# Отчёт по лабораторной работе №8

**Дисциплина: Основы информационной безопасности**

Мишина Анастасия Алексеевна

# Содержание

# Список иллюстраций

# Список таблиц

# 1 Цель работы

Освоить на практике применение режима однократного гаммирования на примере кодирования различных исходных текстов одним ключом [1].

# 2 Выполнение лабораторной работы

Рассмотрим файл main2.py, в котором написана наша программа (рис. 2.1). Создаем функцию encrypt(), которая будет шифровать заданный текст с помощью однократного гаммирования. Также есть возможность подавать на вход определенный ключ шифрования. Если ключа нет, то он генерируется рандомно. Сначала исходный текст и ключ шифрования преобразуются в 16-ую систему счисления, затем, применяется операция XOR для каждого элемента ключа и текста. Полученный шифротекст декодируется из 16-ой СС и получается набор из символов. В функции generate_key() происходит генерация ключа (если он не задан) из ascii-символов и цифр.

Работа программы: сначала создается случайный ключ и с этим ключом шифруются два текста p1 и p2 (в переменные c1 и c2). Далее шифротекст c1 шифруется по ключу c2. Полученный шифротекст c1_c2 шифруется по ключу открытого текста, в результате мы получаем второй открытый текст, ранее неизвестный.

```python
import string


def generate_key(length: int):
    '''
    Генерация рандомного ключа длины length
    '''
    return random.sample(string.ascii_letters + string.digits, length)


def encrypt(text: str, key: list = None):
    '''
    Выводит шифротекст для заданного текста.
    Если ключа нет, то генерируется случайный ключ
    '''
    if not key:
        key = generate_key(length=len(text))

    text_16 = [ord(char) for char in text]
    key = [ord(el) for el in key]

    print(f"Ключ шифрования:", ' '.join(str(s) for s in key))
    print(f"Исходный текст:", text)

    encrypted_text = []
    for i in range(len(text)):
        encrypted_text.append(text_16[i] ^ key[i])

    ciphertext = ''.join([chr(i) for i in encrypted_text])
    print(f'Шифротекст: {ciphertext}\n\n')

    return ciphertext


p1 = 'НаВашисходящийот1204'
p2 = 'ВСеверныйфилиалБанка'
key = generate_key(20)

c1 = encrypt(p1, key=key)
c2 = encrypt(p2, key=key)

c1_c2 = encrypt(c1, key=c2)

encrypt(c1_c2, p1)
encrypt(c1_c2, p2)
```

Рис. 2.1: Исходный код программы

Запустим файл main2.py и просмотрим результат работы программы (рис. 2.2).

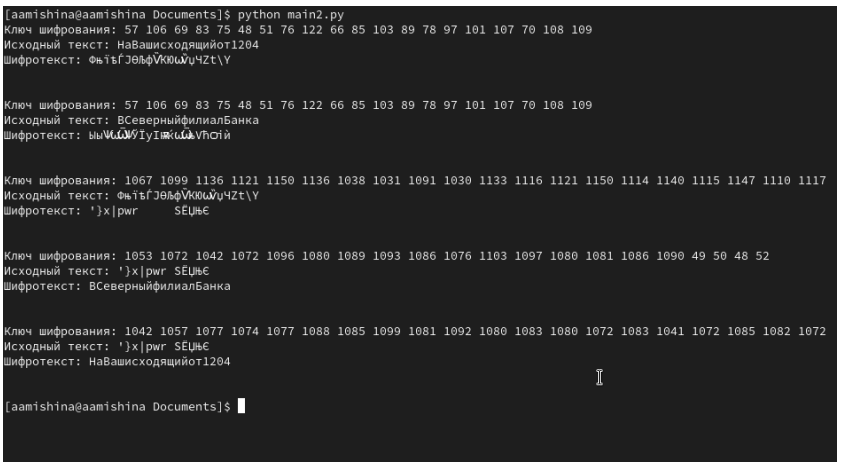


Рис. 2.2: Работа программы

# 3 Контрольные вопросы

1. Как, зная один из текстов (P1 или P2), определить другой, не зная при этом ключа?

Нужно применить XOR для двух шифротекстов, а к полученному результату применить XOR с ключом, равным известному открытому тексту. Тогда результатом будет второй открытый текст

2. Что будет при повторном использовании ключа при шифровании текста?

Шифрование будет небезопасным, т.к. с помощью шифротекстов и одного открытого текста можно дешифровать другой текст

3. Как реализуется режим шифрования однократного гаммирования одним ключом двух открытых текстов?

Каждый текст шифруется однократным гаммированием отдельно с использованием этого ключа

4. Перечислите недостатки шифрования одним ключом двух открытых текстов

Главный недостаток - можно дешифровать открытый текст без знания ключа

5. Перечислите преимущества шифрования одним ключом двух открытых текстов.

Т.к. ключей используется меньше, то тратится меньше памяти на хранение и передачу ключей

# 4 Выводы

В ходе выполнения данной лабораторной работы я освоила на практике применение режима однократного гаммирования на примере кодирования различных исходных текстов одним ключом.

# Список литературы

1. Кулябов Д.С., Королькова А.В., Геворкян М.Н. Информационная безопасность компьютерных сетей. Лабораторные работы, учебное пособие. Москва: РУДН, 2015. 64 с.