Отчет по лабораторной работе №8

Информационная безопасноть

Коломиец Мария Владимировна НПИбд-01-18

Содержание

Список иллюстраций

1 Цель работы

Освоить на практике применение режима однократного гаммирования на примере кодирования различных исходных текстов одним ключом.

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

Два текста кодируются одним ключом (однократное гаммирование). Требуется, не зная ключа и не стремясь его определить, прочитать оба текста.

1. Создаем алфавит из русских букв и цифр. Задаем входные данные из условия лабораторной работы. (рис. 3.1).

```
a = ord("a")
alphabeth = [chr(i) for i in range(a, a + 32)]
a = ord("0")
for i in range(a, a+10):
    alphabeth.append(chr(i))

a = ord("А")
for i in range(1040, 1072):
    alphabeth.append(chr(i))
print(alphabeth)
P1 = "НаВашисходящийот1204"
P2 = "ВСеверныйфилиалБанка"

key = "05 0C 17 7F 0E 4E 37 D2 94 10 09 2E 22 57 FF C8 0B B2 70 54"
```

Рис. 3.1: Создание алфавита

2. Функция “vzlom”, которая получив два открытых сообщения и объединив их получает гамму. (рис. 3.2).

```
def vzlom(P1, P2):
    code = []
    for i in range(20):
        code.append(alphabeth[(alphabeth.index(P1[i]) + alphabeth.index(P2[i]))  % len(alphabeth)])
    print(code)
    print(code[16], " и ", code[19])
    p3 = "".join(code)
    print(p3)

vzlom(P1, P2)
```

```
['а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и', 'й', 'к', 'л', 'м', 'н', 'о', 'п', 'р', 'с', 'т', 'у', 'ф', 'х', 'ц',
'ч', 'ш', 'щ', 'ъ', 'ы', 'ь', 'э', 'ю', 'я', '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'А', 'Б', 'В', 'Г',
'Д', 'Е', 'Ж', 'З', 'И', 'Й', 'К', 'Л', 'М', 'Н', 'О', 'П', 'Р', 'С', 'Т', 'У', 'Ф', 'Х', 'Ц', 'Ч', 'Ш', 'Щ', 'Ъ',
'Ы', 'Ь', 'Э', 'Ю', 'Я']
['щ', 'С', '3', 'в', 'э', 'ш', 'ю', 'Ж', 'ч', 'ш', '7', '4', 'р', 'й', 'щ', 'У', '1', 'Е', 'А', '4']
1  и  4
цС3вэшюЖчш74рйщУ1ЕА4
```

Рис. 3.2: Функция “vzlom”

3. Функция “shifr”, которая получает исходные сообщения (рис. 3.3).

```
def shifr(P1):
    # создаем алфавит
    dicts = {"а": 1, "б": 2, "в": 3, "г": 4, "д": 5, "е": 6, "ё": 7, "ж": 8, "з": 9, "и": 10, "й": 11, "к": 12, "л":
             "м": 14, "н": 15, "о": 16, "п": 17,
             "р": 18, "с": 19, "т": 20, "у": 21, "ф": 22, "х": 23, "ц": 24, "ч": 25, "ш": 26, "щ": 27, "ъ": 28,
             "ы": 29, "ь": 30, "э": 31, "ю": 32, "я": 32, "А":33 , "Б": 34, "В": 35 , "Г":36 , "Д":37 , "Е":38 , "Ё"
              "И":42,"Й":43 , "К":44 , "Л":45 , "М":46 , "Н":47 , "О":48 , "П":49 , "Р":50 , "С":51 , "Т":52 , "У"
    "Ш":58,"Щ":59 , "Ъ":60 , "Ы":61 , "Ь":62 , "Э":63 , "Ю":64 , "Я":65 , "1":66 , "2":67 , "3":68 , "4":69 , "5":70
    }
    dict2 = {v: k for k, v in dicts.items()}
    text = P1
    gamma = input("Введите гамму(на русском языке!")
    listofdigitsoftext = list()
    listofdigitsofgamma = list()

    for i in text:
        listofdigitsoftext.append(dicts[i])
    print("Числа текста", listofdigitsoftext)
    for i in gamma:
        listofdigitsofgamma.append(dicts[i])
    print("числа гаммы", listofdigitsofgamma)
    listofdigitsresult = list()
    ch = 0
    for i in text:
        try:
            a = dicts[i] + listofdigitsofgamma[ch]
        except:
            ch = 0
            a = dicts[i] + listofdigitsofgamma[ch]
        if a > 75:
            a = a%75
            print(a)
        ch += 1
        listofdigitsresult.append(a)
    print("Числа зашифрованного текста", listofdigitsresult)
```

Рис. 3.3: Функция “shifr”

4. Алгоритм расшифровки, вывод программы. (рис. 3.4).

```
    listofdigits1 = list()
    for i in listofdigits:
        try:
            a = i - listofdigitsofgamma[ch]
        except:
            ch=0
            a = i - listofdigitsofgamma[ch]
        if a < 1:
            a = 75 + a
        listofdigits1.append(a)
        ch += 1
    textdecrypted = ""
    for i in listofdigits1:
        textdecrypted += dict2[i]
    print("Расшифрованный текст:", textdecrypted)

shifr(P1)
```

```
Введите гамму(на русском языке)цСЗвзшюЖчш74рйцУ1ЕА4
Числа текста [47, 1, 35, 1, 26, 10, 19, 23, 16, 5, 32, 27, 10, 11, 16, 20, 66, 67, 75, 69]
числа гаммы [27, 51, 41, 3, 31, 26, 32, 40, 25, 26, 72, 69, 18, 11, 27, 53, 66, 38, 33, 69]
1
29
21
57
30
33
63
Числа зашифрованного текста [74, 52, 1, 4, 57, 36, 51, 63, 41, 31, 29, 21, 28, 22, 43, 73, 57, 30, 33, 63]
Зашифрованный текст:  9ТагЧГСЭЗэыуъфИВЧьАЗ
Расшифрованный текст: НаВашисходящийот1204
```

Рис. 3.4: Результат”

4 Выводы

На основе проделанной работы освоила на практике применение режима однократного гаммирования на примере кодирования различных исходных текстов одним ключом.

5 Контрольные вопросы

1. Как, зная один из текстов (P1 или P2), определить другой, не зная при этом ключа?

Можно использовать комбинацию, при которой ключ не будет использоваться, для прочтения неизвестного текста:

$$C_1 \oplus C_2 \oplus P_1 = P_1 \oplus P_2 \oplus P_1 = P_2.$$

C1 и C2 – известный шифротекст

2. Что будет при повторном использовании ключа при шифровании текста?

Текст расшифруется.

3. Как реализуется режим шифрования однократного гаммирования одним ключом двух открытых текстов?

Используется следующая комбинация:

$$C_1 = P_1 \oplus K_i$$

$$C_2 = P_2 \oplus K_i$$

Pi – открытый текст, Ci – зашифрованный текст, K – ключ шифрования.

4. Перечислите недостатки шифрования одним ключом двух открытых текстов.

ключ, попав не в те руки, даст возможность злоумышленнику расшифровать оба текста; можно расшифровать с помощью открытого текста другие известные шифротексты; можно узнать часть текста, используя заранее известный шаблон и формат другого текста.

5. Перечислите преимущества шифрования одним ключом двух открытых текстов.

Преимущества шифрования одним ключом заключаются в том что, скорость шифрования выше; алгоритм шифрования простой; а так же шифротекст сильно меняется, если изменяется ключ или открытый текст.☒