



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 №2.2 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. Шифрование и расшифровка произвольного файла с использованием алгоритма DES.

1.2 Содержание работы

Для выполнения данной лабораторной работы необходимо решить следующие задачи:

1. реализовать программу шифрования алгоритмом DES;
2. обеспечить возможность шифрования и расшифровки произвольного файла;
3. предусмотреть работу программы с пустым и однобайтовым файлом;
4. протестировать работу программы на текстовых файлах различного содержания.

2 Теоретическая часть

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

Симметричное шифрование делится на два типа:

- **Поточные шифры** — шифруют данные побитово или побайтово. Шифртекст получается путём сложения (обычно по модулю 2) открытого текста и псевдослучайной гаммы, генерируемой на основе ключа. Пример: RC4.
- **Блочные шифры** — открытый текст делится на блоки фиксированного размера (например, 64 или 128 бит), и каждый блок шифруется с использованием одного и того же ключа. Примеры: DES, AES.

Схема блочного шифра:

$$\underbrace{M_1, M_2, \dots, M_n}_{\text{блоки открытого текста}} \xrightarrow{\text{Шифрование DES}} \underbrace{C_1, C_2, \dots, C_n}_{\text{блоки шифртекста}}$$

2. Опишите алгоритм шифрования DES.

Algorithm 1 Алгоритм шифрования DES

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

Ensure: Зашифрованный файл F_{enc}

- 1: Разбить F_{in} на блоки по 64 бита.
 - 2: **for** каждый блок M_i **do**
 - 3: Выполнить начальную перестановку битов.
 - 4: Разделить блок на левую L и правую R части по 32 бита.
 - 5: **for** $i = 1$ **to** 16 **do**
 - 6: Вычислить $f(R_{i-1}, K_i)$, где K_i — подключ, полученный из K .
 - 7: $L_i := R_{i-1}$
 - 8: $R_i := L_{i-1} \oplus f(R_{i-1}, K_i)$
 - 9: **end for**
 - 10: Объединить L_{16} и R_{16} .
 - 11: Выполнить финальную перестановку.
 - 12: Записать результат в выходной поток.
 - 13: **end for**
 - 14: 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 шифрования, при которых изменяется порядок символов или битов открытого текста, но сами символы остаются без изменений. *Пример:* шифр маршрутной перестановки.
- **Алгоритмы подстановки** — методы шифрования, при которых каждый символ или группа символов заменяется другим символом (или группой), согласно ключу. *Пример:* шифр Цезаря.
- **Алгоритмы, использующие оба подхода** — современные блочные шифры, сочетающие перестановки и подстановки для повышения криптостойкости. *Пример:* DES.

3 Практическая часть.

Листинг 3.1 – Файл main.py,

```
1 from Crypto.Cipher import DES
2 from Crypto.Random import get_random_bytes
3 import os
4
5 def pad(data: bytes) -> bytes:
6     while len(data) % 8 != 0:
7         data += b'␣'
8     return data
9
10 def encrypt_file(input_file: str, output_file: str, key: bytes):
11     cipher = DES.new(key, DES.MODE_ECB)
12
13     with open(input_file, 'rb') as f:
14         plaintext = f.read()
15
16     padded_data = pad(plaintext)
17     ciphertext = cipher.encrypt(padded_data)
18
19     with open(output_file, 'wb') as f:
20         f.write(ciphertext)
21
22 def decrypt_file(input_file: str, output_file: str, key: bytes):
23     cipher = DES.new(key, DES.MODE_ECB)
24
25     with open(input_file, 'rb') as f:
26         ciphertext = f.read()
27
28     decrypted_data = cipher.decrypt(ciphertext)
29     decrypted_data = decrypted_data.rstrip(b'␣')
30
31     with open(output_file, 'wb') as f:
32         f.write(decrypted_data)
33
34
35 if __name__ == "__main__":
36     key = b'8bytekey'
37
38     encrypt_file("input_rus.txt", "enc_rus.bin", key)
```

```
39 decrypt_file("enc_rus.bin", "dec_rus.txt", key)
40
41 encrypt_file("input_eng.txt", "enc_eng.bin", key)
42 decrypt_file("enc_eng.bin", "dec_eng.txt", key)
43
44 print("Шифрование и дешифрование завершено.")
```

4 Пример работы программы

```
lab2.2 > src > input_eng.txt
1 function function example encryption file number information world information result structure data computer number information model
```

Рисунок 4.1 – Файл до шифрования на английском (10000+ символов)

```
lab2.2 > src > input_rus.txt
1 ель информация программа система лабораторная слово текст алгоритм число проверка результат работа пример компьютер данные алгоритм
```

Рисунок 4.2 – Файл до шифрования на русском (10000+ символов)

В результате шифрования получаются бинарные файлы следующего содержания:

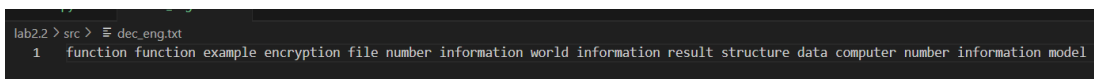


Рисунок 4.3 – Бинарный файл после шифрования на английском



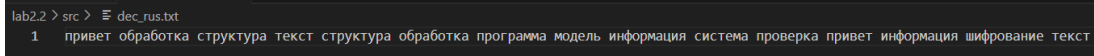
Рисунок 4.4 – Бинарный файл после шифрования на русском

После, расшифруем бинарные файлы:



```
lab2.2 > src > dec_eng.txt
1 function function example encryption file number information world information result structure data computer number information model
```

Рисунок 4.5 – Расшифрованный файл на английском



```
lab2.2 > src > dec_rus.txt
1 привет обработка структура текст структура обработка программа модель информация система проверка привет информация шифрование текст
```

Рисунок 4.6 – Расшифрованный файл на русском