Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»
КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Отчет по лабораторной работе №2.3
по дисциплине «Защита информации»

Студент Лукьяненко В.А.

Группа ИУ7-71Б

Преподаватель Руденкова Ю.С.

2025 г.

1 Задание

1.1 Цель работы

Цель работы: разработка алгоритма симметричного шифрования (DES). Шифрование и расшифровка архивных файлов.

1.2 Содержание работы

Для выполнения данной лабораторной работы необходимо решить следующие задачи:

1. реализовать программу шифрования симметричным алгоритмом DES;
2. обеспечить шифрование и расшифровку архивного файла (rar, zip или др.) с использованием разработанной программы;
3. провести тестирование программы на различных архивных файлах.

2 Теоретическая часть

Вопросы для защиты работы

1. Виды симметричного шифрования (поточные и блочные). Приведите схему для одного из видов.

Симметричное шифрование делится на два основных типа:

— **Поточные шифры** — данные шифруются последовательно, побитово или побайтово. Для генерации ключевой последовательности используется генератор псевдослучайной гаммы. Шифртекст получается как $C_i = M_i \oplus K_i$. Пример: RC4.

— **Блочные шифры** — открытый текст делится на блоки фиксированного размера (например, 64 бита в DES или 128 бит в AES). Каждый блок шифруется с использованием одного и того же ключа. Примеры: DES, AES.

Схема блочного шифра:

$$M_1, M_2, \ldots, M_n \xrightarrow{DES} C_1, C_2, \ldots, C_n$$

2. Опишите алгоритм шифрования DES.

Algorithm 1 Алгоритм шифрования DES

Require: Входной файл F_{in}, секретный ключ K
Ensure: Зашифрованный файл F_{enc}

Разбить F_{in} на блоки по 64 бита.
for каждый блок M_i **do**
    Выполнить начальную перестановку битов.
    Разделить блок на левую L и правую R части по 32 бита.
    for $i = 1$ **to** 16 **do**
        Вычислить $f(R_{i-1}, K_i)$, где K_i — подключ, полученный из ключа K.
        $L_i := R_{i-1}$
        $R_i := L_{i-1} \oplus f(R_{i-1}, K_i)$
    end for
    Объединить L_{16} и R_{16}.
    Выполнить финальную перестановку.
    Записать результат в F_{enc}.
end for
Сохранить F_{enc}.

3. Дайте определения алгоритмов перестановки и подстановки. Приведите примеры каждого из этих видов алгоритмов. Приведите пример алгоритма, использующего оба подхода.

— **Алгоритмы перестановки** — методы, при которых изменяется порядок символов или битов открытого текста без их замены. Пример: шифр маршрутной перестановки.

— **Алгоритмы подстановки** — методы, при которых каждый символ заменяется другим по определённому правилу. Пример: шифр Цезаря.

— **Алгоритмы, сочетающие оба подхода** — современные блочные шифры, использующие и перестановки, и подстановки для повышения стойкости. Пример: DES, AES.

3 Практическая часть.

Листинг 3.1 – Файл `main.py`,

```python
from Crypto.Cipher import DES
import os

def pad(data: bytes) -> bytes:
    padding_len = 8 - (len(data) % 8)
    return data + bytes([padding_len] * padding_len)

def unpad(data: bytes) -> bytes:
    padding_len = data[-1]
    return data[:-padding_len]

def encrypt_file(input_file: str, output_file: str, key: bytes):
    cipher = DES.new(key, DES.MODE_ECB)

    with open(input_file, "rb") as f:
        plaintext = f.read()

    padded_data = pad(plaintext)
    ciphertext = cipher.encrypt(padded_data)

    with open(output_file, "wb") as f:
        f.write(ciphertext)

def decrypt_file(input_file: str, output_file: str, key: bytes):
    cipher = DES.new(key, DES.MODE_ECB)

    with open(input_file, "rb") as f:
        ciphertext = f.read()

    decrypted_data = cipher.decrypt(ciphertext)
    unpadded_data = unpad(decrypted_data)

    with open(output_file, "wb") as f:
        f.write(unpadded_data)


if __name__ == "__main__":
    key = b"8bytekey"
```

```
39
40      encrypt_file("input.zip", "encrypted.bin", key)
41      decrypt_file("encrypted.bin", "output.zip", key)
42
43      print("Шифрование и расшифровка завершены.")
```

4 Пример работы программы

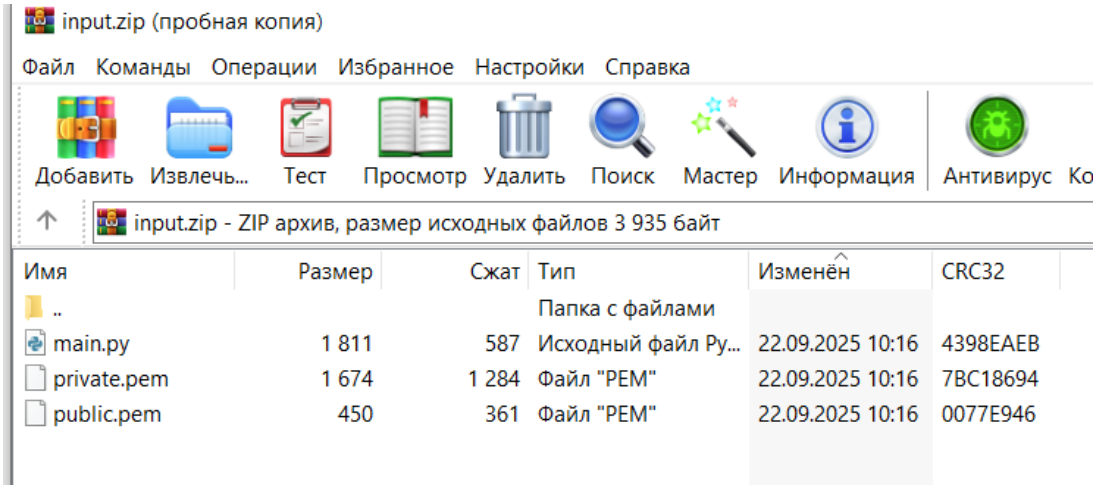

Рисунок 4.1 – Архив до шифрования

В результате шифрования получаются бинарный файл следующего содержания:

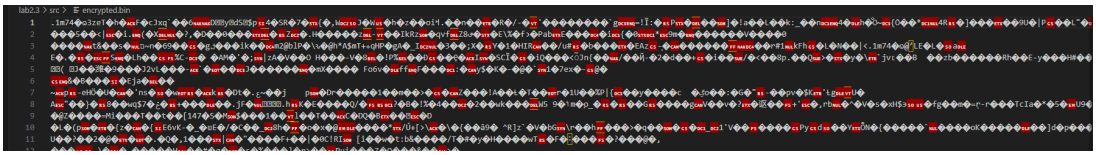

Рисунок 4.2 – Бинарный файл после шифрования

После, расшифруем бинарный файл:

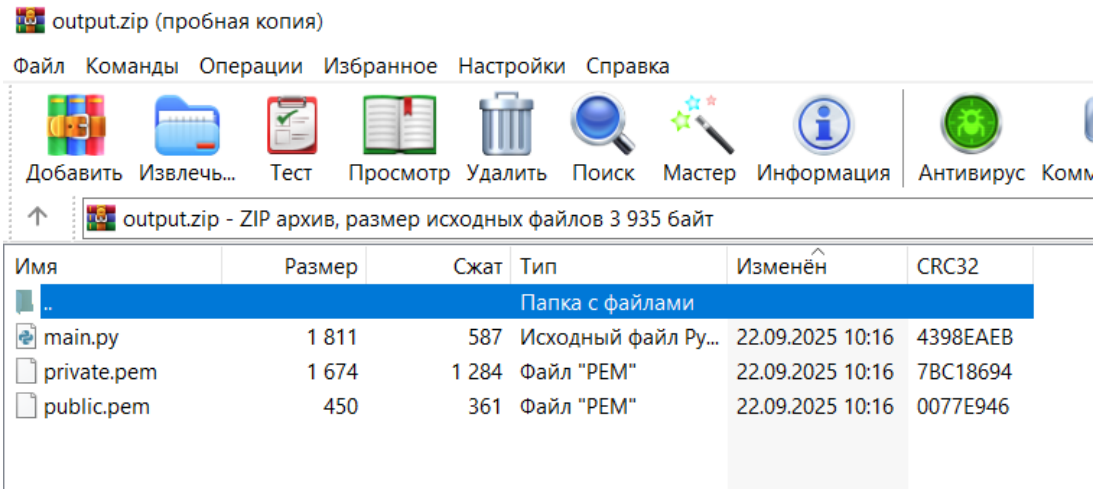

Рисунок 4.3 – Расшифрованный файл