

# **Лабораторная работа №7**

**Элементы криптографии. Однократное гаммирование**

Парфенова Елизавета Евгеньевна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>7</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>8</b>
<b>5</b>	<b>Контрольные вопросы</b>	<b>11</b>
<b>6</b>	<b>Выводы</b>	<b>14</b>
	<b>Список литературы</b>	<b>15</b>

# Список иллюстраций

4.1	Результат работы программы . . . . .	10
-----	--------------------------------------	----

## **Список таблиц**

# **1 Цель работы**

Освоить на практике применение режима однократного гаммирования.

## 2 Задание

Нужно подобрать ключ, чтобы получить сообщение «С Новым Годом, друзья!». Требуется разработать приложение, позволяющее шифровать и дешифровать данные в режиме однократного гаммирования. Приложение должно:

1. Определить вид шифротекста при известном ключе и известном открытом тексте.
2. Определить ключ, с помощью которого шифротекст может быть преобразован в некоторый фрагмент текста, представляющий собой один из возможных вариантов прочтения открытого текста.

### 3 Теоретическое введение

**Криптография** — наука о методах обеспечения конфиденциальности, целостности данных, аутентификации, шифрования[1].

**Гаммирование**, или Шифр XOR, — метод симметричного шифрования, заключающийся в «наложении» последовательности, состоящей из случайных чисел, на открытый текст. Последовательность случайных чисел называется гамма-последовательностью и используется для зашифровывания и расшифровывания данных[2].

С точки зрения теории криптоанализа метод шифрования однократной случайной равновероятной гаммой (однократное гаммирование) той же длины, что и открытый текст, является невскрываемым[3]. Даже при раскрытии части последовательности гаммы нельзя получить информацию о всём скрываемом тексте.

## 4 Выполнение лабораторной работы

В ходе лабораторной работы было разработано требуемое приложение. Оно имеет следующие функции:

- **generate\_random\_key** - функция генерирует ключ шифрования на основе изначального текста. Ключ шифрования состоит из букв кириллицы, латиницы, цифр и различных символов;
- **xor** - функция, которая выполняет само гаммирование, накладывая гамму на скрываемый текст (по сути происходит сложение по модулю 2);
- **encrypt** - функция, шифрующая текст по сгенерированному ключу. Она совершает гаммирование и формирует сам шифротекст;
- **decrypt** - функция, которая, наоборот, расшифровывает шифротекст по определенному ключу. Работает аналогично encrypt, только формируя в конце открытый текст;
- **find\_key** - функция, которая ищет ключ по фрагменту изначального текста и шифротексту.

Вторая часть программы производит вызов всех созданных функций в определенной последовательности и совершает вывод необходимой информации: сгенерированного ключа, изначального текста, зашифрованного текста, расшифрованного текста, а также возможных вариантов прочтения шифротекста по фрагменту открытого текста. Причем, при расшифровке текста по фрагменту



с помощью найденного ключа (расшифрованный фрагмент всегда оставался неизменным (ни больше, ни меньше), какие бы манипуляции с кодом и выводом ни происходили) была добавлена зашифрованная часть этого же текста. Еще раз подчеркну, что поиск ключа на основе фрагмента шифротекста и на основе всего шифротекста не давал никаких различий, все равно расшифровывался только изначальный фрагмент, что характерно для однократного гаммирования.

Далее следует листинг описанной программы:

```
import random

def generate_random_key(text):
    possible_symbol = list(range(32, 127)) + list(range(1040, 1104))
    key_str = ''.join(chr(random.choice(possible_symbol)) for _ in range(len(text)))
    return key_str

def xor(text, key):
    return [ord(s1)^ord(s2) for s1,s2 in zip(text, key)]

def encrypt(text, key):
    chiphr = xor(text, key)
    chiphrotext = ''.join(chr(i) for i in chiphr)
    return chiphrotext

def decrypt(chiphro, key):
    decrypted = xor(chiphro, key)
    opentext = ''.join(chr(i) for i in decrypted)
    return opentext

def find_key(chiphrotext, text_fragment):
    chipher_fragment = chiphrotext[:len(text_fragment)]
```

```

key_f = xor(text_fragment, chipher_fragment)
found_key = ''.join(chr(i) for i in key_f)
return found_key

text = "С Новым Годом, друзья!"
text_fragment = "С Новым"

key = generate_random_key(text)
print("Созданный ключ: ", key)

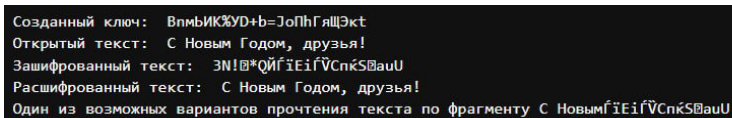
chiphrottext = encrypt(text, key)
print('Открытый текст: ', text)
print('Зашифрованный текст: ', chiphrottext)

opentext = decrypt(chiphrottext, key)
print('Расшифрованный текст: ', opentext)

found_key = find_key(chiphrottext, text_fragment)
open_fragtext = decrypt(chiphrottext[:len(text_fragment)], found_key)
print('Один из возможных вариантов прочтения текста по фрагменту', open_fragtext + chi

```

При запуске программы мы получили следующий вывод (рис. 4.1).



```

Созданный ключ: ВпмЫК%УD+b=JoPhГяЩЭкт
Открытый текст: С Новым Годом, друзья!
Зашифрованный текст: ЗН!@*Qйf'iEifVCпкSBauU
Расшифрованный текст: С Новым Годом, друзья!
Один из возможных вариантов прочтения текста по фрагменту С Новымf'iEifVCпкSBauU

```

Рис. 4.1: Результат работы программы

Здесь мы видим, что функции сработали так, как мы предполагали и текст парвильно зашифровался по сгенерированному ключу и расшифровался, а так- же правильно был найден ключ шифрования для фрагмента этого же текста и нормально произведена его расшифровка.

## 5 Контрольные вопросы

1. Поясните смысл однократного гаммирования.

Однократное гаммирование — это метод симметричного шифрования, который заключается в наложении случайной последовательности (гаммы) на открытый текст с целью получения шифротекста. Гамма должна быть такой же длины, как и открытый текст, и используется только один раз для каждого сообщения. Этот метод обеспечивает высокий уровень безопасности, так как при правильном использовании он является невскрываемым, что означает, что даже если часть шифротекста будет известна, это не даст информации о других частях

2. Перечислите недостатки однократного гаммирования.

- Необходимость в длинной гамме: Для каждого сообщения требуется новая гамма той же длины, что и открытый текст, что делает процесс обмена ключами сложным и неудобным.
- Сложность генерации случайных чисел: Генерация качественной гаммы требует использования аппаратных генераторов случайных чисел, что может быть затруднительно.
- Уязвимость при повторном использовании гаммы: Если гамма будет использована повторно, это приведет к возможности криптоанализа

3. Перечислите преимущества однократного гаммирования.

- Абсолютная стойкость: При соблюдении условий (гамма случайна и используется только один раз) шифр является абсолютно стойким согласно теории Шеннона.
- Простота реализации: Шифрование и дешифрование выполняются одной и той же операцией (XOR), что упрощает реализацию алгоритма.
- Отсутствие зависимости от структуры данных: Метод не зависит от частотного анализа текста, так как гамма полностью случайна

4. Почему длина открытого текста должна совпадать с длиной ключа?

Длина открытого текста должна совпадать с длиной ключа (гаммы), чтобы каждый бит открытого текста мог быть зашифрован соответствующим битом гаммы. Это обеспечивает полное наложение и делает невозможным восстановление информации без полного доступа к гамме

5. Какая операция используется в режиме однократного гаммирования, назовите её особенности?

В режиме однократного гаммирования используется операция исключающего ИЛИ (XOR). Эта операция имеет следующие особенности:

- Она является симметричной:
- Позволяет легко восстанавливать исходные данные при наличии ключа:

6. Как по открытому тексту и ключу получить шифротекст?

Шифротекст можно получить с помощью операции XOR между открытым текстом и ключом

7. Как по открытому тексту и шифротексту получить ключ?

Ключ можно восстановить по битам шифротекста и открытого текста с помощью XOR

8. В чем заключаются необходимые и достаточные условия абсолютной стойкости шифра?

Необходимые и достаточные условия для абсолютной стойкости шифра включают:

- Гамма должна быть случайной и независимой от открытого текста.
- Длина гаммы должна быть не меньше длины защищаемого сообщения.
- Гамма должна использоваться только один раз для каждого сообщения

## **6 Выводы**

В ходе лабораторной работы мы на практике освоили применение режима однократного гаммирования.

## Список литературы

1. Гаммирование [Электронный ресурс]. Wikimedia Foundation, Inc., 2023. URL: <https://ru.wikipedia.org/wiki/Гаммирование>.
2. Криптография [Электронный ресурс]. Wikimedia Foundation, Inc., 2024. URL: <https://ru.wikipedia.org/wiki/Криптография>.
3. Глава 7. Прикладные задачи шифрования [Электронный ресурс]. 2024. URL: <https://bugtraq.ru/library/books/crypto/chapter7/>.