Лабораторная работа №7

Дисциплина: Информационная безопасность

Губина Ольга Вячеслвовна

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

Освоить на практике применение режима однократного гаммирования.

2 Задание

- Написать программу осуществляющую шифровние.

3 Теоретическое введение

Гаммирование, или Шифр XOR, — метод симметричного шифрования, заключающийся в «наложении» последовательности, состоящей из случайных чисел, на открытый текст. Последовательность случайных чисел называется гамма-последовательностью и используется для зашифровывания и расшифровывания данных. Суммирование обычно выполняется в каком-либо конечном поле. Например, в поле Галуа GF(2) суммирование принимает вид операции «исключающее ИЛИ (XOR)».[1]

Шифры гаммирования (аддитивные шифры) являются самыми эффективными с точки зрения стойкости и скорости преобразований (процедур зашифрования и дешифрования). По стойкости данные шифры относятся к классу совершенных. Для зашифрования и дешифрования используются элементарные арифметические операции – открытое / зашифрованное сообщение и гамма, представленные в числовом виде, складываются друг с другом по модулю (mod). Напомним, что результатом сложения двух целых чисел по модулю является остаток от деления (например, 5+10 mod 4 = 15 mod 4 = 3).

В литературе шифры этого класса часто называют потоковыми, хотя к потоковым относятся и другие разновидности шифров. В шифрах гаммирования может использоваться сложение по модулю N (общий случай) и по модулю 2 (частный случай, ориентированный на программно-аппаратную реализацию).[2]

В этом способе шифрование выполняется путем сложения символов исходного текста и ключа по модулю, равному числу букв в алфавите. Если в исходном алфавите, например, 33 символа, то сложение производится по модулю 33. Та-

кой процесс сложения исходного текста и ключа называется в криптографии наложением гаммы.

Пусть символам исходного алфавита соответствуют числа от 0 (А) до 32 (Я). Если обозначить число, соответствующее исходному символу, x, а символу ключа – k, то можно записать правило гаммирования следующим образом:

$z = x + k \pmod N$, где z – закодированный символ, N - количество символов в алфавите, а сложение по модулю N - операция, аналогичная обычному сложению, с тем отличием, что если обычное суммирование дает результат, больший или равный N, то значением суммы считается остаток от деления его на N. Например, пусть сложим по модулю 33 символы Г (3) и Ю (31):

$3 + 31 \pmod{33} = 1$, то есть в результате получаем символ Б, соответствующий числу 1.[3]

4 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: 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.

Был написан код на языке Python, предоставленный на рис. 4.1 - 4.5.

```
gamma = "безопаснаяпередачаданных"
text = "сновымгодом"

alphabet = {"а" :0, "б" :1 , "в" :2 ,"г" :3 ,"д" :4 ,"е" :5 ,"ё" :6 ,"ж": 7,
            "з": 8, "и": 9, "й": 10, "к": 11, "л": 12, "м": 13, "н": 14,
            "о": 15, "п": 16, "р": 17, "с": 18, "т": 19, "у": 20, "ф": 21,
            "х": 22, "ц": 23, "ч": 24, "ш": 25, "щ": 26, "ъ": 27,"ы": 28,
            "ь": 29, "э": 30, "ю": 31, "я": 32
            } # это алфавит, где буква - ключ, а номер - значение

# создаем словарь для обратного определения букв по числу
alphabet_2 = {v: k for k, v in dict.items()}
```

Рис. 4.1: Созлание алфафита для осуществления кодирования, гамма и текст для шифрования

```
text_nums = list() # числа букв по алфавиту
gamma_nums = list() # то же самое для гаммы

# в созданный список запишем числа букв из текста
for i in text:
  text_nums.append(alphabet[i])

# print("Числа текста", text_nums)

# в созданный список запишем числа букв из гаммы
for i in gamma:
  gamma_nums.append(alphabet[i])
```

Рис. 4.2: Сопоставление текста с нумерацией

```
result_nums = list() # результат

ch = 0 # шифруем по принципу остатка от деления числа буквы, на ко-во символов
# в алфавите
for i in text:
  try:
    a = alphabet[i] + gamma_nums[ch]
  except:
    ch=0
    a = alphabet[i] + gamma_nums[ch]
  if a >= 33:
    a = a % 33
  ch+=1
  result_nums.append(a)
```

Рис. 4.3: Шифрование цифрами по принципу сложения по модулю 33 - число знаков алфавита

```
# теперь обратно числа представим в виде букв
text_encrypted = ""

for i in result_nums:
  text_encrypted += alphabet_2[i]

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

Рис. 4.4: Создание кодированного текста через полученные цифры

```python
#теперь приступим к реализации алгоритма дешифровки
listofdigits = list()
for i in text_encrypted:
  listofdigits.append(alphabet[i])
ch = 0
listofdigits1 = list()
for i in listofdigits:
    a = i - gamma_nums[ch]
    #проблемы тут могут быть
    if a < 0:
        a = 32 + a
    listofdigits1.append(a)
    ch+=1
textdecrypted = ""
for i in listofdigits1:
    textdecrypted+=alphabet_2[i]
print("Расшифрованный текст:", textdecrypted)
```

Рис. 4.5: Дешифрование текста

Результатом дешифрования фразы *сновымгодом* является следующее (рис. 4.6):

```
Зашифрованный текст: ттцркмфьднь
Расшифрованный текст: сновъмгоднм
```

Рис. 4.6: Результат дешифрования текста “сновымгодом”

Текст дешифруется не совсем корректно. Это обуславливается тем, что полученный фрагмент текста может быть неверным из-за случайных совпадений частот символов в шифротексте. В этом случае можно использовать дополнительные методы для уточнения ключа и расшифровки текста.

Так например, короткие слова дешифруются корректно (рис. 4.7)

```
Зашифрованный текст: ойъ
Расшифрованный текст: нет
```

Рис. 4.7: результат дешифрования теkаста “нет”

5 Выводы

Освоила на практике применение режима однократного гаммирования.

Список литературы

1. Гаммирование [Электронный ресурс]. 2023. URL: https://ru.wikipedia.org/wiki/%D0%93%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5.
2. Основы шифрования [Электронный ресурс]. 2023. URL: https://sites.google.com/site/anisimovkhv/learning/kripto/lecture/tema6.
3. Простейшие методы шифрования с закрытым ключом [Электронный ресурс]. 2023. URL: https://intuit.ru/studies/mini_mba/5398/courses/547/lecture/12373?page=4.