Лабораторная работа №8

Элементы криптографии. Шифрование (кодирование) различных исходных текстов одним ключом

Беличева Дарья Михайловна

Содержание

Список иллюстраций

1 Цель работы

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

2 Теоретическое введение

Гаммиирование, 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[1].

3 Выполнение лабораторной работы

Два текста кодируются одним ключом (однократное гаммирование). Требуется не зная ключа и не стремясь его определить, прочитать оба текста. Необходимо разработать приложение, позволяющее шифровать и дешифровать тексты P1 и P2 в режиме однократного гаммирования. Приложение должно определить вид шифротекстов C1 и C2 обоих текстов P1 и P2 при известном ключе ; Необходимо определить и выразить аналитически способ, при котором злоумышленник может прочитать оба текста, не зная ключа и не стремясь его определить.

Создадим функцию `key_gen()` для генерации случайного ключа, с помощью которого происходит шифрование. Ключ рандомно генерируется из строчных и заглавных букв русского алфавита, а также из специальных символов. Самое главное в генерации ключа – он такой же длины, что и текст, который шифруется.

Далее создаем функцию `xor()` для применения созданного ключа к исходному тексту (собственно однократное гаммирование), чтобы в итоге получить зашифрованный текст. Здесь у нас выполняется поэлементное сложения по модулю 2.

```
def key_gen(text):
    cirillic = [chr(i) for i in range(1040,1104)]
    symbols = [chr(i) for i in range(32,65)]
    all_characters = cirillic + symbols
    return ''.join([random.choice(all_characters) for i in range(len(text))])
```

```
def xor(text,key):
    return ''.join([chr(ord(a)^ord(b)) for a,b in zip(text,key)])
```

Сгенерируем ключ и зашифруем два сообщения:

```
P1 = "ВЗападныйФилиалБанка"
P2 = "ВСеверныйФилиалБанка"
key = key_gen(P1)
C1 = xor(P1, key)
C2 = xor(P2, key)
```

Опишем случай, когда злоумышленник может прочитать оба текста, не зная ключа и не стремясь его определить. Предположим, что одна из телеграмм является шаблоном – т.е. имеет текст фиксированного формата, в который вписываются значения полей. Допустим, что злоумышленнику этот формат известен. Тогда он получает достаточно много пар $C1 \oplus C2$ (известен вид обеих шифровок). Тогда зная P1 имеем:

$$C1 \oplus C2 \oplus P1 = P1 \oplus P2 \oplus P1 = P2.$$

Покажем этот процесс на практике. Применим наши функции к заданному сообщению. Допустим нам известна часть второго сообщения. В цикле `while` в интерактивном режиме будет отгадывать части сообщений, пока не угадаем их полностью:

```
fragment = "ВСев"


msg2 = fragment
c1, c2 = C1, C2
length = len(msg2)
while length <= len(P1):
    C12 = xor(C1[:length], C2[:length])
```

```
msg1 = xor(C12, msg2)
print("Расшифрованный текст:")
display(msg1 + c1[length:])
if length >= len(P1) - 1:
        break
print("Введите продолжение текста: ")
msg1 += input()
length = len(msg1)
display(msg1 + c1[length:])

msg1, msg2 = msg2, msg1
c1, c2 = c2, c1
```

Результат работы программы выглядит следующим образом (рис. 3.1).

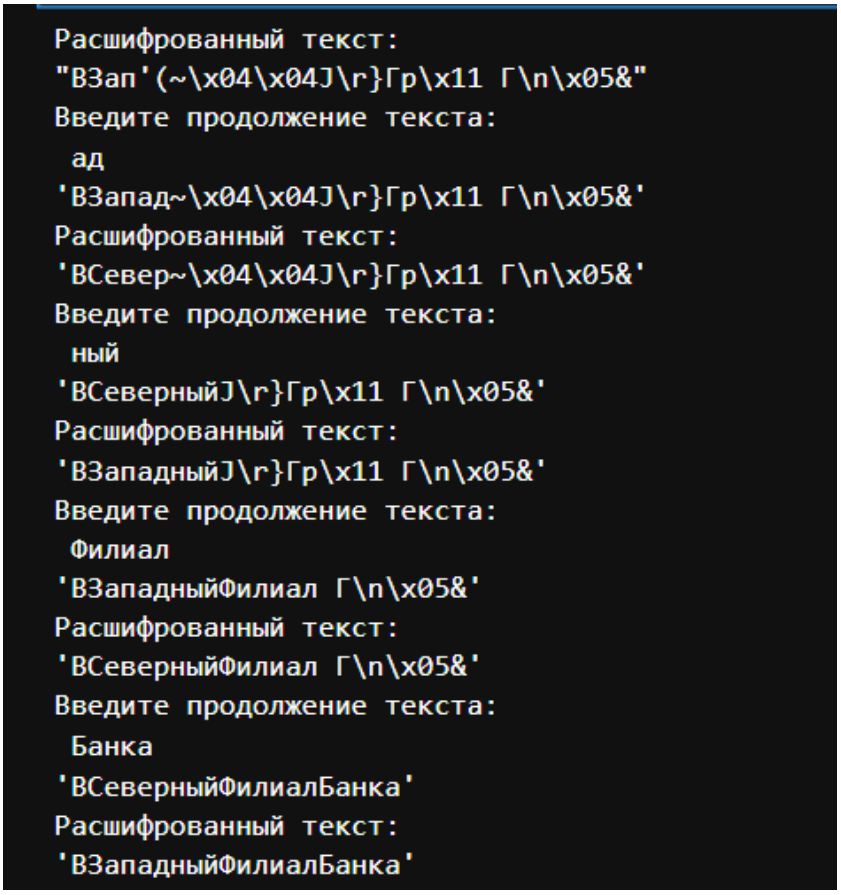

Рис. 3.1: Результат работы программы

4 Контрольные вопросы

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

Предположим, что одна из телеграмм является шаблоном – т.е. имеет текст фиксированный формат, в который вписываются значения полей. Допустим, что злоумышленнику этот формат известен. Тогда он получает достаточно много пар $C1 \oplus C2$ (известен вид обеих шифровок). Тогда зная P1 имеем:

$$C1 \oplus C2 \oplus P1 = P1 \oplus P2 \oplus P1 = P2.$$

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

Текст вернется к исходному виду.

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

К обоим текстам применяется один и тот же ключ.

4. Перечислите недостатки шифрования одним ключом двух открытых текстов.

Главным недостатком является повышение уязвимости. Если злоумышленник узнает один из исзодных текстов или даже его часть, то он может узнать и второй текст.

5. Перечислите преимущества шифрования одним ключом двух открытых текстов.

Ключи могут занимать большое количество памяти и долго генерироваться, поэтому использование одного ключа оптимизирует шифрование. Также это упрощает дешифровку.

5 Выводы

В результате выполнения данной лабораторной работы я освоила на практике применение режима однократного гаммирования на примере кодирования различных исходных текстов одним ключом.

Список литературы

1. Гаммирование [Электронный ресурс]. 2023. URL: https://ru.wikipedia.org/wiki/Гаммирование.