

IV054: Coding, Cryptography and Cryptographic Protocols

Exercise Book II.

Jozef Gruska, Lukáš Boháč, Libor Čaha, Luděk Matyska, Matej Pivluska

Faculty of Informatics, Masaryk University, Brno

December 2019

Preface

This is an extension and continuation of the lecture notes, entitled “IV054: Coding, Cryptography and Cryptographic Protocols, Exercise Book” of co-authors Jozef Gruska, Lukáš Boháč, Luděk Matyska, and Matej Pivoluska, that was put into a general use in FI MU in 2016, especially for students of the course IV054 — Coding, cryptography and cryptography protocols.

The first part of this lecture notes contains new interesting and more illuminating exercises and their stimulating solutions for the first ten lectures of the course IV054 that are dealt with practically in all years the course is hold.

The second part of the lecture notes deals with short presentations of the last three chapters of the course that were not so far accompanying with homeworks and also several exercises for those subjects with their solutions.

In these three new chapters, emphasis are mainly on history of cryptography in a broad sense, on applications and also on the use of quantum phenomena to process information especially in quantum cryptography. This is especially booming area of cryptography and requires special knowledge and expertise. Basics of that are in the booklet presented.

Main goal of this lecture notes is again to help future students of Faculty of Informatics, especially those concentrating on a very attractive area of security, and in a special way to help students of the course IV054.

The collective of authors of this lecture notes consists of prof. Jozef Gruska, who has been given the course for many years and the rest of authors were taken from the best students of the course, especially in handling exercises in the recent past.

This lecture notes are again the product of the project MUNI/FR/1192/2018 of the Masaryk University and this support is to be acknowledged and much thanked.

The authors, December 2019

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	7
2	Linear Codes	17
2.1	Introduction	17
2.1.1	Basic properties	17
2.1.2	Encoding and decoding with linear codes	18
2.1.3	Hamming codes	18
2.2	Exercises	19
2.3	Solutions	23
3	Cyclic Codes	33
3.1	Introduction	33
3.1.1	Polynomials over $GF(q)$	33
3.1.2	Rings of polynomials	33
3.1.3	Algebraic characterization of cyclic codes	34
3.1.4	Check polynomials and parity check matrices for cyclic codes	34
3.2	Exercises	35
3.3	Solutions	38
4	Secret Key Cryptosystems	49
4.1	Introduction	49
4.1.1	Basics of the design and analysis of cryptosystems	49
4.1.2	Basic classical secret-key cryptosystems	50
4.1.3	Product cryptosystems	51
4.1.4	Perfect secrecy	51
4.1.5	Unicity distance	52
4.2	Exercises	52
4.3	Solutions	56
5	Public-Key Cryptography, I.	65
5.1	Introduction	65
5.1.1	Diffie-Hellman protocol	65
5.1.2	Blom's key pre-distribution protocol	65
5.1.3	Knapsack cryptosystem	66
5.1.4	McEliece cryptosystem	66
5.1.5	RSA cryptosystem	66
5.1.6	Rabin-Miller's prime recognition	67

5.2	Exercises	67
5.3	Solutions	70
6	Public-Key Cryptography, II.	78
6.1	Introduction	78
6.1.1	Rabin Cryptosystem	78
6.1.2	ElGamal cryptosystem	78
6.1.3	Shanks' algorithm for discrete logarithm	78
6.1.4	Perfect security of cryptosystems	79
6.1.5	Blum-Goldwasser cryptosystem	79
6.1.6	Hash functions	79
6.2	Exercises	80
6.3	Solutions	83
7	Digital Signatures	94
7.1	Introduction	94
7.1.1	Signature schemes – basic ideas and goals	94
7.1.2	Digital signature scheme – definition	94
7.1.3	Attacks	95
7.1.4	Examples	95
7.2	Exercises	97
7.3	Solutions	100
8	Elliptic Curve Cryptography	106
8.1	Introduction	106
8.1.1	Elliptic curves	106
8.1.2	Group structure and addition law	106
8.1.3	Elliptic curve cryptography	107
8.1.4	Factorization	107
8.2	Exercises	108
8.3	Solutions	111
9	Identification, Authentication and Secret Sharing	120
9.1	Introduction	120
9.1.1	User identification	120
9.1.2	Message authentication	121
9.1.3	Secret sharing	122
9.2	Exercises	123
9.3	Solutions	128
10	Coin Tossing, Bit commitment, Oblivious Transfer, Zero-knowledge Proofs and Other Crypto-protocols	136
10.1	Introduction	136
10.1.1	Coin-flipping protocols	136
10.1.2	Bit commitment protocols	136
10.1.3	Oblivious transfers	137
10.1.4	Interactive and zero-knowledge proofs	137
10.2	Exercises	138
10.3	Solutions	142

11 Steganography and Watermarking	150
11.1 Introduction	150
11.1.1 Steganography	150
11.1.2 Watermarking	151
11.1.3 Parameters of stego- and watermarking systems	151
11.1.4 Breaking cryptography, steganography, and watermarking systems	152
11.2 Exercises	152
11.3 Solutions	155
12 Quantum cryptography	160
12.1 Introduction	160
12.1.1 Basics of quantum mechanics	160
12.1.2 Quantum cryptography	161
12.2 Exercises	164
12.3 Solutions	166
13 From Theory to Practice in Cryptography	175
13.1 Introduction	175
13.1.1 Linear-feedback shift registers	175
13.1.2 Confusion and diffusion	175
13.1.3 Feistel encryption/decryption scheme	176
13.1.4 DES cryptosystem	176
13.1.5 Operational modes of DES	177
13.1.6 AES cryptosystem	177
13.1.7 Hash functions	178
13.2 Exercises	178
13.3 Solutions	180
A	186
A.1 Introduction	186
A.2 Notation	186
A.3 Central concepts and principles of modern cryptography	186
A.4 Groups	187
A.4.1 Groups \mathbb{Z}_n and \mathbb{Z}_n^*	187
A.4.2 Order of the group	187
A.4.3 Properties of the group \mathbb{Z}_n^*	187
A.5 Rings and fields	188
A.5.1 Finite fields	188
A.6 Arithmetics	188
A.6.1 Ceiling and floor functions	188
A.6.2 Modulo operations	189
A.6.3 Exponentiation	189
A.6.4 Euclid algorithm for GCD - I.	189
A.6.5 Extended Euclid algorithm	190
A.7 Basics of the number theory	190
A.7.1 Primes	190
A.7.2 Chinese Remainder Theorem (CRT)	190
A.7.3 Euler totient function	191
A.7.4 Euler and Fermat Theorems	191
A.7.5 Discrete logarithms and square roots	191
A.7.6 Quadratic residues and nonresidues	192
A.7.7 Blum integers	192

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:

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

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, \dots, x_n with probabilities $p_1 \geq \dots \geq p_n$. 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:

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

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:

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

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

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

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 $C = \{x \in \mathbb{Z}_{10}^9 \mid \sum_{i=1}^9 ix_i \equiv 0 \pmod{10}\}$. 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:

$x \mapsto x$	with probability $1 - p$
$x \mapsto x + 1 \pmod{5}$	with probability $p/4$
$x \mapsto x + 2 \pmod{5}$	with probability $p/4$
$x \mapsto x + 3 \pmod{5}$	with probability $p/4$
$x \mapsto x + 4 \pmod{5}$	with probability $p/4$.

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

012000011001230122302240144423333340023.

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 $q \geq 2$. 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 $A_q(n + 1, d)$;
- (d) $A_q(2n, 2)$ and $A_{2q}(n, 4)$;
- (e) $A_2(n, 2d)$ and $4A_2(n - 3, 2d - 1)$.

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 $(n, M, 7)$ -code. There are only two possible values of n . Find them.

1.12. Show that for $t > 0$ there is no perfect binary $(n, M, 2t + 1)$ -code, such that n is even.

1.13. 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.

* **1.14.** 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?

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

(a)

$$A = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix}, \quad B = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \end{pmatrix},$$

(b)

$$A = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \end{pmatrix}, \quad B = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix},$$

(c)

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

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

1.16.

(a) Give an example of a ternary $(6, 7, 4)$ -code, in which all words are palindromes (*i.e.* their i -th letter is equal to their $(7 - i)$ -th letter for $i \in \{1, 2, 3\}$).

(b) Give an example of four binary pairwise disjoint $(4, 4, 2)$ -codes.

1.17. Alice and Bob are communicating with each other. Since their communication channel is noisy, they use the following code:

$$\begin{aligned} 00 &\rightarrow 010010 \\ 01 &\rightarrow 000101 \\ 10 &\rightarrow 101000 \\ 11 &\rightarrow 111111 \end{aligned}$$

Bob receives the message 001101. Assume that the probability of a bit error is $p = 0.01$.

- What is Bob's decoding of the message?
- What is the probability that Bob's interpretation of the message is the same as Alice intended?
- What is the probability that Alice actually wanted to send the message 11?

1.18. Consider a channel characterized by the following conditional probabilities, where X and Y are the random variables of the input and output, respectively.

$$\begin{aligned} P(Y = 0|X = 0) &= 1 \\ P(Y = 0|X = 1) &= p \\ P(Y = 1|X = 0) &= 0 \\ P(Y = 1|X = 1) &= 1 - p, \end{aligned}$$

for some $0 < p < 1$.

- Calculate the probability of t errors in n received bits if the input distribution is $P(X = 0) = q$ and $P(X = 1) = 1 - q$, $0 \leq q \leq 1$, independently for every bit.
- For which q will this value be minimal?

1.19. Consider the binary code of length 12 defined as

$$\{x_1x_2 \cdots x_{12} \mid 3x_1 + x_2 + 3x_3 + x_4 + \dots + 3x_{11} + x_{12} \equiv 0 \pmod{10}\}.$$

Is it possible to detect all adjacent transposition errors with this code?

1.20. Give an example of a 4-ary $(10, 10, 7)$ code such that each of its words contains exactly one 0, two 1's, three 2's and four 3's.

* **1.21.** Consider a family of codes C_{2n} with the following encoding function:

$$\begin{aligned} 0 &\mapsto \overbrace{00 \dots 0}^{2n \text{ times}} \\ 1 &\mapsto \overbrace{11 \dots 1}^{2n \text{ times}} \end{aligned}$$

Consider a binary symmetric channel with error $p \leq \frac{1}{2}$ and the maximum likelihood decoding strategy, *i.e.* every $k < n$ errors can be corrected.

- What are the (n, M, d) parameters of C_{2n} ?
- What is the probability of correct decoding $P_{\text{corr}}(C_{2n})$?
- Calculate $\lim_{n \rightarrow \infty} P_{\text{corr}}(C_{2n})$.
- What is the code rate $R(C_{2n})$?

(e) Calculate $\lim_{n \rightarrow \infty} R(C_{2n})$.

Hint: $\binom{2n}{n} \leq \frac{4^n}{\sqrt{3n+1}}$.

1.22. Let C_1 and C_2 be two block codes of length n . Show that the following holds

(a) $h(C_1 \cap C_2) \geq \max\{h(C_1), h(C_2)\}$.

(b) $h(C_3) = h(C_1) + h(C_2)$, where

$$C_3 = \{x_1y_1x_2y_2 \dots x_ny_n \mid x_1 \dots x_n \in C_1, y_1 \dots y_n \in C_2\}.$$

1.23. Any two equivalent q -ary codes have the same (n, M, d) parameters. Are any two q -ary codes with the same (n, M, d) parameters equivalent? Prove your answer.

1.24. Consider an employee in a supermarket finding a product with corrupted EAN-13 barcode 858x035361404. Help him find the missing digit x .

1.25. You are about to pay a conference fee for a coding theory conference. The organizers decided to test your knowledge by sending you an incomplete bank account number CZ85 2018 0000 0012 3423 x091 written in the International Bank Account Number (IBAN) format. Determine the missing number.

1.26. Consider the following codes and calculate their (n, k, d) parameters.

(a) **m -fold repetition.** Each ℓ bit message is copied m times to create a codeword. Formally,

$$C_{\ell, m} = \{w^m \mid w \in \{0, 1\}^\ell\}.$$

(b) **Checksum of neighboring bit pairs.** Each $\ell \geq 3$ bit message is appended with XORs (denoted \oplus) of all *neighboring* bit pairs. Note that the last bit is XORed with the first one. Formally,

$$C_\ell = \{w_0 \dots w_{\ell-1} b_0 \dots, b_{\ell-1} \mid (w_0 \dots w_{\ell-1}) \in \{0, 1\}^\ell, b_i = w_i \oplus w_{(i+1) \bmod \ell}\}.$$

(c) **Checksum of all bit pairs.** Each $\ell \geq 3$ bit message is appended with XORs (denoted \oplus) of *all* bit pairs. Formally,

$$C_\ell = \{w_0 \dots w_{\ell-1} b_{0,1} \dots b_{i,j} \dots \mid (w_0 \dots w_{\ell-1}) \in \{0, 1\}^\ell, b_{i,j} = w_i \oplus w_j\}.$$

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)

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

(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

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

(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 :

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

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

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

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

Since we know that $\sum_{i=1}^9 ix_i \equiv 0 \pmod{10}$, in order to find a transposition error which cannot be corrected it suffices to find j, k, x_j, x_k , such that $(k-j)(x_j - x_k) \equiv 0 \pmod{10}$. 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.
- (d) In order to prove this, we need to construct q -ary $(2n, M, 4)$ -code from a given $2q$ -ary $(n, M, 4)$ -code. Let C_1 be the given $(n, M, 4)$ -code over Σ_1 , with $|\Sigma_1| = 2q$. Let us label the elements of Σ_1 as $a_1, b_1, \dots, a_q, b_q$. Now we want to construct a code C_2 over Σ_2 , with $|\Sigma_2| = q$. Let us label the elements of Σ_2 as c_1, \dots, c_q . Let us now define a function:

$$\begin{aligned}\varphi : \Sigma_1 &\mapsto \Sigma_2 \times \Sigma_2 \\ \varphi : a_i &\mapsto c_i c_i \\ \varphi : b_i &\mapsto c_i c_{i+1},\end{aligned}$$

where we use identity $q + 1 = 1$. By natural extension of φ from Σ_1 to strings over Σ_1 , we obtain the code $C_2 = \varphi(C_1) = \{\varphi(x) | x \in C_1\}$. Let $x, y \in C_1$. Then we have

$$x_i = y_i \iff \varphi(x)_{2i} = \varphi(y)_{2i} \wedge \varphi(x)_{2i+1} = \varphi(y)_{2i+1}.$$

Therefore we have that $\forall x, y \in C_1$

$$h(x, y) \leq h(\varphi(x), \varphi(y))$$

It immediately follows that

$$A_q(2n, 4) \geq A_{2q}(n, 4).$$

- (e) $A_2(n, 2d) \leq 4A_2(n - 3, 2d - 1)$. As shown in the lecture slides if the distance is odd then $A_2(n, d) = A_2(n + 1, d + 1)$, so we can rewrite $4A_2(n - 3, 2d - 1)$ as $4A_2(n - 2, 2d)$. Now we look at the left hand side and take the first two bits from each word and we divide the words in groups, based on what were the first 2 bits. We get 4 groups that have maximum of $A_2(n - 2, 2d)$ words and we are done. Notice that the hamming distance in each group is unchanged because all words from same group had the same first 2 bits.

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:

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

where the first equality follows from the binomial theorem.

- (c) Binary repetition code of 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.

We need to find an n , for which the sphere packing bound attains the equality:

$$M \left\{ \binom{n}{0} + \binom{n}{1} + \binom{n}{2} + \binom{n}{3} \right\} = 2^n$$

$$M(6 + 6n + 3n^2 - 3n + n^3 - 3n^2 + 2n) = 2^n$$

Solving the equation yields $M = \frac{6 \cdot 2^n}{n^3 + 5n + 6}$. We require M to be a natural number and this is fulfilled only for $n = 7$ and $n = 23$.

1.12.

A perfect binary $(n, M, 2t+1)$ -code is a code such that

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

$$= M \cdot \left[1 + n + \frac{n!}{(n-2)!2!} + \cdots + \frac{n!}{(n-t)!t!} \right]$$

$$= M \cdot \left[1 + n \left(1 + \frac{(n-1)!}{(n-2)!2!} + \cdots + \frac{(n-1)!}{(n-t)!t!} \right) \right].$$

Let us denote the term in the square brackets as N . We can see that N is odd, because n is even and we are adding 1 to a multiple of n . Therefore the equivalence cannot be achieved, as the left hand side of this equivalence is a power of 2 and doesn't have any odd divisors, unlike the right hand side of the equivalence.

1.13.

- (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.

1.14.

- (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 $\left\lfloor \frac{2(r-\lambda)-1}{2} \right\rfloor$.

1.15.

- (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 .

1.16.

- (a) An example of a ternary $(6, 7, 4)$ -code whose words are palindromes is

$$\{aaaaaa, abccba, acbbca, baccab, cabbac, bcaacb, cbaabc\}$$

- (b) An example of four binary pairwise disjoint $(4, 4, 2)$ -codes is:

$$C_1 = \{0000, 0011, 0110, 1010\}$$

$$C_2 = \{0001, 0111, 1011, 1101\}$$

$$C_3 = \{0010, 0100, 1000, 1110\}$$

$$C_4 = \{0101, 1001, 1100, 1111\}$$

1.17.

- (a) Hamming distances of the message from all of the codewords are:

$$\begin{aligned}h(001101, 010010) &= 5 \\h(001101, 000101) &= 1 \\h(001101, 101000) &= 3 \\h(001101, 111111) &= 3\end{aligned}$$

Therefore, by the principle of majority voting, Bob will interpret the message as 01.

- (b) Since $h(001101, 000101) = 1$, if the original message was 01, exactly 1 error occurred, and it was on the third bit. Probability of this is $p \cdot (1 - p)^5 = 0.01 \cdot (1 - 0.01)^5 = 0.009509900499$.
- (c) In case Alice wanted to send the message 11, in order for 111111 to change into 001101, there must be an error on 1st, 2nd and 5th bit. The probability of this is $p^3 \cdot (1 - p)^3$, which for $p = 0.01$ is roughly $9.7 \cdot 10^{-7}$.

1.18.

- (a) In our channel, error occurs only when we send 1 and receive 0. The probability of this is

$$P(Y = 0|X = 1) \cdot P(X = 1) = p(1 - q).$$

Because the input bits are independent of each other and the same holds for the channel, the probability of t errors in n bits is just

$$\binom{n}{t} (1 - q)^t p^t (1 - (1 - q)p)^{n-t}.$$

- (b) This value will be lowest for $q = 1$ independent of p because for this value of q we only send the bit 0 and no error can occur.

1.19. Actually, this defines the UPC (Universal Product Code, UPC-A). This code detects nearly all transposition errors on adjacent positions.

If the digits x_i and x_{i+1} are interchanged then the check sum would change by either

$$3x_i + x_{i+1} - 3x_{i+1} - x_i = 2(x_i - x_{i+1}) \text{ or}$$

$$x_i + 3x_{i+1} - x_{i+1} - 3x_i = 2(x_{i+1} - x_i)$$

Therefore, only if $|x_i - x_{i+1}| = 5$, the error would not be detected.

1.20.

$$\{0112223333, 3301122233, 3333011222, 2233330112, 1222333301, \\ 3310232312, 1233102323, 2312331023, 2323123310, 1023231233\}$$

1.21.

- (a) $(2n, 2, 2n)$

(b) $P_{corr}(C_{2n}) = \sum_{i=n+1}^{2n} \binom{k}{i} p^{2n-i} (1 - p)^i = 1 - \sum_{i=n}^{2n} \binom{2n}{i} p^i (1 - p)^{2n-i}$

- (c) Let us rewrite the limit as

$$\lim_{n \rightarrow \infty} 1 - \sum_{i=n}^{2n} \binom{2n}{i} p^i (1 - p)^{2n-i},$$

thus reducing the problem to calculating the limit of the sum $\sum_{i=n}^{2n} \binom{2n}{i} p^i (1-p)^{2n-i}$. Next, let us divide the sum into summands $s_i = \binom{2n}{i} p^i (1-p)^{2n-i}$. First note that each summand is a product of positive numbers, thus it holds that $\forall i, s_i \geq 0$.

Let us now prove that for $i \in n, n+1, \dots, 2n-1$ it holds that

$$s_i \geq s_{i+1}. \tag{1.4}$$

We have a following series of reductions

$$\begin{aligned} \binom{2n}{i} p^i (1-p)^{2n-i} &\geq \binom{2n}{i+1} p^{i+1} (1-p)^{2n-i-1} \\ \frac{(2n)!}{i!(2n-i)!} (1-p) &\geq \frac{(2n)!}{(i+1)!(2n-i-1)!} p \\ \frac{1}{2n-i} (1-p) &\geq \frac{1}{(i+1)} p \\ \frac{1}{2n-i} &\geq \left[\frac{1}{2n-i} + \frac{1}{i+1} \right] p \\ 1 &\geq \left[1 + \frac{2n-i}{i+1} \right] p. \end{aligned}$$

The last inequality holds, because the largest value of the right hand side within the allowed values of i and p is achieved by $i = n$ and $p = \frac{1}{2}$.

The last step is to show that $\lim_{n \rightarrow \infty} s_n = 0$. Together with the fact that for all $i \in \{n+1, \dots, 2n\}$, $s_i \geq 0, s_i > s_{i+1}$, this implies that the $\lim_{n \rightarrow \infty} \sum_{i=n}^{2n} s_i = 0$, and therefore $\lim_{n \rightarrow \infty} (C_{2n}) = 1$.

$$\begin{aligned} \lim_{n \rightarrow \infty} s_n &= \lim_{n \rightarrow \infty} \binom{2n}{n} p^n (1-p)^n \\ &\leq \lim_{n \rightarrow \infty} \frac{4^n}{\sqrt{3n+1}} (p(1-p))^n \\ &\leq^* \lim_{n \rightarrow \infty} \frac{4^n}{\sqrt{3n+1} 4^n} \\ &= 0. \end{aligned}$$

The last inequality follows from the fact that $\max_p p(1-p) = \frac{1}{4}$ and is achieved by setting $p = \frac{1}{2}$.

- (d) $1/2n$
- (e) 0

1.22.

- (a) Assume $C_3 = C_2 \cap C_1$ has at least two words. Then this property certainly holds, because $\forall x, y \in C_3, x, y \in C_1$ and $x, y \in C_2$. In particular, $h(x, y) \geq h(C_1)$ and $h(x, y) \geq h(C_2)$.
- (b) The property does not hold. It holds that $h(C_1) > 0$ and $h(C_2) > 0$. Now consider three words $x \in C_1$ and $y, z \in C_2$, such that $h(y, z) = h(C_2)$. Then $x_1 y_1 \dots x_n y_n \in C_3$ and $x_1 z_1 \dots x_n z_n \in C_3$. It is easy to see that $h(x_1 y_1 \dots x_n y_n, x_1 z_1 \dots x_n z_n) = h(C_2)$, which is a contradiction to the original claim.

1.23. This does not hold. Consider the following codes, both with $(n, M, d) = (4, 3, 1)$:

$$C_1 = \{0000, 0001, 1100\}, C_2 = \{0000, 1000, 0100\}$$

If they were equivalent, we should be able to obtain one from the other by distance preserving operations. However, let us first calculate the distances between codewords.

$$\begin{aligned} h(0000, 0001) &= 1, h(0000, 1100) = 2, h(0001, 1100) = 3 \\ h(0000, 1000) &= 1, h(0000, 0100) = 1, h(1000, 0100) = 2. \end{aligned}$$

Since the ordered sequences of distances are different, we cannot obtain C_1 from C_2 and they are not equivalent.

1.24. The checksum of EAN-13 is calculated as a sum of products - taking an alternating weight value (3 or 1 starting with 1) times the value of each data digit. The checksum digit is the digit, which must be added to this checksum to get a number divisible by 10. We therefore start by summing the odd positions as $(8 + 8 + 0 + 5 + 6 + 4) = 31$ then from even positions we have $3(5 + x + 3 + 3 + 1 + 0) = 36 + 3x$. Together we have $67 + 3x + 4 = 0 \pmod{10}$. We now solve for x to obtain $x = 3$.

1.25. An IBAN is validated by converting it into an integer and performing a basic modulo 97 operation on it. If the IBAN is valid, the remainder equals 1. The algorithm of IBAN validation is as follows:

1. Check that the total IBAN length is correct as per the country. If not, the IBAN is invalid
2. Move the four initial characters to the end of the string
3. Replace each letter in the string with two digits, thereby expanding the string, where $A = 10, B = 11, \dots, Z = 35$
4. Interpret the string as a decimal integer and compute the remainder of that number after division by 97.

If the remainder is 1, the check digit test is passed and the IBAN might be valid.

We therefore first convert $C = 12$ and $Z = 35$ and move it to the end of the string to obtain 2018 0000 0012 3423 x 091 123585. Trying all options for x reveals that the only solution giving $2018 0000 0012 3423 x091 123585 \equiv 0 \pmod{97}$ is $x = 8$.

1.26.

- (a) $(m\ell, 2^\ell, m)$. The first two parameters are self evident. Since each pair of messages differs in at least one bit, m fold repetition results in minimum distance of m .
- (b) $(2\ell, 2^\ell, 3)$. Since there are as many pairs of neighboring bits as the message bits, the length is clearly 2ℓ . Yet again, calculating the distance takes a little bit of work. Two words with the smallest distance differ in one bit in their first half w and w' . WLOG assume that the difference is in the i th position and $w_i = 0, w'_i = 1$. The only bits in the second half that depend on w_i and w'_i are b_{i-1}, b_i and b'_{i-1}, b'_i . Since $b_{i-1} = w_{i-1} \oplus x \neq w'_{i-1} \oplus x = b'_{i-1}$ and $b_i = w_i \oplus y \neq w'_i \oplus y = b'_i$, for all possible values of x and y , we have that the distance of the two codewords is at least 3. The rest of the positions are calculated from the positions in w and w' which are identical, and therefore the distance is exactly 3. Using similar arguments it is easy to see that if w and w' differ in more than one position, the distance of their corresponding codewords is larger than 3.

- (c) $\left(\frac{\ell^2+\ell}{2}, 2^\ell, \ell\right)$. Let us first consider calculation of n . There are $\binom{\ell}{2} = \frac{\ell(\ell-1)}{2}$ different pairs of bits in a message w , which is of length ℓ . Therefore $n = \ell + \frac{\ell^2-\ell}{2} = \frac{\ell^2+\ell}{2}$. The minimum distance is calculated similarly to the previous case. Assume one bit difference between w and w' in i th bit. Then w_i and w'_i influence exactly $\ell - 1$ check bits. Since all other positions of w and w' are equal, the check bits must be different, resulting in distance ℓ . Increasing the number of bits different in w and w' only increases the number of differences in check bits.

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)–(e) 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 its 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 its parity check matrix is called a binary Hamming code and denoted by $\text{Ham}(r, 2)$.

The code $\text{Ham}(r, 2)$ 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, then 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

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

Show that C is a linear code and find its 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

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

- Find $n, k, h(C)$ and $|C|$.
- Find the standard form generator matrix for C .
- Prove that $C^\perp \subset C$.
- 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 $C' = \{x_1 \dots x_n x_{n+1} \mid x_1 \dots x_n \in C, x_{n+1} = \sum_{i=1}^n x_i\}$, where the addition modulo 2. Show that:

- if C is a linear code, then C' is also a linear code;
- 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

- If $C_1 \subseteq C_2$, then $C_2^\perp \subseteq C_1^\perp$;
- $(C_1 \cap C_2)^\perp = C_1^\perp \cup C_2^\perp$.

* **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:

-

$$G_3 = [G_1 | G_2]$$

-

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

* **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				
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	1	
7	1								1					1	1			1		1	1		1	1	
8	1									1				1	1	1			1		1	1		1	
9	1										1				1	1	1			1		1	1	1	
10	1											1		1	1	1	1				1	1	1	1	
11													1	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

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

2.17.

- (a) What is the maximum number of codewords in a linear binary code of length 8 with minimal distance of 3 bits?
- (b) What is the maximum dimension of a linear ternary code of length 4 in which the Hamming distance between every two of its distinct words is odd?

2.18. Consider the following 7-ary codes C_1, C_2 and C_3 of length 3 such that

- (a) $a_1 a_2 a_3 \in C_1 \iff a_1 \cdot a_2 + a_3 \equiv 0 \pmod{7}$;
- (b) $a_1 a_2 a_3 \in C_2 \iff a_1 + a_2 + a_3 \equiv 0 \pmod{7}$;
- (c) $a_1 a_2 a_3 \in C_3 \iff a_1 + a_2 + a_3 \equiv 3 \pmod{7}$.

Decide whether they are linear codes.

2.19. What is the number of different binary self-dual $[4, 2]$ -codes.

2.20. Let C be a linear code over \mathbb{F}_q , where q is a prime. Show that either all codewords of C begin with 0 or exactly $\frac{1}{q}$ of codewords of C begin with 0.

Then the generator matrix of G_{24} is $G = [B|I]$ and the parity check matrix is $H = [I|B]$, where I is a 12×12 identity matrix. For this formulation of G_{24} we have the following decoding algorithm.

Let v be the received vector, e the error vector and $c = v + e$ the decoded word. Also the functional $w(x)$ denotes the Hamming weight of vector x , e_i is a vector of length 12 with 1 in the i -th position and 0's elsewhere and b_i is the i -th row of matrix B .

Step 1. Compute the syndrome $s = wH^\top$.

Step 2. If $w(s) \leq 3$ then $e = [s, 000000000000]$.

Step 3. If $w(s + b_i) \leq 2$ for some b_i then $e = [s + b_i, e_i]$.

Step 4. Otherwise, compute the second syndrome sB .

Step 5. If $w(sB) \leq 3$ then $e = [000000000000, sB]$.

Step 6. If $w(sB + b_i) \leq 2$ for some b_i then $e = [e_i, sB + b_i]$.

Step 7. If e is not yet determined then there were more than 3 errors and the user should request retransmission.

(a) Decode $v = 001101110111000110000000$

(b) Decode $v = 101010110000110101101000$

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:

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

(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 $(u_1+v_1)+2(u_2+v_2)+3(u_3+v_3)+4(u_4+v_4) = 0 \pmod{5}$. Also we have that for any scalar w and corresponding word wu it holds that $w(u_1+2u_2+3u_3+4u_4) = 0 \pmod{5}$.

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 its parity check matrix $H = (1, 2, 3, 4)$. Its 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.

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

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 $\forall c \in C c \cdot c$. 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 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:

$$C^\perp = \{0000000, 1101001, 1110010, 0111100, 1001110, 0011011, 1010101, 0100111\}$$

$$C = \{0000000, 1000011, 0100111, 0010110, 0001101, 1100100, 1010101, 1001110, 0110001, 0101010, 0011011, 1110010, 1101001, 1011000, 0111100, 1111111\}$$

(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 .

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

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 $\begin{pmatrix} x \\ y \end{pmatrix}$, 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

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

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

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

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 Mersenne 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 2.6 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 must contain all the words orthogonal to $L|R$, so in particular it contains $L'|R'$.

2.17.

- (a) The volume of the space of 8-bit binary words is the number of points, $2^8 = 256$. If no two codewords are within distance of 3, then the spheres of radius 1.5 about any two codewords must be disjoint. The volume of a sphere is the number of points in the sphere. Within distance 1.5 of any 8-bit word are 9 words - the word itself and those words that differ from it in exactly one bit. Since $\frac{256}{9} = 28.444$, there can be at most 28 codewords all of which are distance 3 from each other. Therefore, for a binary linear code there are at most $2^k \leq 28.444$ codewords implying $k \leq 4$. The codewords 11100000, 10011000, 10000110, 00101011 generate the code with minimal distance $d = 3$, therefore the maximum number of codewords in a linear binary code of length 8 and minimal distance of 3 bits is 16.
- (b) We will show by contradiction that the dimension of such a code has to be less than 3. There is a total of $4 \cdot 2^3 + 8 = 40$ words of length 4 and odd distance from the word 0000 (which has to be contained in the code). But the code can contain only one cyclic permutation of these words: 1000, 2000, 1110, 2220, 1210, 2120. We cannot use the other permutations of these words, which means we cannot use at least $6 \cdot 3 = 18$ words, which leaves us with less than $40 - 18 = 22$ words. Thus the code cannot have dimension 3 (otherwise it would have to contain $3^3 = 27$ words). On the other hand, an example of such a code of dimension 2 is the one generated by 1110 and 1201.

2.18.

- (a) We can easily see that $246, 356 \in C_1$ as $2 \cdot 4 + 6 \equiv 0 \pmod{7}$ and $3 \cdot 5 + 6 \equiv 0 \pmod{7}$. But their sum $525 \notin C_1$ as $5 \cdot 2 + 5 \equiv 1 \pmod{7}$.
- (b) Let $a_1a_2a_3, b_1b_2b_3 \in C_2, k, l \in \mathbb{F}_7$. We look whether $k \cdot a_1a_2a_3 + l \cdot b_1b_2b_3$ is in C_2 . $ka_1 + lb_1 + ka_2 + lb_2 + ka_3 + lb_3 \equiv k(a_1 + a_2 + a_3) + l(b_1 + b_2 + b_3) \equiv k \cdot 0 + l \cdot 0 \equiv 0 \pmod{7}$. So the linear combination is in C_2 and thus C_2 is a linear code.
- (c) We can easily see the zero codeword is not in C_3 as $0 + 0 + 0 \equiv 0 \pmod{7}$. This means C_3 cannot be a linear code.

2.19. Since C is self-dual, 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 two codewords of weight 2. This follows from the fact that each pair of such words has to have the 1's in different positions.
- The previous fact implies that C contains the all 1's codeword. Together with the all 0's codeword we now know weights of all the codewords.

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

$$\frac{\binom{4}{2}}{2} = 3.$$

2.20. It is clear that if every codeword of some *basis* of C begins 0 then every codeword of C begins with 0. Now consider C with a generator matrix G with at least one codeword not beginning in 0. Let k be the dimension of C . We will count the number of codewords of C beginning with 0. Every codeword of C is a linear combination of codewords in G . Let

$$n = n_1 + n_2 + \cdots + n_{q-1}$$

be the number of codewords in G not beginning with 0, where n_i is the number of codewords in G beginning with i . Codeword w begins with 0 if it is the sum of arbitrary linear combination of words from G beginning in 0 and a linear combination of all other codewords such that it begins in 0. The number of linear combinations of codewords in G beginning with 0 is q^{k-n} . We need to count the number of linear combinations of words not beginning with 0 such that the result begins with 0. Consider such linear combination v , v is a sum of a linear combination of codewords beginning with $1, 2, \dots, q-2$ and a linear combination of codewords beginning with $q-1$ such that the sum of the first positions of these two combinations is 0. So the first linear combination can begin in any number $a \in \mathbb{F}_q$ but this determines the second part of the linear combination begins in $-a$. The number of all possible linear combinations of codewords in G beginning in $1, 2, \dots, q-2$ is $q^{n_1+n_2+\cdots+n_{q-2}}$. All we need now is the number of linear combinations of the codewords in G beginning in $q-1$ such that it begins in certain number $-a$. We can repeat the previous argument for the number of linear combinations of all the words not beginning in 0 such that it begins in 0, only now we have only words beginning in $q-1$ and the combination begins in $-a$ instead of 0, but that changes nothing. The linear combinations of the first $n_{q-1}-1$ codewords can begin in any $b \in \mathbb{F}_q$ and the last element of the combination must begin in $-b$. The first part gives us $q^{n_{q-1}-1}$ combinations and the second part only one. So the total number is $q^{n_{q-1}-1}$. This gives us the total number of linear combinations of codewords not beginning in 0 such that it begins in 0

$$q^{n_1+n_2+\cdots+n_{q-2}} \cdot q^{n_{q-1}-1} = q^{n_1+n_2+\cdots+n_{q-1}-1}.$$

And this in turn gives us the total number of linear combinations of codewords in G such that it begins in 0

$$q^{n_1+n_2+\cdots+n_{q-1}-1} \cdot q^{k-n} = q^{k-1}.$$

This is exactly $1/q$ of the total number of codewords q^k .

2.21. “ \Rightarrow ”: Consider a codeword $c \in C$. It's a codeword of C so it must hold that $Hc^T = 0$. The non-zero entries of c determine a linearly dependent columns of matrix H . Now if $w(c) < s$ we have a set of $s-1$ (if $w(c)$ is smaller than $s-1$ we can just add arbitrary columns) columns of matrix H which are linearly dependent. But that is a contradiction with our preposition. So $w(c) \geq s$ for any codeword $c \in C$, in other words $w(C) \geq s$.

“ \Leftarrow ”: Denote $H = (h_1, h_2, \dots, h_n)$ the columns of H . Assume that $w(C) \geq s$ and that there is some set of $s-1$ columns of H that are linearly dependent, i.e. there exists a set of indices $I \in \{1, 2, \dots, n\}$, $|I| = s-1$, and coefficients $c'_i \in F_q$ such that

$$\sum_{i \in I} c'_i h_i = 0.$$

We can now construct a codeword c with the following entries c_i :

$$c_i = \begin{cases} 0 & \text{if } i \notin I \\ c'_i & \text{if } i \in I \end{cases}.$$

Clearly $cH^T = 0$ so $c \in C$. But c has only $s-1$ non-zero entries so $w(C) < s$. That is a contradiction and so our set of linearly dependent columns cannot exist.

2.22. First we show that $w(C) \geq 4$ using the result from previous exercise. All we need to show is that there is not a set of 3 columns which are linearly dependent.

It is clear that the set of the first four columns is linearly independent so any 3 of them are also independent.

A linear combination of any two of them can have non zero entries only at two positions at most and so a set of any two of them and the fifth or sixth column is also independent.

The last case is a set with both the fifth and sixth column and one extra column. This again cannot be not independent, because any non-zero linear combination of the fifth and sixth column has at least two non-zero entries and so it cannot be equal to any other column, as they all have a single non-zero entry.

Thus any set of 3 columns of H is linearly independent and $w(C) \geq 4$. At the same time the set of the first, second, third and fifth column is clearly linearly dependent. Since the size of this set is 4, $w(C) \leq 5$. And so $w(C) = 4$. And because $h(C) = w(C)$ for linear codes we have that the minimum distance of C is 4.

2.23.

- (a) Let V be an arbitrary vector space and $\varphi : V \rightarrow V$ linear map on V . Recall that the kernel of φ is

$$\text{Ker}(\varphi) = \{u \in V \mid \varphi(u) = 0\}.$$

Denote the matrix of φ in standard basis by A . Then it holds that

$$\text{Ker}(\varphi) = \{u \in V \mid A \cdot u^T = 0\}.$$

And since for $u \in \{0, 1\}^n$ it holds that $H \cdot u^T = 0$ if and only if $u \in C$, then the kernel of H is the code C , i.e. $\text{Ker}(H) = C$.

- (b) The *Rank-nullity theorem* claims that

$$\text{Rank}(H) + \text{Dim}(\text{Ker}(H)) = \text{Col}(H)$$

where Dim stands for dimension and Col stands for number of columns. From (a) follows that $\text{Dim}(\text{Ker}(H)) = k$. Further, $\text{Col}(H) = n$ and thus $\text{Rank}(H) = n - k$.

2.24. Fixing $k = 1$, we can take the code C generated by the word $\overbrace{11 \dots 1}^n$. This is an $[n, 1]$ code with $d(C) = n$, so it is MDS (because $M = q = q^{n-n+1} = q^{q-d+1}$). Its dual code is the checksum code $C^\perp = c_1 c_2 \dots c_n \in \mathbb{F}_q^n \mid c_1 + c_2 + \dots + c_n = 0$, which is an $[n, n - 1]$ code (it is a subspace of \mathbb{F}_q^n determined by one linear equation) and $d(C^\perp) = 2$ (to see this, note that C^\perp contains the word $\overbrace{1(-1)00 \dots 0}^{n-2}$, but clearly cannot contain any word of weight 1 because of the checksum condition). Therefore $d(C^\perp) = 2 = n - (n - k) + 1$ and C^\perp is again MDS.

2.25.

- (a) Yes. We need to prove two propositions:

- (i) $\forall u, v \in C = C_1 \cap C_2; u + v \in C$. *Proof:* $u, v \in C_1 \cap C_2 \Rightarrow (u, v \in C_1 \wedge u, v \in C_2)$. Since C_1 and C_2 are linear, we have that $u, v \in C_1 \Rightarrow u + v \in C_1$ and $u, v \in C_2 \Rightarrow u + v \in C_2$. Finally, $(u + v \in C_1 \wedge u + v \in C_2 \Rightarrow u + v \in C_1 \cap C_2)$.
- (ii) $\forall u \in C, a \in GF(q) : au \in C$. *Proof:* $u \in C_1 \cap C_2 \Rightarrow (u \in C_1 \wedge u \in C_2)$. Since C_1 and C_2 are linear, we have $(u \in C_1 \wedge a \in GF(q) \Rightarrow au \in C_1)$ and $(u \in C_2 \wedge a \in GF(q) \Rightarrow au \in C_2)$. Finally, $(au \in C_1 \wedge au \in C_2) \Rightarrow au \in C_1 \cap C_2$.

- (b) No. Consider the following counterexample. Let $C_1 = \{00, 10\}$ and $C_2 = \{00, 01\}$. Obviously, both C_1 and C_2 are linear codes, but $01 + 10 = 11$ and $11 \notin C_1 \cup C_2$.
- (c) Let $k \in \mathbb{F}_q$, $u_1, u_2 \in C_1$, $v_1, v_2 \in C_2$ be arbitrary, then

$$k(u_1 \cdot v_1) = (ku_1) \cdot (kv_1) \in C_1 \cdot C_2$$

and

$$(u_1 \cdot v_1) + (u_2 \cdot v_2) = (u_1 + u_2) \cdot (v_1 + v_2) \in C_1 \cdot C_2,$$

therefore $C_1 \cdot C_2$ is linear.

- (d) The code is not linear because $00\dots 0 \notin C_1 \triangle C_2$.
- (e) It depends. For example, if $C_1 = C_2 = C_3$ then $C_1 \triangle C_2 \triangle C_3 = C_1$ is linear, however if

$$C_1 = \{000, 100\}, C_2 = \{000, 010\}, C_3 = \{000, 001\}$$

then $C_1 \triangle C_2 \triangle C_3 = \{000, 100, 010, 001\}$ is not linear.

2.26. Let the code be C an $[n, k, d]$ linear code, then we know $B(n, d) = q^k$. We want to get an $[n - 1, k^*, d]$ code, what we can achieve with code shortening. If we have a full 0 column (special case, this position is useless), we can shorten the code without losing any base code words, and we get an $[n - 1, k, d]$ code, and the equality clearly holds (this gives the largest possible k). More generally (no all 0 column), we can transform the codes generator matrix to have only one 1bit in its last column. With shortening we receive an $[n - 1, k - 1, \geq d]$ code, which has q^{k-1} code words. Substituting into the non-equivalence we get:

$$\begin{aligned} B_q(n, d) &\leq qB_q(n - 1, d) \\ q^k &\leq q \cdot q^{k-1} \\ q^k &\leq q^k \end{aligned}$$

which is true.

2.27.

- (a) In this case $vH^\top = 000100000001$. Therefore according to the decoding algorithm $e = 000100000001000000000000$ and the decoded word $c = v + e = 001001110110000110000000$. The second half of c shows that this is a sum of fourth and fifth row of G .
- (b) In this case we have $s = vH^\top = 111010110010$. The first condition therefore fails. Let us check the second condition

$$\begin{aligned} s + b_1 &= 001101\dots \\ s + b_2 &= 0101001\dots \\ s + b_3 &= 10011\dots \\ s + b_4 &= 000010011\dots \\ s + b_5 &= 001011\dots \\ s + b_6 &= 0110000001\dots \\ s + b_7 &= 111\dots \\ s + b_8 &= 110001\dots \\ s + b_9 &= 1011\dots \\ s + b_{10} &= 01011\dots \\ s + b_{11} &= 10000101\dots \\ s + b_{12} &= 000101001\dots \end{aligned}$$

The second condition therefore fails. We now need to calculate $sB = 100001010011$. The third condition fails, but we find that $sB + b_5 = 010000001000$, therefore the fourth condition holds and $e = 000010000000010000001000$. Thus we decode $c = v + e = 101000110000100101100000$. This is a sum of rows 1, 4, 6 and 7 of the generator matrix G .

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 possess 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 $g(x) \equiv h(x) \pmod{f(x)}$), 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 $x^n \equiv 1 \pmod{(x^n - 1)}$, we have that

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

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

$$g(x) = g_0 + g_1x + \cdots + g_r x^r.$$

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

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

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 $c(x)h(x) \equiv 0 \pmod{x^n - 1}$. Also, if $h(x) = h_0 + h_1x + \cdots + h_k x^k$, then

- (i) a parity-check matrix for C is

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

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

$$\bar{h}(x) = h_k + h_{k-1}x + \cdots + h_0x^k,$$

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 $C_2 = \{x \mid x \in \mathbb{Z}_3^5 \wedge w(x) \equiv 0 \pmod{3}\}$

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

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

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 $01111111110000 \in C$. 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 $c_0c_1 \dots c_{n-1} \in C$ satisfy $c_0 + c_1 + \dots + c_{n-1} = 0$, 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 $g(x) = g_kx^k + \dots + g_1x^1 + g_0 \neq 0$ 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 $111 \dots 11$.

*** 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.

3.18. Let $q, n \in \mathbb{N}$, where q is a prime and let C_1, C_2 be cyclic q -ary codes of length n . In each of the following cases, determine if C_3 is necessarily a cyclic code.

- (a) $C_3 = C_1 \setminus C_2$;
- (b) $C_3 = (C_1 \cup C_2) \setminus (C_1 \cap C_2)$;

(c) $C_3 = \{a_1b_1a_2b_2 \dots a_nb_n \mid a_1a_2 \dots a_n \in C_1, b_1b_2 \dots b_n \in C_2\}$;

(d) $C_3 = \{a_1b_1a_2b_2 \dots a_nb_n \mid a_1a_2 \dots a_n, b_1b_2 \dots b_n \in C_1\}$;

(e) $C_3 = \{w_1 - w_2 \mid w_1 \in C_1, w_2 \in C_2\}$.

* **3.19.** Determine the number of

(a) all cyclic ternary codes of length 16;

(b) all cyclic quaternary codes of length 12.

* **3.20.** Let C be a cyclic code of length n over \mathbb{F}_q with generator polynomial $g(x)$. Let $v(x)$ be a polynomial in R_n such that $\gcd(v(x), x^n - 1) = g(x)$ over $\mathbb{F}_q[x]$. Show that $v(x)$ is the generator polynomial of C as well.

3.21. Find cyclic codes equivalent to the following binary codes:

(a) $C_1 = \{0000, 1001, 0110, 1111\}$

(b) $C_2 = \{100, 010, 001\}$

(c) $C_3 = \{11111\}$

3.22. Consider a binary cyclic code C of odd length n with generator polynomial $g(x)$. Prove that if $1 + x \mid g(x)$ then $11 \dots 1 \notin C$.

3.23. Find the generator polynomial of the binary single-parity-check code (a code consisting of all codewords with parity 0) of length $n \geq 2$.

3.24. Consider binary codes of length 7 containing 16 codewords. Count the number of

(a) all such codes;

(b) such linear codes;

(c) such cyclic codes.

3.25. Prove two following statements:

(a) For any polynomial $f(x) \in \mathbb{F}_q[x]$ the fact that $f(x)$ is irreducible implies that $f(x)$ has no roots in \mathbb{F}_q .

(b) For $f(x) \in \mathbb{F}_q[x]$ with $\deg(f(x)) \leq 3$ it holds that $f(x)$ has no roots in \mathbb{F}_q implies that $f(x)$ is irreducible.

* **3.26.** For any $m, n, k, d \in \mathbb{N}$, $d > 1$ and q a power of a prime, show that if a cyclic q -ary $[n, k, d]$ -code exists then a cyclic q -ary $[mn, mk, d]$ -code exists as well.

3.27. For each code C with codewords of length n a reverse code \bar{C} is defined as

$$\bar{C} = \{x_1x_2 \dots x_n \mid x_nx_{n-1} \dots x_1 \in C\}.$$

(a) Show that for each cyclic code C , its reverse code \bar{C} is also cyclic.

(b) Show that for each binary cyclic code C with codewords of length $n \leq 6$, $C = \bar{C}$.

(c) Find an example of a binary code with codewords of length 7, such that $C \neq \bar{C}$.

3.28. Prove the following. Let C_1 and C_2 be cyclic codes with generator polynomials $g_1(x)$ and $g_2(x)$. Then $C_1 \subseteq C_2$ if and only if $g_2(x)$ divides $g_1(x)$.

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 $z_1 + z_2 + z_3 + z_4 + z_5 = (x_1 + y_1) + (x_2 + y_2) + (x_3 + y_3) + (x_4 + y_4) + (x_5 + y_5) = (x_1 + x_2 + x_3 + x_4 + x_5) + (y_1 + y_2 + y_3 + y_4 + y_5) \equiv 0 \pmod{3}$, since $x_1 + x_2 + x_3 + x_4 + x_5 \equiv 0 \pmod{3}$ and $y_1 + y_2 + y_3 + y_4 + y_5 \equiv 0 \pmod{3}$.
 - 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 $1 \cdot 1 + 2 \cdot 1 + 3 \cdot 2 + 4 \cdot 3 + 5 \cdot 4 = 41 \equiv 6 \pmod{7}$.

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 its 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 its second row to obtain the second row of G .
3. Add the second row of F to its third row to obtain the third row of G .
4. Add the third row of F to its 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 01111111110000 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:

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

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

$$H = (1 \ 2 \ 1 \ 2 \ 1 \ 2).$$

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

$$H = (1 \ 1 \ 1 \ 1 \ 1 \ 1).$$

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,

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

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 $g(x) = \text{gcd}(g_1(x), g_2(x))$. 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 $a_0 + a_1x + \dots + a_{n-1}x^{n-1}$, 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 $C = \{d(x)(x - 1) \mid d(x) \in D\}$. 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 $d_{n-1} - d_0 + d_0 - d_1 + \dots + d_{n-2} - d_{n-1} = 0$

- “ \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 $g(x) = a_0 + a_1x + \dots + a_{n-1}x^{n-1}$, and $a_0 + \dots + a_{n-1} = 0$. This implies that 1 is a root of $g(x)$: $g(1) = a_0 + \dots + a_{n-1} = 0$. 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 $g(x) = x(g_kx^{k-1} + \dots + g_1)$. 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 $\forall f(x) \in C : 2 \mid w(f(x)) \Leftrightarrow f(1) = 0$, 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 $a_0 + a_1 + \dots + a_{n-1} \equiv 1 \pmod{2}$. Because C is cyclic, it also contains all cyclic shifts of a and their sum w . Every position of w can be expressed as $a_0 + a_1 + \dots + a_{n-1}$, 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 its 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.

3.18.

- (a) C_3 is not necessarily a cyclic code. For example if we consider $C_1 = \mathbb{F}_q^n$ and $C_2 = \{00 \dots 0\}$ then $C_1 \setminus C_2$ is not even a linear code much less cyclic (it does not contain the all zero word).
- (b) The same example as in the previous case can be applied here since $C_1 = (C_1 \cup C_2)$ and $C_2 = (C_1 \cap C_2)$. So again C_3 is not necessarily a cyclic code.
- (c) Again we can consider $C_1 = \mathbb{F}_q^n$ and $C_2 = \{00 \dots 0\}$. This code is linear but we can show that it is not cyclic. C_3 has 0 on all even positions, therefore cyclic shift of a word $1010 \dots 10 \in C_3$ does not belong to C_3 . So we get that C_3 is not necessarily a cyclic code.
- (d) We want to show that in this case the code C_3 is cyclic. First we prove that it is linear.

$$\begin{aligned} & a_1b_1a_2b_2 \dots a_nb_n, c_1d_1c_2d_2 \dots c_nd_n \in C_3 \\ \implies & a_1a_2 \dots a_n, b_1b_2 \dots b_n, c_1c_2 \dots c_n, d_1d_2 \dots d_n \in C_1 && \text{definition of } C_3 \\ \implies & (a_1 + c_1)(a_2 + c_2) \dots (a_n + c_n), (b_1 + d_1)(b_2 + d_2) \dots (b_n + d_n) \in C_1 && \text{linearity of } C_1 \\ \implies & (a_1 + c_1)(b_1 + d_1)(a_2 + c_2)(b_2 + d_2) \dots (a_n + c_n)(b_n + d_n) \in C_3 && \text{definition of } C_3 \end{aligned}$$

Similarly

$$\begin{aligned} & a_1b_1a_2b_2 \dots a_nb_n \in C_3, t \in \mathbb{F}_q \\ \implies & a_1a_2 \dots a_n, b_1b_2 \dots b_n \in C_1, t \in \mathbb{F}_q && \text{definition of } C_3 \\ \implies & (ta_1)(ta_2) \dots (ta_n), (tb_1)(tb_2) \dots (tb_n) \in C_1 && \text{linearity of } C_1 \\ \implies & (ta_1)(tb_1)(ta_2)(tb_2) \dots (ta_n)(tb_n) \in C_3 && \text{definition of } C_3 \end{aligned}$$

Now we want to prove that

$$a_1b_1a_2b_2 \dots a_nb_n \in C_3 \implies b_1a_2b_2 \dots a_nb_na_1, b_na_1b_1a_2b_2 \dots a_n \in C_3.$$

It is now sufficient to prove that $a_1a_2b_2 \dots a_nb_na_1 \in C_3$.

$$\begin{aligned} & a_1b_1a_2b_2 \dots a_nb_n \in C_3 \\ \implies & a_1a_2 \dots a_n, b_1b_2 \dots b_n \in C_1 && \text{definition of } C_3 \\ \implies & b_1b_2 \dots b_n, a_2 \dots a_na_1 \in C_1 && \text{cyclicity of } C_1 \\ \implies & b_1a_2b_2 \dots a_nb_na_1 && \text{definition of } C_3 \end{aligned}$$

We get that C_3 is necessarily a cyclic code.

(e) Linearity:

$$\begin{aligned} & x, y \in C_3 \\ \implies & \exists a, b \in C_1, \exists c, d \in C_2 : a - c = x, b - d = y && \text{definition of } C_3 \\ \implies & a + b \in C_1, c + d \in C_2 && \text{linearity of } C_1 \text{ and } C_2 \\ \implies & (a + b) - (c + d) = (a - c) + (b - d) = x + y \in C_3 && \text{definition of } C_3 \end{aligned}$$

and similarly

$$\begin{aligned} & x \in C_3, t \in \mathbb{F}_q \\ \implies & t \in \mathbb{F}_q, \exists a \in C_1, \exists b \in C_2 : a - b = x && \text{definition of } C_3 \\ \implies & ta \in C_1, tb \in C_2 && \text{linearity of } C_1 \text{ and } C_2 \\ \implies & ta - tb = t(a - b) = tx \in C_3 && \text{definition of } C_3 \end{aligned}$$

So C_3 is a linear code. Now we prove cyclicity.

$$\begin{aligned} & x_1x_2 \dots x_n \in C_3 \\ \implies & \exists a_1a_2 \dots a_n \in C_1, \exists b_1b_2 \dots b_n \in C_2 : a_i - b_i = x_i, i \in \{1, 2, \dots, n\} && \text{definition of } C_3 \\ \implies & a_na_1 \dots a_{n-1} \in C_1, b_nb_1 \dots b_{n-1} \in C_2 && \text{cyclicity of } C_1 \text{ and } C_2 \\ \implies & (a_n - b_n)(a_1 - b_1) \dots (a_{n-1} - b_{n-1}) = x_nx_1 \dots x_{n-1} \in C_3 && \text{definition of } C_3 \end{aligned}$$

This proves that code C_3 is necessarily a cyclic code.

3.19. From number theory we know that there are $\frac{1}{n} \sum_{d|n} \mu\left(\frac{n}{d}\right) p^d$ irreducible polynomials of degree n over \mathbb{F}_p (where p is prime and μ is the Möbius function.)

- (a) We want to factorize $x^{16} - 1$ over \mathbb{F}_3 . From the formula $a^2 - 1 = (a + 1)(a - 1)$ we get $x^{16} - 1 = (x^8 + 1)(x^4 + 1)(x^2 + 1)(x + 1)(x - 1)$. Quadratic and cubic polynomials are irreducible if and only if they do not have a root. Polynomials of degree 4 and 5 are irreducible if and only if they do not have a root and are not divisible by quadratic irreducible polynomial. These are conditions that are quite easy to check. $x^2 + 1$ does not have a root so it is irreducible. We know that there are exactly 3 irreducible monic quadratic polynomials. The others are $x^2 \pm x - 1$. Factorization

$$x^4 + 1 = (x^2 + x - 1) \cdot (x^2 - x - 1)$$

gives us a hint for $x^8 + 1$. We get

$$x^8 + 1 = (x^4 + x^2 - 1) \cdot (x^4 - x^2 - 1)$$

and since we know that both of these polynomials cannot have a root (because $x^8 + 1$ does not have one), we must check if they are not a product of two (monic) irreducible quadratic polynomials. From the last coefficient we know that one of the factors has to be $x^2 + 1$ (since the last coefficient in others is -1 and $-1 \cdot (-1) \cdot 6 = -1$). We obtain

$$\begin{aligned} x^4 + x^2 - 1 &= (x^2 + 1)x^2 - 1, \\ x^4 - x^2 - 1 &= (x^2 + 1)(x^2 + 1) + 1, \end{aligned}$$

so both of these polynomials are irreducible. We get factorization of the original polynomial:

$$x^{16} - 1 = (x^4 + x^2 - 1)(x^4 - x^2 - 1)(x^2 + x - 1)(x^2 - x - 1)(x^2 + 1)(x + 1)(x - 1).$$

We get 27 distinct factors of $x^{16} - 1$, so there are 27 cyclic ternary codes of length 16.

- (b) Again we want to factorise $x^{12} - 1$ over $\mathbb{F}_4 = \mathbb{F}_2[y]/(y^2 + y + 1)$. We use the formula $a^2 - 1 = (a + 1)(a - 1)$ and the fact that $1 \equiv -1$.

$$x^{12} - 1 = (x^6 - 1)^2 = (x^3 - 1)^4 = (x - 1)^4 \cdot (x^2 + x + 1)^4$$

since \mathbb{F}_4 is splitting field of polynomial $x^2 + x + 1$ it has to have both roots in this field. If we take $\mathbb{F}_4 = \{0, 1, y, y + 1\}$ we have $x^{12} - 1 = (x - 1)^4 \cdot (x - y)^4 \cdot (x - (y + 1))^4$. Without using any advanced algebra of finite fields, we can just use the definition of \mathbb{F}_4 from the lecture. We know that $y \cdot (y + 1) = y^2 + y = y + 1 + y = 1$. So we get $(x - y)(x - (y + 1)) = x^2 + x(y + y + 1) + y(y + 1) = x^2 + x + 1$ which is exactly what we expected. From the factorization of $x^{12} - 1$ over \mathbb{F}_4 we get 5^3 distinct factors. That gives us 125 cyclic quaternary codes.

3.20. First we prove that for codes C_1, C_2 and their generator polynomials $g_1(x), g_2(x)$ it holds that

$$g_1(x) \in C_2 \iff C_1 \subseteq C_2.$$

Let $f(x) \in C_1$. We know that $f(x) = g_1(x) \cdot r(x)$ for some $r(x) \in R_n$. Since $g_1(x) \in C_2$ we get $g_1(x) = g_2(x) \cdot q(x)$. From this we get $f(x) = g_2(x) \cdot q(x)r(x)$ so $f(x) \in C_2$.

Let $D = \langle v(x) \rangle$ be a cyclic code generated by $v(x)$. We want to show that $C = D$. We know that

$$f(x) \in C \iff g(x) | f(x) \quad (\text{divisibility in } R_n)$$

and $g(x)$ is (the greatest) common divisor of $v(x)$ and another polynomial, we get that $g(x) | v(x)$. Therefore

$$v(x) \in C$$

and we get that $D \subseteq C$ (since C contains the generator of D). Now the greatest common divisor can be considered over $\mathbb{F}_q[x]$ (which is Euclidean domain) or over R_n (which is not even an integral domain).

In the case of R_n we have $\gcd(v(x), xn - 1) = \gcd(v(x), 0)$. It need not be unique (even if we consider only monic polynomials)! If $g(x) = \gcd(v(x), 0)$, we have that $g(x) | v(x)$ (it is divisor) and since $v(x) | v(x)$ and $v(x) | 0$ it also must hold that $v(x) | g(x)$ (it is the greatest common divisor). From this we immediately get that $g(x) \in D$ so $C = D$.

In the case of $\mathbb{F}_q[x]$ we know that we have Euclidean algorithm and Bezout's identity. So we have

$$g(x) = v(x) \cdot a(x) + (x^n - 1) \cdot b(x),$$

where $a(x), b(x) \in \mathbb{F}_q[x]$ are suitable polynomials. Looking at this equation modulo $x^n - 1$ (i.e. in R_n) we have

$$g(x) = v(x)a(x) \implies v(x) | g(x) \quad (\text{in } R_n) \implies g(x) \in D \implies C \subseteq D.$$

Which gives us $C = D$ and the proof is complete.

3.21.

- (a) If we swap the 1st and 2nd bits we obtain an equivalent cyclic code $\{0000, 0101, 1010, 1111\}$.
- (b) C_2 has 3 codewords, therefore the dimension of its cyclic equivalent must be $\log_2 3 \approx 1,58$ which is not a valid dimension for a linear code. As every cyclic code is linear there is no binary cyclic code with 3 codewords. Therefore there is no cyclic code equivalent to C_2 .
- (c) Since C_3 has only 1 codeword it's cyclic equivalent must also have only 1 codeword. The only cyclic code of length 5 with only 1 codeword is the code $\{00000\}$.

3.22. It suffices to show that every codeword in C has even weight, then clearly for odd n holds that $\underbrace{11\dots 1}_n \notin C$. Consider arbitrary word $w = (w_0, w_1, \dots, w_{n-1}) \in C$ with corresponding polynomial $w(x)$. Since $w(x) = g(x) \cdot r(x)$ for some $r(x) \in \mathcal{R}_n$, it holds that $1+x \mid w(x)$. Hence $w(1) = 0$. It also holds that

$$w(1) = 0 \iff w_0 + w_1 + \dots + w_{n-1} = 0.$$

And thus

$$w(1) = 0 \iff w \text{ has even weight.}$$

All together, if $1+x \mid g(x)$, then arbitrary $w \in C$ has even weight and so $\underbrace{11\dots 1}_n \notin C$.

3.23. We start with simple generator matrix for our code

$$G = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & 1 \\ 0 & 1 & 0 & \dots & 0 & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 & 1 \end{bmatrix}.$$

Clearly, with this matrix we can generate any codeword of parity 0, our code. Unfortunately this generator matrix is not in a form from which we could read the generator polynomial. But we can use it to find such matrix. Consider a new generator matrix G' for our code created from G in the following way: to every row, but the last row, add the next row.

$$G' = \begin{bmatrix} 1 & 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 & 1 \end{bmatrix}.$$

We were only adding rows of the original generator matrix and so G' is still generates our code. But G' is now in the form as if it were created by a generator polynomial. This polynomial is

$$g(x) = 1 + x,$$

regardless of n . And so the generator polynomial of the binary single-parity-check code is always $g(x) = 1 + x$ for any length $n \geq 2$.

3.24.

- (a) There are $\binom{2^7}{16} = 93343021201262180000$ combinations for choosing 16 codewords from $2^7 = 128$ binary codewords of length 7.

- (b) We need to count the number of vector subspaces of dimension 4 (to have $2^4 = 16$ elements) in \mathbb{F}_2^7 . We can use the Gaussian binomial coefficient to count k -dimensional subspaces of n -dimensional space. Such coefficients can be rewritten as follows (reflecting the way of specifying k linearly independent vectors):

$$\binom{n}{k}_q = \frac{(q^n - 1)(q^n - q)(q^n - q^2) \cdots (q^n - q^{k-1})}{(q^k - 1)(q^k - q) \cdots (q^k - q^{k-1})}$$

In our case:

$$\binom{7}{4}_2 = \frac{(2^7 - 1)(2^7 - 2)(2^7 - 2^2)(2^7 - 2^3)}{(2^4 - 1)(2^4 - 2)(2^4 - 2^2)(2^4 - 2^3)} = 11811.$$

- (c) We need 16 codewords so we search for a divisor polynomial of $x^7 - 1$ over \mathbb{F}_2 with degree $7 - 4 = 3$. We have $x^7 - 1 = (x + 1)(x^3 + x + 1)(x^3 + x^2 + 1)$, therefore our code is generated either with $(x^3 + x + 1)$ or $(x^3 + x^2 + 1)$. There are 2 such cyclic codes.

3.25.

- (a) According to Euclidean division of polynomials introduced in the lecture slides, each polynomial $f(x)$ can be written as $f(x) = q(x)b(x) + r(x)$, for arbitrary $b(x)$ with $\deg(b(x)) > \deg(r(x))$ and $\deg(f(x)) > \deg(b(x))$. For the sake of constructing a contradiction, assume a is the root of $f(x)$. Then we can write $f(x) = g(x)(x - a) + r$. Note that r has to be constant (degree less than 1) and therefore, since a is a root, $f(a) = 0$ and therefore $r = 0$, showing that $f(x)$ can be written as $f(x) = g(x)(x - a)$ and thus being reducible.

- (b) Let's prove the contrapositive: $f(x)$ being reducible implies $f(x)$ has roots.

- (i) $\deg(f(x)) = 2$. Then reducible $f(x)$ can be written as $f(x) = (x - a)(x - b)$ and both a and b are roots of $f(x)$.
- (ii) $\deg(f(x)) = 3$. Then reducible $f(x)$ can be written as $f(x) = g(x)b(x)$, where $g(x)$ is of degree 2 and $b(x)$ is of degree 1, i.e. $b(x) = (x - a)$ and a is the root of $f(x)$.
- (iii) $\deg(f(x)) = 1$. $f(x)$ is of the form $f(x) = (x - a)$ and clearly a is the root of $f(x)$.

3.26. Let C be a q -ary cyclic $[n, k, d]$ -code with generator polynomial $g(x)$. Thus, we have $\deg(g(x)) = n - \dim(C) = n - k$. Denote by C' the q -ary cyclic code with length mn whose generator polynomial is $f(x) = g(x^m)$. Note that if $g(x)$ is monic, then so is $g(x^m)$, and that $x^{n-1} = g(x)h(x)$ implies $x^{mn} - 1 = (x^m)^n - 1 = g(x^m)h(x^m) = f(x)h(x^m)$, so $f(x)$ is indeed a generator polynomial, as needed. Moreover, we have $\dim(C') = mn - \deg(f) = mn - m \cdot \deg(g) = mn - m(n - k) = mk$, as required. It remains to prove that $h(C') = w(C') = d$:

Since $h(C) = w(C) = d$, there is a polynomial $u(x) = g(x)p(x) \in C$ with $w(u(x)) = d$. Then, we have $u(x^m) = g(x^m)p(x^m) = f(x)p(x^m) \in C'$ (since $\deg(u(x^m)) = m \cdot \deg(u(x)) < mn$ as $\deg(u(x)) < n$) and $w(u(x^m)) = d$ (since the coefficients of $u(x^m)$ are those of $u(x)$ interleaved with $m - 1$ zeros each time). Now, let $u(x) = f(x)p(x) \in C'$ be any non-zero polynomial. We have to prove that $w(u(x)) \geq d$. We have $\deg(p(x)) = \deg(u(x)) - \deg(f(x)) = \deg(u(x)) - (mn - mk) < mk$ as $\deg(u(x)) < mn$. Let us write

$$\begin{aligned} p(x) &= a_0 + a_1x + a_2x^2 + \cdots + a_{mk-1}x^{mk-1} \\ &= (a_0 + a_mx^m + a_{2m}x^{2m} + \cdots + a_{(k-1)m}x^{(k-1)m}) + \\ &\quad + (a_1x + a_{m+1}x^{m+1} + a_{2m+1}x^{2m+1} + \cdots + a_{(k-1)m+1}x^{(k-1)m+1}) + \\ &\quad \vdots \\ &\quad + (a_{m-1}x^{m-1} + a_{2m-1}x^{2m-1} + a_{3m-1}x^{3m-1} + \cdots + a_{mk-1}x^{mk-1}) \\ &= p_0(x^m) + p_1(x^m)x + \cdots + p_{m-1}(x^m)x^{m-1} \end{aligned}$$

where $p_i(x) = \sum_{j=0}^{k-1} a_{jm+i}x^j$ for each $i \in \{0, 1, \dots, m-1\}$. Thus, we have

$$\begin{aligned} u(x) &= f(x)p(x) \\ &= g(x^m)(p_0(x^m) + p_1(x^m)x + \dots + p_{m-1}(x^m)x_{m-1}) \\ &= g(x^m)p_0(x^m) + g(x^m)p_1(x^m)x + \dots + g(x^m)p_{m-1}(x^m)x^{m-1}. \end{aligned}$$

For each $i \in 0, 1, \dots, m-1$, writing $r_i(x) = g(x)p_i(x) \in C$ (since $\deg(r_i(x)) = \deg(g(x)) + \deg(p_i(x)) = n - k + \deg(p_i(x)) < n$ as $\deg(p_i(x)) < k$), we get

$$u(x) = r_0(x^m) + r_1(x^m)x + \dots + r_{m-1}(x^m)x^{m-1}.$$

Moreover, we clearly have $w(u(x)) = w(r_0(x)) + w(r_1(x)) + \dots + w(r_{m-1}(x))$. Finally, at least one of the polynomials $r_i(x)$ is non-zero (otherwise $u(x)$ would also be the zero polynomial), so we get from $w(C) = d$ that $w(r_i(x)) \geq d$, which implies the wanted $w(u(x)) \geq w(r_i(x)) \geq d$.

3.27.

- (a) Trivial. However it is important to note that if C is generated by $g(x) \sum_{i=0}^k a_i x_i$, \bar{C} is generated by its reciprocal polynomial $\bar{g}(x) = \sum_{i=0}^k a_{k-i} x_i$.
- (b) This is the full set of decompositions of $x_n - 1$ over \mathbb{F}_2 up to $n = 6$.

$$\begin{aligned} x^2 - 1 &\text{ is irreducible} \\ x^3 - 1 &= (x + 1)(x^2 + x + 1) \\ x^4 - 1 &= (x + 1)^4 \\ x^5 - 1 &= (x^4 + x^3 + x^2 + x + 1)(x + 1) \\ x^6 - 1 &= (x + 1)^2(x^2 + x + 1)^2 \end{aligned}$$

Note that each of the possible generating polynomials for these codeword sizes is a palindrome, i.e. $g(x) = \bar{g}(x)$ and therefore $C = \bar{C}$. For reference:

$$\begin{aligned} (x + 1)^2 &= x^2 + 1 \\ (x + 1)^3 &= x^3 + x^2 + x + 1 \\ (x + 1)^4 &= x^4 + 1 \\ (x + 1)^5 &= x^5 + x^4 + x + 1 \\ (x + 1)^6 &= x^6 + x^4 + x^2 + 1 \\ (x + 1)^7 &= x^7 + x^6 + x^5 + x^4 + x^3 + x^2 + x + 1 \\ (x + 1)^8 &= x^8 + 1 \\ (x^2 + x + 1)^2 &= x^4 + x^2 + 1 \end{aligned}$$

- (c) Since $x^7 - 1 = (x^3 + x + 1)(x^3 + x^2 + 1)(x + 1)$, the example is a code C is generated by $g(x) = x^3 + x + 1$ and \bar{C} is generated by $\bar{g}(x) = x^3 + x^2 + 1$.

3.28. This follows since $C_1 \subseteq C_2 \iff \langle g_1(x) \rangle \subseteq \langle g_2(x) \rangle \iff g_2(x)$ divides $g_1(x)$.

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 encrypted 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 possess 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 replaced 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 cryptanalytic 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 published 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 $M_n^{-1} \pmod{26}$ exists. Encryption is done by multiplication $c = M_n p$ and decryption by another multiplication $p = M_n^{-1} c$. 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 replaced 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 $c = aw + b \pmod{26}$; decryption by the computation $w = a^{-1}(c - b) \pmod{26}$, 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 i th 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 plaintext 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 .

4.1.5 Unicity distance

The unicity distance of a cipher encrypting natural language plaintexts is the minimum of cryptotexts required for computationally unlimited adversaries to decrypt cryptotext uniquely (to recover uniquely the key that was used).

The expected unicity distance $U_{C,K,L}$ of a cipher C and a key set K for a plaintext language L can be shown to be:

$$U_{C,K,L} = \frac{H_K}{D_L}$$

where H_K is the entropy of the key space (e.g 128 for 2^{128} equiprobably keys), D_L is the plaintext redundancy in bits per character (3.2 for English).

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 QSFCD F 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
 QDEEFDSZHQGAFFVDJDZNGSDDGFIGBSHGFFHQGAFF4GARFQGNSPDYKP
 GAFKSDAJHQZRAXCDSUDGZPDPDQDKNGKDZFYXQJDQNZRSHEGZYDGHQ
 TKDRVXGGAFF4GARFQGNSPKRGAFVDJDZNGSDSFRXJJFQYZGADGBXJFQ
 ZAXNCYZGNYP64DSGZHQRCNYHQTRXXVHQTYSFZZHQTJDZZDTFDQYGA
 FESFEDSDGHXQXMEFSMNJFZGAFCHZGDCZXHQRCNYFZZXJFCFZZXKUH
 XNZDSGZHQRCNYHQTRXQONSHQTRAFZZKXXVKHQYHQTDQYRDSEFQGSP
 QNJKFS45XQGAFFCHZGHZJCFRRAHGDUHVDCEDGAFDSGXMZFRSFGBSHG
 HQTDYUXRDGFYHQXSIFYSGXAFCEBXJFQRXQRFDCGAFYFGDHCZXMGAFH
 SCHDHZXQZXQFXMGAFSFRXJJFQYFYGFRAQHLNFZHQUXCUFZSDQYXJC
 PEDHSHQTCFGGFSZXMGAFDCEADKFGDQYGAFFQZNKZGHGNGHQTFDRACF
 GGFSHQGAFFXSHTHQDCJFZZDTFBHGAHGZEDSGQFS

* 4.8. A simple substitution cipher $f : \mathbb{Z}_{26} \rightarrow \mathbb{Z}_{26}$ (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. $f(f(x)) = x$.

- How many different self-inverting substitution ciphers are there?
- What is the proportion of self-inverting substitution ciphers to all simple substitution ciphers?
- 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.

- the Caesar cryptosystem;
- monoalphabetic substitution cryptosystem;
- transposition cryptosystem with a block length not greater than the number of symbols in the alphabet
- the Affine cryptosystem;
- the Vigenère cryptosystem with a key of known length.

4.10. Find the unicity distance of the following ciphers, consider the keys are chosen uniformly at random.

- the original pigpen cipher;
- the Vigenere cipher with keylength 7;
- transposition cipher with period 7;
- the one-time pad.

4.11. For 26-letter alphabet, determine how many Affine ciphers are there that leave

- no characters fixed;
- exactly one character fixed;

(c) at least two characters fixed.

4.12. Consider the Affine cryptosystem with the encryption function $e(x) = ax + b \pmod{26}$ where $a, b \in \{0, 1, \dots, 25\}$ and $\gcd(a, 26) = 1$.

Find all possible values of a, b such that for all $x \in \{0, 1, \dots, 25\}$ the following holds:

(a) $e(e(e(x))) = e^3(x) \equiv x \pmod{26}$;

(b) $e^5(x) \equiv x \pmod{26}$.

4.13. Consider a p -ary alphabet where p is prime. What is the size of the keyspace in

(a) the Affine cryptosystem?

(b) the Hill cryptosystem?

(c) and the Hill cryptosystem over the 26-ary alphabet?

4.14. 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.

* **4.15.** 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?

* **4.16.** 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 TJSYG LPCDS ROPCQ VYZLG MGMBV CCTMX HCXGC
SXKEW RLINY VRKFJ OELNM RYQK 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
```

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

4.18. 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.

4.19. 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.

4.20. To save resources used by sending the keys, the following simplified one-time pad cryptosystem was implemented. The cryptosystem uses a randomly generated key k_1 during its first use, but to encrypt the i th plaintext w_i , $i > 1$, instead of using new key k_i , the previous plaintext is used, i.e. $k_i = w_{i-1}$, $i > 1$. You managed to intercept all the cryptotexts c_i created by this cryptosystem and also the third plaintext w_3 . Find the first plaintext w_1 and the original key k_1 .

4.21. Consider two Hill cryptosystems described with matrices G and H .

(a) Consider another Hill cryptosystem with matrix M , constructed from G and H , such that

$$e_M(m) = e_H(e_G(m)) \text{ and } d_M(c) = d_G(d_H(c)).$$

Determine M and M^{-1} in terms of G and H .

(b) Prove that if both H and G set up valid Hill cryptosystem, the cryptosystem from (a) is also valid.

4.22. Consider the Gronsfeld cipher. Assuming the corresponding plaintext contains word 'the'. Decrypt the following cryptotext. Do not use brute force.

PFJWBWYIJHYNW

The Gronsfeld cipher is a variant of the Vigenère cipher where numbers $0, \dots, 9$ are used as the key instead of letters. Each plaintext character is shifted along by the corresponding number from the key.

* **4.23.** 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.

4.24. 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.

* **4.25.** 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.

4.26. Suppose that in a symmetric key cryptosystem $P = C = \{0, \dots, n - 1\}$. Suppose that the possible encryption functions are all the permutations of P .

- (a) What is the size of the key set K ?
 (b) Show that if the encryption functions are chosen uniformly, then this cryptosystem achieves perfect secrecy.

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 $c \rightarrow AH$, $r \rightarrow DG$, and so on until we get the resulting cryptotext “AHDGEICKDICFCIBGEI”.

- (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.

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

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

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

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

The cryptotext is therefore “BLPDEDIFWW”.

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

abcdefghijklmnopqrstu
vwxyzSHIFABCDEGJKL

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.

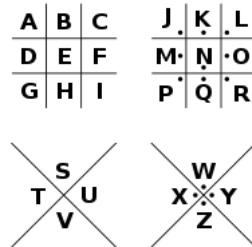
Let π be a permutation of the set $\{1, \dots, n\}$. We can define an $n \times n$ permutation matrix $M_\pi = (m_{ij})$ as follows:

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

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 $e_{(a,b)}(o) = Z$ and $e_{(a,b)}(T) = I$ 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 occurrences the most frequent trigram, followed up with the trigram HQT with 10 occurrences. 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 $t \rightarrow G$ and $h \rightarrow A$ we get $a = 7$ and $b = 3$. These coefficients are valid for transformation $e \rightarrow F$ (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 $f(0) = i$, $i \in \{1, \dots, n - 1\}$, 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

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

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.

We use the redundancy of English text ≈ 3.2 bits per character.

- (a) The **original pig pen cipher** has only one key thus the unicity distance is $\log_2 1/3.2 = 0$.
- (b) **Vigènere with key-length 7** has 26^7 possible keys and the unicity distance is therefore $\log_2 26^7/3.2 \approx 10.3$.
- (c) **Transposition cipher of period 7** has the size of the key space $7!$ and the unicity distance is $\log_2 7!/3.2 \approx 3.84$.
- (d) **The one-time pad** has unbounded key size, i.e., for a plaintext with N bites the size of the key space is 2^N and $\lim_{N \rightarrow \infty} 2^N = \infty$. Thus, the unicity distance is ∞ .

4.11.

- (a) There are 12 possible values for a (because $\gcd(a, 26) = 1$ therefore a is odd) and 26 possible values for b , ie. $12 \cdot 26 = 312$ keys in total.
In order that $x \in \{0, \dots, 25\}$ is fixed, it must hold that $ax + b \equiv x \pmod{26}$. We are interested in the case where $ax + b \not\equiv x \pmod{26}$ which can be rewritten as

$$(a-1)x \not\equiv -b \pmod{26}.$$

For $a = 1$ this allow $b \in \{1, \dots, 25\}$. Now, we use the fact that the congruence $e \cdot x \equiv f \pmod{n}$ has a solution if and only if $\gcd(e, n) | f$, moreover the number of unique solutions is equal to $\gcd(e, n)$.

For all possible $a > 1$ we have $\gcd(a-1, 26) = 2$ and therefore we have to choose b odd so as the congruence has no solution. Together, we have $25 + 11 \cdot 13 = 168$ keys such that no characters are fixed.

- (b) We can use the fact from part (a), ie. that the number of solutions is equal to $\gcd(a-1, 26)$ for $a > 1$. For $a = 0$ and $b = 0$ there are 26 solutions. There is no key with exactly one solution, ie. one fixed character.

- (c) We need to subtract results from (a) and (b) from the total number of keys: $312 - (168 + 0) = 144$.

4.12. First a quick observation: if $cx + d \equiv 0 \pmod{26}$ for all $x \in \{0, 1, \dots, 25\}$, then necessarily $c \equiv 0 \pmod{26}$ (otherwise cx would take at least two different values for varying $x \in \{0, 1, \dots, 25\}$, and the operation of adding d is invertible) and consequently also $d \equiv 0 \pmod{26}$.

- (a) The equality $e(e(e(x))) = x$ can be rewritten as $(a^3 - 1)x + b(a^2 + a + 1) \equiv 0 \pmod{26}$. By the observation above, this implies that $a^3 - 1 \equiv 0 \pmod{26}$ and $b(a^2 + a + 1) \equiv 0 \pmod{26}$. Since $\gcd(3, \varphi(26)) = 3$, there are exactly three third roots of unity modulo 26, namely 1, 3, 9. By solving the second congruence, we can see that all the possible pairs (a, b) are $(1, 0)$, $(3, 0)$, $(3, 2k)$, $(9, 2k)$, where $k \in \{1, 2, \dots, 12\}$ can be arbitrary.
- (b) The equality $e^5(x) = x$ can be rewritten as $(a^5 - 1)x + b(a^4 + a^3 + a^2 + a + 1) \equiv 0 \pmod{26}$. By the observation above, this implies $a^5 - 1 \equiv 0 \pmod{26}$ and $b(a^4 + a^3 + a^2 + a + 1) \equiv 0 \pmod{26}$. But since $\gcd(5, \varphi(26)) = 1$, the map of taking the fifth power modulo 26 is injective, hence the only fifth root of unity modulo 26 is 1. Consequently the only possible pair (a, b) is $(1, 0)$.

4.13.

- (a) In affine cryptosystem the keys are tuples $(a, b) \in \mathbb{Z} \times \mathbb{Z}$ such that $0 \leq a, b < p$ and $\gcd(a, p) = 1$. This gives us p possible values for b and $\varphi(p)$ possible values for a where φ denotes Euler's totient function. Overall it's

$$p \cdot \varphi(p) = p(p - 1).$$

- (b) In Hill cryptosystem the keys are invertible $n \times n$ matrices. Since \mathbb{Z}_p is a field, the invertible matrices are exactly those of rank n , i.e. they have n linearly independent rows.

For the first row we have $p^n - 1$ possible values (all non-zero p -ary words of length n). For the second row we have $p^n - p$ possible values (all p -ary words of length n that are linearly independent from the first row). In the third row we have $p^n - p^2$ possible values and so on. Overall it's

$$\prod_{i=0}^{n-1} (p^n - p^i).$$

- (c) Let $M_n(R)$ denote the ring of $n \times n$ matrices over the ring R . From Chinese remainder theorem for rings we get the isomorphism

$$M_n(\mathbb{Z}_{26}) \cong M_n(\mathbb{Z}_2) \times M_n(\mathbb{Z}_{13}).$$

Let $A = (A_1, A_2) \in M_n(\mathbb{Z}_2) \times M_n(\mathbb{Z}_{13})$ be some invertible matrix. Its inverse has to be of the form $A^{-1} = (A_1^{-1}, A_2^{-1}) \in M_n(\mathbb{Z}_2) \times M_n(\mathbb{Z}_{13})$, i.e. the matrix A is invertible modulo 26 iff A_1 is invertible modulo 2 and A_2 is invertible modulo 13.

There are $\prod_{i=0}^{n-1} (2^n - 2^i)$ possible values for A_1 and $\prod_{i=0}^{n-1} (13^n - 13^i)$ possible values for A_2 . Overall there are

$$\prod_{i=0}^{n-1} (2^n - 2^i)(13^n - 13^i)$$

invertible $n \times n$ matrices over the 26-ary alphabet.

4.14.

- (a) The keyspace is the set $\{(a, b) \in \mathbb{Z}_{126} \times \mathbb{Z}_{126} \mid \gcd(a, 126) = 1\}$. 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 $23^{-1} = 11 \pmod{126}$.

$$d(y) = 11(y - 7) \pmod{126}$$

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

4.15. One has to find 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.

4.16. 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 concurrence 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.

4.17. (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 doutkala 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 maminciny 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 vážly, pak však došlo k zlomu. Vymyslili 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 :).

4.18. 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 $k_1 = (k_{1_1}, \dots, k_{1_m})$, $k_2 = (k_{2_1}, \dots, k_{2_m})$ be keys of length m . Then $k = (k_{1_1} + k_{2_1} \pmod{n}, \dots, k_{1_m} + k_{2_m} \pmod{n}) \in K$.
- (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 $k = (a_1 a_2, a_1 b_2 + b_1) \in K$,

4.19. 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.

4.20. Using the definition of the cryptosystem we have

$$w_1 \oplus k_1 = c_1$$

$$w_2 \oplus w_1 = c_2$$

$$w_3 \oplus w_2 = c_3$$

Taking the sum of all 3 equations we receive

$$w_3 \oplus k_1 = c_1 \oplus c_2 \oplus c_3,$$

which gets us $k_1 = c_1 \oplus c_2 \oplus c_3 \oplus w_3$ and we know everything on the RHS so we now know k_1 . To get w_1 we can add the second and third equation to get

$$w_3 \oplus w_1 = c_2 \oplus c_3,$$

which lets us get the first plaintext as $w_1 = c_2 \oplus c_3 \oplus w_3$.

4.21. The solutions can be obtained by using properties of linear transformations.

- (a) The composition of two linear transformations can be obtained by multiplication of their matrices in the reversed order, i.e.

$$M = H \cdot G.$$

Also it holds that

$$(H \cdot G)^{-1} = G^{-1} \cdot H^{-1} = M^{-1}.$$

- (b) From linear algebra we have that $\det(A \cdot B) = \det(A) \cdot \det(B)$. And the matrix A is invertible mod 26 if and only if $\gcd(\det(A), 26) = 1$. The multiplication preserves this property.

4.22. We assume that the plaintext contains the word 'the'.

The letter "T" is encrypted as one of the letters from the set {T, U, V, W, X, Y, Z, A, B, C}.

The letter "H" is encrypted as one of the letters from the set {H, I, J, K, L, M, N, O, P, Q}.

The letter "E" is encrypted as one of the letters from the set {E, F, G, H, I, J, K, L, M, N}.

The only consecutive three letters in the ciphertext satisfying these conditions are WIT. Subtracting the cryptotext WIT and the plaintext THE results in the possible key 315. Applying this key to the whole ciphertext yields the plaintext MEET AT THE EXIT.

4.23.

- (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:

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

Therefore

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

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.

4.24.

(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).

4.25.

(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.

4.26.

1. Size of K is factorial $n!$

2. to show perfect secrecy we first need to calculate both $p_C(c)$ and $p_{C|P}(c|w)$ for each pair of messages c and w . Let us first start with $p_C(c)$ for each $w \in P$ there exist exactly $(n-1)!$ permutations that map w to c . Therefore $p_C(c)$ can be written as

$$\begin{aligned} p_C(c) &= \sum_{w \in P} p_P(w) \frac{(n-1)!}{n!} \\ &= \frac{(n-1)!}{n!} \sum_{w \in P} p_P(w) \\ &= \frac{(n-1)!}{n!} = \frac{1}{n}, \end{aligned}$$

where $\frac{(n-1)!}{n!}$ is the probability to choose a key that maps w to c and the second to last equality follows from the fact that probabilities sum up to 1. Let us now examine $p_{C|P}(c|w)$. For a fixed pair c and w this is the probability that a key mapping w to c is chosen. As argued before this probability is $\frac{(n-1)!}{n!} = \frac{1}{n}$. Now from the Bayes' rule it follows that $p_{P|C}(w|c) = \frac{p_P(w) \frac{1}{n}}{\frac{1}{n}} = p_P(w)$, which shows perfect secrecy.

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

5.1.2 Blom's key pre-distribution protocol

Let a large prime $p > n$ be publicly known. Steps of the protocol follow:

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

- If Alice (A) wants to send a message to Bob (B), then Alice computes her key $K_{AB} = g_A(r_B)$ and Bob computes his key $K_{BA} = g_B(r_A)$.

5.1.3 Knapsack cryptosystem

Knapsack cryptosystem is based on the infeasibility of solving the general *knapsack 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 $X' = (x'_1, \dots, x'_n)$, $x'_i = ux_i \pmod{m}$.

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 $u^{-1} \pmod{m}$, $c' = u^{-1}c \pmod{m}$ and solve the knapsack problem with X and c' .

5.1.4 McEliece cryptosystem

Just like in the knapsack cryptosystem, McEliece cryptosystem is based on transforming 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 $w = w_1S^{-1}$.

5.1.5 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 $e = d^{-1} \pmod{\phi(n)}$. 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 $c = w^e \pmod{n}$.

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

5.1.6 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 $1 \leq x < n$, by $C(x)$ the condition:*

$$x^d \not\equiv 1 \pmod{n} \text{ and } x^{2^r d} \not\equiv -1 \pmod{n} \text{ for all } 0 \leq r < s.$$

If $C(x)$ holds for some $1 \leq x < n$, 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) Use the Euclidean algorithm to find the $\gcd(4757, 4087)$.
- (b) Use the extended Euclidean algorithm to find an inverse of 97 in $(\mathbb{Z}_{977}, \cdot)$.

5.2.

- (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.

5.3. 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.

5.4. 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?

5.5. Consider the Blom'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.

5.6. In the Blom's key pre-distribution protocol (see the lecture slides) show that the key generated by A and B is the same, *i.e.* show that $K_{AB} = K_{BA}$.

5.7. 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 $c_2 = 383$.

5.8. You are given RSA modulus $n = 53916647$. Determine p and q knowing the fact that the difference between factors is small.

5.9. Let $n = pq$ where p, q are primes with $p > q$.

1. Show that $p - q = \sqrt{(p + q)^2 - 4n}$.

2. Express p and q in terms of n and $\varphi(n)$.

5.10. 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 $c = 0110110$.

5.11. 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.

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

5.13. 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.

5.14. Suppose that Eve receives a cryptotext $c = m^e \pmod{n}$ 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 .

5.15. 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.

5.16. 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.

5.17. Consider the following public-key cryptosystem that allows Bob to send encrypted message m to Alice:

- Alice chooses Galois field \mathbb{F}_q .
- Alice chooses l polynomials in n variables P_1, \dots, P_l , such that $P_i(v_1, \dots, v_n) = 0$ for some $v = (v_1, \dots, v_n) \in \mathbb{F}_q^n$, for all $1 \leq i \leq l$.
- Alice makes \mathbb{F}_q and (P_1, \dots, P_l) public.
- To send a message m , Bob chooses l polynomials with n variables Q_1, \dots, Q_l and encrypts m using the function

$$f : m \mapsto f(m) = m + \sum_{j=1}^l Q_j P_j.$$

- Bob sends $f(m)$, the polynomial that is the encrypted message m , to Alice.

The function f is a trapdoor function. Find the decryption process and the trapdoor information Alice needs to perform the decryption.

* **5.18.** 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 $c_B = 14702$.

* **5.19.** 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:

16278490204355400279.

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

* **5.20.** Let (e, n_1) and (e, n_2) be Alice's and Bob's RSA public keys and let their encryption exponent be $e = 3$. Charlotte sends both of them the same short secret message m . Suppose n_1 and n_2 are coprimes and $m^e \ll n_1 n_2$.

- (a) Show how Eve, who intercepted both cryptotexts, reconstructs m . (Do not use brute force.)
- (b) Calculate m given public moduli $n_1 = 1363$, $n_2 = 2419$ and cryptotexts $c_1 = 18$ and $c_2 = 325$.

* **5.21.** Consider an RSA cryptosystem with public encryption key $e = 3$ and modulus of length 4096 bits where plaintexts are encrypted in blocks of length 1365 bits. Explain why this system is not secure.

5.22. Consider the Diffie-Hellman key exchange protocol with $p > 5$ a safe prime, i.e. there exists a prime r such that $p = 2r + 1$. Let q be a primitive root modulo p . Suppose that Alice and Bob both choose their secret exponents x, y uniformly in the range $1 \leq x, y \leq p - 2$. Calculate the probability that the shared secret $q^{xy} \pmod p$ is equal to 1.

5.3 Solutions

5.1.

(a) We use the Euclidean algorithm to find $\gcd(4757, 4087)$:

$$4757 = 1 \cdot 4087 + 670$$

$$4087 = 6 \cdot 670 + 67$$

$$670 = 10 \cdot 67 + 0.$$

We got a remainder of 0 in the last step so $\gcd(4757, 4087) = 67$.

(b) We are looking for the inverse of 97 in $(\mathbb{Z}_{977}, \cdot)$, which means looking for $x \in \mathbb{Z}_{977}$ such that $97x \equiv 1 \pmod{977}$ which is equivalent to $97x + 977y = 1$ for some $y \in \mathbb{Z}$, which is the Bezout's identity. We can use the extended Euclidean algorithm to find x and y . First we use the normal Euclidean algorithm:

$$977 = 10 \cdot 97 + 7$$

$$97 = 13 \cdot 7 + 6$$

$$7 = 1 \cdot 6 + 1.$$

To make things easier, we now express the individual remainders in each of the equation above:

$$7 = 977 - 10 \cdot 97$$

$$6 = 97 - 13 \cdot 7$$

$$1 = 7 - 1 \cdot 6.$$

Now we just traverse the equations obtained from the basic Euclidean algorithm in reverse:

$$\begin{aligned} 1 &= 7 - 1 \cdot 6 = 7 - 1 \cdot (97 - 13 \cdot 7) = 14 \cdot 7 - 1 \cdot 97 \\ &= 14 \cdot (977 - 10 \cdot 97) - 1 \cdot 97 = 14 \cdot 977 - 141 \cdot 97, \end{aligned}$$

Which gives us $97^{-1} \equiv -141 \equiv 836 \pmod{977}$.

5.2.

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

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

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

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

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

It can easily be seen that $k = k_A = k_B = k_C = q^{xyz} \pmod p$ 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.

5.3. 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.

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

$$X = qx \pmod p$$

$$Y = qy \pmod p.$$

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 $k = xyq \pmod p$. This simplification is therefore not secure as the inverse to multiplication by known number is easy to compute.

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

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

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

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

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

And Bob does the same

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

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

5.6.

$$\begin{aligned}
K_{AB} &= g_A(r_B) \\
&= a_A + b_A r_B \\
&= a + br_A + (b + cr_A)r_B \\
&= a + br_A + br_B + cr_A r_B \\
&= a + br_B + (b + cr_B)r_A \\
&= a_B + b_B r_A \\
&= g_B(r_A) \\
&= K_{BA}.
\end{aligned}$$

5.7.

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

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

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

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 $w_2 = 011011$.

5.8. If difference $|p - q|$ is small then in order to factor n , it is enough to test $x > \sqrt{n}$ until x is found such that $x^2 - n$ is a square, say y^2 . In such a case, $p + q = 2x$ and $p - q = 2y$ and therefore

$$p = x + y,$$

$$q = x - y.$$

In our case $\sqrt{53916647} = 7342.795584789$, so we test $7343^2 - 53916647 = 3002$ that is not a square. Next, we try $7344^2 - 53916647 = 17689 = 133^2$. Therefore, $p = 7344 + 133 = 7477$ and $q = 7344 - 133 = 7211$.

5.9.

1.

$$\begin{aligned}
n &= pq \\
-4n + p^2 + q^2 &= -4pq + p^2 + q^2 \\
-4n + p^2 + 2pq + q^2 &= p^2 - 2pq + q^2 \\
(p + q)^2 - 4n &= (p - q)^2 \\
p - q &= \sqrt{(p + q)^2 - 4n}
\end{aligned}$$

(We know the sign of the square root since $p > q$.)

2. We have a nonlinear system of two equations and two variables, p and q :

$$\begin{aligned}
n &= pq, \\
\phi(n) &= (p - 1)(q - 1).
\end{aligned}$$

From these we immediately get $(p + q) = pq - (p - 1)(q - 1) + 1 = n - \phi(n) + 1$. Now using the formula from the previous exercise we can obtain the value of $(p - q)$ and then

$$\begin{aligned}
p &= \frac{(p + q) + (p - q)}{2} \\
&= \frac{(p + q) + \sqrt{(p + q)^2 - 4n}}{2} \\
&= \frac{n - \phi(n) + 1 + \sqrt{(n - \phi(n) + 1)^2 - 4n}}{2}
\end{aligned}$$

and

$$\begin{aligned}
q &= \frac{(p + q) - (p - q)}{2} \\
&= \frac{(p + q) - \sqrt{(p + q)^2 - 4n}}{2} \\
&= \frac{n - \phi(n) + 1 - \sqrt{(n - \phi(n) + 1)^2 - 4n}}{2}.
\end{aligned}$$

5.10.

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

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

(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_1S^{-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 $w = w_1S^{-1} = (1101)$.

5.11.

(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 $e = d^{-1} \pmod{\phi(n)}$. 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 :

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

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 $c^d \pmod{n}$:

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

So the whole decrypted message is 397569169.

5.13. 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 $p = pq - 1287 - q = 1363 - 1287 - q = 76 - q$. Plugging this into to the second equation we get the quadratic equation $q^2 - 76q + 1363 = 0$. 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 $1363 = 29 \cdot 47$.

5.13. 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.

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

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

5.15. 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

$$\begin{aligned} \gcd(8844679, 11316499) &= 3169 \text{ which gives us a divisor of } 8844679 \text{ and } 11316499. \\ \gcd(18761893, 21799573) &= 4219 \text{ which gives us a divisor of } 18761893 \text{ and } 21799573. \\ \gcd(22862761, 18761893) &= 4219. \text{ which gives us a divisor of } 22862761. \\ \gcd(43831027, 58354333) &= 7057 \text{ which gives us a divisor of } 43831027 \text{ and } 58354333. \end{aligned}$$

5.16. 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 $x^d \pmod n$ for all the integers:

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

Already we can see that $C(x_3)$ does not hold. The possible r for the second part of the condition $x^{2^r d} \pmod n$ are $r_0 = 0$, $r_1 = 1$, and $r_2 = 2$. From the previous calculation, we can see that the condition is not violated for r_0 and we compute

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

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

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 $\frac{1}{8}$.

5.17. The trapdoor information is $v = (v_1, \dots, v_n)$ because using it Alice can do the following with the encrypted message $f(m)$:

$$f(m)(v) = m + \sum_{j=1}^l Q_j(v)P_j(v) = m + \sum_{j=1}^l Q_j(v) \cdot 0 = m$$

to get the decrypted message.

5.18.

- (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

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

- (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 :

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

5.19. 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

$$m x + e_B y = 1,$$

for some integers x , y . Using this identity we have $e_B y \equiv 1 - m x \equiv 1 \pmod{\phi(n)}$ 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:

$$16278^{2807399} \pmod{99443} = 73327$$

$$49020^{2807399} \pmod{99443} = 67986$$

$$43554^{2807399} \pmod{99443} = 69328$$

$$279^{2807399} \pmod{99443} = 97985$$

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

5.20.

1. Eve intercepted two cryptotexts c_1 and c_2 . Since the message is very short $m^e \ll n_1 n_2$ and $\gcd(n_1, n_2) = 1$, she can use Chinese remainder theorem to reconstruct m^3 which is a unique solution $\pmod{n_1 n_2}$ of the system of simultaneous congruences:

$$m^3 \equiv c_1 \pmod{n_1} \tag{5.1}$$

$$m^3 \equiv c_2 \pmod{n_2}.$$

The solution is

$$m^3 = c_1 n_2 (n_2^{-1} \pmod{n_1}) + c_2 n_1 (n_1^{-1} \pmod{n_2}) \pmod{n_1 n_2}. \tag{5.2}$$

- 1) It is a valid solution, since:

$$m^3 \equiv c_1 n_2 (n_2^{-1} \pmod{n_1}) + c_2 n_1 (n_1^{-1} \pmod{n_2}) \equiv c_1 n_2 (n_2^{-1} \pmod{n_1}) \equiv c_1 \pmod{n_1}.$$

Similarly for $\pmod{n_2}$.

- 2) Consider there being another solution to (5.1) we will denote it by x . Since it is a solution, it has the same remainder $\pmod{n_1}$ and therefore $m^3 - x$ is a multiple of n_1 , and similarly for n_2 . Moreover since n_1 and n_2 are coprimes, it has to be multiple of $n_1 n_2$. Therefore $x \equiv m^3 \pmod{n_1 n_2}$ and m^3 is a unique solution modulo $n_1 n_2$.

Since the message is very short it is enough to calculate integer cube root to recover m .

2. We use the equation (5.2) to calculate $m^3 = 18 \times 2419 \times 626 + 325 \times 1363 \times 1308 = 2744 \pmod{2419 \times 1363}$. We calculate integer cube root and receive the secret message 14.

5.21. The maximum cryptotext value for message of length of 1365 bits is $(2^{1365} - 1)^3$. This is obviously less than $(2^{1365})^3 = 2^{4095}$ which is the minimal value for key of length 4096. Therefore no modular reduction happens and we can easily calculate the third root of encrypted message to get the plaintext.

5.22. Since q is a primitive element of the \mathbb{Z}_p^* , we have $q^{xy} = 1$ in \mathbb{Z}_p^* if and only if $p - 1 | xy$, i.e. $2r | xy$. Since r is a prime and $1 \leq x, y \leq p - 2 = 2r - 1 < 2r$, we get that one of x, y must be r . Then, it is clearly necessary and sufficient that the other integer be even. Together, the probability is

$$2 \cdot \frac{1}{2r-1} \cdot \frac{r-1}{2r-1} = \frac{2(r-1)}{(2r-1)^2} = \frac{p-3}{(p-2)^2}.$$

where $\frac{1}{2r-1}$ is the probability of choosing $x = r$, $\frac{r-1}{2r-1}$ is the probability of choosing even y . It can be vice versa as well, therefore the multiplication by 2; note that r is odd, so we do not count any possibility twice.

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 $c = w^2 \pmod n$.

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 $M = \prod_{i=1}^t m_i$, $M_i = \frac{M}{m_i}$, $N_i = M_i^{-1} \pmod{m_i}$ 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 $y = q^x \pmod p$. 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 $a = q^r \pmod p$ and $b = y^r w \pmod p$. The ciphertext is then $c = (a, b)$.

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

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 $q^{mj} \pmod p$, for all $0 \leq j \leq m - 1$.
- Create the list L_1 of m pairs $(j, q^{mj} \pmod p)$, $0 \leq j \leq m - 1$, sorted by the second item.
- Compute $yq^{-i} \pmod p$, for all $0 \leq i \leq m - 1$.
- Create the list L_2 of m pairs $(i, yq^{-i} \pmod p)$, $0 \leq j \leq m - 1$, 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 $f : \mathbb{N} \rightarrow \mathbb{R}$ 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 $s_0 \in \{0, 1, \dots, n\}$
- For $i = 1, 2, \dots, m + 1$ compute $s_i = s_{i-1}^2 \pmod n$ and σ_i , the least significant bit of s_i .

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

Decryption of the ciphertext (r, y)

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

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

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. Let $f(n)$ be a negligible function and $g(n)$ not be a negligible function. Show that

$$g(n) - f(n)$$

is not negligible.

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

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

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

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

6.8. 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 $(2218, 1001)$.

6.9. Consider any two strongly collision resistant hash functions $h_1 : \{0, 1\}^n \rightarrow \{0, 1\}^m$ and $h_2 : \{0, 1\}^n \rightarrow \{0, 1\}^m$ such that $h_1(x) \neq h_2(x)$ for any $x \in \{0, 1\}^n$. Now consider the following hash functions $h : \{0, 1\}^n \rightarrow \{0, 1\}^m$ and $h' : \{0, 1\}^n \rightarrow \{0, 1\}^{2m}$:

$$h(x) = h_1(x) \oplus h_2(x),$$

$$h'(x) = h_1(x) \parallel h(x).$$

Determine whether h, h' has to be

1. pre-image resistant,
2. weakly collision resistant,
3. strongly collision resistant.

Explain your reasoning.

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

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

where x_1 is the first half of x and x_2 is the second half of x . Prove that h' is strongly collision-free hash function.

* **6.11.** Suppose you know a valid plaintext-ciphertext pair $w_1 = 457, (a_1, b_1) = (663, 2138)$, constructed using ElGamal cryptosystem with public information $p = 6661, q = 6, y = 6015$. You also know that instead of using a new random r to encrypt each new message, the sender just increments the previous one, i.e. $r_2 = r_1 + 1$.

- (a) With this knowledge, decrypt the following ciphertext $(a_2, b_2) = (3978, 1466)$ without calculating discrete logarithms.
- (b) Show that the same attack is possible for any linear update function of the random seed, i.e. whenever $r_2 = kr_1 + \ell \pmod{p-1}$.

* **6.12.** 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

$$c = a^x \pmod{p}.$$

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

6.13. 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.

* **6.14.** 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 $a = \frac{x_1 + x_2 + \dots + x_n}{n}$ without revealing any other information about any x_i . Observe that a might not be an integer.

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

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

* **6.16.** What is the smallest number of people in a group so that the probability that two people in the group have birthday within the interval of k days is at least $\frac{1}{2}$? Calculate this number for $k = 1, \dots, 15$.

* **6.17.** Determine all odd quadratic residues modulo 2^n for $n \geq 3$, i.e. odd numbers k such that

$$x^2 \equiv k \pmod{2^n}$$

has a solution for $x \in \mathbb{Z}$.

- (a) Find all odd quadratic residues modulo 8.
- (b) Show that for $n > 3$, the congruence $x^2 \equiv k \pmod{2^n}$ has either zero or exactly four solutions for $x \in \mathbb{Z}$. *Hint:* You can use without proof the fact that for $n > 3$, any odd positive integer $m < 2^n$ satisfies the congruence $m \equiv (-1)^{e_1} 5^{e_2} \pmod{2^n}$ for a unique $e_1 \in \{0, 1\}, e_2 \in \{0, 1, \dots, 2^{n-2}\}$.
- (c) Using (a) and (b), describe all odd quadratic residues modulo 2^n for $n > 3$ by a single congruence.

* **6.18.** Using a primitive root of \mathbb{Z}_{43}^* , solve the following congruence

$$x^{19} \equiv 38 \pmod{43}.$$

Avoid the exhaustive search for a primitive root.

* **6.19.** Consider the following cryptosystem. Let $n = pq$ where p and q are primes. The value n is made public, $(n, \phi(n))$ forms private key.

- *Encryption:* To encrypt a message $m \in \mathbb{Z}_n$, choose a random $r \in \mathbb{Z}_n^*$ and compute

$$c = (1 + n)^m r^n \pmod{n^2}$$

- *Decryption:* To decrypt a ciphertext c , compute

$$m = \frac{(c^{\phi(n)} \bmod n^2) - 1}{n} \phi(n)^{-1} \bmod n$$

where integer division is used.

- Let $n = 3953$. Use $r = 1111$ to encrypt $m = 2019$. Decrypt your ciphertext using the fact $n = 59 \cdot 67$.
- Decrypt $c = 4354044$ without using the private key, only with the knowledge of the plaintext-ciphertext pair from (a).
- Prove the following fact that the described cryptosystem exploits: For integers n and a , $0 \leq a \leq n$, prove that $(1 + n)^a = 1 + an \pmod{n^2}$.

* **6.20.** Consider the following cryptosystem:

- *Key generation:* Let k be an integer. Pick two different odd primes p and q , an element $e \in \mathbb{Z}_n$ such that $\gcd(e, \phi(n)) = 1$. Let $n = pq$ and $d = e^{-1} \bmod \phi(n)$.
- *Public key:* (e, n)
- *Secret key:* (d, n)
- *Encryption:* To encrypt a message $m \in \mathbb{Z}_n$, one picks a random $r \in \mathbb{Z}_n^*$ and computes the ciphertext $c = r^e(1 + mn) \bmod n^2$.

Find the decryption algorithm.

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

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

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

$$s = a_1 M_1 N_1 + a_2 M_2 N_2 \bmod n,$$

where

$$\begin{aligned} M_1 &= \frac{n}{m_1} = 13, & M_2 &= \frac{n}{m_2} = 11, \\ N_1 &= M_1^{-1} \bmod m_1 = 6, & N_2 &= M_2^{-1} \bmod m_2 = 6, \\ a_1 &= s \bmod 11, & a_2 &= s \bmod 13. \end{aligned}$$

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 below

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 choose a random x and ask for the decryption of $c = x^2 \pmod n$. 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

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

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

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

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

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

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

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

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

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

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 $c' = (a_1 a_2, n b_1 b_2) = (q^{r_1} q^{r_2}, n y^{r_1} y^{r_2} m_1 m_2)$, $n \in \mathbb{Z}_p^*$. Decrypting c' we get

$$n b_1 b_2 (a_1 a_2)^{-x} = n y^{r_1} y^{r_2} m_1 m_2 (q^{r_1} q^{r_2})^{-x} = n q^{x(r_1+r_2)} m_1 m_2 q^{-x(r_1+r_2)} = n m_1 m_2.$$

So c' is the correct encryption of the message $n m_1 m_2$ 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 $(j, q^{mj} \pmod p)$, 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
$q^{mj} \pmod p$	1	117	65	7	33	62	49	100	41	81	45	25

Next we create the list L_2 of m tuples $(i, yq^{-i} \pmod p)$, 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
$yq^{-i} \pmod p$	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

$$\begin{aligned} (5, 62) & \quad (10, 62), \\ (11, 25) & \quad (3, 25), \\ (0, 1) & \quad (5, 1). \end{aligned}$$

This gives us three solutions to the discrete logarithm in the form $\log_q y = mj + i$. The first pair gives us

$$\log_5 112 = (12 \cdot 5 + 10) \pmod{130} = 70.$$

The second

$$\log_5 112 = (12 \cdot 11 + 3) \pmod{130} = 5.$$

And the third pair gives the same solution as the second one

$$\log_5 112 = (12 \cdot 0 + 5) \pmod{130} = 5.$$

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. Let $f(n)$ be a negligible function, $g(n)$ be a not negligible function and

$$h(n) = g(n) - f(n)$$

be negligible. Then $g(n) = h(n) + f(n)$. We will show that the sum of two negligible functions has to be a negligible function.

Since $h(n)$ and $f(n)$ are negligible we know that $\forall c \exists n_0$ such that $\forall n \geq n_0 \quad f(n), h(n) < n^{-(c+1)}$. Now $\forall n \geq n_0$ it holds $f(n) + h(n) \leq n^{-(c+1)} + n^{-(c+1)} = 2n^{-(c+1)} \leq n \cdot n^{-(c+1)} = n^{-c}$. (This only holds if $n_0 \geq 2$ but if that's not true, we just take $\forall n \geq 2$ instead). But this is just the definition of $f(n) + h(n)$ being negligible. This gives us a contradiction and thus $g(n) - f(n)$ must be not negligible.

6.7. Because for every $\rho, \rho' \in G$ it holds $\text{Im } \rho, \text{Im } \rho' \subseteq \mathbb{N}$ it also holds $\text{Im}(\rho \circ \rho') \subseteq \mathbb{N}$. 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 $n \in \mathbb{N}$

$$\begin{aligned} ((f \circ g) \circ h)(n) &= (f \circ g)(h(n)) = f(g(h(n))) \\ (f \circ (g \circ h))(n) &= f((g \circ h)(n)) = f(g(h(n))) \end{aligned}$$

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 $\text{Im } \rho \subseteq \mathbb{N}$.

6.8.

- (a) The public key is $n = pq = 43 \cdot 59 = 2537$. We need to calculate s_i for $i \in \{1, 2, \dots, 5\}$ using the formula $s_i = s_{i-1}^2 \pmod n$.

$$\begin{aligned} s_1 &= 1337^2 \pmod{2537} = 1521 \\ s_2 &= 1521^2 \pmod{2537} = 2234 \\ s_3 &= 2234^2 \pmod{2537} = 477 \\ s_4 &= 477^2 \pmod{2537} = 1736 \\ s_5 &= 1736^2 \pmod{2537} = 2277. \end{aligned}$$

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

$$\begin{aligned} r_p &= r^{((p+1)/4)^m} \pmod p = 2218^{(11^4)} \pmod{43} = 13 \\ r_q &= r^{((q+1)/4)^m} \pmod q = 2218^{(15^4)} \pmod{59} = 46. \end{aligned}$$

Now we obtain s_1 by computing

$$s_1 = q(q^{-1} \pmod p)r_p + p(p^{-1} \pmod q)r_q \pmod n = 59 \cdot 35 \cdot 43 + 43 \cdot 11 \cdot 21 \pmod{2537} = 400.$$

We continue by calculating $s_{i+1} = s_i^2$ for $i \in \{1, 2, 3\}$.

$$\begin{aligned} s_2 &= 400^2 \pmod{2537} = 169 \\ s_3 &= 169^2 \pmod{2537} = 654 \\ s_4 &= 654^2 \pmod{2537} = 1500 \end{aligned}$$

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

$$m = y \oplus \sigma_1\sigma_2\sigma_3\sigma_4 = 1001 \oplus 0100 = 1101.$$

6.9. The hash function h does not have to have any of the desired properties. Consider the following example: Let h_1 be any strongly collision-resistant hash function from $\{0, 1\}^n$ to $\{0, 1\}^m$. Now let

$$h_2(x) = h_1(x) \oplus 1^m.$$

Clearly h_2 is also strongly collision-resistant hash function. This is because if we could find two messages that would break its strongly collision-resistant hash function property, they would also do the same to h_1 which is strongly collision-resistant, thus a contradiction. This h_2 also satisfies the condition $h_1(x) \neq h_2(x)$ for any $x \in \{0, 1\}^n$. Now look at h :

$$h(x) = h_1(x) \oplus h_1(x) \oplus 1^m = 1^m.$$

Since h is now just a constant function, it clearly does not satisfy any of the desired properties.

Let us assume that h' is not strongly collision-free. Then it is feasible to find words w, w' such that $w \neq w'$ and $h'(w) = h'(w')$. Then

$$h_1(w) \parallel h(w) = h_1(w') \parallel h(w')$$

But this gives us that

$$\begin{aligned} h_1(w) &= h_1(w') \\ h(w) &= h(w') \end{aligned}$$

But using the strongly collision-free property of h_1 we get that $w = w'$, which is a contradiction. So h' is a strongly collision-free hash function.

6.10. 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

$$h(h(w_1) \parallel h(w_2)) = h(h(w'_1) \parallel h(w'_2)).$$

But h is strongly collision-free so it has to hold

$$h(w_1) \parallel h(w_2) = h(w'_1) \parallel h(w'_2),$$

otherwise the two words $h(w_1) \parallel h(w_2)$ and $h(w'_1) \parallel h(w'_2)$ 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

$$\begin{aligned} w_1 &= w'_1 \\ w_2 &= w'_2, \end{aligned}$$

which is a contradiction as it gives us $w = w'$. So h' is a strongly collision-free hash function.

6.11.

1. The first step is to realize that we can express

$$y^r = b_1 w_1^{-1} \pmod{p}.$$

Subsequently, we get that

$$\begin{aligned} w_2 &= b_2 y^{-(r_1+1)} \pmod{p} \\ &= b_2 y^{-r_1} y^{-1} \pmod{p} \\ &= b_2 w_1 b_1^{-1} y^{-1} \pmod{p}. \end{aligned}$$

In our case

$$w_2 = 1466 \cdot 457 \cdot 4153 \cdot 464 = 888 \pmod{6661}.$$

2. Note that for more general update function, we still can express

$$y^r = b_1 w_1^{-1} \pmod{p}.$$

With this knowledge we have

$$\begin{aligned} w_2 &= b_2 y^{-(kr_1 + \ell)} \pmod{p} \\ &= b_2 y^{-kr_1} y^{-\ell} \pmod{p} \\ &= b_2 (y^{r_1})^{-k} y^{-\ell} \pmod{p} \\ &= b_2 w_1^k b_1^{-k} y^{-\ell} \pmod{p}. \end{aligned}$$

Since all b_1, b_2, w_1, k, ℓ and y are known, plaintext w_2 can be efficiently calculated.

6.12. We will use Euler's criterion to find the least significant bit of x . First we calculate $c^{\frac{p-1}{2}} \pmod{p}$ 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 $c = a^{2k} \pmod{p}$ 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.

6.13. 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.

6.14.

(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 $i + 1 \pmod n$, 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 first 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.

6.15.

(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 $p(m) = a_m m^m + \dots + a_0$, $a_m, \dots, a_0, m \in \mathbb{N}$. Now consider the function $r_p : \mathbb{N} \rightarrow \mathbb{N}$, $r_p(n) = (\sum_{i=0}^m |a_i|) n^m$. 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 $n^{-\log \log n} \leq \frac{1}{r_p(n)}$:

$$\frac{1}{n^{\log \log n}} \leq \frac{1}{r_p(n)}$$

$$r_p(n) \leq n^{\log \log n}$$

$$\left(\sum_{i=0}^m |a_i| \right) n^m \leq n^{\log \log n}$$

$$\begin{aligned} \log \left[\left(\sum_{i=0}^m |a_i| \right) n^m \right] &\leq \log(n^{\log \log n}) \\ \log \left(\sum_{i=0}^m |a_i| \right) + m \log n &\leq (\log \log n) \log n \\ m &\leq \log \log n - \frac{\log \left(\sum_{i=0}^m |a_i| \right)}{\log n} \end{aligned}$$

We now take the limit of the right hand side

$$\lim_{n \rightarrow \infty} \log \log n - \frac{\log \left(\sum_{i=0}^m |a_i| \right)}{\log n} = \infty.$$

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.

6.16. This problem is usually denoted as the almost birthday problem. Let us denote the probability that, in a group of n people, no two birthdays lie within the interval of k days as $A_k(n)$. Recall that $A_1(n)$ corresponds to the standard birthday problem probability of which is computed as

$$P(A_1(n)) = \frac{365!}{(365 - n)!365^n}.$$

Further we rewrite the probability $P(A_k(n))$ using the conditional probability as:

$$P(A_k(n)) = P(A_k(n)|A_1(n))P(A_1).$$

Conditioning $A_k(n)$ on $A_1(n)$ and preventing coincidental birthdays simplify the following analysis.

We find the number of possible orderings of birthdays that satisfy the conditions that no two birthdays lie in the same interval of k days. We can rewrite a potential ordering of birthday and non-birthday days satisfying this condition as the following sequence:

$$1, \underbrace{0, 0, \dots, 0}_{k-1}, 1, \underbrace{0, 0, \dots, 0}_{k-1}, 1, \underbrace{0, 0, \dots, 0}_{k-1}, 0*, 0*, \dots$$

where 1's represent birthdays, 0's represent the first $k - 1$ non-birthday days after a birthday, and 0's with asterisk represent extra non-birthday days after the first $k - 1$. We should treat such sequence as "cyclic" to be able to catch birthdays at the turn of the year.

To obtain the probability of not having any birthdays within k days of each other given that there are no coincidental birthdays, we divide the number of distinct orderings that take the required form by the total number of orderings. We fix the first birthday as well and do not allow it to be permuted to eliminate rotated orderings.

Now, we group $[1, \underbrace{0, 0, \dots, 0}_{k-1}]$, treat them as a unit and permute them together with $0*$'s. The number of $0*$'s is $365 - kn$, the number of $[1, \underbrace{0, 0, \dots, 0}_{k-1}]$ units is $n - 1$ (recall that the first such unit is fixed). The number of these groupings is given as

$$\binom{(365 - kn) + (n - 1)}{n - 1} = \binom{364 - kn + n}{n - 1}.$$

Total number of orderings is as $\binom{364}{n-1}$ (recall again that the first birthday is fixed). Together, we can write the conditional probability as:

$$P(A_k(n)|A_1(n)) = \frac{\binom{364-kn+n}{n-1}}{\binom{364}{n-1}}$$

and

$$P(A_k(n)) = \frac{\binom{364-kn+n}{n-1}}{\binom{364}{n-1}} \cdot \frac{365!}{(365-n)!365^n}$$

which can be simplified to:

$$P(A_k(n)) = \frac{(364 - kn + n)!}{(365 - kn)!365^{n-1}}.$$

To answer the question in the exercise we need to compute the probability of the complement event: $1 - P(A_k(n))$ and find n such that this probability is greater than $\frac{1}{2}$.

The results are given in the following table:

k	n
1	23
2	14
3	11
4	9
5	8
6	8
7	7
8	7
9	6
10	6
11	6
12	6
13	5
14	5
15	5

6.17.

(a) The only odd quadratic residue modulo 8 is 1, as

$$(\pm 1)^2 \equiv (\pm 3)^2 \equiv 1 \pmod{8}.$$

(b) Let $n > 3$ and suppose that $x^2 \equiv k \pmod{2^n}$ has a solution $x \in \mathbb{Z}$ for an odd integer k . Then $k \equiv m^2 \pmod{2^n}$ for some odd positive integer $m < 2^n$. Using the hint, we know that there exist unique $e_1 \in \{0, 1\}, e_2 \in \{0, 1, \dots, 2^{n-2}\}$ such that

$$m \equiv (-1)^{e_1} 5^{e_2} \pmod{2^n}.$$

Then

$$m^2 \equiv 5^{2e_2} \equiv (-1)^{f_1} 5^{f_2} \pmod{2^n}$$

for unique $f_1 \in \{0, 1\}, f_2 \in \{0, 1, \dots, 2^{n-2}\}$. This implies that

$$5^{2e_2 - f_2} (-1)^{f_1} \equiv 1 \equiv 5^0 (-1)^0 \pmod{2^n},$$

hence $f_1 = 0$ and $2e_2 \equiv f_2 \pmod{2^{n-2}}$ (here we are using the fact that the order of 5 modulo 2^n is 2^{n-2} , which follows directly from the hint). Let $f_2 = 2g_2$ for some $g_2 \in \mathbb{Z}$ (here we are using the fact $n > 3$). Then we obtain $e_2 \equiv g_2 \pmod{2^{n-3}}$, so either $e_2 \equiv g_2 \pmod{2^{n-2}}$ or $e_2 \equiv g_2 + 2^{n-3} \pmod{2^{n-2}}$. Thus there are exactly two possibilities for e_2 and also two possibilities for e_1 (it could be either 0 or 1), hence four possible square roots of k modulo 2^n .

- (c) If an odd integer k is a quadratic residue modulo 2^n for $n > 3$, it is also a quadratic residue modulo 8, so in particular $k \equiv 1 \pmod{8}$ by a). On the other hand, there are 2^{n-1} positive odd integers less than 2^n , and since the squaring function modulo 2^n for $n > 3$ is four-to-one by b), there are exactly $2^{n-3} = \frac{2^n}{8}$ quadratic residues modulo 2^n for $n > 3$. But this is exactly the number of positive integers $k < 2^n$ satisfying $k \equiv 1 \pmod{8}$. Thus we can conclude that an odd integer k is a quadratic residue modulo 2^n for $n > 3$ if and only if $k \equiv 1 \pmod{8}$.

6.18. We use the following lemma

Lemma 6.2.1. - Let $m \in \mathbb{Z}$, such that there exist private roots modulo m . Write $\varphi(m) = q_1^{\alpha_1} \dots q_k^{\alpha_k}$, where q_1, \dots, q_k are primes and $\alpha_1, \dots, \alpha_k \in \mathbb{N}$. Then for arbitrary $g \in \mathbb{Z}$, $(g, m) = 1$ it holds that g is a primitive root modulo m if and only if

$$g^{\frac{\varphi(m)}{q_1}} \not\equiv 1 \pmod{m}, \dots, g^{\frac{\varphi(m)}{q_k}} \not\equiv 1 \pmod{m}.$$

Since $\varphi(43) = 42 = 2 \cdot 3 \cdot 7$, we are looking for such $g \in \mathbb{Z}$, $(g, 43) = 1$ such that

$$g^{21} \not\equiv 1 \pmod{43}, g^{14} \not\equiv 1 \pmod{43}, g^6 \not\equiv 1 \pmod{43}.$$

We soon find that $g = 3$ (among the others). Next, we notice that $3^4 \equiv 38 \pmod{43}$. Thus, the given congruence is equivalent to the congruence

$$3^{19t} \equiv 3^4 \pmod{43}.$$

Which is equivalent to

$$19t \equiv 4 \pmod{42}.$$

And thus $t \equiv 40 \pmod{42}$. When going backwards we get $x \equiv 3^{40} \equiv 24 \pmod{43}$.

6.19. The cryptosystem is the simple variant of the Paillier cryptosystem.

1. $n^2 = 15626209$, $\phi(n) = 58 \cdot 66 = 3826$, $\phi(n)^{-1} = 3700$

Encryption:

$$c = (1 + 3953)^{2019} 1111^{3953} \pmod{15626209} = 7334081$$

Decryption:

$$m = \frac{7334081^{3826} - 1}{3953} \cdot 3700 \pmod{3953} = 617 \cdot 3700 \pmod{3953} = 2019$$

2. The cryptosystem has the following homomorphic property:

$$E(m_1) \cdot E(m_2) = E(m_1 + m_2)$$

because

$$((1 + n)^{m_1} r_1^n) \cdot ((1 + n)^{m_2} r_2^n) = ((1 + n)^{m_1+m_2} \pmod{n} (r_1 r_2)^n) \pmod{n^2}.$$

Notice that $c = 4354044 = 2019^2 \pmod{n^2}$, therefore decryption of c is $2019+2019 \pmod{3953} = 85$.

3. This is clear from the binomial theorem:

$$(1+n)^a = \sum_{k=0}^a \binom{a}{k} n^k = 1 + an + \binom{a}{2} n^2 + \text{higher powers of } n$$

6.20. We start the decryption by finding the random r . First we note that $c \pmod n \equiv r^e(1+mn) \pmod n$ and so

$$c^d \equiv r^{ed}(1+mn)^d \equiv r \pmod n.$$

Now using r we can obtain $1+mn$ as

$$cr^{-e} \equiv r^e r^{-e}(1+mn) \equiv 1+mn \pmod{n^2}.$$

While n isn't invertible modulo n^2 , we can still obtain m as

$$m = L(cr^{-e} \pmod{n^2}),$$

where $L(x) = \frac{x-1}{n}$ in which $\frac{a}{b}$ denotes the quotient of a divided by b .

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 $sig_k : \{0, 1\}^* \times M \rightarrow S$, and for each key $k_v \in K_v$, there exists a single and easy to compute verification mapping $ver_{k_v} : M \times S \rightarrow \{true, false\}$ 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 $s = sig(m) = m^d \pmod n$. The signature s for the message m is valid if $s^e \pmod n = m$.

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 $y = q^x \pmod p$, where $1 < x < p$ is the secret key.

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

Rabin signatures

A collision-resistant hash function $h : \{0, 1\}^* \rightarrow \{0, 1\}^k$ is used for some fixed k . The signer chooses primes p, q of size approximately $k/2$ and computes public key $n = pq$. The pair (p, q) is kept secret. To sign a message w , the signer chooses a random string U and calculates $h(w, U)$ (if $h(w, U) \notin QR(n)$, the signer picks a new U and repeats the process). The signer solves the equation

$x^2 = h(w, U) \pmod n$. The pair (U, x) is the signature of w . The verifier computes x^2 and $h(w, U)$ and verifies that they are equal.

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 , ($r = h^{\frac{p-1}{q}} \pmod p$, where h is a primitive element in \mathbb{Z}_p), x is a random integer such that $0 < x < q$ and $y = r^x \pmod p$. 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 $s = \text{sig}(m, k) = (a, b)$, where $a = (r^k \pmod p) \pmod q$ and $b = k^{-1}(m + xa) \pmod q$, where $kk^{-1} = 1 \pmod q$. The signature $s = (a, b)$ is valid if $(r^{u_1}y^{u_2} \pmod p) \pmod q = a$, where $u_1 = mz \pmod q$, $u_2 = az \pmod q$ and $z = b^{-1} \pmod q$.

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 $f : Y \rightarrow Z$ 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: $\text{sig}(x_1, \dots, x_k) = (y_{1,x_1}, \dots, y_{k,x_k})$

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 , $s_i = \sqrt{v_i^{-1}} \pmod n$. 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 $x_i = r_i^2 \pmod n$, 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 $y_i = r_i \prod_{j=1}^k s_j^{b_{ij}} \pmod n$ and sends w , all b_{ij} , all y_i and h to Bob. Bob finally computes z_1, \dots, z_k , where $z_i = y_i^2 \prod_{j=1}^k v_j^{b_{ij}} \pmod n = x_i$ 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, $\text{sig}_B(m)$ of the message m , Alice chooses a random r and asks Bob to sign $m' = mr^e \pmod n$ and to send her back his signature $\text{sig}_B(m')$. Alice then gets easily $\text{sig}_B(m) = r^{-1}\text{sig}_B(m')$.

Ong-Schnorr-Shamir subliminal channel

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.

Let n be a large integer and let k be an integer such that $\gcd(n, k) = 1$. Calculate $h = k^{-2} \pmod n$. Both h and n form public key whereas k is kept secret as trapdoor information.

Let w be a secret message Alice wants to send with $\gcd(w, n) = 1$. Let w' be a harmless message with $\gcd(w', n) = 1$.

To sign w' Alice computes $S_1 = \frac{1}{2}(\frac{w'}{w} + w) \bmod n$ and $S_2 = \frac{k}{2}(\frac{w'}{w} - w) \bmod n$. Signature of the harmless message is verified as $w' = S_1^2 - hS_2^2 \bmod n$. The secret is decrypted by Bob as $w = \frac{w'}{(S_1 + k^{-1}S_2)}$.

7.2 Exercises

7.1. Alice and Bob are using the RSA signature scheme. Alice's public key is $(n, e) = (899, 17)$. Malignant 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 these 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 a 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

(a) $b = (w - ra)x^{-1} \pmod{p-1}$

(b) $b = xa + rw \pmod{p-1}$

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 $a = q^k \pmod{p}$ 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, \dots, 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, \dots, y_l) . The secret key is an l -tuple (x_1, x_2, \dots, x_l) such that $f(x_i) = y_i$ for each $i \in \{1, 2, \dots, l\}$. The signature of a message $m_1 m_2 \dots m_l \in \{0, 1\}^l$ 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 $x = q_i n_i + r_i$, $0 \leq r_i < n_i$ (we know r_i, q_i are unique). Then

$$g_i(x) = \begin{cases} q_i n_i + f_i(r_i) & \text{if } (q_i + 1)n_i \leq 2^b \\ x & \text{otherwise} \end{cases}$$

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?

7.15. Consider a signature scheme based on the Rabin cryptosystem with secret primes p, q and public information $n = pq$. Signature of a message w are its four square roots modulo n .

- (a) Which messages can be signed?
- (b) Is the proposed signature scheme secure?

7.16. Consider the Ong-Schnorr-Shamir subliminal channel with public key $(h, n) = (36606, 47371)$. Alice wanted to be sure her secret message gets to Bob so she sent the same secret message w twice using the signed messages $(11587, 46420, 41083)$ and $(3561, 41492, 25348)$. Perform the following tasks:

- (a) Verify the signature for both messages.
- (b) Without using brute force, find the secret message w and the secret key k .

7.17. Find a way to use Rabin signatures as a subliminal channel that allows Alice to send at least two secret bits to Bob with every signed message. This channel must look like a normal channel with Rabin signatures so that the warden Walter must not be able to retrieve the secret message.

7.18. Bob is using a single RSA scheme to both decrypt encrypted messages and create signatures (with the same set of public and private keys). You have intercepted an encrypted message c addressed to Bob. Use his signature scheme to make him help you decrypt c without him noticing.

7.19. Consider the ElGamal signature scheme.

- (a) Show that the scheme is vulnerable to existential forgery. Show that an adversary can produce a combination of message w and a correct signature (a, b) , but cannot choose the value of w .
- (b) Show that given a valid signature (a, b) of a message w , an adversary can compute signatures for messages of the form $w' = (w + \beta b)\alpha \pmod{p-1}$, for an arbitrarily chosen $\beta \in \mathbb{Z}_p^*$ and $\alpha = q^\beta \pmod{p}$.

7.20. ElGamal signature scheme with public key $(p, q, y) = (97, 10, 7)$ was used to sign messages $w_1 = 54$ and $w_2 = 13$ with $sig(w_1) = (40, 38)$ and $sig(w_2) = (40, 25)$. Without calculating the discrete logarithm or using other kind of brute force, find the secret key x .

7.21. Consider the following signature scheme. Choose primes p, q such that $q | p - 1$. Choose a generator $g \in \mathbb{Z}_p^*$ of order q . Choose a random $x \in \mathbb{Z}_q^*$ and compute $y = g^x \pmod{p}$. The value x serves as a secret key, while p, q, g and y are public.

To sign a message m , choose a random $k \in \mathbb{Z}_q^*$ and compute $r = g^k \pmod{p}$ and $s = k - H(m||r)x \pmod{q}$ where $H : \{0, 1\}^* \rightarrow \mathbb{Z}_q$ is a cryptographic hash function. The pair (r, s) is the signature of m .

- (a) Provide a verification procedure for the proposed scheme and prove that it is correct.
- (b) Show that a private key x can be recovered if the same k is reused.

* **7.22.** Consider Alice and Bob use the DSA signature scheme with $p = 2347, q = 23, r = 266, x = 11$ and $y = 864$. Alice signs $w = 1000$ and sends the corresponding signature $sig(w) = (7, 20)$ to Bob. Surprisingly, Bob shares with Alice her secret key x and his purpose is not only verification of the signature. Is Alice trying to communicate something to Bob?

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 $m = m_1 m_2 \pmod n$ as $s = s_1 s_2 \pmod n$. This is the first case and the signature of $m_a = 195 = 13 \times 15$ is $s_a = \text{sig}(m_1) \times \text{sig}(m_2) \pmod n = 524$. Verification: $(s_a)^e \pmod n = 524^{17} \pmod{899} = 195$ 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 $s^{-1} \pmod n$ is the signature of the message $m^{-1} \pmod n$. Using the Extended Euclidean Algorithm we get $1 = -362 \times 524 + 211 \times 899$. Hence $s_b = -362 \pmod n = 537$. Verification: $(s_b)^e \pmod n = 537^{17} \pmod{899} = 627 = m_b$.

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) $a^a y^b \equiv q^w \pmod p$
- (b) $y^a a^w \equiv q^b \pmod p$

Recall that $y = q^x \pmod p$ and $a = q^r \pmod p$. Now we prove correctness of the proposed verification method:

- (a) $a^a y^b \equiv y^{ar} y^b \equiv y^{ar+w-ar+k(p-1)} \equiv q^w \pmod p$
- (b) $y^a a^w \equiv q^{ax} q^{rw} \equiv q^{ax+rw} \equiv q^b \pmod p$

7.6. If the signer signs two messages w_1 and w_2 he obtains signatures (a, b_1) and (a, b_2) where $b_1 = k^{-1}(w_1 - xa) \pmod{p-1}$ and $b_2 = k^{-1}(w_2 - xa) \pmod{p-1}$. We have $xa = w_1 - kb_1 = w_2 - kb_2$ and $k = (w_1 - w_2)(b_1 - b_2)^{-1} \pmod{p-1}$. Now we can calculate the secret key as $x = (w_1 - kb_1)a^{-1} \pmod{p-1}$. There are $d = \text{gcd}(a, p-1)$ 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 $c = w_2 w_1^{-1} \pmod{p-1}$. It holds that

$$q^{w_2} \equiv q^{c w_1} \equiv y^{a_1 c} a_1^{b_1 c} \pmod{p}$$

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 $b_2 = b_1 c \pmod{p-1}$.

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 $b_{i,j}$ and y_i , she can construct the following system of congruences:

$$y_1 \equiv r_1 s_1^{b_{1,1}} s_2^{b_{1,2}} \dots s_k^{b_{1,k}} \pmod{n}$$

$$y_2 \equiv r_2 s_1^{b_{2,1}} s_2^{b_{2,2}} \dots s_k^{b_{2,k}} \pmod{n}$$

...

$$y_t \equiv r_t s_1^{b_{t,1}} s_2^{b_{t,2}} \dots s_k^{b_{t,k}} \pmod{n}$$

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 $d = e^{-1} \pmod{\phi(n)}$, thus $d = 59^{-1} \pmod{9600} = 1139$.

- Alice computes $m' = m k^e \pmod{n} = 4242 \cdot 142^{59} \pmod{9797} = 808$ and sends it to Bob.
- Bob computes $s' = (m')^d \pmod{n} = 808^{1139} \pmod{9797} = 6565$ and sends it to Alice.
- Alice now computes signature s of message m , $s = k^{-1} s' \pmod{n} = 69 \cdot 6565 \pmod{9797} = 2323$.

Message can be verified as $m = s^e \pmod{n}$.

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_1j} 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)

$$\begin{aligned} N &= 2, M = 2^N = 4, \\ y_1 &= H^M(x_1) = H^4(x_1), \\ y_2 &= H^M(x_2) = H^4(x_2) \end{aligned}$$

(Signing)

$$\begin{aligned} n &= 3, \\ s_1 &= H^n(x_1) = H^3(x_1), \\ s_2 &= H^{M-n-1}(x_2) = H^0(x_2) = x_2 \end{aligned}$$

(Verification)

$$\begin{aligned} y_1 &= H^{M-n}(s_1) = H(s_1) = H(H^3(x_1)) = H^4(x_1), \\ y_2 &= H^{n+1}(s_2) = H^4(s_2) = H^4(x_2) \end{aligned}$$

(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 .

$$\begin{aligned} s &= k^{-1}(h - xr) \pmod q \\ sk &= h - xr \pmod q \\ k &= s^{-1}(h - xr) \pmod q \end{aligned}$$

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.

7.15.

(a) One can sign quadratic residues modulo n , ie. messages w for which there exists $x \in \mathbb{Z}_n^*$ such that $x^2 = w \pmod n$.

- (b) The scheme is not secure. Knowing signatures we know $x, y, x \not\equiv \pm y \pmod{n}$ such that $x^2 \equiv y^2 \pmod{n}$. Now we can factorize the modulus n . We have $x^2 - y^2 \equiv 0 \pmod{n}$, thus $n|x^2 - y^2 = (x + y)(x - y)$. Computing $\gcd(x \pm y, n)$ gives p or q .

7.16.

- (a) Walter verifies that

$$w' = S_1^2 - h \cdot S_2^2 \pmod{n}.$$

In this case:

$$46420^2 - 36606 \cdot 41083 = 11587 \pmod{47371},$$

$$41492^2 - 36606 \cdot 25348 = 3561 \pmod{47371}.$$

- (b) Signatures are calculated as

$$S_1 = \frac{1}{2} \left(\frac{w'}{w} + w \right) \pmod{n}$$

$$S_2 = \frac{k}{2} \left(\frac{w'}{w} - w \right) \pmod{n}$$

The scheme was used twice for the same message w . Now we have 4 equations with 3 unknown variables.

We can take the following

$$46420 \equiv \frac{1}{2}(11587w^{-1} + w) \pmod{n}$$

$$41083 \equiv \frac{k}{2}(11587w^{-1} - w) \pmod{n}$$

$$41492 \equiv \frac{1}{2}(3561w^{-1} + w) \pmod{n}$$

and transform into

$$w \equiv 45469 - 11587w^{-1} \pmod{n}$$

$$34795 \equiv k \cdot (11587w^{-1} - w) \pmod{n}$$

$$35613 \equiv (3561w^{-1} + w) \pmod{n}$$

to obtain $9586 \equiv 8026w^{-1} \pmod{n}$. Now we can easily calculate $w^{-1} = 44067$, $w = 5778$ and $k = 5492$.

7.17. During the signing procedures, Alice is asked to find the square root of $h(wU) \pmod{n}$. Since $n = pq$, there are 4 possible square roots and she can pick any of them for the signature. Now her choice of one of four possible choices is exactly two bits of information. She can for example order them by their magnitudes, so picking one corresponds to picking a number from 1 to 4, i.e. two bits. Now if Bob also knows the factorization of $n = pq$, he can also find all the square roots of $h(wU) \pmod{n}$ and recover the information about Alice's choice, the two bits. So all they need to do to use Rabin signatures for subliminal channel is share the factorization of the public moduli as a trapdoor information. They would use the channel normally, only deliberately pick the square roots instead of randomly.

Since they use the signature scheme properly, every message will be correctly signed in Walter's eyes. All the additional work is done locally on Bob's side after the communication and Walter has no information about that. Walter also cannot feasibly discover the other square roots (he would be able to factorize n in that case), so he also cannot retrieve the message.

7.18. We will use the blind signature scheme to make Bob sign a message that will allow us to decrypt c while he has no idea what he is signing. (We could make him just sign c to decrypt it but he could realize what he is doing)

Let n be the public modulus and e and d the public and private keys respectively, we pick a random r such that $\gcd(r, n) = 1$ and make Bob sign

$$c' = cr^e = (mr)^e \pmod{n},$$

where m is the original message. When Bob signs our message we get

$$\begin{aligned} s &= c'^d \pmod{n} \\ &= ((mr)^e)^d \pmod{n} \\ &= (mr)^{ed} \pmod{n} \\ &= mr \pmod{n} \end{aligned}$$

Now we can easily obtain the original message by computing $m = sr^{-1} \pmod{n}$ which is simple since we know both s and r .

7.19.

(a) Let $a = q^\alpha y^\beta$, $b = -a\beta^{-1}$, $w = \alpha b$. Indeed, the verification procedure give:

$$\begin{aligned} y^a a^b &= y^{q^\alpha y^\beta} (q^\alpha y^\beta)^{-q^\alpha y^\beta \beta^{-1}} \pmod{p} \\ &= y^{q^\alpha y^\beta} y^{-q^\alpha y^\beta} q^{-\alpha q^\alpha y^\beta \beta^{-1}} \pmod{p} \\ &= q^{-\alpha q^\alpha y^\beta \beta^{-1}} \pmod{p} \\ &= q^{\alpha b} \pmod{p} \\ &= q^w \pmod{p}. \end{aligned}$$

(b) We have that $a = q^r$ and $b = (w - xa)r^{-1}$ are a valid signature for w . The signature for $w' = (m + \beta b)\alpha \pmod{(p-1)}$ is $a' = \alpha a$ and $b' = \alpha b$. Indeed, the verification is as follows:

$$\begin{aligned} y^{a'} a'^{b'} &= q^{x\alpha a} (\alpha a)^{\alpha b} \pmod{p} \\ &= q^{x\alpha a} q^{\beta\alpha b} q^{r\alpha(w-xa)r^{-1}} \pmod{p} \\ &= q^{x\alpha a} q^{\beta\alpha b} q^{\alpha(w-xa)} \pmod{p} \\ &= q^{\beta\alpha b} q^{\alpha w} \pmod{p} \\ &= q^{\alpha(w+\beta b)} \pmod{p} \\ &= q^{w'} \pmod{p}. \end{aligned}$$

7.20. First we notice that for $\text{sig}(w_i) = (a_i, b_i)$ we have $a_1 = a_2 = a$. That means the same random r was used for both signatures. Now taking $b_1 - b_2 \equiv (w_1 - w_2)r^{-1} \pmod{p-1}$. In our case $b_1 - b_2 = 13$ has an inverse modulo 96 $(b_1 - b_2)^{-1} = 37$, so $r = (b_1 - b_2)^{-1}(w_1 - w_2) \pmod{96} = 77$.

With the secret $r = 77$ we can now calculate the secret key x . We have $w_1 - b_1 r \equiv xa \pmod{p-1}$. In our case

$$8 = 40x \pmod{96}.$$

Unfortunately, 40 does not have an inverse modulo 96 so we have to use the extended Euclidian algorithm to find all the candidates for x satisfying this congruence. We get $x = 5 - 12l$ for $l \in \mathbb{Z}$. Because $1 \leq x \leq p$ we get these possible candidates $x \in \{5, 17, 29, 41, 53, 65, 77, 89\}$, where only 53 satisfies $7 = y = q^x \pmod{p}$, so the secret key is

$$x = 53.$$

7.21.

- (a) We compute $e = H(m||r)$, $r_v = g^s y^e \pmod p$, $e_v = H(M||r_v)$ and verify $e_v = e$. Indeed, $r_v = g^s y^e = g^{k-ex} g^{ex} = g^k = r \pmod p$ and therefore $e_v = H(M||r_v) = H(m||r) = e$.
- (b) We can subtract s values: $s' - s \equiv (k' - k) + x(e' - e) \equiv (k' - k) + x(H(m'||r') - H(m||r)) \pmod q$. If the same k is reused then $k = k'$ and $r = r'$ as well. If $m \neq m'$ then we can easily calculate x from $s' - s \equiv x(H(m'||r) - H(m||r)) \pmod q$.

7.22. Alice and Bob use the following subliminal channel: Alice did not use a random value for k , instead she set this parameter with a subliminal message w' which Bob would be able to recover with the knowledge of Alice's secret key. To recover $k = w'$, Bob calculates:

$$k = b^{-1}(w + xa) \pmod q = 15(1000 + 11 \cdot 7) \pmod{23} = 9.$$

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

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

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 $f : \mathbb{Z}_n \rightarrow \mathbb{Z}_n$ 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:

$$\begin{aligned} a^2 + b^2 &= c^2 \\ \frac{1}{2}ab &= n \end{aligned}$$

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 $g : B \rightarrow A$ 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

$$L = i(p - 1), \quad L = j(q - 1) + k \quad \text{and} \quad a^k \not\equiv 1 \pmod{q}.$$

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 $f(x) = x^2 + c \pmod{n}$ with a randomly chosen c , $0 \leq c < n$. 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 $b \geq 0$ 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.

8.21. Find two elliptic curves over \mathbb{F}_5 , with 8 points but a different group structure.

8.22. Consider the following cryptosystem using a non-singular elliptic curve E_p with n points, secret key $d < n$ and public key (P, Q) , where $Q = dP$ are two points on E_p : To encrypt a message $m = (m_1, m_2)$, $1 \leq m_1, m_2 < p$, pick a random integer $1 \leq k < n$ and compute $O = kP$, $y_1 = c_1 m_1 \pmod{p}$ and $y_2 = c_2 m_2 \pmod{p}$, where $(c_1, c_2) = kQ$. The encrypted message is then (O, y_1, y_2) . Find the decryption procedure.

8.23. Consider an elliptic curve $E : y^2 = x^3 + 8$ over \mathbb{R} . Show that E does not have multiple roots. Algebraically determine the number of roots E has.

8.24. Use the Quadratic sieve method to factorize $n = 713$.

8.25. Find all points of the elliptic curve $E : y^2 + xy = x^3 + 1$ over the field \mathbb{F}_4 .

8.26. Find a factor of 119 without using brute force, if you know that the function $29^x \bmod 119$ has a period $r = 16$ (Remark: This is a classical subroutine in the Shor's quantum polynomial time algorithm for integer factorization).

8.27. Consider elliptic version of the ElGamal cryptosystem. Public key is as follows: $p = 11$, $E : y^2 = x^3 + 3x + 6 \pmod{11}$, $P = (2, 8)$, $Q = (2, 3)$. Show computation steps.

(a) Encrypt the message $m = (5, 6)$ with $r = 2$.

(b) Decrypt the ciphertext, computed in (a), with private key $x = 4$.

8.28. Consider the elliptic curve version of the ElGamal digital signature.

(a) Show that the private key a can be recovered if the adversary learns r .

(b) Show that the private key a can be recovered if the same r is used to generate signatures on two messages.

(c) Let $E : y^2 = x^3 + x + 4 \pmod{23}$ and let $P = (0, 2)$. Show that an adversary can forge valid signature on any message of their choice. Propose a method to prevent such an attack.

8.29. Propose a simple method to compute mP on an elliptic curve with approximately $\log_2 m$ point doublings and $\frac{1}{2} \log_2 m$ or less additions on average.

8.30. Find all points of order 2, *ie.* points P such that $2P = \mathcal{O}$, of the elliptic curve $E : y^2 = x^3 - x$ over \mathbb{R} .

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 $y^2 + uxy + vy = x^3 + ax^2 + bx + c$ 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:

$$x \rightarrow x - \frac{d}{3}$$

For E we have $x \rightarrow x - \frac{6}{3} = x - 2$ ($3^{-1} \equiv 10 \pmod{29}$) 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 $n_a P = (10, 25)$.
 (2) Bob computes $n_b P = (24, 22)$.
 (3) Alice and Bob interchange values $n_a P$ and $n_b P$.
 (4) Alice computes $n_a(n_b P) = (20, 3)$.
 (5) Bob computes $n_b(n_a P) = (20, 3)$.

- (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 $(M + kP_U) - s_U(kG) = M + kP_U - k(s_U G) = M + kP_U - kP_U = M$.
 (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 $1 \leq b < n$ 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, $2^{kl} - 1 = (2^k - 1)(2^{l(k-1)} + 2^{l(k-2)} + \dots + 1)$. 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 $k \geq 1$. We have $\binom{n}{p} = \frac{n!}{(n-p)!p!} = \frac{n(n-1)(n-2)\dots(n-p+1)}{p!} = \frac{p^k q(n-1)(n-2)\dots(n-p+1)}{p!} = \frac{p^{k-1} q(n-1)(n-2)\dots(n-p+1)}{(p-1)!}$. 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 $(n-1)(n-2)\dots(n-p+1)$ is not divisible by p , q is not divisible by p and $\binom{n}{p} = \frac{p^{k-1} q(n-1)(n-2)\dots(n-p+1)}{(p-1)!}$ is not divisible by p^k which gives that $\binom{n}{p}$ is not divisible by n .

- (b) \Rightarrow :

We have $\binom{n}{k} = \frac{n!}{(n-k)!k!} = \frac{n(n-1)\dots(n-k+1)}{k!}$ and since n is a prime and $k < n, k, k-1, \dots, 1$ are coprime to n and therefore $\gcd(k!, n) = 1$. Since $\binom{n}{k}$ is an integer, $k! \mid (n-1)\dots(n-k+1)$. Let $\frac{(n-1)\dots(n-k+1)}{k!} = m$. Then $\binom{n}{k} = mn$ and $n \mid \binom{n}{k}$.

\Leftarrow :

We use obversion: If n is not a prime, then there exists $k \in \{1, 2, \dots, n-1\}$ 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 $(x+a)^n \equiv x^n + a \pmod{n}$.

\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, \dots and for every x_i it computes $\gcd(x_i - x_0, n), \gcd(x_i - x_1, n), \dots, \gcd(x_i - x_{i-1}, n)$ 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 $k > 2$.

$$\begin{array}{l} kP = (x_k, y_k) \\ \hline P = (0, 2) \\ 2P = (1, 548) \\ 3P = (24, 118) \\ 4P = (382, 172) \\ 5P = (136, 275) \\ 6P = (507, 297) \\ 7P = (260, 539) \\ 8P = (516, 314) \\ 9P = (477, 94) \\ 10P = (214, 547) \\ 11P = (495, 326) \\ 12P = (171, 397) \end{array}$$

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 $x^{2^a} \equiv 1 \pmod{2^{a+2}}$.

8.21. Two curves we are searching for are

$$E_1 : y^2 = x^3 + 4x + 1 \pmod{5} \quad \text{and} \quad E_2 : E_1 : y^2 = x^3 + 4x \pmod{5}.$$

Let us investigate E_1 first. Quadratic residues mod 5 are 1 and 4, and 0 has a square root of zero. Let us check whether $x^3 + 4x + 1 \pmod{5}$ is a quadratic residue for different values of x .

x	$x^3 + 4x + 1 \pmod{5}$	QR?	y
0	1	YES	(1, 4)
1	1	YES	(1, 4)
2	2	NO	-
3	0	-	0
4	1	YES	(1, 4)

From this table we can list all the 8 points – all of those in the table, plus the point in infinity \mathcal{O} :

$$E_1 = \{(0, 1), (0, 4), (1, 1), (1, 4), (3, 0), (4, 1), (4, 4), \mathcal{O}\}.$$

The group structure of E_1 is isomorphic to $(\mathbb{Z}_8, +)$. In order to prove this claim let us find the isomorphism from $(E_1, +)$ to $(\mathbb{Z}_8, +)$. Let's calculate multiples of the point $P = (0, 1)$:

n	nP
1	(0, 1)
2	(4, 1)
3	(1, 4)
4	(3, 0)
5	(1, 1)
6	(4, 4)
7	(0, 4)
0	\mathcal{O}

Since the whole group is generated by $(0, 1)$ we see that the table also defines the desired isomorphism and each point is mapped to the corresponding number in the first column. Indeed it is easy to verify that each point Q can be expressed as qP and therefore $Q + R = qP + rP = (q+r)P$, which is the condition for the mapping function to be a homomorphism. The bijectivity of this mapping is trivial.

Let us investigate E_2 in the same way:

x	$x^3 + 4x \pmod{5}$	QR?	y
0	0	-	0
1	0	-	0
2	1	YES	(1, 4)
3	4	YES	(2, 3)
4	0	-	0

The list of points is therefore

$$E_2 = \{(0, 0), (1, 0), (2, 1), (2, 4), (3, 2), (3, 3), (4, 0), \mathcal{O}\}.$$

In order to prove that the group structure is not equivalent to $(\mathbb{Z}_8, +)$ it is enough to note that for all points $P \in \{(0, 0), (1, 0), (4, 0), \mathcal{O}\}$ we have that $2P = \mathcal{O}$. Since \mathcal{O} always corresponds to 0, it is enough to note that in $(\mathbb{Z}_8, +)$ there are only two points with this property (0 and 4). That leaves us with two other options. Either $(E_2, +)$ is equivalent to $(\mathbb{Z}_4 \times \mathbb{Z}_2, +)$, or $(\mathbb{Z}_2^3, +)$, since there are no other Abelian groups with 8 elements. Of these two for all 8 elements of $(\mathbb{Z}_2^3, +)$ it holds that $2a = 0, \forall (\mathbb{Z}_2^3, +)$, therefore the only option left is $(\mathbb{Z}_4 \times \mathbb{Z}_2, +)$. Indeed, the elements corresponding to $\{(0, 0), (1, 0), (4, 0), \mathcal{O}\} \subset E_2$ are $(0, 0), (2, 0), (0, 1), (2, 1)$.

8.22. Upon receiving the encrypted message (O, y_1, y_2) , we can calculate

$$dO = d(kP) = k(dP) = kQ = (c_1, c_2)$$

using our secret d . Using that we can easily calculate $c_1^{-1} \pmod p$ and $c_2^{-1} \pmod p$ to obtain

$$m_1 = y_1 c_1^{-1} \pmod p \quad \text{and} \quad m_2 = y_2 c_2^{-1} \pmod p,$$

with $m = (m_1, m_2)$.

8.23. Elliptic curve is non-singular if its discriminant $\Delta = -16(4a^3 + 27b^2)$ is non-zero. Plugging in $a = 0$ and $b = 8$, we receive $\Delta = -27648$. Hence, E is nonsingular and therefore has no multiple roots. Moreover, since the discriminant is negative E has only one root.

8.24. In order to factorize $n = 713$ by Quadratic sieve method let us choose $m = \lfloor \sqrt{713} \rfloor = 26, k = 6$, and the factorization basis $\{2, 3, 5, 7, 11, 13, 17\}$.

u	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6
$(m+u)^2 - n$	-313	-272	-229	-184	-137	-88	-37	16	71	128	187	248	311
sieve with 2		-17		-23		-11		1		1		31	
sieve with 3													
sieve with 5													
sieve with 7													
sieve with 11						-1					17		
sieve with 17		-1									1		

We can see that

$$\begin{aligned} 21^2 - 713 &= -272 = -2^4 \cdot 17, \\ 25^2 - 713 &= -88 = -2^3 \cdot 11, \\ 29^2 - 713 &= 128 = 2^7, \\ 30^2 - 713 &= 187 = 11 \cdot 17. \end{aligned}$$

and so

$$(21^2 - 713)(25^2 - 713)(29^2 - 713)(30^2 - 713) = 2^{14} \cdot 11^2 \cdot 17^2 = 23936^2 \equiv 407^2 \pmod{713}.$$

On the other hand,

$$(21^2 - 713)(25^2 - 713)(29^2 - 713)(30^2 - 713) \equiv 456750^2 \equiv 430^2 \pmod{713}.$$

It follows that 713 divides $(403 - 407)(403 + 407) = 23 \cdot 837$. We have $\gcd(713, 23) = 23 \neq 1$. We obtain prime factors $713 = 23 \cdot 31$.

8.25. Let us denote elements of the field \mathbb{F}_4 : $0, 1, a, b$. This field has the following additive and multiplicative structure:

+	0	1	a	b
0	0	1	a	b
1	1	0	b	a
a	a	b	0	1
b	b	a	1	0

·	0	1	a	b
0	0	0	0	0
1	0	1	a	b
a	0	a	b	1
b	0	b	1	a

We can examine cases:

$x = 0$, the only possibility is $y = 0$,
 $x = 1$, possible y is $y = 0$ or $y = 1$,

$x = a$, possible $y = 0$ or $y = a$,
 $x = b$, possible $y = 0$ or $y = b$,
 $x = \mathcal{O}$.

To conclude, there are 8 points on $y^2 + xy = x^3 + 1$ over \mathbb{F}_4 : $(0, 0)$, $(1, 0)$, $(1, 1)$, $(a, 0)$, (a, a) , $(b, 0)$, (b, b) , and \mathcal{O} .

8.26. Since $r = 16$ is even, we know that $29^8 \equiv 50 \pmod{119}$ is a nontrivial solution of $x^2 \equiv 1 \pmod{119}$, which means that $119 \mid (49 \cdot 51)$. We can now compute $\gcd(49, 119) = 7$ and obtain a factor of $119 = 7 \cdot 17$.

8.27.

- (a) To encrypt the message we have to calculate $A = rP$ and $B = m + rQ$, which represent the encrypted message as (A, B) .

We calculate A as:

$$\lambda = \frac{3 \cdot 2^9 + 3}{2 \cdot 8} = 4 \cdot 9 = 3,$$

$$a_x = \lambda^2 - 2 - 2 = 5, \quad a_y = \lambda(2 - 5) - 8 = 5,$$

$$A = rP = (a_x, a_y) = (5, 5);$$

and B as follows. First we can use the calculated result to obtain rQ :

$$2(2, 3) = -2(2, 8) = -(5, 5) = (5, 6).$$

We add m and rQ to obtain B :

$$\lambda = \frac{3 \cdot 5^2 + 3}{2 \cdot 6} = 1$$

$$b_x = 1 - 5 - 5 = 2, \quad a_y = 5 - 2 - 6 = 8,$$

$$B = m + rQ = (b_x, b_y) = (2, 8).$$

- (b) Decryption of the message m follows:

$$m = B - xA = (2, 8) - 4(5, 5) = -7(2, 8) = -2(2, 8) = 2(2, 3) = (5, 6).$$

8.28.

- (a) Knowing r , an adversary can calculate

$$x^{-1}(m - rs) = x^{-1}(m - rr^{-1}(m - ax)) = x^{-1}(m - m + ax) = a \pmod{n}.$$

- (b) We have

$$s_1 = r^{-1}(m_1 - ax) \pmod{n},$$

$$s_2 = r^{-1}(m_2 - ax) \pmod{n}.$$

We can rewrite the equations as

$$rs_1 = m_1 - ax \pmod{n},$$

$$rs_2 = m_2 - ax \pmod{n}.$$

Subtraction gives $r(s_1 - s_2) = m_1 - m_2 \pmod{n}$. If $s_1 \not\equiv s_2 \pmod{n}$ then $r = (m_1 - m_2)(s_1 - s_2)^{-1} \pmod{n}$. Knowing r , an adversary continues as in (a).

- (c) An adversary can forge valid signature for a message m by choosing any r with $\gcd(r, n) = 1$. The pair $(0, r^{-1}m)$ is valid signature for the message m because

$$xQ + sR = 0 + r^{-1}m \cdot rP = mP.$$

Therefore, a verifier should reject signatures with $x \equiv 0 \pmod{n}$.

8.29. *Double-and-add* method, similar to modular exponentiation by squaring, can be used. We can write m in the binary form:

$$m = \sum_{i=0}^n m_i 2^i, \quad \text{where } m_i = \{0, 1\}.$$

Moreover, we have $2^n \leq m < 2^{n+1}$ and $n < \log_2 m$. We need to perform n doublings which is approximately $\log_2 m$ doublings. Now we sum $m_i 2^i$ and, assuming distribution of m is uniform, half of m_i will be zeroes and half will be ones on average, resulting in approximately $\frac{1}{2} \log_2 m$ additions.

The method above can be enhanced allowing $m_i = \{-1, 0, 1\}$: subtraction of points takes the same amount of time as point addition. There will be $\frac{2}{3}$ of m_i equal to zero on average, resulting in $\frac{1}{3} \log m$ point additions or subtractions.

8.30. $P = -P$ is defined as $(x, y) = (x, -y)$ and that's true only if $y = 0$, therefore we have to find such x that $y = 0$. Such x are $-1, 0, 1$ and therefore 2-torsion points are $(-1, 0), (0, 0), (1, 0), \mathcal{O}$.

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 $x = r^2 \pmod{n}$ and sends x to Bob.
2. Bob sends to Alice a random bit b as a challenge.
3. Alice sends Bob as the response $y = rs^b \pmod{n}$.
4. Bob identifies the sender as Alice if and only if $y^2 = xv^b \pmod{n}$.

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 $auth_k : \{0, 1\}^* \times M \rightarrow T$ and a single and easy to compute verification mapping $ver_k : M \times T \rightarrow \{true, false\}$. 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, $P = \{P_i \mid 1 \leq i \leq n\}$, in such a way that any t , or more, participants can compute the value S , but no group of $t - 1$, or less, 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 , $1 \leq i \leq n$, 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 $1 \leq i \leq n$, D computes the shares $y_i = f(x_i)$, where

$$f(x) = S + \sum_{j=1}^{t-1} a_j x^j \pmod{p}.$$

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 the shared secret S . Since $f(x)$ has degree $t - 1$, $f(x)$ has the form $f(x) = f_0 + a_1 x + \dots + a_{t-1} x^{t-1}$, 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:

$$S = a_0 = f(0) = \sum_{i \in J} f_i \prod_{j \in J, j \neq i} \frac{j}{j - i}.$$

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:

- Bob randomly chooses a bit string r and sends it to Alice.
- Alice computes $r \oplus k$ and sends it to Bob.
- 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:

- $A \rightarrow B$: "Alice"
 - $B \rightarrow A$: N_B
 - $A \rightarrow B$: $\{N_B\}_{K_{AS}}$
 - $B \rightarrow S$: "Bob", {"Alice", $\{N_B\}_{K_{AS}}\}_{K_{BS}}$
 - $S \rightarrow B$: $\{N_B\}_{K_{BS}}$
 - verify N_B
- Show that a malicious user Mallot can impersonate Alice to Bob.
 - 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) $A \rightarrow B : \text{"Alice"}, N_A$
- (ii) $B \rightarrow A : \{N_A\}_{K_{AB}}, N_B$
- (iii) $A \rightarrow B : \{N_B\}_{K_{AB}}$

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 them 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?

$h \backslash m$	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. Consider the Shamir's (n, t) -threshold scheme. Show that, in the secret reconstruction, a dishonest party can exclusively recover the secret, while forcing other honest parties to derive a faked secret.

9.17. Let (G, \cdot) be a group and $s \in G$ be a secret. Propose a perfect (n, n) -threshold scheme based on G .

9.18. Consider a group of $2^n - 1$ users, $n \geq 1$, trying to share a secret. These users are organized in a perfect binary tree hierarchy. Design a secret sharing scheme that will allow the recovery of the secret only to the groups which contain users that form a path from the root of the binary tree to one of its leaves (or more precisely the subgraph induced by this group contains such path).

9.19.

Consider the following secret sharing scheme. Arthur, Barbara, Clark, Donald, Elisabeth and Fay – each is given a different piece of information about a secret natural number n :

- Arthur knows that n is prime.
- Barbara knows that the binary representation of n contains at most 11 digits.
- Clark knows that $n \equiv 2 \pmod{5}$.
- Donald knows that $n \equiv 5 \pmod{503}$.
- Elisabeth knows that the binary representation of n contains at least 11 digits.
- Fay knows that n is a divisor of 60510.

Find n and determine all possible combinations of persons who are able to determine the secret together with certainty.

9.20.

Can a secret sharing scheme for five participants A, B, C, D, E and an access structure generated by the authorized sets $\{A, B\}, \{B, C, D\}, \{A, D, E\}$ be implemented using only one instance of a threshold scheme? Prove your answer.

9.21. Consider the Okamoto identification scheme simplified in the following way: we completely omit the numbers α_2, a_2, k_2 and y_2 . This means our computation will change in the following way:

$$\begin{aligned}v &= \alpha^{-a_1} \pmod{p}, \\ \gamma &= \alpha^{k_1} \pmod{p}, \\ y_1 &= k_1 + a_1 r \pmod{q}\end{aligned}$$

and verification will be

$$\gamma \equiv \alpha_1^{y_1} v^r \pmod{p}.$$

(We can consider this the case of the original protocol where it always holds $a_2 = k_2 = 0$.)

Show that if Alice now chooses unfortunate a_1 and k_1 such that

$$v\gamma \equiv 1 \pmod{p},$$

then Bob can discover the secret key a_1 .

9.22. Find an example of an orthogonal array $\text{OA}(3, 4, 1)$ or at least prove that such exists.

9.23. Consider the general form of orthogonal arrays.

(a) For all t and all n , there exists a t - $(n, t + 1, 1)$ array.

- (b) If there exists a t - (n, k, λ) orthogonal array, then there exists a $(t - 1)$ - $(n, k - 1, \lambda)$ orthogonal array.

9.24. Consider an orthogonal array $OA(n, k, \lambda)$ as it was defined in the lecture. Such orthogonal array has strength $s = 2$, in any two columns of the array every one of the possible n^2 pairs of symbols occurs in exactly λ rows. Generally, for an orthogonal array of strength s , in any s columns of the array every one of the possible n^s ordered s -tuples of symbols occurs in exactly λ rows.

Give an example of orthogonal array $OA(2, 4, 1)$ of strength 3.

9.25. Show how to construct a $(k - 1, s)$ -threshold secret sharing scheme from an orthogonal array $OA(n, k, 1)$ of strength s .

9.26. You have received the following card allowing you to open the safe-deposit box. It is clear that you need a password to open the box. Unfortunately, you do not know this password. At the same time, your colleague received a similar card for the same safe-deposit box...



9.27. Consider the following function f that computes the message authentication code of a message m comprising blocks $m_1 || m_2 || \dots || m_n$ using a secret key k and a block cipher E :

$$f_1 = E_k(m_1),$$

$$f_i = E_k(f_{i-1} \oplus m_i) \text{ for } i = 2, \dots, n$$

Show that f_n is not a secure message authentication code.

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 $y_1 \neq y_2$:

$$y_1 - y_2 \equiv ar_1 - ar_2 \pmod{q}$$

$$y_1 - y_2 \equiv a(r_1 - r_2) \pmod{q}$$

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) $M \rightarrow B : \text{"Alice"}, N_E$

(ii) $B \rightarrow M : \{N_M\}_{K_{AB}}, N_B$

(iii) $M \rightarrow B : \text{"Alice"}, N_B$ (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) $A \rightarrow B : \text{"Alice"}, N_A$

(ii) $B \rightarrow A : \{\text{"Alice"}, N_A\}_{K_{AB}}, N_B$

(iii) $A \rightarrow B : \{\text{"Bob"}, N_B\}_{K_{AB}}$

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) $B \rightarrow A : \{\text{“Alice”}, N_A, N_B\}_{K_{AB}}$

(iii) $A \rightarrow B : \{N_B, N_A\}_{K_{AB}}$

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:

$h \backslash m$	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}, \dots, I_{i_t} , the secret s is recovered using the Chinese Remainder Theorem, as the unique integer solution of the system of modular equations

$$x \equiv I_{i_1} \pmod{a_{i_1}}$$

...

$$x \equiv I_{i_t} \pmod{a_{i_t}}.$$

Moreover, s lies in $\mathbb{Z}_{a_{i_1} \dots a_{i_t}}$ because $s < \prod_{i=1}^t a_i$.

On the other hand, having only $t - 1$ distinct shares $I_{i_1}, \dots, I_{i_{t-1}}$, we obtain only that $s \equiv x_0 \pmod{a_{i_1} \dots a_{i_{t-1}}}$, where x_0 is the unique solution modulo $a_{i_1} \cdot a_{i_2} \cdot \dots \cdot a_{i_{t-1}}$ of the resulted system ($S > \prod_{i=1}^{t-1} a_{n-i+1} \geq a_{i_1} \cdot a_{i_2} \cdot \dots \cdot a_{i_{t-1}} > x_0$).

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. A dishonest cheater can always exclusively derive the secret by presenting a faked share and make the other honest shareholders to get nothing but a faked secret.

Let P_1, \dots, P_n be the participants. Suppose that P_1 cheats by presenting an incorrect share $(1, s'_1)$. The participants compute $s' = s + s'_1 r_1 - s_1 r_1$ and P_1 can easily compute s .

If t parties cooperate to recover the secret, that is find out the polynomial $f(x)$ of degree $t - 1$ and evaluating it at $f(0)$, and one of them is dishonest and provides a fake share, then the resulting interpolated polynomial will be different and the secret recovered will be faked without any of the parties being able to recognize it.

9.17. We define a (n, n) -threshold scheme as follows.

1. Give an element $g_i \in G$ chosen uniformly at random to the first $n - 1$ participants.
2. Give the share $g_n = g_1 \cdot g_2 \cdot \dots \cdot g_{n-1} \cdot s$ to the last participant.

The set of all participants can compute $s = g_{n-1}^{-1} \cdot \dots \cdot g_1^{-1} \cdot g_n$ to recover the secret s . However, the share vector of any set of $m < n$ participants, takes any value in G^m with the same probability, and thus give no information about k .

9.18. We will refer to the users as nodes in their binary tree hierarchy. We will use several instances of the Shamir's $(4, 3)$ -threshold scheme. One instance of this scheme will be used for every parent and its two children, with parent getting two shares of the secret and each child one of the remaining two shares. This means only a parent and at least one of its children will be able to construct the secret of this sub-scheme.

So all interior (non-root and non-leaf) nodes will be a part of two such sub-schemes. Once as a parent and once as a child. For one interior node we can call them its *parent-scheme* and *child-scheme* respectively.

The idea now is that the share of an interior node for the child-scheme is not directly known to this node, but instead it is the secret from its parent-scheme. Now, because the leaf nodes do not have their parent-scheme, they start with the shares for their child-scheme. The root has only its parent scheme and its secret will be the actual secret we want to share with the whole group. All nodes have their two shares in their parent-scheme from the start.

We can now see that a node gets its children-scheme share if and only if at least one of its children is part of the group and it can get its own children scheme share. Starting with the root we get the whole secret if and only if the root is in the group and one of its children is also in the group and it got its own children-scheme share. But that means that one of the children of this child must also be in the group and have its own children-scheme share. So recursively we get to one of the leaves that must be in the group and it already has its children-scheme share.

More precisely, a group gets the whole secret if and only if it contains the root P_0 , $P_1 = \text{child of } P_0$, $P_2 = \text{child of } P_1$, \dots , $P_{n-1} = \text{child of } P_{n-2}$ which has to be a leaf. But that is just stating that the group has to contain (or rather the induced subgraph) a path from root to a leaf.

9.19. As we will see below, $n = 2017$. First let us rephrase some points.

- B knows that $n < 2^{11} = 2048$.
- E knows that $1024 = 2^{10} \leq n$.
- F knows that $n \in \{1, 2, 3, 5, 6, 10, 15, 30, 2017, 4034, 6051, 10085, 12102, 20170, 30255, 60510\}$ (since $60510 = 2 \cdot 3 \cdot 5 \cdot 2017$).
- C and D together know that $n \equiv 2017 \pmod{2515}$ (using the Chinese remainder theorem).

From the Dirichlet theorem on arithmetic progressions, it is clear that A,C,D,E cannot recover n together, as it could be arbitrarily large. Thus we can now consider only coalitions containing either B or F.

First let us consider the coalitions containing F. From the list of divisors of 60510 above, it is easy to see that D and F can together recover n , B,E,F can together recover n and A,E,F can together recover n . On the other hand, C,E,F cannot together recover n (it could be both 2017 and 12102) and A,B,C,F cannot together recover n (it could be both 2 and 2017). This concludes the discussion of the cases where F is present.

Now let us consider the cases where B is present. It is clear that it is not possible to recover n if B joins with exactly one of A,C,D,E. Also even A,B,C,E cannot together recover n (it could be both 2017 and 2027) Thus it now suffices to only consider the cases where B and D are present (and F is not, as we already know). Since $2515 > 2048$, B,C,D can together recover n . But A,B,D cannot together recover n (it could be both 5 and 2017) and also B,D,E cannot together recover n (it could be both 1514 and 2017).

That exhausts all the possibilities, so we can summarize: a subset of $\{A, B, C, D, E, F\}$ can recover n if and only if it contains at least one of the following subsets:

$$\{D, F\}, \{A, E, F\}, \{B, E, F\}, \{B, C, D\}.$$

9.20. No, it is not possible. Proof by contradiction:

Let t be the threshold and a, b, c, d, e the numbers of shares belonging to the participants A, B, C, D, E respectively.

Because the sets $\{B, C, D\}$ and $\{A, D, E\}$ are authorized, it holds that

$$b + c + d \geq t \tag{9.1}$$

and

$$a + d + e \geq t. \tag{9.2}$$

However, the sets $\{B, E, D\}$ and $\{A, D, C\}$ are not authorized, therefore

$$b + e + d < t \tag{9.3}$$

and

$$a + d + c < t. \tag{9.4}$$

From (9.1) and (9.3) we get

$$c \geq e$$

and from (9.2) and (9.4) we get

$$e \geq c$$

which means that

$$c = e.$$

Substituting $c = e$ in (9.1) gives us

$$b + e + d \geq t$$

which contradicts (9.3).

9.21. When $v\gamma \equiv \alpha^{k_1 - a_1} \equiv 1 \pmod{p}$ we get

$$k_1 - a_1 \equiv 0 \pmod{q}.$$

Now because $0 \leq a_1, k_1 \leq q - 1$ it must hold $a_1 = k_1$. In that case

$$y_1 = a_1 + a_1 r = a_1(r + 1) \pmod{q}.$$

So to get a_1 , Bob just needs to calculate

$$a_1 = y_1(r + 1)^{-1} \pmod{q}.$$

9.22. We know that if p is a prime and $d \leq 2$ is an integer then there exists an orthogonal array

$$\text{OA} \left(p, \frac{p^d - 1}{p - 1}, p^{d-2} \right).$$

For $p = 3$ and $d = 2$ we have that $\text{OA}(3, 4, 1)$ exists. To construct such array we can extend the $\text{OA}(3, 3, 1)$ with the fourth column as follows:

$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 2 & 2 & 2 & 0 \\ 0 & 1 & 2 & 1 \\ 1 & 2 & 0 & 1 \\ 2 & 0 & 1 & 1 \\ 0 & 2 & 1 & 2 \\ 1 & 0 & 2 & 2 \\ 2 & 1 & 0 & 2 \end{pmatrix}$$

9.23.

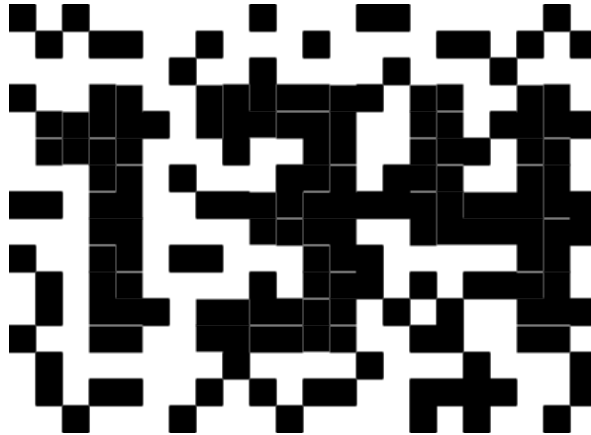
- Take \mathbb{Z}_n and to all t -tuples $(x_1, \dots, x_t) \in \mathbb{Z}_n^t$ add a parity element, ie. $x_{t+1} = -(x_1 + \dots + x_t) \pmod{n}$. These $(t + 1)$ -tuples form t - $(n, t + 1, 1)$ orthogonal array.
- Take a t - (n, k, λ) OA and choose an element e in it. Now take all rows where e is in the first position and delete the first position to obtain a subarray. This subarray forms $(t - 1)$ - $(n, k - 1, \lambda)$ OA.

9.24.

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

9.25. The construction of the threshold scheme starting from the orthogonal array proceeds as follows. The first column of the OA corresponds to the dealer and the remaining $k - 1$ columns correspond to the $k - 1$ participants. To distribute a secret S , the dealer selects a random row of the OA such that S appears in the first column and gives out the remaining $k - 1$ elements of the row as the shares. When s participants later pool their shares, the collective information will determine a unique row of the OA (as $\lambda = 1$) and hence they can compute S as the value of the first element in the row. A group of $s - 1$ participants is not able to compute S . Any possible value of the secret along with the actual shares of these $s - 1$ participants determine a unique row of the OA. Hence, no value of the secret can be ruled out. Moreover, it is clear that the $s - 1$ participants can obtain no information about the secret.

9.26. When these two patterns are combined and stacked on top of each other (either we can use transparencies or we just let light pass through papers) we obtain the following result.



9.27. The function f is CBC-MAC. Given two message-tag pairs (m, t) and (m', t') then t' is valid authentication code for the following message as well:

$$m'' = m || [(m'_1 \oplus t) || m'_2 || \dots || m'_x].$$

We have $E_k(m) = t$ and in the next step we apply exclusive or operation on the value t with $m_1 \oplus t$ canceling out t therefore the tag t' is valid for m'' .

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 $V = \{1, \dots, n\}$. The following steps are then repeated t times (where t is a chosen security parameter).

- (1) Peggy chooses a random permutation π of $V = \{1, \dots, n\}$ 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 $V = \{1, \dots, n\}$ 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 , $x_0 = x^2 \pmod n$, and $x_1 = x_0^2 \pmod n$. 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 $x_0 = x^2 \pmod n$ and $x_1 = x_0^2 \pmod n$. 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 $commit(b, r) = y^{br^2} \pmod n$ 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, \dots, 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 $j \in \{1, \dots, n\}$. 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 $j \in \{1, \dots, n\}$. 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 $y = x^2 \pmod{n}$ and sends y to Peggy.
 - (2) Peggy computes all four square roots of $y \pmod{n}$, 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 dishonest 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

$$\prod_{j=1}^{i-1} g^{x_j} \Big/ \prod_{j=i+1}^n g^{x_j} .$$

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.)

10.16. Consider the following commitment scheme, with public information as follows: p a large prime, q a large prime dividing $(p - 1)$, $g \in \mathbb{Z}_p^*$ of order q , and $h = g^k \pmod p$, with $0 < k < q$ a random integer not known to any party. The commitment function is

$$\text{commit}(r, x) = g^r h^x \pmod p,$$

where x is the committed bit and $0 < r < q$ is a random integer.

- (a) Define the reveal phase of this protocol.

- (b) Discuss the binding and hiding properties of this protocol. Are they computationally/information theoretically secure?
- (c) What happens if Bob (the receiver) knows $\log_g h$?
- (d) What happens if Alice (the sender) knows $\log_g h$?

10.17. Consider the following coin flip protocol. Let $n = pq$, where p and q are large primes.

- i. Alice generates a random integer $0 \leq a < n - 1$ and sends $c = a^2 \pmod n$ as her commitment to Bob.
- ii. Bob guesses the parity of a and tells his guess to Alice.
- iii. Alice reveals a to Bob and Bob checks that $c = a^2 \pmod n$.
- iv. If Bob guessed correctly, he wins the coin flip.

Is the proposed protocol fair?

10.18. Alice and Bob have a boolean function $f : \{0, 1\} \times \{0, 1\} \rightarrow \{0, 1\}$ known to both of them. Show that if f is not a constant function there is no protocol that lets them perfectly evaluate $f(a, b)$ on all respective secret inputs a and b without neither party obtaining any information about the other party's secret.

10.19. As the semester comes to an end, you and your colleague want to find out whether both of you received the same grade from the cryptographic course without disclosing the grade itself. However, you do not want to use any mathematical technique. Provide a simple non-cryptographic way to achieve this task.

10.20. Victor is color-blind and cannot distinguish between colors at all. Peggy who can see colors has two apples, one green and one red, but otherwise identical. Design a zero-knowledge protocol that allows Peggy to convince Victor that the apples have different colors.

10.21. Assume that $f : \{0, 1\}^n \rightarrow \{0, 1\}^n$, for some large enough n , is a bijective one-way function, such that absolutely no information can be obtained about the input from a given output, known to both Alice and Bob. Design a coin tossing protocol between Alice and Bob that uses f and no other one-way function. Show that neither of the players can cheat in your protocol.

10.22. Secure function evaluation is a task in which Alice has an input $x \in \mathcal{X}$, Bob has an input $y \in \mathcal{Y}$ and they want to evaluate a function $g(x, y)$ in such a way, they both learn the outcome, but they do not learn each others inputs. Show that a 1-out-of- k oblivious string transfer protocol, in which k values input by Alice are bit-strings of arbitrary length can be used to implement secure function evaluation.

10.23. Alice and Bob are trying to use a binary symmetric channel with error probability $p = \frac{1}{2}$ to implement coin-tossing in the following way:

Alice chooses a random bit b and sends it to Bob through the binary symmetric channel. Bob receives the bit b' and then sends it back to Alice using different channel without errors. Now Alice takes both bits and calculates the output of the coin-tossing protocol as $b \oplus b'$. After this she also sends b to Bob through the perfect channel so he can calculate the same output. Assuming neither party can influence the binary symmetric channel (other than giving the input for Alice or accessing the output for Bob), discuss the security of this protocol.

10.24. Propose a generalization of a 1-out-of-2 oblivious transfer – implementation that combines a public-key cryptosystem and a secret-key cryptosystem – to enable a k -out-of- n oblivious transfer.

10.25. You are given a black box that implements the following protocol between two parties: One party has no inputs and has two outputs, random bit strings s_0 and s_1 of length n . The other party has input bit c and its output is the string s_c . Use this black box to implement a 1-out-of-2 oblivious transfer for sending messages of n bits.

Assuming the black box works exactly as described (the first party does not know c and the second party does not get to know $s_{c \oplus 1}$), show that your implementation is secure, *ie.* the sender of the oblivious transfer cannot learn which message was received and the receiver cannot learn both messages.

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 (*ie.* $p \equiv q \equiv 3 \pmod{4}$). 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 $x^4 \equiv y^4 \equiv x_1 \pmod{n}$. If $y_0 \equiv y^2 \pmod{n}$ 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 $i \in \{0, 1, \dots, p-1\}$, and also

$$c \equiv g^r m \equiv g^j \equiv g^{r'} g^i \equiv g^{r'} m' \pmod{p}$$

for appropriate $j, r' \in \{0, 1, \dots, p-1\}$, $j \equiv r' + i \pmod{p-1}$. 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 $B = \text{commit}(0, x)$ for some $x \in X$, there must exist an x' , such that $B = \text{commit}(1, x')$. 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 $\text{commit}(b, r)$ 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 $\text{commit}(b, r)$ 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 $-b$, she would have to find such r'_i that $r'_i \oplus -bc_i = r_i \oplus bc_i$. 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 $x_c = x_{-b} = g$. 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-of-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 $\sigma : \{1, \dots, n\} \mapsto \{1, \dots, n\}$.
- (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[3]. As $3\text{-SAT} \in \text{NP}$, 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 $\text{gcd}(r-x, n)$ 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[6].

- (a) We have $y_i = \sum_{j < i} x_j - \sum_{j > i} x_j$.

$$\begin{aligned} \sum_i x_i y_i &= \sum_i \sum_{j < i} x_i x_j - \sum_i \sum_{j > i} x_i x_j = \sum_{j < i} x_i x_j - \sum_{i < j} x_i x_j \\ &= \sum_{j < i} x_i x_j - \sum_{j < i} x_j x_i = 0 \end{aligned}$$

- (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.

10.16.

- (a) Alice simply reveals r and x . Bob checks whether $g^r h^x \pmod p$ is equal to the message he received in the commitment phase.
- (b) *Binding is computational.* This scheme is computationally binding (under the assumption of hardness of discrete logarithms), which can be seen as follows. Suppose it is computationally feasible to compute $r, r' \in \mathbb{Z}_p^*$ such that $\text{commit}(r, x) = \text{commit}(r', 1 - x)$. That means that

$$\begin{aligned} g^r h^x &= g^{r'} h^{1-x} \pmod p \\ g^r g^{kx} &= g^{r'} g^{k(1-x)} \pmod p \\ g^{r+kx} &= g^{r'+k(1-x)} \pmod p \\ r + kx &= r' + k(1-x) \pmod q \\ k(2x - 1) &= (r' - r) \pmod q \\ k &= (r' - r)(2x - 1)^{-1} \pmod q. \end{aligned}$$

Therefore, being able to open the commitment in both ways is as hard as calculating the discrete log problem for h with basis g in \mathbb{Z}_p^* .

Hiding is information theoretic. This can easily be seen by the fact that the distribution $g^r h^x$ is independent of x , i.e. g^r and $g^r h$ are statistically indistinguishable, because the value r is chosen at random.

- (c) This doesn't help Bob at all. Knowing k does not help him distinguish g^r from $g^{r+kx} \pmod q$. The reason for this that for each r there exists $r = r' + kx$. Since exponent's r are chosen uniformly at random, Bob cannot decide whether r was chosen and commitment is 0 or r' was chosen and commitment is 1.
- (d) In this case Alice can cheat. Assume she commits with $g^r h^x \pmod p$. Then she can find r' such that $g^{r'} h^{1-x} = g^r h^x$ as follows:

$$\begin{aligned} g^r h^x &= g^{r'} h^{1-x} \pmod p \\ g^r g^{kx} &= g^{r'} g^{k(1-x)} \pmod p \\ g^{r+kx} &= g^{r'+k(1-x)} \pmod p \\ r + kx &= r' + k(1-x) \pmod q \\ r + k(2x - 1) &= r' \pmod q. \end{aligned}$$

10.17. The protocol is not fair. In step three Alice can decide whether to reveal a or $-a \pmod n$. Since n is odd, a and $-a \pmod n$ have different parities.

10.18. Let inputs of $f(a, b)$ be that of Alice and Bob, respectively. For such function it must hold that $f(0, 0) = f(0, 1)$, otherwise Alice would be able to obtain b from a and $f(a, b)$: if $f(0, 0) = 0$ and $f(0, 1) = 1$ then $b = f(0, b)$; if $f(0, 0) = 1$ and $f(0, 1) = 0$ then $b = 1 - f(0, b)$. Analogically, it must hold $f(1, 0) = f(1, 1)$. Now considering Bob's point of view, it must hold that $f(0, 0) = f(1, 0)$, otherwise Bob who could obtain a . Together,

$$f(0, 0) = f(0, 1) = f(1, 0) = f(1, 1).$$

10.19. You get 6 lockable money-boxes (with small slots) and label them with each possible grade, ie. A to F. You throw away all keys except the one that corresponds to your grade. You leave the boxes and your colleague prepares small pieces of papers, writes “+” sign to one of them and “-” sign to other. Then he puts, through small slot, the paper with “+” sign to the locked box corresponding to his grade and papers with “-” sign to other locked boxes. Your colleague leaves the boxes, you return and open the box you have the only key for to find out whether there is a paper with “+” sign inside which would mean you and your colleague have received the same grade. Both you and your colleague are supposed to be honest.

10.20. Victor holds each apple in one hand, then places his hands behind his back and either switches the apples or not. Then he puts the hands in front of him and Peggy tells him if he switched them or not (which is trivial for her). This does not give Victor any information about their colors at all. After n repetitions of this protocol, Victor can be sure with probability $1 - \frac{1}{2^n}$ that Peggy can really distinguish the colors, hence that they must be different.

10.21. The protocol starts with Alice choosing a random input $x \in \{0, 1\}^n \rightarrow \{0, 1\}^n$. Alice then computes $f(x)$ and sends it to Bob. Bob now guesses the parity of Alice’s input and tells Alice. If he guesses correctly, Bob wins, otherwise Alice wins. Bob can verify the result by making Alice send him the input x .

This protocol is secure. Since the function is bijective, there’s no other $y \neq x$ such that $f(x) = f(y)$ and thus Alice commits to her x by sending $f(x)$ and cannot change her choice later to cheat. On the other hand, since no information can be gained about the input of function f from its input, Bob has no way of computing the parity of x from $f(x)$.

10.22. Alice calculates all $g(x, y_1), g(x, y_2), \dots, g(x, y_{|Y|})$, expresses them in binary and inputs $g(x, i)$ as the i -th input into the 1 -out-of- k OST protocol. Bob upon inputting y learns $g(x, y)$ and communicates it to Alice. Because OST protocol does not reveal x to Bob, all he can learn about x is can be deduced from $g(x, y)$. Reversely, since OST protocol does not reveal anything about Bob’s choice, all Alice learns about y can be deduced from $g(x, y)$. This is therefore an instance of a secure function evaluation for arbitrary function g .

10.23. This protocol is not secure, Alice has total control over the outcome of the protocol. Assuming Alice wants the outcome o , she can just claim this outcome and then send to Bob $o \oplus b'$ instead of b . Now Bob will calculate

$$o \oplus b' \oplus b' = o$$

and will also get the same outcome o .

10.24.

1. Alice and Bob agree on a public-key cryptosystem P and a secret-key cryptosystem S .
2. Alice chooses n public/private key pairs (e_i, d_i) , $1 \leq i \leq n$, from P and makes e_i public.
3. Bob chooses k symmetric keys s_j , $1 \leq j \leq k$, from S .
4. Bob chooses randomly k of Alice’s public keys, let us denote them e_{x_j} , encrypts all his secret keys s_j with Alice’s public keys e_{x_j} , respectively, and sends encryptions to Alice.
5. Alice does not know which public keys Bob has used so she decrypts $e_{x_j}(s_j)$ with each of her private keys d_i (exactly one of such decryptions will be Bob’s key s_j) and uses them to encrypt each of her n messages and sends them to Bob.
6. Bob knows for which message sent by Alice he can use s_j for decryption.
7. Steps 5. and 6. are repeated for all k Bob’s messages $e_{x_j}(s_j)$, therefore Bob learns k out of n messages.

10.25. First, the two parties use the black box, with the sender of the oblivious transfer being the party with no inputs into the black box. The sender now has the strings s_0 and s_1 , the receiver of the oblivious transfer has the string b_c . The sender now wants to send messages m_0 and m_1 . He first calculates $m'_0 = m_0 \oplus s_0$ and $m'_1 = m_1 \oplus s_1$. He now sends the pair (m'_0, m'_1) to the receiver. He can now use the string s_c to get the message m_c :

$$m_c = m_c \oplus s_c \oplus s_c = m'_c \oplus s_c.$$

Now this is exactly the 1-out-of-2 oblivious transfer protocol, sender sends messages m_0 and m_1 , while the receiver gets only the message m_c according to his choice c . This implementation is also secure from both sides.

If the sender can learn the choice c she can also learn the bit c that's the output of the black box since these bits are the same, but we are assuming she cannot learn this bit (and since the receiver has no additional output, she could just simulate our 1-out-of-2 oblivious transfer protocol to break the black box). Now again if the receiver can learn both messages m_0 and m_1 , then he can also learn both of the strings s_0 and s_1 which we again assume is impossible.

Chapter 11

Steganography and Watermarking

11.1 Introduction

Steganography and *Watermarking* are very important areas of the art, science, and technologies of *information hiding*. They differ from cryptography in specific ways:

Cryptography goals are to make transmitted messages *unreadable* by undesirable parties but not merely to hide the communication itself, while

Steganography and watermarking goals are to *hide or conceal messages* against undesirable parties in the case of watermarking in an un-removable way.

11.1.1 Steganography

Stegosystem consist of:

Set of cover text (cover-data or cover-object) is set of objects used to communicate a hidden message.

Set of messages to be communicated.

Embedding process serves to hide a message by embedding it into a cover object to obtain stego-text (stego-data or stego-object).

Stegotext (stego-data or stego-object) is the message that comes out of the embedding process and contains the hidden message.

Recovering process (or extraction process) serves to recover the hidden message from the stego-object.

Security requirement: third person watching such a communication should not be able to find out whether the sender has been steganographically active, in the sense that she/he embedded a message in the cover-data. In other words, stego-data should be *indistinguishable* from cover-data even if some powerful technology is used.

Basics steganographic techniques:

Substitution technique replace redundant (noisy) parts of the cover-object with the message.

Transformed domain techniques: the message is embedded in some transform space of the data (e.g. in the frequency domain).

Cover generation techniques: the message is not embedded in randomly chosen (or predetermined) cover-object, but cover is created or synthesized to fit the message.

Statistical techniques: the message is embedded by changing some statistical properties of the cover-object and recovered using hypothesis-testing methods in the extraction process.

Examples of stegosystems

Linguistic (or text) steganography: allow to hide messages in formatted texts, e.g.:

Acrostic: A message is hidden into certain letters of the text, for example in the first (last or other predetermined) letters of some words.

Changes in stylistic features: Stylistic features (at predetermined points) can be used to embed a message. Some elementary techniques of this type are: (1) Lines shifting; (2) Words shifting; (3) Data hiding through justifications. (4) Data hiding through features encoding (for example, using slightly different directions of the vertical lines of letters **b, d, h, k**)¹.

Unfortunately the text steganography (a really good one) is considered to be very difficult due to the lack of redundancy in texts comparing to images or audio.

Hiding message in noisy (parts of) data. Perhaps the most basic and general methods of steganography is to utilize the existence of redundant information in communication channels/media. Noise is especially relevant to steganography because seemingly useless stochastic components in natural digital images or audio data could convey messages very well.

Least significant bit (LSB) embedding: Consider a natural image represented by RGB pixels we can assume that least significant bits of each pixel color represent noise. In this embedding we replace them with bits of a message. Unfortunately, this method does not provide high level of security and can change significantly statistical properties of the cover-data as well.

Other methods include hiding messages to the *frequency domain* using, e.g., discrete cosine transformation (DCT) or using techniques from linear codes, i.e., *matrix embedding*.

Covert channels examples are: *timing based* (modifying timing of performed operations) or *storage based* (modifying storage location of data). Used e.g. in network steganography exploiting elements of communication protocols to cover hidden data.

Combination of cryptography with steganography

It is often useful to enhance steganography with cryptosystems. We distinguish *secret-key stegosystems* and *public-key stegosystems* which use secret-key and public-key cryptography to encrypt the message before it is embedded.

11.1.2 Watermarking

Watermarking is usually used for embedding information called *watermarks* into *data* to create *watermarked data*, in order to identify authors or owners or to authenticate the origin (or augment transmitted data) of data in an *un-removable and irreplaceable* way. The main differences between watermarking systems and stego-systems are: in watermarking systems the input data are important (we are not using synthesis of cover data). The embedded data are usually small and should be embedded in a fairly robust way (if data is copied so is the embedded watermark and its detectability is not increased and hence secrecy should not change).

11.1.3 Parameters of stego- and watermarking systems

Important parameters of data hiding systems are *security, robustness, and capacity*. These parameters are inversely related creating so called *data-hiding dilemma*.

¹F. Bacon was likely first who realised that by using italic or normal fonts one can encode binary representation of letters.

11.1.4 Breaking cryptography, steganography, and watermarking systems

- A *cryptographic system* is broken when the attacker can *read* the secret message.
- Breaking of a *steganographic/watermarking system* has two stages:
 1. The attacker is able to *detect* that steganography/watermarking has been used;
 2. The attacker is able to *read, modify or remove* the hidden message.

A steganography/watermarking system is considered as insecure already in the case that a detection of an application on the received data of some steganography/watermarking process has been, with a large probability, determined.

11.2 Exercises

11.1. You received the following email from a friend of yours.

Hello!

Eve is coming home tomorrow. Last week she called me. Please take care of her. Maybe you can go to a cinema or theater. Enjoy the time you spent together!

11.2.

Pius gloriosus	Conditor gloriosus
Aeternus dominus	Rector magnus
Pacificus imperator	Redemptor imperator
Gloriosus illustrator	Gloriosus illustrator
Piissimus fortissimus	Opifex fortissimus
Auctor creator	Auctor creator
Deus pacificus	Clemens redemptor
Conseruator iudex	Omnipotens iudex
Princeps gubernator	Princeps fortissimus
Aeternus redemptor	Rector redemptor
Gloriosus opifex	Gloriosus opifex
Clemens pastor	Clemens pastor

11.3. On your trips, you find the following Latin inscription.

BELGRADI CAESA EST LVNA PER EVGENIVM

You are educated in history so you know that the translation from Latin to English reads “At Belgrade the moon was defeated by Eugene.” which means that Prince Eugene of Savoy defeated the Turkish army (symbolized with the crescent) at Belgrade. However, you cannot remember when this event happened. . .

11.4.

Poor soul, the centre of my sinful earth,
 [...] these rebel powers that thee array;
 Why dost thou pin within and suffer dearth,
 Painting thy outward walls so costly gay?
 Why so large cost, having so short a lease,
 Dost thou upon thy fading mansion spend?
 Shall worms, inheritors of this excess,
 Eat up thy cherge? Is this thy body's end?
 Then sol, live thou upon thy servantes loss,
 And let that pine to aggravate thy store;
 Buy terms diviene in selling hours of dross;
 Within be fed, without be rich no more:
 So shalt thou feed on Death, that feeds on man,
 And Death once dead, there's non more dying then.

(Shakespeare: The Sonnets, 1609)

11.5. *(You need the electronic version to solve this exercise.)*

Steganography also includes the practice of writing in invisible ink. As far back as the first century A.D., Pliny the Elder explained how the “milk” of the tithymalus plant could be used as an invisible ink. Although the ink is transparent after drying, gentle heating chars it and turns it brown. Many organic fluids behave in a similar way, because they are rich in carbon and therefore char easily. Indeed, it is not unknown for modern spies who have run out of standard-issue invisible ink to improvise by using their own urine.

(Simon Singh: The Code Book)

11.6. Whoever opined, “Money can’t buy you happiness,” obviously had far too much of the stuff.

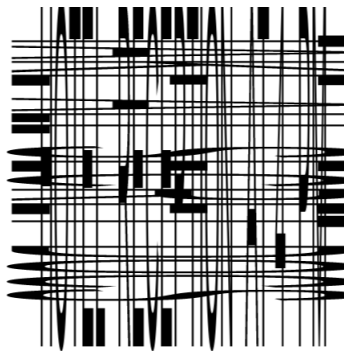
(David Mitchell: Cloud Atlas)

11.7. You are reading the newspapers and the following invitation to an art exhibition attracts your attention. You quickly know that you should come to another place.


MODERNART

exhibition

10-15 April, City Hall



11.8. A microdot. *(Might not be visible in a printed version.)*

11.9. You are given the following picture  (PDF attachment tested on macOS Adobe Acrobat Reader 19; macOS/Ubuntu Firefox 71) recover the secret message hidden in the least significant bits of the RGB channels.



11.10. Alice and Bob agreed to use LSB embedding to insert a secret data into an image. Carl suggested to them to use PNG or JPEG file formats to save amount of transferred data. Is it a good idea? Explain.

11.11. Alice and Bob are using the following message structure to communicate. Determine at least three ways how Alice can embed a hidden message for Charlie into her communication with Bob without disturbing the data.

1. header
 - (a) time & date
 - (b) message structure info
 - i. version of the message structure (currently 0) (1 byte)
 - ii. info about data encryption and authentication (4 bytes)
 - iii. reserved (2 bytes), in the current version omitted by the receiver
 - iv. data offset (2 bytes)
 - v. data length (6 bytes)
 - (c) padding checked by receiver to be all zeros padding the whole message to powers of two
2. data payload including error correction

Watermarkings

11.12. Explore and learn about security features used in Czech banknotes. Document² at least five examples.

²Please always follow the current laws and regulations.

11.13. Yellow dots or printer watermark. We have found a pattern captured in the Figure 11.1 in a document⁵. Can you determine the time & date the document was obtained and whether it was printed or copied and the serial number of the printer?

(Important note: as we found, this printer has a bit different pattern than what is publicly known to the best of our knowledge. Please use only 4 least significant bits of information from the month region to determine the month correctly.)

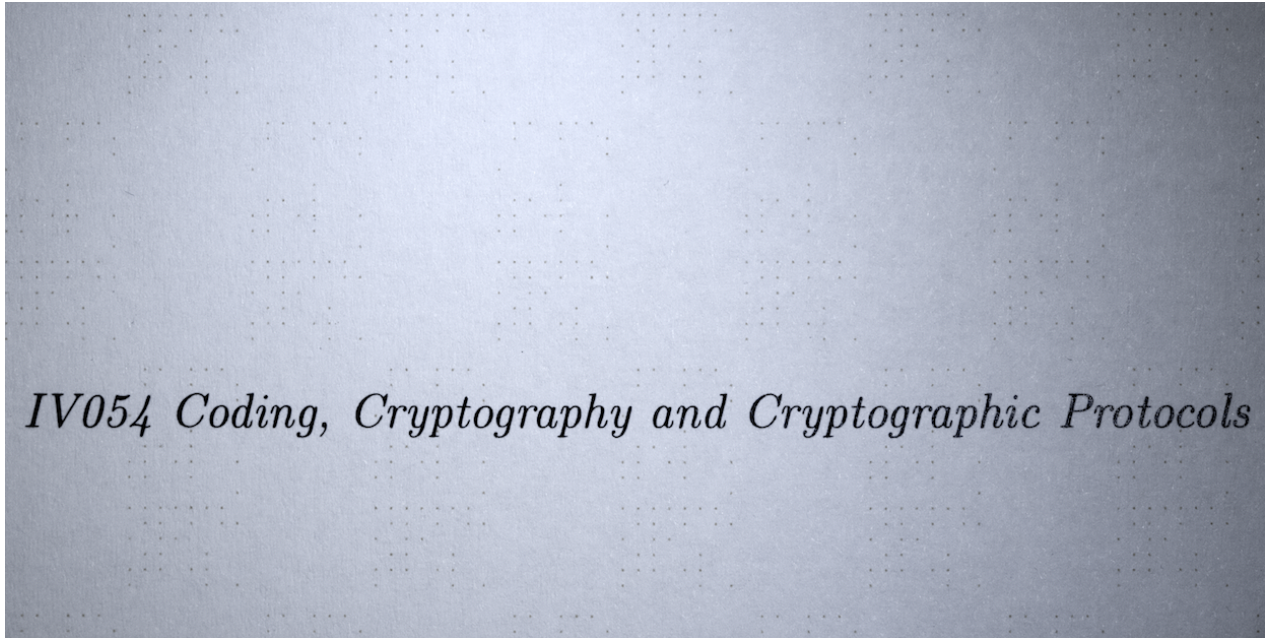


Figure 11.1: Post-processed photo of the document with a yellow dots pattern. The diameter of a single dot is roughly 0.1 mm. If you would like to try to see it for yourself, you can use a blue light, which makes the yellow dots look darker, and examine a paper printed on a color printer (some of them prints similar watermark).

11.3 Solutions

11.1. This is so-called acrostic code. First letters of sentences forms message: “HELP ME”.

11.2. This is Ave Maria Cipher, where cryptotext looks innocently and one can think it is merely a prayer in Latin. There exists a codebook: for each plaintext letter there are several Latin words that can be used to encrypt it. Solution of the exercise is: “COME TO OLD PUB AT SIX PM TODAY”.

11.3. This is a chronogram, an inscription where several letters are interpreted as numbers. Usually Roman letters are used, ie. I, V, X, C, D and M. The letter “U” is usually replaced with “V”.

BELGRADI CAESA EST LVNA PER EVGENIVM

	M	1000
	D	500
	C	100
LDICLVVIVM = MDCLLVVVII	LL	50 + 50 = 100
	VVV	5 + 5 + 5 = 15
	II	1 + 1 = 2
		1717

⁵We want to thank Daniel Reitzner for helping took photos and post-processed the image.

11.4. The given text is the 164th sonnet from Shakespeare, but some words contain typos. The original version reads

Poor soul, the centre of my sinful earth,
 [...] these rebel powers that thee array;
 Why dost thou **pine** within and suffer dearth,
 Painting thy outward walls so costly gay?
 Why so large cost, having so short a lease,
 Dost thou upon thy fading mansion spend?
 Shall worms, inheritors of this excess,
 Eat up thy **charge**? Is this thy body's end?
 Then **soul**, live thou upon thy **servant's** loss,
 And let that pine to aggravate thy store;
 Buy terms **divine** in selling hours of dross;
 Within be fed, without be rich no more:
 So shalt thou feed on Death, that feeds on **men**,
 And Death once dead, there's **no** more dying then.

(Shakespeare: The Sonnets, 1609)

Taking the first letters of the words following the words with typo's we receive the hidden message "William".

11.5. There is a hidden message written in the background color between words. Hint: Try copying the text from PDF into your clipboard and then pasting it into a text file. The following can be seen:

AlthoughItheNinkVisItransparentSafterIdrying,Bgentle heatingLcharsEitIandNturnsKit
 The message reads: "INVISIBLE INK".

11.6. There is a hidden message encoded in tiny rotations of letters. The message is itself is in character code, while a presence of rotation represents 1 and its absence 0. We can see the tiny rotations on the overlap of the quotation with an unmodified text:

Whoever opined, "Money can't buy you happiness," obviously had far too much of the stuff.

If the letter in the original text is rotated, write down 1, otherwise write 0. The hidden binary sequence reads:

101001010001011101011101000101100000110011001000101100010000000000000000,

and the hidden message is "REVEALED" (in 7-bit ASCII).

11.7. When you tilt the paper, you may be able to read a secret message. You should look within an acute angle with the paper almost parallelly. The hidden message reads:

"HOTEL RECEPTION AT 4 PM TAKE ALL SECRET DOCS".

11.8. There is a message hidden in a microdot in the last full stop hiding the text "steganographia" written phonetically in Greek alphabet *στεγανογραφια*.



11.9. We can use Wolfram Mathematica to recover the least significant bits and the secret message repeating “IV054” in the 8-bit ASCII code.

11.10. Both PNG and JPEG file formats use compression. While the PNG file format uses lossless compression and hence is suitable for LSB embedding the JPEG file format uses lossy compression and is not suitable for this task.

11.11. We found the following three embeddings.

1. Time based embedding. Alice can choose to transmit data at certain time frames to encode information.
2. Using of the reserved bits. There are reserved bytes which are omitted by the receiver, Alice can use them to hide some information.
3. Length of the padding. Alice can alter the length of the padding to hide some information.

11.12. We choose to document security features in 1000 Czech banknote. The following pictures are after ČNB³, and are used with their kind permission. For a printed version please always respect the current laws and regulations (currently [Dec 2019] vyhláška č. 274/2011 Sb. – část sedmá⁴). We document 6 security features: watermark (no. 1 in the following picture), windowed thread with microtext (no. 2), coloured fibres (no. 3), iridescent strip (no. 7), colour-shifting ink (no. 6), and front-to-back register (no. 4).



Watermark. There is stepped watermark (combination of lighter and darker) portrait of František Palacký and negative watermark of 1000 and the leaf visible when the banknote is hold against a light.



Windowed thread with microtext There is metallic plastic, 3 mm wide, strip with the microtext “1000 ČNB” embedded into the banknote. When viewed against a light we can see the continuous strip.

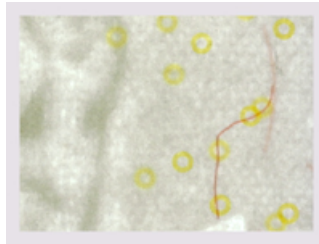
³Available online [accessed: November 27th, 2019]

<https://www.cnb.cz/en/banknotes-and-coins/banknotes/protective-elements-czk-1000/>

⁴https://www.cnb.cz/export/sites/cnb/cs/legislativa/.galleries/vyhlasiky/vyhlasika_274_2011_uplne_zneni.pdf



Coloured fibres There are orange fibers 6mm long embedded into the paper.



Iridescent strip There is an iridescent strip with a metallic gold and violet reflection when the banknote is tipped against a light.



Colour-shifting ink There is a leaf printed with a special ink that change color from gold to green (due to optical effect) when examined under different angles of a light.



Front-to-back register There are portions of the following symbol printed on each side of the banknote. The complete mark “ČR” is visible when hold against a light.



11.13. To decode the pattern, we will use Buck P., Reverse Engineering the Machine Identification Code, 2018, doi: 10.13140/RG.2.2.28980.76169 and the references herein especially DocuColor Tracking Dot Decoding Guide, EFF 2005⁶. This particular pattern is used, e.g., by Xerox printers.

⁶Online in archive accessed 30/11/2019 (the original document seems not to be on the EFF server any more) <https://web.archive.org/web/20180102231955/https://w2.eff.org/Privacy/printers/docucolor/>

The data are encoded in binary format with parity bits and we display them in the following matrix. We marked the parity bits by gray, there are parity row and parity column in the encoding. Taking the dots as ones and the empty spaces as zeroes the pattern encodes 14 numbers each in 7 bits. The 2nd column encodes hours, the 5th encodes minutes, the 6th day, the 7th month (we take only the 4 least significant bits), and the 8th year. The serial number of the printer is encoded in the columns 13, 12, and 11, and its product number (which we are not ask to determine) in columns 15 and 14. Moreove, the 10th column is (excluding the parity bit) all ones whenever the document was printed and all zeroes whenever it was copied.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
2^6	•	•	•				•		•		•				•
2^5	•									•			•	•	•
2^4	•	•			•	•	•	•		•	•			•	
2^3	•						•			•		•			
2^2	•					•				•	•	•	•	•	
2^1	•	•				•	•	•		•	•		•	•	
2^0	•			•			•	•		•		•		•	•

Therefore, we can decode:

Time & date: 16:18, 22/11/2019,
 Serial number: 381386,
 Print or copy: print.

Chapter 12

Quantum cryptography

12.1 Introduction

Quantum cryptography is an area of science and technology that utilizes potential of quantum phenomena for getting better security for some cryptography tasks. A new, and of the key importance, feature of quantum cryptography is that security of quantum cryptographical protocols is based on the laws of nature — of quantum physics, and not on unproven assumptions about computational complexity of certain tasks as we could see in case of classical cryptography.

In this chapter we introduce basics of quantum mechanics and two cryptography protocols — Quantum Key Distribution (QKD) and Quantum one-time pad.

12.1.1 Basics of quantum mechanics

The basic information unit in quantum information is called a *qubit* (quantum bit). Mathematically, the space of all qubits is represented by a 2 dimensional complex vector space, so called the *inner-product space*, H_2 , on which binary function called the *inner product*) is defined as follows:

$$\forall \psi, \phi \in H_2, \psi \cdot \phi = \psi_1^* \phi_1 + \psi_2^* \phi_2,$$

where $\psi = \begin{pmatrix} \psi_1 \\ \psi_2 \end{pmatrix}$, $\phi = \begin{pmatrix} \phi_1 \\ \phi_2 \end{pmatrix}$ and $*$ is a complex conjugation. This allows to define on H_2 *norm of vectors*

$$\|\phi\| = \sqrt{|\phi \cdot \phi|}.$$

Vectors $\phi \in H_2$, with $\|\phi\| = 1$ are called *pure qubit states*. Generally you can therefore think of a qubit state as a complex vector $\begin{pmatrix} \alpha \\ \beta \end{pmatrix}$, with $|\alpha|^2 = |\beta|^2 = 1$.

Dirac bra-ket notation

We will be using so called *bra-ket notation* for quantum states and linear functionals $f : H_2 \rightarrow C$.

- $|\psi\rangle$ is called a *ket vector* and it denotes a column vector equivalent to ψ .
- $\langle\psi|$ is called a *bra vector* and it denotes a complexly conjugated row vector ψ .
- $\langle\psi|\phi\rangle$ then denotes the inner product between ψ and ϕ .

Example: For states $\phi = \begin{pmatrix} \phi_1 \\ \phi_2 \end{pmatrix}$ and $\psi = \begin{pmatrix} \psi_1 \\ \psi_2 \end{pmatrix}$ we have

$$|\phi\rangle = \begin{pmatrix} \phi_1 \\ \phi_2 \end{pmatrix}, \langle\phi| = (\phi_1^*, \phi_2^*); \langle\phi|\psi\rangle = \sum_{i=1}^2 \phi_i^* \psi_i, |\phi\rangle\langle\psi| = \begin{pmatrix} \phi_1 \psi_1^* & \phi_1 \psi_2^* \\ \phi_2 \psi_1^* & \phi_2 \psi_2^* \end{pmatrix}$$

Quantum projective measurements

Two pure qubit states are called orthogonal in case their scalar product is zero. A very important fact is that two unknown pure qubit states are physically perfectly distinguishable if and only if they are orthogonal. A *basis* of the Hilbert space H_2 is a set of two mutually orthogonal pure qubit states. The canonical basis has special labels $|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$. Together with the information presented earlier we recover the typical notation for a qubit

$$|\psi\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \alpha |0\rangle + \beta |1\rangle.$$

Nevertheless, there are infinitely many other bases, and a pure state ψ can be expressed in basis $\{|b_1\rangle, |b_2\rangle\}$ as:

$$|\psi\rangle = \langle b_1|\psi\rangle |b_1\rangle + \langle b_2|\psi\rangle |b_2\rangle.$$

Numbers $\langle b_1|\psi\rangle$ and $\langle b_2|\psi\rangle$ are called *amplitudes*.

Now we are ready to define *projective measurements*. To each such measurement a basis $\{|b_1\rangle, |b_2\rangle\}$ is associated. There are two outcomes of a projection measurement of a state $|\phi\rangle$ with respect to a basis $\{|b_1\rangle, |b_2\rangle\}$:

1. Classical result of such a measurement is information into which subspace (spanned either by $|b_1\rangle$ or $|b_2\rangle$) projection of $|\psi\rangle$ was made.
2. Quantum result of such a measurement is the projection (collapse) into the particular subspace.

If the state $|\psi\rangle$ is measured with respect to the basis $\{|b_1\rangle, |b_2\rangle\}$ then the state $|\psi\rangle$ collapses into the state $|b_i\rangle$ with the probability $|\langle b_i|\psi\rangle|^2$. The classical outcome of the measurement of the state $|\psi\rangle$ with respect to the basis $\{|b_1\rangle, |b_2\rangle\}$ is the index i of that state $|b_i\rangle$ into which the state $|\psi\rangle$ collapses.

12.1.2 Quantum cryptography

There are two important postulates about quantum mechanics that quantum cryptography utilizes.

- There is no way in general to measure quantum states without destroying or disturbing them.
- So called *No-cloning theorem* says that there is no physical way to make a perfect copy of unknown quantum state.

This means that if a party (an eavesdropper) sees a state known to him to be from a small set of not mutually orthogonal set of states, then the eavesdropper has no way to copy such a state and to identify perfectly this state by a repeated measurement.

Quantum Key Distribution

Quantum protocols for using quantum systems to achieve unconditionally secure generation of secret (classical) random keys by two parties are one of the main achievements of quantum information processing and communication research and development. We will present QKD protocol called BB84 (due its authors Charles Bennett and Gilles Brassard), for quantum generation of a classical binary random and shared key of length n . It has several phases:

Raw key generation phase:

- 1.) In each of $2n$ rounds Alice randomly prepares one of the following pure states:

$$\left\{ |0\rangle, |1\rangle, |+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), |-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \right\}.$$

- 2.) Alice creates a bit string of length $2n$, where i th position has value 0 if she sent $|0\rangle$ or $|+\rangle$ or 1 if she sent $|1\rangle$ or $|-\rangle$.

- 3.) Bob measures each received qubit in randomly chosen basis $\{|0\rangle, |1\rangle\}$, or $\{|+\rangle, |-\rangle\}$ and records the outcome in the following way. If the measurement outcome was $|0\rangle$ or $|+\rangle$ he writes down 0 and he writes 1 otherwise.
- 4.) At the end of the transmission Bob informs Alice about the sequence of $2n$ measurement settings he has chosen and Alice tell him in which cases he used the same basis for decoding as she used for encoding. Both of them delete from their respective strings those bits where different bases were used.

The resulting bit string should be of roughly length n and is called a *raw key*. If the quantum channel they were using was perfect, their strings are identical, random and private. Nevertheless if it contains errors, they can be used to detect an adversary.

Parameter estimation:

Alice and Bob select a small portion of raw keys and publicly compare them in order to estimate the number of errors in their remaining bits and the amount of bits the adversary can know about Alice's string. If too many errors are estimated, they terminate the protocol, otherwise they continue.

Error correction: Alice and Bob exchange some additional information in order to correct the errors in their remaining raw strings K_a and K_b . Two protocols that achieve this task are described below. After this step both Alice and Bob have identical strings, which may be partially known to the adversary at the cost of revealing additional information to the adversary.

Privacy amplification: If after the error correction the adversary does not know the whole Alice's string K_a (say the adversary does not know k bits), Alice chooses randomly a hash function from a 2-universal family of hash functions mapping strings to k bits (see below), applies it to her string K_a and publishes her choice. Bob uses the same hash function on his (now identical) string. The resulting strings are (much) shorter, but the adversary has no information about them.

Error Correction I: Cascade protocol

Interactive binary reconciliation protocol that works first by estimating the error rate and then splitting the raw strings into blocks of size N such that each block has not more than one error on average. the protocol between Alice and Bob is then:

- Step 1.** Alice and Bob compare parities of every block using a classical public channel.
- Step 2.** Every block where the parities disagree is split into two blocks of length $N/2$ and the parity of the first sub-block is compared, this parity reveals which sub-block has at least one error.
- Step 3.** The binary search continues in the sub-block where the parities disagree until an error is found and corrected.
- Step 4.** After every initial block has been searched the key is randomly permuted and the binary search is repeated with double the block size. The protocol is repeated until the block size reaches the raw string length. In every pass following the first, every error found must have been masked by another error in the previous passes, this second error can then be found by correcting the original error in the previous passes and comparing the new parities, lowering the amount of parity bits required to find this error.

The adversary gains k -bit knowledge about the final string, where k is the number of parities disclosed during the run of the cascade protocol.

Error Correction II: Using ECC

Non-interactive information reconciliation protocol can be formulated with the use of error correcting codes. Assume Alice and Bob estimate that their raw strings of length n contain t errors. The error correction protocol is then as follows:

- Step 1.** Alice generates a random bit string x of length k .
- Step 2.** Alice uses a generator matrix G of an error correcting (n, k, d) code, with $2t + 1 \leq d$ to encode x and gets the code word c .
- Step 3.** Alice uses the raw key K_a to do bitwise XOR operation with the code string c to get $K_a \oplus c$. Then she transmits it to Bob.
- Step 4.** Bob does the same operation to the received string with K_b and gets $(c \oplus K_a) \oplus K_b = c \oplus e$, where e is the error vector of weight t . He uses the parity check matrix H and $c \oplus e$ to calculate the syndrome s . Using s , he gets the error vector e and the codeword c . Then he gets the random bit string x by decoding c .
- Step 5.** Bob calculates $(c \oplus K_a) \oplus c$ to obtain Alice's raw key K_a .

Privacy Amplification: 2-Universal hash functions

Consider a hash family $H = \{h_i\}_{i=1}^k$ where $h_i : U \mapsto \{0, \dots, n - 1\}$ for each i . H is 2-universal if for all $x, y \in U$ such that $x \neq y$ we have

$$\Pr_{h \in_r H} (h(x) = h(y)) \leq \frac{1}{n}$$

where $h \in_r H$ means that h is selected uniformly at random from H .

Quantum one-time pad cryptosystem

Quantum one-time pad cryptosystem is defined as follows:

- plaintext:** an n -qubit string $|p\rangle = |p_1\rangle \dots |p_n\rangle$
- shared key:** two n -bit strings k, k'
- cryptotext:** an n -qubit string $|c\rangle = |c_1\rangle \dots |c_n\rangle$
- encoding:** $|c_i\rangle = \sigma_x^{k_i} \sigma_z^{k'_i} |p_i\rangle$
- decoding:** $|p_i\rangle = \sigma_z^{k'_i} \sigma_x^{k_i} |c_i\rangle$

where $|p_i\rangle = \begin{pmatrix} a_i \\ b_i \end{pmatrix}$ and $|c_i\rangle = \begin{pmatrix} d_i \\ e_i \end{pmatrix}$ are qubits and $\sigma_x = X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ (bit flip) with $\sigma_z = Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$ (phase flip) are Pauli matrices.

Unconditional security of quantum encryption (informal)

Generally, quantum encryption protocol S works with a plaintext Hilbert space H_P , a cryptotext Hilbert space H_C and a keyspace K . In any quantum encryption protocol a key $k \in K$ determines an encryption (unitary) operation E_k and a decryption operation D_k such that for any plaintext $|\psi\rangle \in H_P$, $E_k |\psi\rangle$ is the corresponding cryptotext and it holds

$$|\psi\rangle = D_k(E_k |\psi\rangle).$$

In the case of encryption of a qubit

$$|\phi\rangle = \alpha|0\rangle + \beta|1\rangle$$

if the key is chosen uniformly from the key space, the following classical mixture of quantum states is being transmitted through the channel, each with probability $\frac{1}{|K|}$

$$\{E_1 |\psi\rangle, E_2 |\psi\rangle, \dots, E_{|K|} |\psi\rangle\}$$

Typically, classical mixture of quantum states is denoted in form of a *density matrix*. State $|\psi\rangle$ is represented by a density matrix $|\psi\rangle\langle\psi|$. A classical distribution of (pure) qubit states $|\psi_i\rangle$, each appearing with probability p_i is represented by a density matrix $\sum_i p_i |\psi_i\rangle\langle\psi_i|$. Note that multiple different mixtures of qubit pure states can have the same density matrix representation. The interpretation is that mixtures of pure states with the same density matrix representation are physically indistinguishable. For this reason density matrices are also considered to be physical states and are called *mixed states*.

The above mixture of states resulting from a qubit quantum encryption protocol has the following density matrix representation:

$$|\Psi_S\rangle\langle\Psi_S| = \frac{1}{|K|} \sum_{k \in K} E_k |\psi\rangle\langle\psi| E_k^\dagger$$

Qubit quantum encryption protocol is considered to be perfectly secure, if $|\Psi_S\rangle\langle\Psi_S|$ is identical to a uniformly random mixture of $|0\rangle$ and $|1\rangle$. This has the following density matrix representation

$$\begin{aligned} \frac{1}{2} |0\rangle\langle 0| + \frac{1}{2} |1\rangle\langle 1| &= \frac{1}{2} \left[\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \right] \\ &= \frac{1}{2} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \end{aligned}$$

The mixed state $\frac{1}{2} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ is called the *maximally mixed state* of qubits, because arbitrary measurement of such state gives a uniformly random outcome.

12.2 Exercises

12.1. Calculate the probabilities of measurement outcomes if the BB84 states

$$\left\{ |0\rangle, |1\rangle, |+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), |-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \right\}$$

are measured in basis:

- (a) $\{|0\rangle, |1\rangle\}$
- (b) $\{|+\rangle, |-\rangle\}$
- (c) $\{|+i\rangle = \frac{1}{\sqrt{2}}(|0\rangle + i|1\rangle), |-i\rangle = \frac{1}{\sqrt{2}}(|0\rangle - i|1\rangle)\}$
- (d) $\{|\varphi_0\rangle = \cos\left(\frac{\pi}{8}\right)|0\rangle + \sin\left(\frac{\pi}{8}\right)|1\rangle, |\varphi_1\rangle = \cos\left(\frac{5\pi}{8}\right)|0\rangle + \sin\left(\frac{5\pi}{8}\right)|1\rangle\}$

12.2. Assume the attacker performs the same so called *intercept-resend attack* in each round of the BB84 QKD protocol, in which the attacker measures the qubit state in the channel and sends the resulting post-measurement state to Bob. Estimate the amount of errors between Alice's and Bob's raw keys in the following cases:

- (a) The adversary measures each round in the $\{|0\rangle, |1\rangle\}$ basis.
- (b) The adversary measures each round in the basis $\{|0\rangle, |1\rangle\}$ or the basis $\{|+\rangle, |-\rangle\}$, each with probability $\frac{1}{2}$.
- (c) The adversary measures in each round in the:

$$\{|\varphi_0\rangle = \cos\left(\frac{\pi}{8}\right)|0\rangle + \sin\left(\frac{\pi}{8}\right)|1\rangle, |\varphi_1\rangle = \cos\left(\frac{5\pi}{8}\right)|0\rangle + \sin\left(\frac{5\pi}{8}\right)|1\rangle\}$$
 basis.

12.3. Assume the adversary uses an intercept-resend attack with measurement in $\{|\varphi_0\rangle, |\varphi_1\rangle\}$ basis each round with probability p .

- (a) How much error is introduced to Bob's raw string?
- (b) How much information can the adversary learn about Alice's raw string?
- (c) Calculate how many errors in the raw strings of Alice and Bob are too many for the key distillation.

12.4. Alice and Bob share the string 10101001 10100001 but Bob's copy has few bit errors and actually looks like 10101001 01110001. Use the Cascade protocol with starting block length of 4 to reconcile the error's on Bob's side. Show each step of the protocol. Use the permutation

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 \\ 15 & 8 & 14 & 2 & 9 & 10 & 3 & 5 & 4 & 11 & 16 & 12 & 13 & 7 & 1 & 6 \end{pmatrix}$$

in the second step of the protocol and omit discarding the bits of the blocks that had its parity revealed.

12.5. Perform the protocol for non-interactive error correction for strings

$$K_a = 101010011010000100110110 \text{ and } K_b = 101000011010010100111110,$$

with the use of Golay code G_{24} defined in exercise 2.27. Let Alice's random choice be

$$x = 100101100000.$$

12.6. Consider the following family of hash functions $h_{ab} : \{1, \dots, N\} \mapsto \{0, \dots, n-1\}$

$$H = \{h_{ab}(x) = [ax + b \pmod p] \pmod n, a \in \{1, \dots, p-1\}, b \in \{0, \dots, p-1\}\},$$

where $N \leq p \leq 2N$ is a prime. Show that H is 2-universal.

12.7. Consider the quantum one-time pad protocol.

1. Using the key $k_1 = 0$ and $k'_1 = 1$ encrypt the plaintext

$$|p_1\rangle = \frac{\sqrt{3}}{2} |0\rangle + \frac{1}{2} |1\rangle.$$

2. Using the key $k_2 = 1$ and $k'_2 = 1$ decrypt the cryptotext

$$|c_2\rangle = \frac{-i}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle.$$

* **12.8.** Prove the unconditional security of the quantum one-time pad cryptosystem by showing that the mixed state of an encrypted arbitrary qubit is exactly the state of a uniformly random bit.

* **12.9.** Show why each use of quantum one-time pad cryptosystem requires new random key.

* **12.10.** Consider a modification of the quantum one-time pad protocol: Instead of using 2 bits to encrypt one qubit $|p\rangle$, we only use one bit k with the following encryption:

- (a)

$$|c\rangle = \begin{cases} \sigma_x |p\rangle & \text{if } k = 0 \\ \sigma_z |p\rangle & \text{if } k = 1. \end{cases}$$

(b)

$$|c\rangle = \begin{cases} |p\rangle & \text{if } k = 0 \\ \sigma_x |p\rangle & \text{if } k = 1. \end{cases}$$

Show that these modifications aren't perfectly secure and find states which remain unencrypted by the protocol, if there are any.

12.3 Solutions

12.1. We need to express all states and measurement bases in the $\{0, 1\}$ basis. Then, since $\langle 0|0\rangle = \langle 1|1\rangle = 1$ and $\langle 0|1\rangle = \langle 1|0\rangle = 0$ we can calculate everything. We have that $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$. Further, recall that an absolute value of an imaginary number $a + ib$ is calculated as $|(a + ib)| = \sqrt{a^2 + b^2}$ and the following trigonometry identities

$$\begin{aligned} 2 \cos(\alpha) \cos(\beta) &= \cos(\alpha - \beta) + \cos(\alpha + \beta), \\ 2 \sin(\alpha) \sin(\beta) &= \cos(\alpha - \beta) - \cos(\alpha + \beta). \end{aligned}$$

Utilizing all these facts we have:

(a)

$$\begin{aligned} |\langle 0|0\rangle|^2 &= 1 \\ |\langle 1|0\rangle|^2 &= 0 \\ |\langle 0|1\rangle|^2 &= 0 \\ |\langle 1|1\rangle|^2 &= 1 \\ |\langle 0|+\rangle|^2 &= \left| \frac{1}{\sqrt{2}} (\langle 0|0\rangle + \langle 0|1\rangle) \right|^2 = \frac{1}{2} \\ |\langle 1|+\rangle|^2 &= \left| \frac{1}{\sqrt{2}} (\langle 1|0\rangle + \langle 1|1\rangle) \right|^2 = \frac{1}{2} \\ |\langle 0|-\rangle|^2 &= \left| \frac{1}{\sqrt{2}} (\langle 0|0\rangle - \langle 0|1\rangle) \right|^2 = \frac{1}{2} \\ |\langle 1|-\rangle|^2 &= \left| \frac{1}{\sqrt{2}} (\langle 1|0\rangle - \langle 1|1\rangle) \right|^2 = \frac{1}{2} \end{aligned}$$

(b)

$$\begin{aligned}
|\langle +|0\rangle|^2 &= \left| \frac{1}{\sqrt{2}} (\langle 0|0\rangle + \langle 1|0\rangle) \right|^2 = \frac{1}{2} \\
|\langle -|0\rangle|^2 &= \left| \frac{1}{\sqrt{2}} (\langle 0|0\rangle - \langle 1|0\rangle) \right|^2 = \frac{1}{2} \\
|\langle +|1\rangle|^2 &= \left| \frac{1}{\sqrt{2}} (\langle 0|1\rangle + \langle 1|1\rangle) \right|^2 = \frac{1}{2} \\
|\langle -|1\rangle|^2 &= \left| \frac{1}{\sqrt{2}} (\langle 0|1\rangle - \langle 1|1\rangle) \right|^2 = \frac{1}{2} \\
|\langle +|+\rangle|^2 &= \left| \frac{1}{2} (\langle 0|0\rangle + \langle 0|1\rangle + \langle 1|0\rangle + \langle 1|1\rangle) \right|^2 = 1 \\
|\langle -|+\rangle|^2 &= \left| \frac{1}{2} (\langle 0|0\rangle + \langle 0|1\rangle - \langle 1|0\rangle - \langle 1|1\rangle) \right|^2 = 0 \\
|\langle +|-\rangle|^2 &= \left| \frac{1}{2} (\langle 0|0\rangle - \langle 0|1\rangle + \langle 1|0\rangle - \langle 1|1\rangle) \right|^2 = 0 \\
|\langle -|-\rangle|^2 &= \left| \frac{1}{2} (\langle 0|0\rangle - \langle 0|1\rangle - \langle 1|0\rangle + \langle 1|1\rangle) \right|^2 = 1
\end{aligned}$$

(c)

$$\begin{aligned}
|\langle +i|0\rangle|^2 &= \left| \frac{1}{\sqrt{2}} (\langle 0|0\rangle - i \langle 1|0\rangle) \right|^2 = \frac{1}{2} \\
|\langle -i|0\rangle|^2 &= \left| \frac{1}{\sqrt{2}} (\langle 0|0\rangle + i \langle 1|0\rangle) \right|^2 = \frac{1}{2} \\
|\langle +i|1\rangle|^2 &= \left| \frac{1}{\sqrt{2}} (\langle 0|1\rangle - i \langle 1|1\rangle) \right|^2 = \frac{1}{2} \\
|\langle -i|1\rangle|^2 &= \left| \frac{1}{\sqrt{2}} (\langle 0|1\rangle + i \langle 1|1\rangle) \right|^2 = \frac{1}{2} \\
|\langle +i|+\rangle|^2 &= \left| \frac{1}{2} (\langle 0|0\rangle + \langle 0|1\rangle - i \langle 1|0\rangle - i \langle 1|1\rangle) \right|^2 = \left| \frac{1-i}{2} \right|^2 = \left(\frac{\sqrt{2}}{2} \right)^2 = \frac{1}{2} \\
|\langle -i|+\rangle|^2 &= \left| \frac{1}{2} (\langle 0|0\rangle + \langle 0|1\rangle + i \langle 1|0\rangle + i \langle 1|1\rangle) \right|^2 = \left| \frac{1+i}{2} \right|^2 = \left(\frac{\sqrt{2}}{2} \right)^2 = \frac{1}{2} \\
|\langle +i|-\rangle|^2 &= \left| \frac{1}{2} (\langle 0|0\rangle - \langle 0|1\rangle - i \langle 1|0\rangle + i \langle 1|1\rangle) \right|^2 = \left| \frac{1+i}{2} \right|^2 = \left(\frac{\sqrt{2}}{2} \right)^2 = \frac{1}{2} \\
|\langle -i|-\rangle|^2 &= \left| \frac{1}{2} (\langle 0|0\rangle - \langle 0|1\rangle + i \langle 1|0\rangle - i \langle 1|1\rangle) \right|^2 = \left| \frac{1-i}{2} \right|^2 = \left(\frac{\sqrt{2}}{2} \right)^2 = \frac{1}{2}
\end{aligned}$$

(d)

$$|\langle \varphi_0|0\rangle|^2 = \left| \cos\left(\frac{\pi}{8}\right) \langle 0|0\rangle + \sin\left(\frac{\pi}{8}\right) \langle 1|0\rangle \right|^2 = \cos^2\left(\frac{\pi}{8}\right) \approx 0.85355$$

$$|\langle \varphi_1|0\rangle|^2 = \left| \cos\left(\frac{5\pi}{8}\right) \langle 0|0\rangle + \sin\left(\frac{5\pi}{8}\right) \langle 1|0\rangle \right|^2 = \cos^2\left(\frac{5\pi}{8}\right) \approx 0.14645$$

$$|\langle \varphi_0|1\rangle|^2 = \left| \cos\left(\frac{\pi}{8}\right) \langle 0|1\rangle + \sin\left(\frac{\pi}{8}\right) \langle 1|1\rangle \right|^2 = \sin^2\left(\frac{\pi}{8}\right) \approx 0.14645$$

$$|\langle \varphi_1|1\rangle|^2 = \left| \cos\left(\frac{5\pi}{8}\right) \langle 0|1\rangle + \sin\left(\frac{5\pi}{8}\right) \langle 1|1\rangle \right|^2 = \sin^2\left(\frac{5\pi}{8}\right) \approx 0.85355$$

$$\begin{aligned} |\langle \varphi_0|+\rangle|^2 &= \left| \frac{1}{\sqrt{2}} \left(\cos\left(\frac{\pi}{8}\right) \langle 0|0\rangle + \cos\left(\frac{\pi}{8}\right) \langle 0|1\rangle + \sin\left(\frac{\pi}{8}\right) \langle 1|0\rangle + \sin\left(\frac{\pi}{8}\right) \langle 1|1\rangle \right) \right|^2 \\ &= \left| \frac{1}{\sqrt{2}} \left(\cos\left(\frac{\pi}{8}\right) + \sin\left(\frac{\pi}{8}\right) \right) \right|^2 = \left| \left(\cos\left(\frac{\pi}{4}\right) \cos\left(\frac{\pi}{8}\right) + \sin\left(\frac{\pi}{4}\right) \sin\left(\frac{\pi}{8}\right) \right) \right|^2 \\ &= \left| \frac{\cos\left(\frac{\pi}{8}\right) + \cos\left(\frac{3\pi}{8}\right)}{2} + \frac{\cos\left(\frac{\pi}{8}\right) - \cos\left(\frac{3\pi}{8}\right)}{2} \right|^2 = \cos^2\left(\frac{\pi}{8}\right) \approx 0.85355 \end{aligned}$$

$$\begin{aligned} |\langle \varphi_1|+\rangle|^2 &= \left| \frac{1}{\sqrt{2}} \left(\cos\left(\frac{5\pi}{8}\right) \langle 0|0\rangle + \cos\left(\frac{5\pi}{8}\right) \langle 0|1\rangle + \sin\left(\frac{5\pi}{8}\right) \langle 1|0\rangle + \sin\left(\frac{5\pi}{8}\right) \langle 1|1\rangle \right) \right|^2 \\ &= \left| \frac{1}{\sqrt{2}} \left(\cos\left(\frac{5\pi}{8}\right) + \sin\left(\frac{5\pi}{8}\right) \right) \right|^2 = \left| \left(\cos\left(\frac{\pi}{4}\right) \cos\left(\frac{5\pi}{8}\right) + \sin\left(\frac{\pi}{4}\right) \sin\left(\frac{5\pi}{8}\right) \right) \right|^2 \\ &= \left| \frac{\cos\left(-\frac{3\pi}{8}\right) + \cos\left(\frac{7\pi}{8}\right)}{2} + \frac{\cos\left(-\frac{3\pi}{8}\right) - \cos\left(\frac{7\pi}{8}\right)}{2} \right|^2 = \cos^2\left(-\frac{3\pi}{8}\right) \approx 0.14645 \end{aligned}$$

$$\begin{aligned} |\langle \varphi_0|-\rangle|^2 &= \left| \frac{1}{\sqrt{2}} \left(\cos\left(\frac{\pi}{8}\right) \langle 0|0\rangle - \cos\left(\frac{\pi}{8}\right) \langle 0|1\rangle + \sin\left(\frac{\pi}{8}\right) \langle 1|0\rangle - \sin\left(\frac{\pi}{8}\right) \langle 1|1\rangle \right) \right|^2 \\ &= \left| \frac{1}{\sqrt{2}} \left(\cos\left(\frac{\pi}{8}\right) - \sin\left(\frac{\pi}{8}\right) \right) \right|^2 = \left| \left(\cos\left(\frac{\pi}{4}\right) \cos\left(\frac{\pi}{8}\right) - \sin\left(\frac{\pi}{4}\right) \sin\left(\frac{\pi}{8}\right) \right) \right|^2 \\ &= \left| \frac{\cos\left(\frac{\pi}{8}\right) + \cos\left(\frac{3\pi}{8}\right)}{2} - \frac{\cos\left(\frac{\pi}{8}\right) - \cos\left(\frac{3\pi}{8}\right)}{2} \right|^2 = \cos^2\left(\frac{3\pi}{8}\right) \approx 0.14645 \end{aligned}$$

$$\begin{aligned} |\langle \varphi_1|-\rangle|^2 &= \left| \frac{1}{\sqrt{2}} \left(\cos\left(\frac{5\pi}{8}\right) \langle 0|0\rangle - \cos\left(\frac{5\pi}{8}\right) \langle 0|1\rangle + \sin\left(\frac{5\pi}{8}\right) \langle 1|0\rangle - \sin\left(\frac{5\pi}{8}\right) \langle 1|1\rangle \right) \right|^2 \\ &= \left| \frac{1}{\sqrt{2}} \left(\cos\left(\frac{5\pi}{8}\right) - \sin\left(\frac{5\pi}{8}\right) \right) \right|^2 = \left| \left(\cos\left(\frac{\pi}{4}\right) \cos\left(\frac{5\pi}{8}\right) - \sin\left(\frac{\pi}{4}\right) \sin\left(\frac{5\pi}{8}\right) \right) \right|^2 \\ &= \left| \frac{\cos\left(-\frac{3\pi}{8}\right) + \cos\left(\frac{7\pi}{8}\right)}{2} - \frac{\cos\left(-\frac{3\pi}{8}\right) - \cos\left(\frac{7\pi}{8}\right)}{2} \right|^2 = \cos^2\left(\frac{7\pi}{8}\right) \approx 0.85355 \end{aligned}$$

12.2.

(a) Alice's raw key is unaffected by the adversary's measurement. We therefore need to estimate the probability of error introduction in Bob's raw key. Since raw key is obtained after the sifting step, we only need to estimate the probability of a wrong outcome in case Bob measures in Alice's preparation basis. There are essentially 4 combinations each with probability $\frac{1}{4}$:

- (1) Alice sends $|0\rangle$ and Bob measures in the basis $\{|0\rangle, |1\rangle\}$.
- (2) Alice sends $|1\rangle$ and Bob measures in the basis $\{|0\rangle, |1\rangle\}$.

- (3) Alice sends $|+\rangle$ and Bob measures in the basis $\{|+\rangle, |-\rangle\}$.
- (4) Alice sends $|-\rangle$ and Bob measures in the basis $\{|+\rangle, |-\rangle\}$.

Since the adversary always measures in $\{|0\rangle, |1\rangle\}$ basis, the first two cases are trivial and no errors are introduced. In the latter two cases, however, after the adversary's interception Bob is measuring a state $|0\rangle$ or $|1\rangle$ instead, each with probability $\frac{1}{2}$.

Looking at the solution of exercise 11.1, we know that measuring both states $|0\rangle$ and $|1\rangle$ in the $\{|+\rangle, |-\rangle\}$ basis yields the same results — outcomes $|+\rangle$ and $|-\rangle$ each with probability $\frac{1}{2}$. This leads to a probability $\frac{1}{2}$ of error in Bob's raw key, whenever he measured $\{|+\rangle, |-\rangle\}$. Since according to the protocol Bob measures each basis with probability $\frac{1}{2}$, the total probability of an error in his raw key is $\frac{1}{4}$. Therefore, if the adversary performs the intercept-resend attack in this way, the expected amount of errors between raw keys of Alice and Bob of length n is $\frac{n}{4}$.

(b) Similarly to the previous case we need to examine the following four cases:

- (1) Alice sends $|0\rangle$ and Bob measures $\{|0\rangle, |1\rangle\}$.
- (2) Alice sends $|1\rangle$ and Bob measures $\{|0\rangle, |1\rangle\}$.
- (3) Alice sends $|+\rangle$ and Bob measures $\{|+\rangle, |-\rangle\}$.
- (4) Alice sends $|-\rangle$ and Bob measures $\{|+\rangle, |-\rangle\}$.

In the first case, with probability $\frac{1}{2}$, the adversary measures in $\{|0\rangle, |1\rangle\}$ basis and does not introduce any errors. Then, with probability $\frac{1}{2}$, they measure $\{|+\rangle, |-\rangle\}$ basis and (see solutions of 11.1) introduce an error with probability $\frac{1}{2}$. Altogether in case Alice sends $|0\rangle$ and Bob measures $\{|0\rangle, |1\rangle\}$ in $n_{|0\rangle}$ rounds, the expected number of errors is $\frac{n_{|0\rangle}}{4}$. It is easy to see that all other cases are symmetric. The adversary does not introduce any errors in case of using the correct basis, which happens with probability $\frac{1}{2}$, and introduces an error with probability $\frac{1}{2}$ otherwise. Putting everything together we can see that the overall probability of introducing an error is $\frac{1}{4}$. Thus in raw keys of size n the expected number of errors is $\frac{n}{4}$.

(c) Similarly to the previous cases we need to examine the following four cases:

- (1) Alice sends $|0\rangle$ and Bob measures $\{|0\rangle, |1\rangle\}$.
- (2) Alice sends $|1\rangle$ and Bob measures $\{|0\rangle, |1\rangle\}$.
- (3) Alice sends $|+\rangle$ and Bob measures $\{|+\rangle, |-\rangle\}$.
- (4) Alice sends $|-\rangle$ and Bob measures $\{|+\rangle, |-\rangle\}$.

Let us now calculate the probabilities of states reaching Bob, which can be obtained by using results of exercise 11.1 and can be given by:

- (1) Bob is measuring $|\varphi_0\rangle$ or $|\varphi_1\rangle$ with probabilities $p_0 \approx 0.85$ and $p_1 \approx 0.15$ respectively (in $\{|0\rangle, |1\rangle\}$ basis).
- (2) Bob is measuring $|\varphi_0\rangle$ or $|\varphi_1\rangle$ with probabilities $p_0 \approx 0.15$ and $p_1 \approx 0.85$ respectively (in $\{|0\rangle, |1\rangle\}$ basis).
- (3) Bob is measuring $|\varphi_0\rangle$ or $|\varphi_1\rangle$ with probabilities $p_0 \approx 0.85$ and $p_1 \approx 0.15$ respectively (in $\{|+\rangle, |-\rangle\}$ basis).
- (4) Bob is measuring $|\varphi_0\rangle$ or $|\varphi_1\rangle$ with probabilities $p_0 \approx 0.15$ and $p_1 \approx 0.85$ respectively (in $\{|+\rangle, |-\rangle\}$ basis).

Let us now examine the probabilities of Bob's measurement outcomes. We need to determine $|\langle x|\varphi_0\rangle|^2$ and $|\langle x|\varphi_1\rangle|^2$ for $x \in \{0, 1, +, -\}$. Here we can utilize the fact that $|\langle x|y\rangle|^2 = |\langle y|x\rangle|^2$ and utilize solutions of 11.1 again. Therefore for four above scenarios we have:

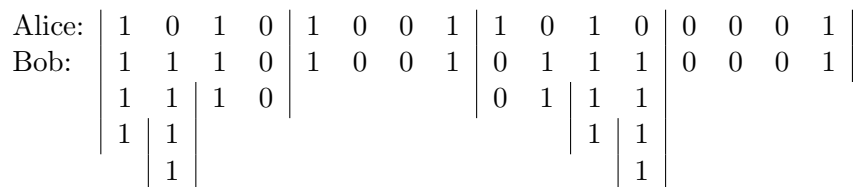
- (1) Bob gets outcome $|0\rangle$ with probability $\cos^2\left(\frac{\pi}{8}\right) |\langle 0|\varphi_0\rangle|^2 + \cos^2\left(\frac{5\pi}{8}\right) |\langle 0|\varphi_1\rangle|^2 = \cos^4\left(\frac{\pi}{8}\right) + \cos^4\left(\frac{5\pi}{8}\right) = 0.75$
- (2) Bob gets outcome $|1\rangle$ with probability $\cos^2\left(\frac{\pi}{8}\right) |\langle 1|\varphi_1\rangle|^2 + \cos^2\left(\frac{5\pi}{8}\right) |\langle 1|\varphi_0\rangle|^2 = \cos^4\left(\frac{\pi}{8}\right) + \cos^4\left(\frac{5\pi}{8}\right) = 0.75$
- (3) Bob gets outcome $|+\rangle$ with probability $\cos^2\left(\frac{\pi}{8}\right) |\langle +|\varphi_0\rangle|^2 + \cos^2\left(\frac{5\pi}{8}\right) |\langle +|\varphi_1\rangle|^2 = \cos^4\left(\frac{\pi}{8}\right) + \cos^4\left(\frac{5\pi}{8}\right) = 0.75$
- (4) Bob gets outcome $|-\rangle$ with probability $\cos^2\left(\frac{\pi}{8}\right) |\langle -|\varphi_1\rangle|^2 + \cos^2\left(\frac{5\pi}{8}\right) |\langle -|\varphi_0\rangle|^2 = \cos^4\left(\frac{\pi}{8}\right) + \cos^4\left(\frac{5\pi}{8}\right) = 0.75$

All other outcomes introduce an error, thus we can conclude that in for raw keys of length n we expect $\frac{n}{4}$ errors.

12.3.

- (a) Using the previous exercise it is easy to see that the probability of an error in Bob’s raw string is $\frac{p}{4}$.
- (b) The first step is to calculate how much information about Alice’s string the adversary learns when they use the attack. Since $|\langle \varphi_0|0\rangle|^2 = |\langle \varphi_0|+\rangle|^2 = |\langle \varphi_1|1\rangle|^2 = |\langle \varphi_1|-\rangle|^2 = \cos^2\left(\frac{\pi}{8}\right) \approx 0.85$, it is easy to see that if the adversary interprets their measurement outcome $|\varphi_0\rangle$ as $|0\rangle$ or $|+\rangle$ and $|\varphi_1\rangle$ as $|1\rangle$ or $|-\rangle$, based on the information gained during the basis reconciliation phase of the protocol, they guess each bit correctly with probability $\cos^2\left(\frac{\pi}{8}\right) \approx 0.85$. In the case they do not use the attack, they need to guess Alice’s bit, which has a probability of success $\frac{1}{2}$. Together, after performing the attack, the adversary knows on average $(0.85p + 0.5(1 - p))n = (0.5 + 0.35p)n$ bits of Alice’s n bit raw key.
- (c) We know that the post processing is possible if and only if Bob knows more about Alice’s string than the adversary. It is clear that with increasing p Bob’s information about Alice’s string decreases as $(1 - 0.25p)n$ and the adversary’s information increases as $(0.5 + 0.35p)n$. Therefore solving $(1 - 0.25p)n = (0.5 + 0.35p)n$ for p gives us a value of p for which Bob’s and the adversary’s information are equal. The solution is $p = \frac{5}{6}$. Any probability larger than this value means that no secret key can be post-processed from the raw keys. The critical amount of errors in Alice’s and Bob’s raw strings is therefore $(1 - \frac{1}{4} \cdot \frac{5}{6})n \approx 0.79n$. If they observe less or equal amount of errors, they should abort the protocol, if they know the adversary is performing the intercept resend attack.

12.4. Alice and Bob start the protocol by dividing the two string into blocks of size 4 and then comparing the parity of each corresponding block, performing binary search in every block with wrong parity. The first pass of the protocol is demonstrated in the diagram below.



Bob has detected two errors, on the 2nd and 12th position, and Bob’s corrected string is now 10101001 01100001.

For the next pass, Alice and Bob permute their strings into 00011100 1000111 and 00001100 101001111 respectively and double the block size to 8, this time however, every time Bob detects an error, he cascades back through the previous pass to correct the error that was previously masking it, this prevents having to disclose additional parity bits in the public channel.

$$\begin{array}{l} \text{Alice:} \\ \text{Bob:} \end{array} \left(\begin{array}{cccc|cccc} 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array} \right) \left(\begin{array}{cccc|cccc} 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array} \right)$$

Now since Bob detected another error, he must find the error that was masking it in the previous pass. This error was on the 9th position in the first pass. During the pass, Bob obtained the parity of the 9th and 10th bit together of Alice’s string, which was odd. With the now corrected 9th bit, he knows that the 10th bit also has an error and corrects it. His string is now 10101001 10100001 and the second block of size 8 now agrees in parity with Alice’s side. This concludes the protocol as increasing the block size further to the whole string wont achieve anything now, so Bob ends with the string 10101001 10100001.

12.5. In Step 2. of the protocol Alice calculates a codeword $c = xG = 101000110000100101100000$. Then in Step 3. she sends to Bob $c \oplus K_a = 000010101010100001010110$. In Step 4. Bob calculates $c \oplus e = 101010110000110101101000$ and finds e using algorithm of exercise 2.27. Calculation yields $e = 000010000000010000001000$ and $c = 101000110000100101100000$. In the last step Bob calculates $(c \oplus K_a) \oplus c = 000010101010100001010110 \oplus 101000110000100101100000$. This results to a string 101010011010000100110110, which is indeed equal to K_a .

12.6. Fix some $x, y \in U$ with $x \neq y$. Without loss of generality assume $x > y$. For any $a \in \{1, \dots, p - 1\}$ and $b \in \{0, \dots, p - 1\}$ let $r = (ax + b) \pmod p$ and $s = (ay + b) \pmod p$. Notice first that $r \neq s$ as otherwise we would have $ax + b \equiv ay + b \pmod p$ so $p|a(x - y)$ which is impossible as both a and $(x - y)$ are positive and less than p . Furthermore, for $r \neq s$ fixed there exist unique solutions for a, b in the desired range as

$$a \equiv (r - s)(x - y)^{-1} \pmod p$$

where x^{-1} here is the multiplicative inverse of $x \pmod p$, which exists as p is prime and $x \neq 0$. This clearly yields a unique solution for $a \in \{1, \dots, p - 1\}$, and so

$$b \equiv r - ax \pmod p$$

which yields a unique solution for $b \in \{0, \dots, p - 1\}$. Thus

$$h_{a,b}(x) = h_{a,b}(y) \iff r - s \equiv 0 \pmod n.$$

Therefore the number of collisions is the size of the set

$$|\{h \in H : h(x) = h(y)\}| = |\{(r, \in \{0, \dots, p - 1\}) : r \neq s \wedge r \equiv s \pmod n\}| \leq p \binom{p-1}{n} \leq \frac{p(p-1)}{n}$$

as for any fixed r there are at most $\binom{p-1}{n}$ values of $s \in \{0, \dots, p - 1\}$ with $r \neq s$ and $r \equiv s \pmod p$. As $|H| = p(p - 1)$ we have

$$\Pr_{h \in_r H} (h(x) = h(y)) = \frac{|\{h \in H : h(x) = h(y)\}|}{p(p - 1)} \leq \frac{p(p-1)}{p(p-1)} = \frac{1}{n}$$

so H is 2-universal, as claimed.

12.7.

1. The encryption using the quantum one-time pad protocol is done by calculating $|c\rangle = \sigma_x^k \sigma_z^{k'} |p\rangle$. So for our case we get

$$|c_1\rangle = \sigma_x^0 \sigma_z^1 \left(\frac{\sqrt{3}}{2} |0\rangle + \frac{1}{2} |1\rangle \right) = \frac{\sqrt{3}}{2} |0\rangle - \frac{1}{2} |1\rangle.$$

2. The decryption is just the inverse of the encryption $|p\rangle = \sigma_z^{k'} \sigma_x^k |c\rangle$, so we decrypt

$$|p_2\rangle = \sigma_z^1 \sigma_x^1 \left(\frac{-i}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle \right) = \frac{1}{\sqrt{2}} |0\rangle + \frac{i}{\sqrt{2}} |1\rangle.$$

12.8. Let $|p\rangle = \begin{pmatrix} a \\ b \end{pmatrix}$ with $|a|^2 + |b|^2 = 1$ be the qubit plaintext. Then the mixed state ρ created by the quantum one-time pad protocol with uniformly random secret key is

$$\begin{aligned} \rho &= \frac{1}{4} \sigma_x |p\rangle \langle p| \sigma_x + \frac{1}{4} \sigma_z |p\rangle \langle p| \sigma_z + \frac{1}{4} \sigma_x \sigma_z |p\rangle \langle p| \sigma_z \sigma_x + \frac{1}{4} |p\rangle \langle p| \\ &= \frac{1}{4} \left[\begin{pmatrix} b \\ a \end{pmatrix} (b^* \ a^*) + \begin{pmatrix} a \\ -b \end{pmatrix} (a^* \ -b^*) + \begin{pmatrix} -b \\ a \end{pmatrix} (-b^* \ a^*) + \begin{pmatrix} a \\ b \end{pmatrix} (a^* \ b^*) \right] \\ &= \frac{1}{4} \left[\begin{pmatrix} |b|^2 & a^* b \\ ab^* & |a|^2 \end{pmatrix} + \begin{pmatrix} |a|^2 & -ab^* \\ -a^* b & |b|^2 \end{pmatrix} + \begin{pmatrix} |b|^2 & -a^* b \\ -ab^* & |a|^2 \end{pmatrix} + \begin{pmatrix} |a|^2 & ab^* \\ a^* b & |b|^2 \end{pmatrix} \right] \\ &= \frac{1}{4} \begin{pmatrix} 2(|a|^2 + |b|^2) & 0 \\ 0 & 2(|a|^2 + |b|^2) \end{pmatrix} \\ &= \frac{1}{2} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \end{aligned}$$

Which is exactly the maximally mixed state.

12.9. To show this we will look at the resulting encrypted mixed state analogously to exercise 8 but this time we will look at two encrypted qubits at once while using the same pair of secret keys for both. Since the keys now aren't independent, the resulting two qubit state won't be just a simple tensor product of one qubit states calculated in exercise 8.

Let $|p_1\rangle |p_2\rangle = \begin{pmatrix} a \\ b \end{pmatrix} \otimes \begin{pmatrix} c \\ d \end{pmatrix}$ be the two qubit plaintext. Because we use the same key to encrypt both qubits we only have four (equally probable) possible encryptions of the qubit pair: $\sigma_x \otimes \sigma_x$, $\sigma_z \otimes \sigma_z$, $\sigma_x \sigma_z \otimes \sigma_x \sigma_z$ and identity.

The resulting encrypted two qubit state ρ is then

$$\begin{aligned}
\rho &= \frac{1}{4} \sigma_x \otimes \sigma_x |p_1\rangle |p_2\rangle \langle p_1| \langle p_2| \sigma_x \otimes \sigma_x + \frac{1}{4} \sigma_z \otimes \sigma_z |p_1\rangle |p_2\rangle \langle p_1| \langle p_2| \sigma_z \otimes \sigma_z \\
&\quad + \frac{1}{4} \sigma_x \otimes \sigma_x \sigma_z \otimes \sigma_z |p_1\rangle |p_2\rangle \langle p_1| \langle p_2| \sigma_z \otimes \sigma_z \sigma_x \otimes \sigma_x + \frac{1}{4} |p_1\rangle |p_2\rangle \langle p_1| \langle p_2| \\
&= \frac{1}{4} \left[\sigma_x |p_1\rangle \sigma_x |p_2\rangle \langle p_1| \sigma_x \langle p_2| \sigma_x + \sigma_z |p_1\rangle \sigma_z |p_2\rangle \langle p_1| \sigma_z \langle p_2| \sigma_z \right. \\
&\quad \left. + \sigma_x \sigma_z |p_1\rangle \sigma_x \sigma_z |p_2\rangle \langle p_1| \sigma_z \sigma_x \langle p_2| \sigma_z \sigma_x + |p_1\rangle |p_2\rangle \langle p_1| \langle p_2| \right] \\
&= \frac{1}{4} \left[\begin{pmatrix} b \\ a \end{pmatrix} \otimes \begin{pmatrix} d \\ c \end{pmatrix} (b^* \ a^*) \otimes (d^* \ c^*) \right. \\
&\quad + \begin{pmatrix} a \\ -b \end{pmatrix} \otimes \begin{pmatrix} c \\ -d \end{pmatrix} (a^* \ -b^*) \otimes (c^* \ -d^*) \\
&\quad + \begin{pmatrix} -b \\ a \end{pmatrix} \otimes \begin{pmatrix} -d \\ c \end{pmatrix} (-b^* \ a^*) \otimes (-d^* \ c^*) \\
&\quad \left. + \begin{pmatrix} a \\ b \end{pmatrix} \otimes \begin{pmatrix} c \\ d \end{pmatrix} (a^* \ b^*) \otimes (c^* \ d^*) \right] \\
&= \frac{1}{4} \begin{pmatrix} |bd|^2 & bb^*c^*d & a^*bdd^* & a^*bc^*d \\ bb^*cd^* & |bc|^2 & a^*bcd^* & a^*bcc^* \\ ab^*dd^* & ab^*c^*d & |ad|^2 & aa^*c^*d \\ ab^*cd^* & ab^*cc^* & aa^*cd^* & |ac|^2 \end{pmatrix} \\
&\quad + \frac{1}{4} \begin{pmatrix} |ac|^2 & -aa^*cd^* & -ab^*cc^* & ab^*cd^* \\ -aa^*c^*d & |ad|^2 & ab^*c^*d & -ab^*dd^* \\ -a^*bcc^* & a^*bcd^* & |bc|^2 & -bb^*cd^* \\ a^*bc^*d & -a^*bdd^* & -bb^*c^*d & |bd|^2 \end{pmatrix} \\
&\quad + \frac{1}{4} \begin{pmatrix} |bd|^2 & -bb^*c^*d & -a^*bdd^* & a^*bc^*d \\ -bb^*cd^* & |bc|^2 & a^*bcd^* & -a^*bcc^* \\ -ab^*dd^* & ab^*c^*d & |ad|^2 & -aa^*c^*d \\ ab^*cd^* & -ab^*cc^* & -aa^*cd^* & |ac|^2 \end{pmatrix} \\
&\quad + \frac{1}{4} \begin{pmatrix} |ac|^2 & aa^*cd^* & ab^*cc^* & ab^*cd^* \\ aa^*c^*d & |ad|^2 & ab^*c^*d & ab^*dd^* \\ a^*bcc^* & a^*bcd^* & |bc|^2 & bb^*cd^* \\ a^*bc^*d & a^*bdd^* & -bb^*c^*d & |bd|^2 \end{pmatrix} \\
&= \frac{1}{2} \begin{pmatrix} |ac|^2 + |bd|^2 & 0 & 0 & 2 \operatorname{Re}(a^*bc^*d) \\ 0 & |ad|^2 + |bc|^2 & 2 \operatorname{Re}(a^*bcd^*) & 0 \\ 0 & 2 \operatorname{Re}(ab^*c^*d) & |ad|^2 + |bc|^2 & 0 \\ 2 \operatorname{Re}(ab^*cd^*) & 0 & 0 & |ac|^2 + |bd|^2 \end{pmatrix}.
\end{aligned}$$

We can now see the resulting state ρ depends on the plaintext $|p_1\rangle |p_2\rangle$ and so the perfect secrecy of the quantum one-time pad is violated as an attacker might obtain information from the cryptotext.

12.10. Just like in the exercise 8 we will again look at the resulting mixed state, only this time we have just 2 possible encryptions. Let $|p\rangle = \begin{pmatrix} a \\ b \end{pmatrix}$ with $|a|^2 + |b|^2 = 1$ be the qubit plaintext. Then the mixed state ρ created by the modified quantum one-time pad protocol with uniformly random secret key is

(a)

$$\begin{aligned}
\rho &= \frac{1}{2} \sigma_x |p\rangle\langle p| \sigma_x + \frac{1}{2} \sigma_z |p\rangle\langle p| \sigma_z \\
&= \frac{1}{2} \left(\begin{pmatrix} |b|^2 & a^*b \\ ab^* & |a|^2 \end{pmatrix} + \begin{pmatrix} |a|^2 & -ab^* \\ -a^*b & |b|^2 \end{pmatrix} \right) \\
&= \frac{1}{2} \begin{pmatrix} 1 & 2\operatorname{Im}(a^*b)i \\ 2\operatorname{Im}(ab^*)i & 1 \end{pmatrix}.
\end{aligned}$$

We again see that this isn't independent of $|p\rangle$ and so information about the plaintext can be obtained from the cryptotext. To see if there is any state not encrypted by this new protocol we solve the equation

$$\frac{1}{2} \begin{pmatrix} 1 & 2\operatorname{Im}(a^*b)i \\ 2\operatorname{Im}(ab^*)i & 1 \end{pmatrix} = \begin{pmatrix} |a|^2 & ab^* \\ a^*b & |b|^2 \end{pmatrix},$$

Which gives us $|a| = |b| = \frac{1}{\sqrt{2}}$ but at the same time

$$\begin{aligned}
\frac{a^*b - ab^*}{2} &= ab^* \\
a^*b &= 3ab^* \\
ab^* &= 0,
\end{aligned}$$

which means there is no solution. This implies that every state is changed by the encryption function.

(b)

$$\begin{aligned}
\rho &= \frac{1}{2} \sigma_x |p\rangle\langle p| \sigma_x + \frac{1}{2} |p\rangle\langle p| = \frac{1}{2} \left(\begin{pmatrix} |b|^2 & a^*b \\ ab^* & |a|^2 \end{pmatrix} + \begin{pmatrix} |a|^2 & ab^* \\ a^*b & |b|^2 \end{pmatrix} \right) \\
&= \frac{1}{2} \begin{pmatrix} 1 & 2\operatorname{Re}(ab^*) \\ 2\operatorname{Re}(a^*b) & 1 \end{pmatrix}.
\end{aligned}$$

This is again not independent of $|p\rangle$. The states not encrypted by this modified protocol are exactly those states unchanged by the σ_x operator, i.e. its eigen vectors, which are $|+\rangle$ and $|-\rangle$.

Chapter 13

From Theory to Practice in Cryptography

13.1 Introduction

In this section we deal with several important issues of applied cryptography.

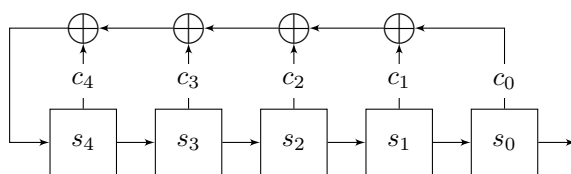
13.1.1 Linear-feedback shift registers

A linear-feedback shift register (LFSR) is a shift register whose input bit is a linear function of its previous state. Linear feedback shift registers are useful in applications such as pseudorandom number generation and are efficient to compute.

Linear feedback shift registers are an efficient way to realize recurrence relations of the type:

$$x_{n+m} = c_0x_n + c_1x_{n+1} + \dots + c_{m-1}x_{n+m-1} \pmod{n}$$

that can be specified by $2m$ bits: c_0, \dots, c_{m-1} and x_1, \dots, x_m . Typically they are represented by the following diagram:



The feedback XOR operations are called taps. LFSRs are initialized by n -bit state vector called seed. Then in each step the rightmost bit is produced as the output, new leftmost bit is calculated and the state vector is shifted to right. The overall output of the LFSR is a periodic bit sequence of period up to $2^n - 1$.

13.1.2 Confusion and diffusion

Two general methods to make cryptoanalysis harder are the following:

Confusion means that each bit of the ciphertext should depend on several parts of the key, obscuring the connections between the two. The property of confusion hides the relationship between the ciphertext and the key. This property makes it difficult to find the key from the ciphertext and if a single bit in a key is changed, most or all the bits in the ciphertext will be affected. Confusion increases the ambiguity of ciphertext and it is used by both block and stream ciphers.

Diffusion means that if we change a single bit of the plaintext, then (statistically) half of the bits in the ciphertext should change, and similarly, if we change one bit of the ciphertext, then approximately one half of the plaintext bits should change. Since a bit can have only two states, when they are all re-evaluated and changed from one seemingly random position to another, half

of the bits will have changed state. The idea of diffusion is to hide the relationship between the ciphertext and the plain text. This will make it hard for an attacker who tries to find out the plain text and it increases the redundancy of plain text by spreading it across the rows and columns; it is achieved through transposition of algorithms and it is used by block cipher only.

13.1.3 Feistel encryption/decryption scheme

This is a general scheme for construction of cryptosystems that was used as the design of several important cryptosystems, such as **DES**. Its main advantage is that encryption and decryption are very similar, and even identical in some cases.

Let F be a so-called **round function** and K_0, K_1, \dots, K_n be sub-keys for rounds $0, 1, 2, \dots, n$. **Encryption** is as follows:

- Split the plaintext into two equal size parts L_0, R_0 .
- For rounds $i \in \{0, 1, \dots, n\}$ compute

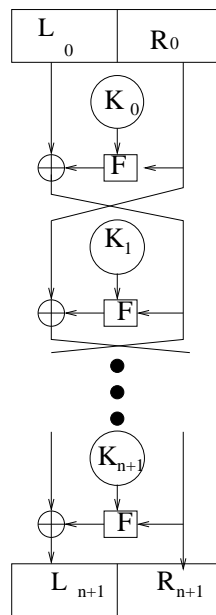
$$L_{i+1} = R_i; R_{i+1} = L_i \oplus F(R_i, K_i)$$

The ciphertext is then: (R_{n+1}, L_{n+1})

Decryption of (R_{n+1}, L_{n+1}) is done by computing, for $i = n, n - 1, \dots, 0$

$$R_i = L_{i+1}, L_i = R_{i+1} \oplus F(L_{i+1}, K_i)$$

and then (L_0, R_0) is the plaintext.



13.1.4 DES cryptosystem

Preprocessing: A secret 56-bit key k_{56} is chosen and a fixed/public permutation ϕ_{56} is applied to get $\phi_{56}(k_{56})$. The first (second) part of the resulting string are taken to get 28-bit blocks $C_0(D_0)$.

Using a fixed/public sequence s_1, \dots, s_{16} of integers, 16 pairs of 28-bit blocks $(C_i, D_i), i = 1, \dots, 16$ are then obtained as follows:

- $C_i(D_i)$ is obtained from $C_{i-1}(D_{i-1})$ by s_i left shifts with fixed/public s_i .

- Using a fixed and public order, a 48-bit block K_i is created from each pair C_i and D_i .

Encryption A fixed+public permutation ϕ_{64} is applied to a 64-bits long plaintext w to get $w' = L_0R_0$, where L_0 and R_0 has 32 bits. Then, 16 pairs of 32-bit blocks $L_i, R_i, 1 \leq i \leq 16$, are computed using the recurrence:

$$\begin{aligned} L_i &= R_{i-1} \\ R_i &= L_{i-1} \oplus f(R_{i-1}, K_i), \end{aligned}$$

where f is a fixed+public and easy-to-implement function.

The cryptotext is then $c = \phi_{64}^{-1}(L_{16}, R_{16})$

Decryption: $\phi_{64}(c) = L_{16}R_{16}$ is computed and then the recurrences

$$\begin{aligned} R_{i-1} &= L_i \\ L_{i-1} &= R_i \oplus f(L_i, K_i), \end{aligned}$$

are used to get $L_i, R_i, 15 \geq i \geq 0$ and finally $w = \phi_{64}^{-1}(L_0, R_0)$.

13.1.5 Operational modes of DES

To encode a sequence x_1, x_2, x_3, \dots of plaintext blocks,

ECB (Electronic Code Book) each x_i is encrypted with the same key.

CBC (Cipher Block Chaining) a c_0 is chosen and each x_i is encrypted to get the cryptotext $c_i = e_k(c_{i-1} \oplus x_i)$.

OFB (Output Feedback) a z_0 is chosen, $z_i = e_k(z_{i-1})$ are computed and each x_i is encrypted to get cryptotext $c_i = x_i \oplus z_i$.

CFB (Cipher Feedback) a c_0 is chosen and each x_i is encrypted to get cryptotext $c_i = x_i \oplus z_i$, where $z_i = e_k(c_{i-1})$.

13.1.6 AES cryptosystem

This is the newest broadly used cryptographic system.

Mathematics behind

- Some operations in AES are defined on **bytes**.
- Bytes are seen as elements of the finite field $GF(2^8)$.
- Bytes are represented either by binary 8-bit strings $b_7b_6b_5b_4b_3b_2b_1b_0$ or by polynomials

$$b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0$$

- Some operations of AES are defined in terms of **4-bytes words**.

Basic ideas and structures

- AES is a substitution-permutation network.
- Basic AES implementations operate on 4×4 matrices of bytes called **states**. A 128-bit message is also written as a 4×4 matrix of bytes.
- Some AES implementations work with states with additional columns in the state matrices.

- Encryption is performed through 10, 12 or 14 rounds depending on whether the key size is 128, 196 or 256 bits.
- Each round (except the final one) consists of four simple transformations:
 1. **SubBytes** - byte-wise substitution defined by a special table of 256 bytes.
 2. **ShiftRows** - circular shifts of i -th rows of the matrix by i positions to the left.
 3. **MixColumns** - a linear transformation on each column defined by a 4×4 matrix of bytes.
 4. **AddRoundKey** - bit-wise XOR with a round key defined by another matrix.

13.1.7 Hash functions

Another very important primitive for information processing that allows efficiently dealing with huge data sets and streams are **hash functions** and **hashes**, and especially **cryptographically secure hash functions** and **universal sets of hash functions**.

A good **cryptographic hash function** f is such a hash function that withstands all known cryptographic attacks. As a minimum, it must have the following properties:

Pre-image resistance: Given a hash h it should be unfeasible (difficult) to find a (message) m such that $h = f(m)$. In such a case it is also said that f should have **one-wayness property**.

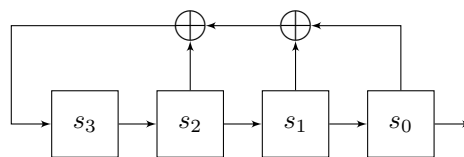
Second pre-image resistance: Given a message m_1 it should be unfeasible (difficult) to find another message m_2 such that $f(m_1) = f(m_2)$. In such a case it is also said that f should be **weakly collision resistant**.

Collision resistance: It should be unfeasible (difficult) to find two messages m_1 and m_2 such that $f(m_1) = f(m_2)$. In such a case it is also said that f should be **strongly collision resistant**.

13.2 Exercises

LFSR

13.1. Consider the following LFSR



- (a) Write down the recursive relation characterizing the register.
- (b) Find the period, generate all the internal states and the output bits if the initial state bits (s_3, s_2, s_1, s_0) are:
 - (1) 0001
 - (2) 0010
 - (3) 1111

13.2. Assume you captured the following ciphertext of one-time pad cryptosystem:

$$c = 110100010000000001111101011101100000$$

Later you found out that the first 10 bits of the plaintext are 1001001101 and the key is a pseudorandom sequence generated by a LSFR of size 5. The 35 bits of the plaintext encode 5 ASCII characters. Find the message.

Confusion/diffusion

13.3. Determine whether the following cryptosystems satisfy the confusion and diffusion properties:

- (a) One-time pad,
- (b) Vigenère cipher,
- (c) Hill,
- (d) DES in CBC operation mode,
- (e) DES in ECB operation mode.

Feistel/DES/(AES)

13.4. Is it possible to distinguish a Feistel cipher from a random function if it has

- (a) one round (with one chosen plaintext)?
- (b) two rounds (with two chosen plaintexts)?
- (c) three rounds (with two chosen plaintexts, access to a decryption oracle and one chosen ciphertext)?

13.5. Consider a two-round Feistel scheme with the round function being one-time pad with the respective keys being $K_0 = 110$ and $K_1 = 100$. Decrypt the cryptotext 001000.

13.6. Consider Alice and Bob using the DES cipher. An eavesdropper Eve has captured a ciphertext c sent by Alice to Bob. She also knows that the corresponding plaintext is m . Now, Eve forces Alice to encrypt the ciphertext c and, surprisingly, the resulting ciphertext c' equals to m . Is Eve able to read subsequent messages encrypted under the same DES key?

13.7. Consider the DES cipher. What happens to the ciphertext block if all bits in both the key and plaintext are inverted?

13.8. Use the result of the previous exercise and try to speed up the DES key exhaustive search in chosen-plaintext attack.

13.9. Consider triple, or 3-DES, cipher:

$$c = E_{k_3}(D_{k_2}(E_{k_1}(m)))$$

Discuss why is there a decryption used in the second operation.

13.10. DES-X is a variant of DES with two additional keys k_1 and k_2 where

$$c = k_2 \oplus E_k(m \oplus k_1).$$

The key size is increased to $56 + 2 \times 64 = 184$ bits. Show that the key size is effectively only about 120 bits.

13.11. Consider double DES, or 2-DES, cipher where two consecutive encryptions, resp. decryptions, with two different keys are performed:

$$c = E_{k_2}(E_{k_1}(m))$$

$$m = D_{k_1}(D_{k_2}(c)).$$

One would expect that brute-forcing of all possible values of k_1 and k_2 gives a total of $2^{|k_1|+|k_2|} = 2^{112}$ operations. Show that we can reduce the number of DES encryptions/decryptions to 2^{57} .

13.12. How is security affected if we omit the *SubBytes* part of AES?

Operation modes

13.13. Describe how the OFB mode can be attacked if the IV is not different for each execution of the encryption operation.

13.14. Suppose a bank is using encryption with ECB operation mode (without any additional message integrity check). Alice is sending 1024 USD to Eve. Eve has a partial knowledge about the structure of the message communicated to the bank. In particular, she knows that 1024 is encrypted in two blocks with inputs 10 and 24. Can Eve, who does not know Alice's password nor a way to break the encryption algorithm, alter the cryptotext to receive more money?

13.15. Alice and Bob use AES with CBC operation mode. Unfortunately Alice's encrypting device is broken and instead of the desired output $c_i = e_k(c_{i-1} \oplus x_i)$ it produced just $c_i = e_k(x_i)$ at the beginning of the communication. What will happen to the rest of the communication? Would a change to the ECB operation mode address this issue?

13.16. A programmer wants to use CBC in order to protect both the integrity and the confidentiality of network packets. She attaches a block of zero bits x_{n+1} to the end of the plaintext divided into n blocks $x_1 \dots x_n$ as redundancy, then encrypts with CBC. At the receiving end, she verifies that the added redundant bits are still all zero after CBC decryption. Does this test ensure the integrity of the transferred message?

Hash functions

13.17. Consider a cryptographic hash function with 32 bits long output. What is the approximate number of random inputs you have to try to find a collision with probability at least $3/4$?

13.3 Solutions

13.1.

(a) $x_{n+4} = x_{n+2} + x_{n+1} + x_n$

(b) Generate all internal states and output bits if the initial state bits are:

(1) The internal state sequence is

$$0001 \rightarrow 1000 \rightarrow 0100 \rightarrow 1010 \rightarrow 1101 \rightarrow 0110 \rightarrow 0011 \rightarrow 0001 \rightarrow \dots$$

The period is therefore 7. Linear feedback shift registers produce the last bit of their internal state as an output in each round, therefore the output of this LFSR is 1000101...

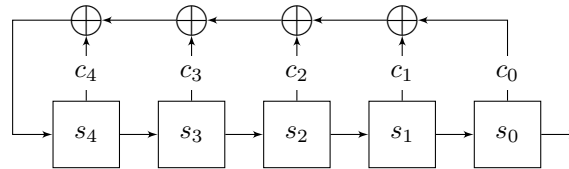
(2) The internal state sequence is

$$0010 \rightarrow 1001 \rightarrow 1100 \rightarrow 1110 \rightarrow 0111 \rightarrow 1011 \rightarrow 0101 \rightarrow 0010 \rightarrow \dots$$

The period is therefore 7 and the output is 0100111...

- (3) There is only single internal state 1111, therefore the period is 1 and the output string is a string containing 1 in each position.

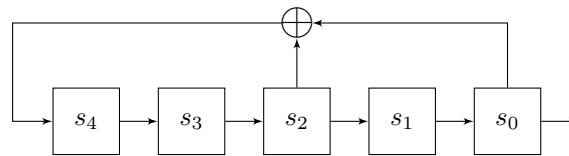
13.2. Using the first 10 bits of the plaintext, we can recover the first 10 bits of key as $1101000100 \oplus 1001001101 = 0100001001$. In order to fully reconstruct LFSR of size n it is enough to have $2n$ bits of its output. Therefore we have enough information to proceed. Let us first use the following variables, where c_i signifies, whether the corresponding i th bit is used for feedback:



Now we can calculate the following table of internal states history of the used LFSR, starting from the initial vector, which is simply the first 5 bits of the key in reverse:

states of cell s_4	states of cell s_3	states of cell s_2	states of cell s_1	states of cell s_0	implies
0	0	0	1	0	
$c_1 = 0$	0	0	0	1	$c_1 = 0$
$c_0 \oplus (c_4 \cdot c_1) = 1$	0	0	0	0	$c_0 = 1$
$c_4 \cdot c_0 = 0$	1	0	0	0	$c_4 = 0$
$c_3 \oplus c_4 \cdot (c_4 \cdot c_0) = 0$	0	1	0	0	$c_3 = 0$
$c_2 \oplus (c_4 \cdot c_3) = 1$	0	0	1	0	$c_2 = 1$

The LFSR used to generate the key sequence can be depicted as:



Having the LFSR and its initial state vector $(s_4, s_3, s_2, s_1, s_0) = (0, 0, 0, 1, 0)$, we can generate the whole key

$$k = 01000010010110011111100110111010100.$$

Now we can recover the plaintext as

$$m = c \oplus k = 10010011010110011000001101010110100.$$

Noticing the length of the message is 35 we will try dividing it into five 7-bit ASCII codes. We recover the message “IV054”.

13.3. If the confusion property holds one bit of the ciphertext should depend on several parts of the key. If the diffusion property holds then a change of a single bit in the plain text should lead to a change of statistically half bits of the cipher text.

- (a) One-time pad satisfies neither confusion nor diffusion property, since each bit is encrypted independently by an independent bit of the key.
- (b) Vigenère cipher satisfies neither confusion nor diffusion property. While substitutions add confusion properties when designing ciphers, a simple polyalphabetic substitution cipher, such as Vigenère, by itself does not satisfy the confusion property, since each output symbol depends only on one key symbol. Diffusion property is also not satisfied, since each plaintext symbol is encrypted independently.

- (c) For clarity of the argument, we will discuss Hill cryptosystem over \mathbb{F}_2 , with the plaintext being binary vector of length n and the key being a random $n \times n$ binary matrix. Confusion property holds, as the i -th bit of the ciphertext depends on all bits in the i -th row of the key matrix.

Diffusion property holds. Ciphertext bits are sum of elements in their corresponding rows. The element of each row is taken into the sum if and only if it is in a position, where the plaintext has value 1 (this is just the definition of matrix multiplication). Changing the plaintext value in i -th position thus changes which elements are taken into the final sum. Changing i -th plaintext bit from 1 to 0 means that the sum takes one less element. Since these elements are chosen in the key uniformly at random, on average half of the ciphertext bits flip their value. On the other hand changing from 0 to 1 adds one more element to each sum. Again, since we are adding a random bit, on average half the position of the ciphertext flip their value.

- (d) First note that single block encryption has both excellent confusion and diffusion properties by AES design. Put simply:
- The *add key* layer ensures the encryption function is only computable by someone who knows the key. This adds some confusion because the key is (psuedo) random
 - The *subBytes* s-box layer creates confusion as each symbol is mapped to another symbol in a way that impedes common methods of cryptanalysis (high resistance to linear and differential cryptanalysis)
 - The *shiftRows* and *mixColumns* operations combine to provide full diffusion over the course of 2 rounds. The state is a 4×4 grid of 8-bit words *mixColumns* operates vertically on each 4 word/32-bit column. One bit difference in any word in the input column will spread to multiple places in multiple words in the output column. *shiftRows* ensures that over the course of successive rounds different words are grouped up as inputs to the *mixColumn* function.

We are therefore interested in arguments about CBC mode of operation. Confusion: Since the key is reused for each block, trivially each block depends on several bits of the key and the confusion property holds. Diffusion: in the CBC operation mode each block depends on the previous block (or IV) as $c_{i+1} = e_k(c_i \oplus x_{i+1})$ hence a single change of a bit in the plaintext leads to statistically significant changes in the ciphertext.

- (e) AES in ECB operation mode has poor diffusion property since each block is encrypted independently. Blocks in ECB mode are encrypted independently, hence a single bit change in the plaintext affects only bits in its corresponding block of the cryptotext. The confusion property holds in the similar way as for AES with CBC.

13.4. Let $(L, R) = (L_0, R_0)$ be the plaintext and let $(S, T) = (L_{n+1}, R_{n+1})$ be the result of a Feistel scheme applied on input (L, R) .

1. Find whether $S = R$. If a Feistel scheme has only one round than this occur with 100% probability.
2. Choose two messages such that $R' = R$ and $L' \neq L$. Now check if $S \oplus S' = L \oplus L'$. This will occur with 100% probability if it is a Feistel scheme with 2 rounds and with probability approximately $\frac{1}{n}$ if it is a random permutation.
3. (a) Choose (L, R) and encrypt it to obtain (S, T) .
 (b) Choose $L' \neq L$, encrypt (L', R) to obtain (S', T') .
 (c) Ask for the decryption of $(S', T' \oplus L \oplus L')$ to obtain (L_d, R_d) . If $R_d = S' \oplus S \oplus R$, then a Feistel scheme has 3 rounds.

13.5. We start with $00100 = (R_2, L_2)$, where $R_2 = 001$ and $L_2 = 000$. Then the first round of the decryption is

$$R_1 = L_2 = 000, \quad L_1 = R_2 \oplus L_2 \oplus K_1 = 001 \oplus 000 \oplus 100 = 101.$$

The second round is

$$R_0 = L_1 = 101, \quad L_0 = R_1 \oplus L_1 \oplus K_0 = 000 \oplus 101 \oplus 110 = 011.$$

Our plaintext is then $(L_0, R_0) = 011101$.

13.6. One of the following four keys, called *weak keys*, were used:

C_0	D_0	64-bit key before parity drop (hexadecimal)	56-bit actual key
all zeroes	all zeroes	0101 0101 0101 0101	0000000 0000000
all zeroes	all ones	1F1F 1F1F 0E0E 0E0E	0000000 FFFFFFFF
all ones	all zeroes	E0E0 E0E0 1F1F 1F1F	FFFFFFFF 0000000
all ones	all ones	FEFE FEFE FEFE FEFE	FFFFFFFF FFFFFFFF

The round keys created from any of these weak keys are the same and have the same pattern as the cipher key. If we encrypt a block with a weak key and subsequently encrypt the result with the same weak key, we get the original block.

13.7. We obtain bitwise complement of the original ciphertext, ie.

$$DES_{\bar{k}}(\bar{m}) = \overline{DES_k(m)}.$$

This is the so called key complementation property.

Proof: It is clear that the initial permutation preserves the bitwise complement (so as the final permutation).

We can see that the key schedule preserves the bitwise complement as well. It is therefore sufficient to show that a single round preserves this property. Consider the application of the i -th round of DES:

$$DES_i(L_{i-1}||R_{i-1}, K_i) = L_i||R_i.$$

We want to show that $DES_i(\overline{L_{i-1}}||\overline{R_{i-1}}, \overline{K_i}) = \overline{L_i}||\overline{R_i}$.

Because $R_{i-1} = L_i$ we obtain $DES_i(\overline{L_{i-1}}||\overline{R_{i-1}}, \overline{K_i}) = \overline{L_i}||R'_i$. To show that $R'_i = \overline{R_i}$ we need to examine the round function f_i which is used to compute $R_i = L_{i-1} \oplus f_i(R_{i-1})$.

Again, it is easy to see that if the expansion permutation applied to R_{i-1} gives V_i then it gives $\overline{V_i}$ when applied to $\overline{R_{i-1}}$. After applying the XOR operation with round key, using the associativity and commutativity of the XOR function, we obtain

$$\overline{V_i} \oplus \overline{K_i} = (V_i \oplus 1) \oplus (K_i \oplus 1) = (V_i \oplus K_i) \oplus (1 \oplus 1) = V_i \oplus K_i.$$

Therefore, the input to the remaining part of the round function f is identical in both cases:

$$f_i(R_{i-1}, K_i) = f_i(\overline{R_{i-1}}, \overline{K_i}).$$

Now we can use the result above to obtain

$$\overline{L_{i-1}} \oplus f_i(\overline{R_{i-1}}, \overline{K_i}) = \overline{L_{i-1}} \oplus f_i(R_{i-1}, K_i) = 1 \oplus L_{i-1} \oplus f_i(R_{i-1}, K_i) = 1 \oplus R_i = \overline{R_i}.$$

We show that after one round, the output is inverted. By induction, it follows that the output is inverted after arbitrary number of rounds as well.

13.8. It suffices to try half of all 2^{56} DES keys.

- We ask for encryption of a plaintext p and its complement \bar{p} to obtain ciphertexts c_1 and c_2 , respectively.

- We try 2^{55} keys k with the most significant bit equal to 0.
- If $DES_k(p) = c_1$, then k is the key.
- If $\overline{DES_k(p)} = c_2$, then \bar{k} is the key.

13.9. This is due to backward compatibility with plain DES, *i.e.* letting $k_1 = k_2 = k_3$ we cancel out first two operations leaving a plain DES.

13.10. It is vulnerable to a sort of meet-in-the-middle attack since one does not really have to brute force both keys k_1 and k_2 .

$c = k_2 \oplus E_k(m \oplus k_1)$ is equivalent to $c \oplus k_2 = E_k(m \oplus k_1)$. We can proceed as follows:

1. Store 2^{64} plaintexts $k_1 \oplus m = m_i$ for each possible value of k_1 .
2. Store 2^{64} ciphertexts $c \oplus k_2$ for each possible value of k_2 .
3. Compute $E_k(m_i)$ for each value of k (*i.e.*, perform 2^{56} encryptions).
4. Compare the list in step 3 against the one in step 2.

13.11. We can mount a so-called meet-in-the-middle attack. We can rewrite the encryption formula as follows:

$$\begin{aligned}c &= E_{k_2}(E_{k_1}(m)) \\D_{k_2}(c) &= D_{k_2}(E_{k_2}(E_{k_1}(m))) \\D_{k_2}(c) &= E_{k_1}(m)\end{aligned}$$

Now we can precompute $E_{k_1}(m)$ for all values of k_1 and $D_{k_2}(c)$ for all possible values of k_2 . This is total of $2^{|k_1|} + 2^{|k_2|}$ operations. For 2-DES this is $2^{56+1} = 2^{57}$. Next we try to find a match between $E_{k_1}(m)$ and $D_{k_2}(c)$ lists. Such values k_1 and k_2 are possibly correct keys. We can verify their validity on another plaintext-ciphertext pair. We decrease the time while increasing the space usage, *i.e.* there is a trade-off between time and space.

13.12. The *SubBytes* operation adds non-linearity to AES and provides confusion in the cipher text. Without *SubBytes*, the whole cipher becomes affine.

Both linear transformations and addition of a constant are kinds of affine transformations, and since the composition of any two affine transformations is itself an affine transformation. It follows that, if the non-affine *SubBytes* step is removed from AES, the whole cipher becomes affine.

13.13. Assuming that the key remains the same, encrypting with the same IV will produce the exact same keystream as previous encryptions.

If no plaintext/ciphertext pairs are known, then there is no way to use this fact to attack the cipher. However, this can be used to mount a chosen plaintext attack. Choosing message m_1 to produce ciphertext c_1 , the attacker can calculate the keystream as $K = m_1 \otimes c_1$. The keystream can then be used to decrypt another message of the same length, because if the users do not change the IV, the keystreams are identical.

13.14. Since blocks are encrypted separately in the ECB operation mode, malicious Eve can exchange blocks encrypting 10 and 24 to receive 2410 USD.

13.15. Let's say that the error appeared in the i -th block. The rest of the message (including the i -th block) is corrupted since the error is propagated to the subsequent blocks, *i.e.* c_i is fed to $c_{i+1} = e_k(c_i \oplus x_{i+1})$. On the other hand, the ECB mode encrypts block of plaintext independently therefore error in i -th block does not affect subsequent blocks.

13.16. This method does not ensure the integrity of the transferred message. Let us denote the corresponding ciphertext blocks as c_1, \dots, c_n, c_{n+1} . From the decryption algorithm of the CBC mode we can see that each plaintext block x_i can be faithfully recovered whenever ciphertext blocks c_{i-1} and c_i are not corrupt. Therefore whenever error happens in blocks c_1, \dots, c_{n-1} , it will not be detected by this method.

13.17. We can use an approximation on the probability of finding collision in n elements after r random trials:

$$p(n, r) \approx 1 - \exp\left(-\frac{r^2}{2n}\right). \quad (13.1)$$

Thus, the approximated number of trials reads:

$$r \approx \sqrt{2n \ln \frac{1}{1 - p(n, r)}}. \quad (13.2)$$

There are 2^{32} possible outcomes of the hash function. Therefore, the number of random trials we have to perform is approximately

$$\sqrt{2 \times 2^{32} \ln \frac{1}{1 - 3/4}} \approx 109125.$$

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 modelled 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 (i.e., either X_n or Y_n).

2. Principles.

- Kerkhoffs principle: The security of a cryptosystem must not depend on keeping secret the encryption algorithm. The security should depend only on keeping secret the key.

3. Murphy 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:** $a \equiv b \pmod{n}$

Let $+_n, \times_n$ denote addition and multiplication modulo n , that is $a +_n b = (a + b) \bmod n$ and $a \times_n b = (ab) \bmod n$.

$\mathbb{Z}_n = \{0, 1, \dots, n-1\}$ is a group under the operation $+_n$. \mathbb{Z}_n^* is a field under the operations $+_n, \times_n$ if n is a prime.

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 integers k such that $a^k = 1$.
- 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 $p-1$ and it holds $a^{p-1} \equiv 1 \pmod{p}$.

A.4.3 Properties of the group \mathbb{Z}_n^*

Definition (1) For any group (G, \circ) and any $x \in G$

$$\text{order of } x = \min\{k > 0 \mid x^k = 1\}$$

(2) The group (G, \circ) is called cyclic if it contains an element g , called 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 multiplicative group $(\mathbb{Z}_n^*, \times_n)$ is cyclic iff n is either $1, 2, 4, p^k$ or $2p^k$ for some $k \in \mathbb{N}^+$ 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 *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

$$n \bmod m = \begin{cases} n - m \lfloor \frac{n}{m} \rfloor & n \neq 0 \\ 0 & m = 0 \end{cases}$$

Example: $7 \bmod 5 = 2$ $122 \bmod 11 = 1$

- Identities:**
- $(a + b) \bmod n = ((a \bmod n) + (b \bmod n)) \bmod n$
 - $(a \cdot b) \bmod n = ((a \bmod n) \cdot (b \bmod n)) \bmod n$
 - $a^b \bmod n = ((a \bmod n)^b) \bmod n$.

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: $a^n \bmod m = ((a \bmod m)^n) \bmod m$

Modular multiplication: $ab \bmod n = ((a \bmod n)(b \bmod n)) \bmod n$

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 $\text{gcd}(m, n), 0 \leq m < n$

EUCLID ALGORITHM

$$\text{gcd}(0, n) = n \tag{A.1}$$

$$\text{gcd}(m, n) = \text{gcd}(n \bmod m, m) \text{ for } m > 0 \tag{A.2}$$

Example $\text{gcd}(296, 555) = \text{gcd}(259, 296) = \text{gcd}(37, 259) = \text{gcd}(0, 37) = 37$.

Theorem $T(n) = \mathcal{O}(\log n)$ for the number of steps of Euclid's algorithm to compute $\text{gcd}(m, n)$ 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 $x = m^{-1} \pmod n$.

An extension of Euclid's algorithm, which computes x and y together with $\gcd(m, n)$ is sometimes referred to as **extended Euclid's 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 n. 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 n. 2 How (difficult is) 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 (CRT)

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, $0 < a_i < m_i, 1 \leq i \leq t$. 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

$$M = \prod_{i=1}^t m_i, M_i = \frac{M}{m_i}, N_i = M_i^{-1} \pmod{m_i}$$

and the solution (\star) is unique up to the congruence modulo M .

Comment: Each integer $0 < x < M$ is uniquely represented by t -tuple: $x \pmod{m_1}, \dots, x \pmod{m_t}$. **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, subtraction and multiplication can be done componentwise in parallel time.

A.7.3 Euler totient function

$$\Phi(n) = |Z_n^*| = |\{m | 1 \leq m \leq n, \gcd(m, n) = 1\}|$$

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)

$$n^{\Phi(m)} \equiv 1 \pmod{m}$$

if $n < m, \gcd(m, n) = 1$

Corollary $n^{-1} \equiv n^{\Phi(m)-1} \pmod{m}$ if $n < m, \gcd(m, n) = 1$

Theorem (Fermat's Little Theorem)

$$a^p \equiv a \pmod{p}$$

if p is prime.

A.7.5 Discrete logarithms and square roots

Three problems are related with the equation $y = x^a \pmod{n}$.

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. 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 the following special 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 known.

Examples

$$\begin{aligned}\{x \mid \sqrt{x} \pmod{15} = 1\} &= \{1, 4, 11, 14\} \\ \{x \mid \sqrt{x} \pmod{15} = 3\} &= \emptyset \\ \{x \mid \sqrt{x} \pmod{15} = 4\} &= \{2, 7, 8, 13\}\end{aligned}$$

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 roots if $k = 2$;
- 4 square roots 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 nonresidues

An integer $x \in \mathbf{Z}_m^*$ is called a quadratic residue modulo m if

$$x \equiv y^2 \pmod{m}$$

for some $y \in \mathbf{Z}_m^*$, otherwise x is a quadratic nonresidue.

Notation: QR_m denotes the set of all quadratic residues modulo m . QR_m is therefore a subgroup of squares in \mathbf{Z}_m .

QNR_m denotes the set of all quadratic nonresidues 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 \mathbf{Z}_p^* , then g^k is a quadratic residue iff k is even.

Theorem If p is a prime, then $a \in \mathbf{Z}_p^*$ 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 $f : QR(n) \rightarrow QR(n)$ defined by $f(x) = x^2$ is a permutation on $QR(n)$.
- The inverse function is $f^{-1}(x) = x^{((p-1)(q-1)+4)/8} \pmod n$

Bibliography

- [1] Ingemar Cox, Matthew Miller, Jeffrey Bloom, Jessica Fridrich, and Ton Kalker. *Digital Watermarking and Steganography*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2 edition, 2007.
- [2] Oded Goldreich. *Foundations of Cryptography: Basic Tools*. Cambridge University Press, New York, NY, USA, 2000.
- [3] 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.
- [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] Feng Hao and Piotr Zieliński. A 2-round anonymous veto protocol. In *the 14th International Workshop on Security Protocols*, 2006.
- [7] Raymond Hill. *A First Course in Coding Theory*. Oxford Applied Linguistics. Clarendon Press, 1986.
- [8] Zuzana Kuklová. Coding theory, cryptography and cryptographic protocols - exercises with solutions, 2007.
- [9] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, Inc., 1997.
- [10] Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomization and Probabilistic Techniques in Algorithms and Data Analysis*. Cambridge University Press, USA, 2nd edition, 2017.
- [11] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.
- [12] Vera Pless. *Introduction to the Theory of Error-correcting Codes*. A Wiley-Interscience publication. John Wiley, 1998.
- [13] Arto Salomaa. *Public-Key Cryptography*. Texts in Theoretical Computer Science. An EATCS Series. Springer Berlin Heidelberg, 1996.
- [14] Bruce Schneier. *Applied Cryptography*. John Wiley & Sons, Inc., 2 edition, 1996.
- [15] Douglas R. Stinson. *Cryptography Theory and Practice*. CRC Press, Inc., 1995.
- [16] Wade Trappe and Lawrence C. Washington. *Introduction to Cryptography with Coding Theory*. Pearson Educational international, Prentice Hall, 2006.

- [17] Serge Vaudenay. *A Classical Introduction to Cryptography: Applications for Communications Security*. Springer US, 2005.