Отчет по лабораторной работе №7

Информационная безопасноть

Коломиец Мария Владимировна НПИбд-01-18

Содержание

Список иллюстраций

1 Цель работы

Освоить на практике применение режима однократного гаммирования.

2 Теоретическое описание

Предложенная Г. С. Вернамом так называемая «схема однократного использования (гаммирования)» является простой, но надёжной схемой шифрования данных. *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. Наложение гаммы по сути представляет собой выполнение операции сложения по модулю 2 (XOR) (обозначаемая знаком $\oplus$) между элементами гаммы и элементами подлежащего сокрытию текста. Напомним, как работает операция XOR над битами: $0 \oplus 0 = 0, 0 \oplus 1 = 1, 1 \oplus 0 = 1, 1 \oplus 1 = 0$. Такой метод шифрования является симметричным, так как двойное прибавление одной и той же величины по модулю 2 восстанавливает исходноезначение, а шифрование и расшифрование выполняется одной и той же программой. Если известны ключ и открытый текст, то задача нахождения шифротекста заключается в применении к каждому

символу открытого текста следующего правила:

$$C_i = P_i \oplus K_i$$

где C_i — i-й символ получившегося зашифрованного послания, P_i — i-й символ открытого текста, K_i — i-й символ ключа, i = 1, m. Размерности открытого текста и ключа должны совпадать, и полученный шифротекст будет такой же длины. Если известны шифротекст и открытый текст, то задача нахождения ключа решается также, а именно, обе части равенства необходимо сложить по модулю 2 с P_i:

$$C_i \oplus P_i = P_i \oplus K_i \oplus P_i = K_i, K_i = C_i \oplus P_i.$$

Открытый текст имеет символьный вид, а ключ — шестнадцатеричное представление. Ключ также можно представить в символьном виде, воспользовавшись таблицей ASCII-кодов. К. Шеннон доказал абсолютную стойкость шифра в случае, когда однократно используемый ключ, длиной, равной длине исходного сообщения, является фрагментом истинно случайной двоичной последовательности с равномерным законом распределения. Криптоалгоритм не даёт никакой информации об открытом тексте: при известном зашифрованном сообщении C все различные ключевые последовательности K возможны и равновероятны, а значит, возможны и любые сообщения P. Необходимые и достаточные условия абсолютной стойкости шифра: – полная случайность ключа; – равенство длин ключа и открытого текста; – однократное использование ключа.

3 Выполнение лабораторной работы

Требуется разработать приложение, позволяющее шифровать и дешифровать данные в режиме однократного гаммирования.

1. Написана функция *to_hex*, трансформирующая текст в шестнадцатиричное представление (рис. 3.1).

```
In [1]: def to_hex (text):
            hexa=[]
            for i in text:
                hexa.append(hex(ord(i))[2:])
            return hexa
```

Рис. 3.1: Код функции *to_hex*

2. Написана функция *encryption*, которая с помощью однократного гаммирования из сообщения и ключа получает шифротекст (рис. 3.2).

```
In [2]: def encryption1 (message, key):
            cypher=[]
            cypher_1=[]
            for i, j in zip(message, key):
                c=hex(int(i,16)^int(j,16))[2:]
                c=(c,'0'+c)[len(c)==1]
                cypher.append(c)
                cypher_1.append(chr(int(i,16)^int(j,16)))
            return cypher, cypher_1
```

Рис. 3.2: Код функции *encryption*

3. Написана функция *gen_key*, генерирующая случайный ключ (рис. 3.2).

```
In [3]: from random import randrange

def gen_key (lenght):
    key=[]
    for _ in range(lenght):
        temp=randrange(256)
        temp=hex(temp)[2:]
        key.append((temp,'0'+temp)[len(temp)==1])
    return ' '.join(key)
#print(gen_key(22))
```

Рис. 3.3: Код функции *gen_key*

4. Определяю вид шифротекста при известном ключе и известном открытом тексте. Применяю к шифротексту ключ снова, чтобы получить исходное сообщение (рис. 3.4).

```
In [4]: message='Лабораторная работа №7, Коломиец Мария Владимировна'
#key='01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01'
key=gen_key(len(message))
#key = ' '.join(to_hex('КоломиецМарияВладимировна'))

print('Применение ключа к исходному сообщению.\nСообщение:\t\t\t %s \nКлюч:\t\t\t\t %s' %(message, key))
key_m=key.split()
message_hex = to_hex(message)

cypher_hex, cypher=encryption1(message_hex, key_m)
cypher=''.join(cypher)
cypher_hex=' '.join(cypher_hex)
#print('Зашифрованное сообщение:\t %s' %cypher)
print('Зашифрованное сообщение:\t %s' %cypher_hex)

print('\n\nПрименение ключа к зашифрованному сообщению.\nЗашифрованное сообщение:\t %s \nКлюч:\t\t\t\t %s' %(cypher_
mess_hex, mess=encryption1(cypher_hex.split(), key_m)
mess=''.join(mess)
print('Расшифрованное сообщение:\t %s' %mess)

Применение ключа к исходному сообщению.
Сообщение:                      Лабораторная работа №7, Коломиец Мария Владимировна
Ключ:                           ac 4f 3e 15 3d 0f 18 0f e1 23 22 bb ab 1d c9 00 a4 7d 7d cc ab 25 1d 29 67 bd 47 4
6 79 f6 f3 1f 5a 18 c0 4a a4 33 23 22 81 42 f0 90 87 24 b4 bd 39 66 48
Зашифрованное сообщение:        4b7 47f 40f 42b 47d 43f 45a 431 4a1 41e 412 4f4 8b 45d 4f9 431 49a 43f 44d ec 21bd
12 31 09 47d 483 47c 478 445 4ce 4c6 459 7a 404 4f0 40a 49c 47c 03 430 4ba 472 4c4 4a8 4bb 41c 4f4 483 40b 45b 478


Применение ключа к зашифрованному сообщению.
Зашифрованное сообщение:        4b7 47f 40f 42b 47d 43f 45a 431 4a1 41e 412 4f4 8b 45d 4f9 431 49a 43f 44d ec 21bd
12 31 09 47d 483 47c 478 445 4ce 4c6 459 7a 404 4f0 40a 49c 47c 03 430 4ba 472 4c4 4a8 4bb 41c 4f4 483 40b 45b 478
Ключ:                           ac 4f 3e 15 3d 0f 18 0f e1 23 22 bb ab 1d c9 00 a4 7d 7d cc ab 25 1d 29 67 bd 47 4
6 79 f6 f3 1f 5a 18 c0 4a a4 33 23 22 81 42 f0 90 87 24 b4 bd 39 66 48
Расшифрованное сообщение:       Лабораторная работа №7, Коломиец Мария Владимировна
```

Рис. 3.4: Получение шифротекста

5. Определяю ключ, с помощью которого шифротекст может быть преобразован в некоторый фрагмент текста, представляющий собой один из возможных вариантов прочтения открытого текста (Нужно подобрать ключ, чтобы получить сообщение «С Новым Годом, друзья!»)(рис. 3.5).

In [5]:
```
test='С Новым годом, друзья!'
new_key_hex, new_key = encryption1(cypher_hex.split(), to_hex(test))
```

In [6]:
```
test_key=' '.join(new_key_hex)
test_key=test_key.split()
print('Подбор ключа.\nЗашифрованное сообщение:\t %s \nТестовый ключ:\t\t\t %s' %(cypher_hex, key))
mess_hex, mess=encryption1(cypher_hex.split(), test_key)
mess=''.join(mess)
print('Возможное сообщение:\t\t %s' %mess)
```

```
Подбор ключа.
Зашифрованное сообщение:         4b7 47f 40f 42b 47d 43f 45a 431 4a1 41e 412 4f4 8b 45d 4f9 431 49a 43f 44d ec 21bd
12 31 09 47d 483 47c 478 445 4ce 4c6 459 7a 404 4f0 40a 49c 47c 03 430 4ba 472 4c4 4a8 4bb 41c 4f4 483 40b 45b 478
Тестовый ключ:                   ac 4f 3e 15 3d 0f 18 0f e1 23 22 bb ab 1d c9 00 a4 7d 7d cc ab 25 1d 29 67 bd 47 4
6 79 f6 f3 1f 5a 18 c0 4a a4 33 23 22 81 42 f0 90 87 24 b4 bd 39 66 48
Возможное сообщение:             С Новым годом, друзья!
```

Рис. 3.5: Один из вариантов прочтения шифротекста

4 Выводы

На основе проделанной работы освоила на практике применение режима однократного гаммирования.

5 Контрольные вопросы

1. Поясните смысл однократного гаммирования. Смысл однократного гаммирования состоит в том, что каждый символ попарно с символом ключа складываются по модулю.

2. Перечислите недостатки однократного гаммирования. Недостатками является то, что ключ нельзя переиспользовать, а также размер ключа должен быть равен размеру текста.

3. Перечислите преимущества однократного гаммирования. Основными преимуществами являются симметричность и криптостойкость.

4. Почему длина открытого текста должна совпадать с длиной ключа? Каждый символ открытого текста должен попарно складываться с символом ключа.

5. Какая операция используется в режиме однократного гаммирования, назовите её особенности? В режиме однократного гаммирования используется сложение по модулю 2: при сложении чисел с другим получается исходное. Например, 0+0 = 0, 0+1=1, 1+0=1, 1+1=0. Если в методе шифрования используется однократная вероятностная гамма той же длины, что и подлежащий сокрытию текст, то текст нельзя раскрыть.

6. Как по открытому тексту и ключу получить шифротекст? Для этого необходимо сложить попарно символы текста с ключом по модулю 2.

7. Как по открытому тексту и шифротексту получить ключ? Для этого необходимо сложить попарно по модулю 2 символы открытого текста с символами

шифротекста.

8. В чём заключаются необходимые и достаточные условия абсолютной стойкости шифра? Необходимые и достаточные условия абсолютной стойкости шифра заключаются в полной случайности ключа; равенстве длин ключа и открытого текста; использовании ключа однократно.