Отчет по лабораторной работе №7

Элементы криптографии. Однократное гаммирование

Евсеева Дарья Олеговна

22 октября, 2022

Содержание

Список иллюстраций

Цель работы

Целью данной работы является освоение на практике применения режима однократного гаммирования.

Задание

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.

Теоретическое введение

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.

В соответствии с теорией криптоанализа, если в методе шифрования используется однократная вероятностная гамма (однократное гаммирование) той же длины, что и подлежащий сокрытию текст, то текст нельзя раскрыть. Даже при раскрытии части последовательности гаммы нельзя получить информацию о всём скрываемом тексте.

Выполнение лабораторной работы

Выполнять работу будем в среде Jupyter Notebook на языке Python.

Для начала импортируем необходимую библиотеку и определим вспомогательные функции для дальнейшей работы.

```python
import numpy as np
```

```python
def check_enc(array):
    arr = [i if len(i)==2 else "0"+i for i in array]
    return arr
```

```python
def generate_key(n):
    k = np.random.randint(0, 255, n)
    key = [hex(i)[2:] for i in k]
    key = check_enc(key)
    return key
```

```python
def encode_str(string):
    s = []
    for i in string:
        s.append(i.encode('cp1251').hex())
    return s
```

Рис. 1: Вспомогательные функции

Далее определим основные функции. Функция cipher_define будет использоваться для определения вида шифротекста при известном ключе и известном открытом тексте, а функция key_define — для определения ключа, с помощью которого шифротекст может быть преобразован в некоторый фрагмент текста, представляющий собой вариант прочтения открытого текста.

```python
def cipher_define(text, key):
    t = encode_str(text)
    c = []
    for i in range(len(t)):
        c.append(hex(int(t[i], 16) ^ int(key[i], 16))[2:])
    c = check_enc(c)
    cipher = bytearray.fromhex("".join(c)).decode('cp1251')
    return cipher
```

```python
def key_define(text, cipher):
    t = encode_str(text)
    c = encode_str(cipher)
    k = []
    for i in range(len(t)):
        k.append(hex(int(t[i], 16) ^ int(c[i], 16))[2:])
    key = check_enc(k)
    return key
```

Рис. 2: Основные функции

Далее проверим работу определенных ранее функций и убедимся в правильности результатов.

```python
initial_str = "С Новым Годом, друзья!"
```

```python
key1 = generate_key(len(initial_str))
print(key1)
```

```
['76', '3a', '37', 'd5', 'ed', 'e7', '09', '5a', 'eb', '81', '35', '9b', '14', 'a2', 'd8', '36', '95', '4e', '61',
'f8', '00', '4d']
```

```python
cipher1 = cipher_define(initial_str, key1)
print(cipher1)
```

```
§ъ;ez(oCшшЋшTeS†яl
```

```python
key2 = key_define(initial_str, cipher1)
print(key2)
```

```
['76', '3a', '37', 'd5', 'ed', 'e7', '09', '5a', 'eb', '81', '35', '9b', '14', 'a2', 'd8', '36', '95', '4e', '61',
'f8', '00', '4d']
```

Рис. 3: Проверка работы

Как мы видим, все функции успешно работают.

Выводы

В результате проделанной работы мы освоили на практике применение режима однократного гаммирования.

Список литературы

- Методические материалы к лабораторной работе, представленные на сайте “ТУИС РУДН” https://esystem.rudn.ru/