Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет   
информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина «Методы защиты информации»

|  |  |
| --- | --- |
|  |  |

**ОТЧЕТ**

к лабораторной работе № 4

на тему «Асимметричная криптография. Алгоритм Мак-Элиса»

Выполнил             Е. А. Киселева

Проверил                           А. В. Герчик

Минск 2024

**СОДЕРЖАНИЕ**

[1 Постановка задачи 3](#_Toc157722973)

[2 Краткие теоретические сведения](#_Toc157722974) 4

[3 Результаты выполнения лабораторной работы 5](#_Toc157722975)

[Выводы](#_Toc157722976) 6

[Список использованных источников 7](#_Toc157722977)

[Приложение А (обязательное) Листинг исходного кода 8](#_Toc157722978)

# **1 ПОСТАНОВКА ЗАДАЧИ**

Целью выполнения данной лабораторной работы является изучение теоретических сведений и реализация криптостойкого программного средства шифрования и дешифрования текстовых файлов при помощи Криптосистемы Мак-Элиса.

# **КРАТКИЕ ТЕОРЕТИЧЕСКИЕ** **СВЕДЕНИЯ**

В системе Мак-Элиса параметрами системы, общими для всех абонентов, являются числа k, n, t. Для получения открытого и соответствующего секретного ключа каждому из абонентов системы следует осуществить следующие действия:

1 Выбрать порождающую матрицу G = Gkn двоичного (n,k)-линейного кода, исправляющего t ошибок, для которого известен эффективный алгоритм декодирования.

2 Случайно выбрать двоичную невырожденную матрицу S = Sk.

3 Случайно выбрать подстановочную матрицу P = Pn.

4 Вычислить произведение матриц G1 = S · G · P.

Открытым ключом является пара (G1, t), секретным – тройка (S, G, P).

Для того чтобы зашифровать сообщение M, предназначенное для абонента A, абоненту B следует выполнить следующие действия:

– представить M в виде двоичного вектора длины k;

– выбрать случайный бинарный вектор ошибок Z длиной n, содержащий не более t единиц;

– вычислить бинарный вектор C = M \* GA + Z и направить его абоненту A.

Получив сообщение C, абонент A вычисляет вектор C1 = C \* P-1, с помощью которого, используя алгоритм декодирования кода с порождающей матрицей G, получает далее векторы M1 и M = M1 \* S-1.

В качестве кода, исправляющего ошибки в системе Мак-Элиса, можно использовать код Гоппы. Известно, что для любого неприводимого полинома g(x) степени t над полем GF(2m) существует бинарный код Гоппы длины n = 2m и размерности k ≥ n – mt, исправляющий до t ошибок включительно, для которого имеется эффективный алгоритм декодирования.

Рекомендуемые параметры этой системы – n = 1024, t = 38, k > 644 – приводят к тому, что открытый ключ имеет размер около 219 бит, а длина сообщения увеличивается при шифровании примерно в 1,6 раза, в связи с чем данная система не получила широкого распространения.

# **3 РЕЗУЛЬТАТЫ ВЫПОЛНЕНИЯ ЛАБОРАТОРНОЙ             РАБОТЫ**

В ходе выполнения лабораторной было реализовано программное средство шифрования и дешифрования текстовых файлов при помощи криптосистемы Мак-Элиса.

Начальный текст находится в файле input.txt. После запуска программа шифрует исходный файл и зашифрованные данные заносит в фай output.txt. Сразу после этого программа расшифровывает данные из файла output.txt и заносит расшифрованный текст в файл decrypt.txt. Работа программы представлен на рисунке 3.1.

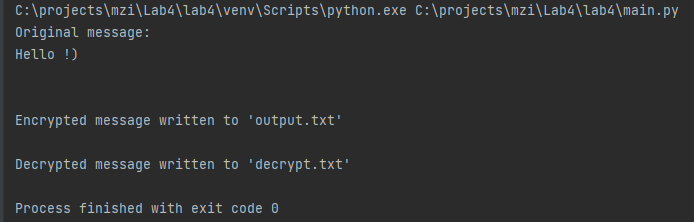


Рисунок 3.1 – Вывод консоли

Таким образом, в ходе данной лабораторной работы было реализовано программное средство шифрования и дешифрования текстовых файлов при помощи криптосистемы Мак-Элиса.

# **ВЫВОДЫ**

В ходе данной лабораторной работы были изучены теоретические сведения и реализовано криптостойкое программное средство шифрования и дешифрования текстовых файлов при помощи Криптосистемы Мак-Элиса.

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

[1] Криптосистема Мак-Элиса [Электронный ресурс]. – Режим доступа: https://cyberleninka.ru/article/n/kriptosistema-mak-elisa-i-problemy-eyo-vnedreni  
ya. – Дата доступа: 08.10.2024.

[2] Криптосистема Мак-Элиса в атаках декодирования классической информации [Электронный ресурс]. – Режим доступа: https://web.snauka.ru/issues/2020/06/92527. – Дата доступа: 09.10.2024.

## **ПРИЛОЖЕНИЕ А**

## **(обязательное)**

## **Листинг исходного кода**

Листинг 1 – Программный код файла main

import os

import numpy as np

import base64

input\_file\_path = "input.txt"

output\_file\_path = "output.txt"

decrypt\_file\_path = "decrypt.txt"

if not os.path.exists(input\_file\_path):

print(f"Input file '{input\_file\_path}' not found.")

else:

with open(input\_file\_path, "r", encoding="utf-8") as file:

original\_message = file.read()

print("Original message:\n" + original\_message)

def generate\_random\_matrix(rows, columns):

return np.random.randint(2, size=(rows, columns), dtype=np.uint8)

def encrypt(message, generator\_matrix):

message\_bytes = message.encode('utf-8')

encrypted\_bytes = bytearray(len(message\_bytes))

rows, columns = generator\_matrix.shape

for i in range(len(message\_bytes)):

encrypted\_bytes[i] = message\_bytes[i] ^ generator\_matrix[i % rows, i % columns]

return base64.b64encode(encrypted\_bytes).decode('utf-8')

def decrypt(encrypted\_message, generator\_matrix):

encrypted\_bytes = base64.b64decode(encrypted\_message)

decrypted\_bytes = bytearray(len(encrypted\_bytes))

rows, columns = generator\_matrix.shape

for i in range(len(encrypted\_bytes)):

decrypted\_bytes[i] = encrypted\_bytes[i] ^ generator\_matrix[i % rows, i % columns]

return decrypted\_bytes.decode('utf-8')

# Размеры матрицы

n = 512

k = 256

generator\_matrix = generate\_random\_matrix(k, n)

encrypted\_message = encrypt(original\_message, generator\_matrix)

with open(output\_file\_path, "w", encoding="utf-8") as file:

file.write(encrypted\_message)

print(f"\nEncrypted message written to '{output\_file\_path}'")

decrypted\_message = decrypt(encrypted\_message, generator\_matrix)

with open(decrypt\_file\_path, "w", encoding="utf-8") as file:

file.write(decrypted\_message)

print(f"\nDecrypted message written to '{decrypt\_file\_path}'")