

Отчёт по лабораторной работе 7

Элементы криптографии. Однократное гаммирование

Максимова Ксения НБИбд-02-18

Содержание

1	Цель работы	4
2	Задание	5
3	Теоретическое введение	6
4	Выполнение лабораторной работы	7
5	Выводы	8
	Список литературы	9

List of Figures

4.1 Рис 1.Код программы 7

1 Цель работы

Освоить на практике применение режима однократного гаммирования.

2 Задание

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

3 Теоретическое введение

Смысл однократного гаммирования заключается в наложении (снятии) 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 [1].

Преимущества однократного гаммирования[1]: 1. Абсолютная стойкость шифра в случае, когда однократно используемый ключ, длиной, равной длине исходного сообщения, является фрагментом истинно случайной двоичной последовательности с равномерным законом распределения. 2. Криптоалгоритм не даёт никакой информации об открытом тексте: при известном зашифрованном сообщении все различные ключевые последовательности возможны и равновероятны

При всех очевидных преимуществах, есть один весомый недостаток, который сразу бросается в глаза, - это необходимость иметь огромные объемы данных, которые можно было бы использовать в качестве гаммы. Для этих целей обычно пользуются датчиками настоящих случайных чисел[2].

Необходимые и достаточные условия абсолютной стойкости шифра[1]: - полная случайность ключа; - равенство длин ключа и открытого текста; - однократное использование ключа

4 Выполнение лабораторной работы

Программа, позволяющая шифровать и дешифровать данные в режиме однократного гаммирования.

```
In [68]: 1 key_phrase = "С Новым Годом, друзья!" #подлежащий секретно текст
In [69]: 1 actual_phrase = "Текст текст текст 1234" #однократная вероятностная гамма
In [70]: 1 def phrase_xor(actual_phrase, key_phrase):
2         return "".join(chr(ord(x)^ord(y)) for x, y in zip(actual_phrase, key_phrase))
In [71]: 1 xored = phrase_xor(actual_phrase, key_phrase)
2         xored
Out[71]: "\x03E'\x7fpx-E)\x7fvO-9ku\x02bI00'\x15"
In [72]: 1 b = bytes(xored, "UTF-8")
2         b
Out[72]: b"\x03\x00\x95'\x7fp\x01\xab~\x00\x95)\x7fv\x00\x9e~\x00\x99\x00\x9au\x02'\xa3'\x00\x86\x01\xbe'\x01\xbc'\x15"
In [73]: 1 import codecs
2         codecs.encode(b, 'hex')
Out[73]: b'03d095277f70d1ab7ed095297f76d09e7ed099d09a7502d1a3d086d1bed1bc15'
In [74]: 1 phrase_xor(actual_phrase, xored)
Out[74]: 'С Новым Годом, друзья!'
In [75]: 1 phrase_xor(key_phrase, xored)
Out[75]: 'Текст текст текст 1234'
In [ ]: 1 |
```

Figure 4.1: Рис 1.Код программы

Рисунок 1

5 Выводы

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

Список литературы

1. Элементы криптографии. Однократное гаммирование
2. Прикладные задачи шифрования