Отчёт по лабораторной работе

Лабораторная работа № 8

Живцова Анна

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

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

2 Задание

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

3 Теоретическое введение

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 [1].

4 Выполнение лабораторной работы

1. Создана программа, шифрующая сообщение по ключу.
2. При шифровке двух текстов одним ключом, создана программа, определяющая один из исходных текстов по двум шифротекстам и одному исходному тексту.
3. Программа проверена на данных из пособия

Листинг кода

```
coding_const = 1040 - 192
def one_in_rus(text1, text2):
    text1 = [ord(i) - coding_const if ord(i) > coding_const else
                                          ord(i)  for i in text1]
    text2 = text2.split(' ')
    return  ' '.join(format(text1[i] ^ int(text2[i], 16), 'X')
                                       for i in range(len(text1)))

def find_text_2(text1, crypt_text_1, crypt_text_2):
    text1 = [ord(i) - coding_const if ord(i) > coding_const else
                                         ord(i) for i in text1]
    crypt_text_1 = crypt_text_1.split(' ')
    crypt_text_2 = crypt_text_2.split(' ')
```

```
    return  ''.join(chr((int(crypt_text_1[i], 16) ^
                int(crypt_text_2[i], 16) ^ text1[i]) + coding_const)
    if int(crypt_text_1[i], 16) ^ int(crypt_text_2[i], 16) ^ text1[i] > 192
    else chr(int(crypt_text_1[i], 16) ^ int(crypt_text_2[i], 16) ^ text1[i])
                                                for i in range(len(text1)))
```

Процесс тестирования

```
key = '05 0C 17 7F 0E 4E 37 D2 94 10 09 2E 22 57 FF C8 0B B2 70 54'
text1 = 'НаВашисходящийот1204'
text2 = 'ВСеверныйфилиалБанка'
crypt_text1 = one_in_rus(text1, key)
crypt_text2 = one_in_rus(text2, key)
find_text_2(text1, crypt_text1, crypt_text2)
```

Вывод ВСеверныйфилиалБанка

5 Выводы

Успешно реализовали метод однократного гаммирования.

Список литературы

1. А.А. Аргановский, Р.А.Хади. Практическая криптография: алгоритмы и их программирование. солон пресс, 2009.