Лабораторная работа №7

Элементы криптографии. Однократное гаммирование

Парфенова Е. Е.

19 октября 2024

Российский университет дружбы народов, Москва, Россия

Информация

Докладчик

- Парфенова Елизавета Евгеньвена
- студент
- Российский университет дружбы народов
- 1032216437@pfur.ru
- https://github.com/parfenovaee

Вводная часть

Актуальность

Важность понимания способов шифрования сообщений для обеспечения их макисмальной безопасности

Цели и задачи

Цель: Освоить на практике применение режима однократного гаммирования.

Задача: 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.

Теоретическое введение

Теоретическое введение(1)

Криптография — наука о методах обеспечения конфиденциальности, целостности данных, аутентификации, шифрования.

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.

Теоретическое введение(2)

С точки зрения теории криптоанализа метод шифрования однократной случайной равновероятной гаммой (однократное гаммирование) той же длины, что и открытый текст, является невскрываемым. Даже при раскрытии части последовательности гаммы нельзя получить информацию о всём скрываемом тексте.

Выполнение лабораторной работы

Функции программы

- generate_random_key - функция генерирует ключ шифрования на основе изначального текста
- xor - функция, которая выполняет само гаммирование
- encrypt - функция, шифрующая текст по сгенерированному ключу
- decrypt - функция, которая, наоборот, расшифровывает шифротекст по определенному ключу
- find_key - функция, которая ищет ключ по фрагменту изначального текста и шифротексту.

Листинг программы

```
import random

def generate_random_key(text):
    possible_symbol = list(range(32, 127)) + list(range(1040, 1104))
    key_str = ''.join(chr(random.choice(possible_symbol))
        for _ in range(len(text)))
    return key_str

def xor(text, key):
    return [ord(s1)^ord(s2) for s1,s2 in zip(text, key)]
```

Листинг программы

```
def encrypt(text, key):
    chiphr = xor(text, key)
    chiphrotext = ''.join(chr(i) for i in chiphr)
    return chiphrotext

def decrypt(chiphro, key):
    decrypted = xor(chiphro, key)
    opentext = ''.join(chr(i) for i in decrypted)
    return opentext
```

Листинг программы

```
def find_key(chiphrotext, text_fragment):
    chipher_fragment = chiphrotext[:(len(chiphrotext))]
    key_f = xor(text_fragment, chipher_fragment)
    found_key = ''.join(chr(i) for i in key_f)
    return found_key

text = "С Новым Годом, друзья!"
text_fragment = "С Новым"

key = generate_random_key(text)
print("Созданный ключ: ", key)
```

Листинг программы

```
chiphrotext = encrypt(text, key)
print('Открытый текст: ', text)
print('Зашифрованный текст: ', chiphrotext)

opentext = decrypt(chiphrotext, key)
print('Расшифрованный текст: ', opentext)

found_key = find_key(chiphrotext, text_fragment)
open_fragtext = decrypt(chiphrotext[:len(text_fragment)], found_key)
print('Один из возможных вариантов прочтения текста по фрагменту',
    open_fragtext + chiphrotext[len(text_fragment):])
```

Результат работы программы

```
Созданный ключ:  BnmЬИК%YD+b=JoПhГяЩЭkt
Открытый текст:  С Новым Годом, друзья!
Зашифрованный текст:  3N!@*QЙЃїEifЎCnќS@auU
Расшифрованный текст:  С Новым Годом, друзья!
Один из возможных вариантов прочтения текста по фрагменту С НовымЃїEifЎCnќS@auU
```

Рис. 1: Результат работы программы

Вывод

Выводы

В ходе лабораторной работы мы на практике освоили применение режима однократного гаммирования.