Лабораторная работа №7

Василий А. Селезнев - студент группы НКНбд-01-18

10.12.2021

Элементы криптографии.

Однократное гаммирование

Цель выполнения лабораторной работы

- Освоить на практике применение режима однократного гаммирования

Задачи выполнения работы

- Написать программу, которая должна определить вид шифротекста при известном ключе и известном открытом тексте
- Также эта программа должна определить ключ, с помощью которого шифротекст может быть преобразован в некоторый фрагмент текста, представляющий собой один из возможных вариантов прочтения открытого текста

Результаты выполнения лабораторной работы

- Написал программу, которая определяет вид шифротекста при известном ключе и известном открытом тексте (рис - @fig:001, рис - @fig:002)

```
In [1]: import numpy as np

In [2]: def encryption(text):
            print("Открытый текст: ", text)
            #задаем массив из символов открытого текста в шестнадцатеричном представлении:
            text_array = []
            for i in text:
                text_array.append(i.encode("cp1251").hex())
            print("\nОткрытый текст в шестнадцатеричном представлении: ", *text_array)

            #задаем случайно сгенерированный ключ в шестнадцатеричном представлении:
            key_dec = np.random.randint(0,255,len(text))
            key_hex = [hex(i)[2:] for i in key_dec]
            print("\nКлюч в шестнадцатеричном представлении: ", *key_hex)

            #задаем зашифрованный текст в шестнадцатеричном представлении:
            crypt_text = []
            for i in range(len(text_array)):
                crypt_text.append("{:02x}".format(int(text_array[i], 16)^ int(key_hex[i], 16)))
            print("\nЗащифрованный текст в шестнадцатеричном представлении: ", *crypt_text)

            #задаем зашифрованный текст в обычном представлении:
            final_text = bytearray.fromhex("".join(crypt_text)).decode("cp1251")
            print("\nЗашифрованный текст: ", final_text)
            return key_hex, final_text
```

Рис. 1: Функция, шифрующая данные

```
In [4]: #Изначальная фраза:
        phrase = "С новым годом, друзья!"
        #получение сгенерированного ключа и зашифрованной фразы:
        crypt_key, crypt_text = encryption(phrase)
```

```
Открытый текст:  С новым годом, друзья!

Открытый текст в шестнадцатеричном представлении:  d1 20 ed ee e2 fb ec 20 e3 ee e4 ee ec 2c 20 e4 f0 f3 e7 fc ff 2
1

Ключ в шестнадцатеричном представлении:  33 bd bd eb 7d ab 2d 75 e8 f4 4f 26 37 22 b1 9f ea 75 96 67 e8 c8

Защифрованный текст в шестнадцатеричном представлении:  e2 9d 50 05 9f 50 c1 55 0b 1a ab c8 db 0e 91 7b 1a 86 71 9b
17 e9

Зашифрованный текст:  вќРџРБU«ИЫ‘{†q›й
```

Рис. 2: Результат работы функции, шифрующей данные

- Написанная мною программа определяет ключ, с помощью которого шифротекст может быть преобразован в некоторый фрагмент текста, представляющий собой один из возможных вариантов прочтения открытого текста (рис - @fig:003, рис - @fig:004)

In [3]:
```
def decryption(text, final_text):
    print("Открытый текст: ", text)
    print("Зашифрованный текст: ", final_text)

    #задаем массив из символов открытого текста в шестнадцатеричном представлении:
    text_hex = []
    for i in text:
        text_hex.append(i.encode("cp1251").hex())
    print("Открытый текст в шестнадцатеричном представлении: ", *text_hex)

    #задаем массив из символов зашифрованного текста в шп:
    final_text_hex = []
    for i in final_text:
        final_text_hex.append(i.encode("cp1251").hex())
    print("Зашифрованный текст в шестнадцатеричном представлении: ", *final_text_hex)

    #найдем ключ
    key = [hex(int(i,16)^int(j,16))[2:] for (i,j) in zip(text_hex, final_text_hex)]
    print("Найденный ключ в шестнадцатеричном представлении: ", *key)
    return key
```

Рис. 3: Функция, дешифрующая данные

```
In [5]: #получение нужного ключа:
        key=decryption(phrase,crypt_text)

        Открытый текст:  С новым годом, друзья!
        Зашифрованный текст:  вќРµРБU«ИЫ‘{†q›й
        Открытый текст в шестнадцатеричном представлении:  d1 20 ed ee e2 fb ec 20 e3 ee e4 ee ec 2c 20 e4 f0 f3 e7 fc ff 2
        1
        Зашифрованный текст в шестнадцатеричном представлении:  e2 9d 50 05 9f 50 c1 55 0b 1a ab c8 db 0e 91 7b 1a 86 71 9b
        17 e9
        Найденный ключ в шестнадцатеричном представлении:  33 bd bd eb 7d ab 2d 75 e8 f4 4f 26 37 22 b1 9f ea 75 96 67 e8 c
        8
```

Рис. 4: Результат работы функции, дешифрующей данные

In [6]:
```
#проверка правильности ключа:
print("ключ верен!") if crypt_key == key else print("ключ неверен!")
```

```
ключ верен!
```

Рис. 5: Сравнение ключей

Таким образом, я освоил на практике применение режима однократного гаммирования.