

Лабораторная работа №7

Кондрашина Мария Сергеевна¹

18.10.2022, Moscow

¹RUDN University, Moscow, Russian Federation

Элементы криптографии.

Однократное гаммирование

Освоить на практике применение режима однократного гаммирования

Теоретические сведения

Гаммирование представляет собой наложение (снятие) на открытые (зашифрованные) данные последовательности элементов других данных, полученной с помощью некоторого криптографического алгоритма, для получения зашифрованных (открытых) данных. Иными словами, наложение гаммы — это сложение её элементов с элементами открытого (закрытого) текста по некоторому фиксированному модулю, значение которого представляет собой известную часть алгоритма шифрования.

Выполнение лабораторной работы

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

Приложение должно:

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

Работа с открытым текстом. Перевод в шестнадцатеричное представление

1) Определить вид шифротекста при известном ключе и известном открытом тексте.

```
Ввод [102]: my_str = "С Новым Годом, друзья!"  
            print("Открытый текст: ", my_str)
```

Открытый текст: С Новым Годом, друзья!

```
Ввод [103]: def to_16(strM):  
            str_16 = []  
            for i in strM:  
                str_16.append((i.encode('cp1251')).hex().upper())  
            return str_16
```

```
Ввод [104]: print("Открытый текст в шестнадцатеричной представлении: ", to_16(my_str))
```

Открытый текст в шестнадцатеричной представлении: ['D1', '20', 'CD', 'EE', 'E2', 'FB', 'EC', '20', 'C3', 'EE', 'E4', 'EE', 'E
C', '2C', '20', 'E4', 'F0', 'F3', 'E7', 'FC', 'FF', '21']

Создание ключа и шифрование текста

Ввод [4]: `import random`

```
key = [hex(random.randint(0,255))[2:].upper() for _ in range(len(my_str))]  
print("Ключ: ", key)
```

Ключ: ['85', 'A4', '40', 'AC', '12', '0', '98', '98', '9F', 'BF', '14', '4B', '3B', '4F', 'D1', 'E4', 'C4', '9B', '24', '9A', '8', '81']

Ввод [5]: `def cipher_text(strM, keyM):`

```
    c_text = []  
    for i,k in zip(strM, keyM):  
        c_text.append('{:02x}'.format(int(i,16) ^ int(k, 16))).upper()  
    return c_text
```

Ввод [6]: `mc_text = cipher_text(to_16(my_str), key)`

```
print("Зашифрованный текст в шестнадцатеричном представлении: ", mc_text)
```

Зашифрованный текст в шестнадцатеричном представлении: ['54', '84', '8D', '42', 'F0', 'FB', '74', 'B8', '5C', '51', 'F0', 'A5', 'D7', '63', 'F1', '00', '34', '68', 'C3', '66', 'F7', 'A0']

Ввод [7]: `new_text = [(bytes.fromhex(e1)).decode('cp1251') for e1 in mc_text]`

```
print("Зашифрованный текст: ", new_text)
```

Зашифрованный текст: ['Т', ' ', 'К', 'В', 'р', 'ы', 'т', 'ё', '\\', 'Q', 'р', 'Г', 'Ч', 'с', 'с', '\\x00', '4', 'h', 'Г', 'f', 'ч', '\\xa0']

Определение ключа по отrypted и зашифрованному тексту

2) Определить ключ, с помощью которого шифротекст может быть преобразован в некоторый фрагмент текста, представляющий собой один из возможных вариантов прочтения открытого текста.

```
Ввод [109]: print("Открытый текст: ", my_str)
            print("Зашифрованный текст: ", new_text)
```

Открытый текст: С Новым Годом, друзья!

Зашифрованный текст: ['E', 'A', '>', 'A', 'M', 'P', '9', 'H', 'P', '\x02', 'X', 'p', 'Й', ':', 'E', '{', 'f', '.', 'h', 'A', 'z', 'y']

```
Ввод [110]: def find_key(str_16, c_text_16):
            key_new = []
            for i,j in zip(str_16, c_text_16):
                key_new.append('{:02x}'.format(int(i,16) ^ int(j, 16))).upper()
            return key_new
```

```
Ввод [111]: print("Найденный ключ: ", find_key(to_16(my_str), to_16(new_text)))
```

Найденный ключ: ['94', 'B3', 'F3', '2E', '5B', '2B', 'D5', 'A4', '93', 'EC', '31', '1E', '25', '16', '98', '9F', '55', '44', '79', '3C', '38', 'D2']

```
Ввод [115]: if (find_key(to_16(my_str), to_16(new_text)) == key):
            print("Ключ совпал с ранее созданным")
        else:
            print("Ключи различны")
```

Ключ совпал с ранее созданным

- Выполнила лабораторную работу №7.
- Освоила на практике применение режима однократного гаммирования

1. Методические материалы курса. “Информационная безопасность компьютерных сетей” Кулябов Д. С., Королькова А. В., Геворкян М. Н.