a_0 \oplus b_1 &= 0 \\ a_1 \oplus b_0 &= 0 \\ a_1 \oplus b_1 &= 1 \end{aligned}$$

Summing them together we get $0 = 1$ which is clearly a contradiction.

(b) Alice inputs $a_0 \oplus a_1$ into the proposed device (usually called a nonlocal or PR-box), receives A and sends $m = A \oplus a_0$. Suppose Bob inputs j into the device and he obtains B. We show that the correct answer is $B \oplus m = B \oplus A \oplus a_0$.
If Bob wants to recover a_0, his input into the device is 0. Since $A \oplus B = (a_0 \oplus a_1) \cdot 0$, we have that $A \oplus B = 0$ and therefore $B \oplus m = a_0$ as required.
If Bob wants to recover a_1, he inputs 1 into the device. Then we have $A \oplus B = (a_0 \oplus a_1) \cdot 1 = a_0 \oplus a_1$ and $B \oplus m = a_0 \oplus a_1 \oplus a_0 = a_1$ as required.
Actually, 1-out-of-2 oblivious transfer is realized with such device. Difference between random access codes and oblivious transfers is that in the latter is required that Bob cannot learn anything about other input bits whereas the former does not have this requirement.

10.10. Non-cryptographic solution goes as follows. Peggy covers the whole paper with even larger piece of paper which is at least double in the width and the height of the original paper, with the small hole in the middle. This hole is only as large as the digit 7. To prove she knows the position of the 7, Peggy moves the cover paper so that the hole is revealing only the number 7 and nothing else is visible from the underlying paper. After that Victor is convinced that Peggy knows the position but he himself has no information about this position.

10.11.Non-cryptographic solution using paper and scissors goes as follows.

(1) Peggy has a sheet of paper on which the puzzle is printed. She then writes down, for every cell with a filled-in value, this filled-in value on the back side of the cell, right behind the printed filled-in value. (The result is that filled-in cells, and only them, have their values written on both sides of the page. Without this step, Peggy can cheat and send the solution to different puzzle.)

(2) Peggy writes down the solution to the puzzle on the printed puzzle keeping this side of the page hidden from Victor.

(3) Victor checks that Peggy wrote the right values on the back of the puzzle.

(4) Victor chooses one out of the following options: rows/columns/subgrids.

(5) Suppose that Victor choice is "rows". Peggy then cuts the puzzle and separates it into n rows. If his choice is "columns", Peggy separates the columns from each other, similarly for subgrids. Peggy then cuts each row/column/subgrid (according to Victor's choice) to separate it into n cells. Peggy shuffles the cells of each row/column/subgrid (separately from the cells of other rows/columns/subgrids) and then hands them to Victor.

(6) Victor checks that

   (i) each row contains all n values,

(ii) in each row the cells whose value is written on both sides agree with the filled-in values of that row in the puzzle, and

(iii) these cells have the same value written on both their sides.

Cryptographic solution:

(1) Peggy chooses a random permutation $\sigma : \{1, \ldots, n\} \mapsto \{1, \ldots, n\}$.

(2) For each cell (i, j) with the value v, Peggy sends to Victor a commitment for the value $\sigma(v)$.

(3) Victor chooses at random one of the following $3n+1$ possibilities: a row, column or subgrid ($3n$ possibilities), or "filled-in cells", and asks the prover to open the corresponding commitments. After the prover responds, in case the verifier chose a row, column or subgrid, the verifier checks that all values are indeed different. In case the verifier chose the filled-in cells option, it checks that cells that originally had the same value still have the same value (although the original value may be different than the committed one), and that cells with different values are still different, i.e. that σ is indeed a permutation over the values in the filled-in cells.

10.12. Under the assumption that there exists a statistically binding and computationally hiding bit commitment scheme, there exists a zero-knowledge proof for any NP language (Goldreich, Micali, Wigderson, 1991). As 3-SAT$\in$ NP, there exists a zero-knowledge proof for 3-SAT.

10.13. No, the proposed protocol is not zero-knowledge.
Indeed, if the congruence $r^2 \equiv y \pmod{n}$ in the step (iii) holds, but r is not congruent to $\pm x$, then Victor knows that x and r are two different square roots of y. With this knowledge, he can factor n to reveal p and q.
If $r \not\equiv \pm x \pmod{n}$, Victor can get factors of n because the following holds:

$$r^2 \equiv x^2 \pmod{n} \Rightarrow r^2 - x^2 \equiv 0 \pmod{n} \Rightarrow (r-x)(r+x) \equiv 0 \pmod{n}$$

By computing $\gcd(r - x, n)$ a factor of n is obtained.
There are four square roots of y and two of them are different from x and $-x$. This means the probability of factoring is $\frac{1}{2}$ in each iteration, so Victor can reveal p and q with probability $1-(\frac{1}{2})^{20} =$ 99.9999% after 20 rounds.

10.14. This is not a zero knowledge protocol. Suppose Victor has a graph H and wants to know that they if $H \cong G_1$ or $H \cong G_2$. Using the proposed protocol, Victor simply sends H to Peggy and from the answer Victor will learn that $H \cong G_j$, or that H is isomorphic to neither G_1 or G_2 (if Peggy happens to abort). The problem with this is to ensure that the verifier does indeed know in advance what the prover will say. To do it in a correct way Peggy have to ask Victor to prove that H is indeed isomorphic to either G_1 or G_2.

10.15. The presented scheme is so-called *anonymous veto* network (Hao, Piotr Zieliński, 2006).

(a) We have $y_i = \sum_{j<i} x_j - \sum_{j>i} x_j$.

$$\sum_i x_i y_i = \sum_i \sum_{j<i} x_i x_j - \sum_i \sum_{j>i} x_i x_j = \sum\sum_{j<i} x_i x_j - \sum\sum_{i<j} x_i x_j$$
$$= \sum\sum_{j<i} x_i x_j - \sum\sum_{j<i} x_j x_i = 0$$

(b) Each participant computes $\prod_i g^{c_i y_i}$. If all participants sent "yes", each of them computes $\prod_i g^{c_i y_i} = \prod_i g^{x_i y_i} = 1$ because $\sum_i x_i y_i = 0$, $\prod_i g^{x_i y_i} = g^{\sum_i x_i y_i} = 1$. If one or more sent "no", $\prod_i g^{c_i y_i} \neq 1$.

(c) Cryptographers who did not pay for the dinner send the "yes" message, the cryptographer who paid the dinner, if there is one, sends the "no" message.

Appendix A

A.1 Introduction

In this appendix some needed notations, concepts and results from the discrete mathematics, algebra and number theory, as well as from the probability are briefly introduced.

A.2 Notation

1. Logarithms.

   - $\log_b a$ - logarithm of a at the base b.
   - $\log n$ - logarithm at the base 10 – decimal logarithm.
   - $\lg n$ - logarithm at the base 2 – binary logarithm
   - $\ln n$ - logarithm at the base e – natural logarithm

A.3 Central concepts and principles of modern cryptography

1. Efficiency and feasibility.

   - **Efficient (feasible) computation** is usually modeled by computations that are polynomial-time in an input (security) parameter.
   - **Efficient (computational) indistinguishability**. We say that probability ensembles $X = \{X_\alpha\}_{\alpha \in S}$ and $Y = \{Y_\alpha\}_{\alpha \in S}$ are computationally indistinguishable if for every family of polynomial-size circuits $\{D_n\}$, every polynomial p, all sufficiently large n and every $\alpha \in \{0,1\}^n \cap S$,
   $$|Pr[D_n(X_\alpha) = 1] - Pr[D_n(Y_\alpha) = 1]| < \frac{1}{p(n)}$$
   where the probabilities are taken over the relevant distribution (*ie.*, either X_n or Y_n).

2. Principles.

   - **Kerkhoffs principle:** The security of a cryptosystem must not depend on keeping secret the encryption algorthm. The security should depend only on keeping secret the key.
   - **Murphys laws:** If there is a single security hole in a cryptosystem, the exposure of a cryptosystem will make sure that someone will eventually find it. Even if this person is honest the discovery may ultimately leak to malicious parties.

A.4 Groups

A *Group* is a set of elements and an operation, denote it $\circ$, with the following properties:

- G **is closed under the operation** $\circ$; that is if $a, b \in G$, so is $a \circ b$.
- The **operation** $\circ$ **is associative**; that is $a \circ (b \circ c) = (a \circ b) \circ c$, for any $a, b, c \in G$.
- G **has an identity element** e such that $e \circ a = a \circ e = a$ for any $a \in G$.
- **Every element** $a \in G$ **has an inverse element** $a^{-1} \in G$, so that $a \circ a^{-1} = a^{-1} \circ a = e$.

A group G is said to be an *Abelian group* if the operation $\circ$ is commutative (that is $a \circ b = b \circ a$ for any $a, b \in G$).

A.4.1 Groups $\mathbb{Z}_n$ and $\mathbb{Z}_n^*$

Two integers a, b are congruent modulo n if $a \bmod n = b \bmod n$. **Notation**: $a \equiv b (\text{mod} n)$

Let $+_n, \times_n$ denote addition and multiplication modulo n, that is $a +_n b = (a + b) \bmod n$ and $a \times_n b = (ab) \bmod n$.

$\mathbb{Z}_n = \{0, 1, \ldots, n-1\}$ is a group under the operation $+_n$. $\mathbb{Z}_n^*$ is a multiplicative group $\pmod n$, *ie.* the set of all elements invertible under the operation $\times_n$, together with operation $\times_n$. Note that if n is a prime $\mathbb{Z}_n^*$ contains all elements $1 \leq i \leq n$, and set $\{0, 1, \ldots, n-1\}$ is a field under the operations $+_n, \times_n$.

For any n in the group $\mathbb{Z}_n^*$ computation of any inverse and also exponentiation can be done in polynomial time.

A.4.2 Order of the group

- If a is an element of a finite group G, then its **order** is the smallest integer k such that $a^k = e$.
- Order of each element of a group G is a divisor of the number of elements of G. This implies that every element $a \in \mathbb{Z}_p^*$, where p is a prime, has order dividing $p - 1$ and it holds $a^{p-1} \equiv 1 \pmod p$.

A.4.3 Properties of the group $\mathbb{Z}_n^*$

The group $(G, \circ)$ is called cyclic if it contains an element g, called the *generator*, such that the order of $g = |G|$.

Theorem. If the multiplicative group $(\mathbb{Z}_n^*, \times_n)$ is cyclic, then it is isomorphic to the additive group $(\mathbb{Z}_{\Phi(n)}, +_{\Phi(n)})$. (However, no effective way is known, given n, to create such an isomorphism!)

Theorem. The mutliplicative group $(\mathbb{Z}_n^*, \times_n)$ is cyclic iff n is either $1, 2, 4, p^k$ or $2p^k$ for some $k \in N^+$ and an odd prime $p > 2$.

Theorem. Let p be a prime. Given the prime factorization of $p - 1$ a generator for group $(\mathbb{Z}_p^*, \times_p)$ can be found in polynomial time by a randomized algorithm.

A.5 Rings and fields

A **ring** R is a set with two operations $+$ (addition) and $\circ$ (multiplication), satisfying the following properties:

- R is closed under $+$ and $\circ$.
- R is an Abelian group under $+$ (with the unity element for addition called **zero**).
- The associative law for multiplication holds.
- R has an identity element 1 for multiplication
- The distributive law holds $(a \circ (b + c) = a \circ b + a \circ c$ for all $a, b, c \in R$.

A ring is said to be a *commutative ring* if multiplication is commutative

A **field** F is a set with two operations $+$ (addition) and $\circ$ (multiplication), with the following properties:

- F is a commutative ring.
- Non-zero elements of F form an Abelian group with respect to multiplication.

A non-zero element g is a **primitive element** of a field F if all non-zero elements of F are powers of g.

A.5.1 Finite fields

Finite field are very well understood.

Theorem. If p is a prime, then the the set of all integers smaller than p, $GF(p)$, constitute a field. Every finite field F contains a subfield that is $GF(p)$, up to relabeling, for some prime p and $p \cdot \alpha = 0$ for every $\alpha \in F$.

If a field F contains a prime field $GF(p)$, then p is called the em characteristic of F.

Theorem.
(1) Every finite field F has p^m elements for some prime p and some m.
(2) For any prime p and any integer m there is a unique (up to isomorphism) field of p^m elements $GF(p^m)$.
(3) If $f(x)$ is an irreducible polynomial of degree m in $F_p[x]$, then the set of polynomials in $F_p[x]$ with additions and multiplications modulo $f(x)$ is a field with p^m elements.

A.6 Arithmetics

A.6.1 Ceiling and floor functions

Floor	$\lfloor x \rfloor$ – the largest integer $\leq x$
Ceiling	$\lceil x \rceil$ – the smallest integer $\geq x$

Example $\lfloor 3.14 \rfloor = 3 = \lfloor 3.75 \rfloor$, $\lfloor -3.14 \rfloor = -4 = \lfloor -3.75 \rfloor$;
$\lceil 3.14 \rceil = 4 = \lceil 3.75 \rceil$, $\lceil -3.14 \rceil = -3 = \lceil -3.75 \rceil$

A.6.2 Modulo operations

The remainder of n when divided by m is defined by

$$n \mod m = \begin{cases} n - m\left\lfloor\frac{n}{m}\right\rfloor & m \neq 0 \\ 0 & m = 0 \end{cases}$$

Example: $7 \mod 5 = 2$ $\quad$ $122 \bmod 11 = 1$

Identities:
- $(a+b) \mod n = ((a \mod n) + (b \mod n)) \mod n$
- $(a \cdot b) \mod n = ((a \mod n) \cdot (b \mod n)) \mod n$
- $a^b \bmod n = ((a \bmod n)^b) \bmod n$.

A.6.3 Exponentiation

Exponentiation (modular) plays the key role in many cryptosystems. If

$$n = \sum_{i=0}^{k-1} b_i 2^i, \quad b_i \in \{0,1\}$$

then

$$e = a^n = a^{\sum_{i=0}^{k-1} b_i 2^i} = \prod_{i=0}^{k-1} a^{b_i 2^i} = \prod_{i=0}^{k-1} (a^{2^i})^{b_i}$$

The above decomposition of n induces the following **Algorithm for exponentiation**

```
begin $e \leftarrow 1$; $p \leftarrow a$;
    for $i \leftarrow 0$ to $k-1$
        do if $b_i = 1$ then $e \leftarrow e \cdot p$;
            $p \leftarrow p \cdot p$
        od
end
```

Modular exponentiation: $a^n \mod m = ((a \bmod m)^n) \bmod m$

Modular multiplication: $ab \mod n = ((a \mod n)(b \mod n) \mod n)$

Examples: $3^{1000} \bmod 19 = 16$; $3^{10000} \bmod 13 = 3$; $3^{340} \bmod 11 = 1$ - $3^{100} \bmod 79 = 51$

A.6.4 Euclid algorithm for GCD - I.

This is algorithm to compute greatest common divisor (gcd) of two integers, in short

to compute $gcd(m,n), 0 \leq m < n$

Euclid algorithm

$$\gcd(0,n) = n \tag{A.1}$$

$$\gcd(m,n) = \gcd(n \mod m, m) \text{ for } m > 0 \tag{A.2}$$

Example $gcd(296,555) = gcd(259,296) = gcd(37,259) = gcd(0,37) = 37$

Theorem $T(n) = \mathcal{O}(\log n)$ for the number of steps of Euclid's algorithm to compute gcd(m,n) for $0 \leq m \leq n$.

A.6.5 Extended Euclid algorithm

Theorem For all $0 < m < n$ there exist integers x and y (that can be computed in polynomial time) such that

$$\gcd(m, n) = xm + yn.$$

this means that if gcd(m,n)=1, then $x = m^{-1} \bmod n$.

An extention of Euclid's algorithm, which computes x and y together with $\gcd(m, n)$ is sometimes referred to as **extended Euclid algorithm**.

A.7 Basics of the number theory

The number theory concepts, methods and results introduced in the following play an important role in modern considerations concerning cryptography, cryptographic protocols and randomness.

The key concepts are that of primality and randomness.

A.7.1 Primes

Primes play key role in the modern cryptography.

A positive integer $p > 1$ is called **prime** if it has just two divisors: 1 and p.

Fundamental theorem of arithmetic: Each integer n has a unique decomposition

$$n = \prod_{i=1}^{k} p_i^{e_i}$$

where $p_i < p_{i+1}$ are primes and e_i are integers.

Basic question no. 1 How many primes $\Pi(n)$ are there among the first n integers?

Prime number theorem.

$$\Pi(n) = \frac{n}{\ln n} + \frac{n}{(\ln n)^2} + \frac{2!n}{(\ln n)^3} + \frac{3!n}{(\ln n)^4} + \Theta\left(\frac{n}{(\ln n)^6}\right)$$

Gauss estimation: $\Pi(n) \doteq \frac{n}{\ln n}$.

Basic question no. 2 How (difficult is it) to determine whether a given integer is a prime?

- Only in 2002 it has been shown that there is a ($O(m^{12})$) deterministic algorithm to recognize whether an m bit integer is a prime.
- There are (very) simple randomized algorithm to decide fast and with large probability correctly whether a given integer is a prime.

A.7.2 Chinese remainder theorem

Theorem Let $m_1, \ldots, m_t$ be integers, $\gcd(m_i, m_j) = 1$ if $i \neq j$ and $a_1, \ldots, a_t$ be integers, $0 < a_i < m_i, 1 \leq i \leq t$. Then the system of congruences

$$x \equiv a_i \pmod{m_i}, 1 \leq i \leq t$$

has the solution

$$x = \sum_{i=1}^{t} a_i M_i N_i \tag{$*$}$$

where

$$M = \prod_{i=1}^{t} m_i, M_i = \frac{M}{m_i}, N_i = M_i^{-1} \bmod m_i$$

and the solution $(*)$ is unique up to the congruence modulo M.

Comment: Each integer $0 < x < M$ is uniquelly represented by t-tuple: $x \mod m_1, \ldots, x \mod m_t$. **For example,** if $m_1 = 2, m_2 = 3, m_3 = 5$, then $(1,0,2)$ represents 27. **Advantage:** With such a modular representation addition, substraction and multiplication can be done componentwise in parallel time.

A.7.3 Euler totient function

$$\Phi(n) = |\mathbb{Z}_n^*| = |\{m | 1 \le m \le n, \gcd(m,n) = 1\}|$$

has the following **properties:**

- $\Phi(1) = 1$
- $\Phi(p) = p - 1$, if p is a prime;
- $\Phi(p^k) = p^{k-1}(p-1)$, if p is prime, $k > 0$;
- $\Phi(nm) = \Phi(n)\Phi(m)$, if $\gcd(m,n) = 1$;

Theorem Computation of $\Phi(n)$ and factorization of n are computationally polynomially related problems.

A.7.4 Euler and Fermat theorems

Theorem (Euler's Totient Theorem)

$$n^{\Phi(m)} \equiv 1 \ (\text{mod} m)$$

if $n < m, \gcd(m,n) = 1$

Corollary $n^{-1} \equiv n^{\Phi(m)-1}$ (mod m) if $n < m, \gcd(m,n) = 1$

Theorem (Fermat's Little Theorem)

$$a^p \equiv a \ (\text{mod} p)$$

if p is prime.

A.7.5 Discrete logarithms and square roots

Three problems are related with the equation $y = x^a$ (mod$n$).

Exponentiation problem: Given x, a, n, compute y. The problem is easy: it can be computed in polynomial time, even its modular version

Discrete logarithm problem: Given x, y, n, compute a. the problem is computationally very hard. Indeed, it is believed that the discrete logarithm problem is **NP**-hard even in the average case. (A formal proof of it would imply that exponentiation is a one-way function.)

Root finding problem: Given y, a, n, compute x. It is also very hard, even in in the following spcial case:

Square root finding problem Given $y, a = 2, n$, compute x: This problem is in general as hard as factorization.

However, the square root finding can be done by a randomized polynomial time algorithm if n is a prime or the prime decomposition of n is know.

Examples
$\{x \mid \sqrt{x}\ (\mathbf{mod}\ 15) = 1\} = \{1, 4, 11, 14\}$
$\{x \mid \sqrt{x}\ (\mathbf{mod}\ 15) = 3\} = \emptyset$
$\{x \mid \sqrt{x}\ (\mathbf{mod}\ 15) = 4\} = \{2, 7, 8, 13\}$

One of basic questions: How many square roots exist?

Theorem
(1) If $p > 2$ is a prime, $k \geq 1$, then any quadratic residue modulo p^k has exactly two distinct square roots $x, -x = p^k - x$

(2) If $p = 2$, $k \geq 1$, then any quadratic residue modulo 2^k has
- 1 square root if $k = 1$;
- 2 square root if $k = 2$;
- 4 square root if $k > 2$.

Theorem If an odd number n has exactly t distinct factors, then any quadratic residue a modulo n has exactly 2^t distinct square roots.

A.7.6 Quadratic residues and non-residues

An integer $x \in \mathbb{Z}_m^*$ is called a quadratic residue modulo m if

$$x \equiv y^2 (\text{mod } m)$$

for some $y \in \mathbb{Z}_m^*$, otherwise x is a quadratic non-residue.

Notation:

- QR_m denotes the set of all quadratic residues modulo m. QR_m is therefore subgroup of squares in $\mathbb{Z}_m$.
- QNR_m denotes the set of all quadratic non-residues modulo m.

One of basic questions concerning quadratic residues: How to decide whether an x is a quadratic residue?

Theorem. If $p > 2$ is a prime and g is a generator of $\mathbb{Z}_p^*$, then g^k is a quadratic residue iff k is even.

Theorem If p is a prime, then $a \in \mathbb{Z}_p^*$ is a quadratic residue iff

$$a^{\frac{p-1}{2}} \equiv 1 \pmod{p}.$$

For any prime p the set QR_p has $\frac{p-1}{2}$ elements.

Theorem – Euler criterion Let $p > 2$ be a prime. Then x is a quadratic residue modulo p if and only if

$$x^{(p-1)/2} \equiv 1 \pmod{p}.$$

A.7.7 Blum integers

If p, q are primes such that $p \equiv 3 \pmod 4$, $q \equiv 3 \pmod 4$ then the integer $n = pq$ is called **Blum integer**. Blum integers n have the following important properties.

- If $x \in QR_n$, then x has exactly four square roots and exactly one of them is in QR_n – this square root is called **primitive square root** of x modulo n.
- Function $f : QR_n \to QR_n$ defined by $f(x) = x^2$ is a permutation on QR_n.
- The inverse function is $f^{-1}(x) = x^{((p-1)(q-1)+4)/8} \bmod n$

Bibliography

[1] Jozef Gruska. Lecture notes for IV054: Coding, Cryptography and Cryptographic Protocols.

[2] Raymond Hill. *A First Course in Coding Theory.* Oxford Applied Linguistics. Clarendon Press, 1986.

[3] Vera Pless. *Introduction to the Theory of Error-correcting Codes.* A Wiley-Interscience publication. John Wiley, 1998.

[4] Jozef Gruska. *Foundations of Computing.* International Thomson Computer Press, Boston, MA, USA, 1997.

[5] Jozef Gruska. *Quantum Computing.* Advanced topics in computer science series. McGraw-Hill, 1999.

[6] Arto Salomaa. *Public-Key Cryptography.* Texts in Theoretical Computer Science. An EATCS Series. Springer Berlin Heidelberg, 1996.

[7] Douglas R. Stinson. *Cryptography Theory and Practice.* CRC Press, Inc., 1995.

[8] Wade Trappe and Lawrence C. Washington. *Introduction to Cryptography with Coding Theory.* Pearson Educational international, Prentice Hall, 2006.

[9] Bruce Schneier. *Applied Cryptography.* John Wiley & Sons, Inc., 2 edition, 1996.

[10] Serge Vaudenay. *A Classical Introduction to Cryptography: Applications for Communications Security.* Springer US, 2005.

[11] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography.* CRC Press, Inc., 1997.

[12] Oded Goldreich. *Foundations of Cryptography: Basic Tools.* Cambridge University Press, New York, NY, USA, 2000.

[13] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs That Yield Nothing but Their Validity or All Languages in Np Have Zero-knowledge Proof Systems. *ACM*, 38(3):690–728, July 1991.

[14] Feng Hao and Piotr Zieliński. A 2-round Anonymous Veto Protocol. In *the 14th International Workshop on Security Protocols*, 2006.

[15] Zuzana Kuklová. Coding Theory, Cryptography and Cryptographic Protocols - Exercises with Solutions (Bachelor Thesis), 2007.

[16] Lukáš Boháč. Quantum Authentication and Signature Protocols (Master Thesis), 2004.