Лабораторная работа №8

Основы Информационной Безопасности

Чистов Даниил Максимович

Содержание

Список иллюстраций

1 Цель работы

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

2 Выполнение лабораторной работы

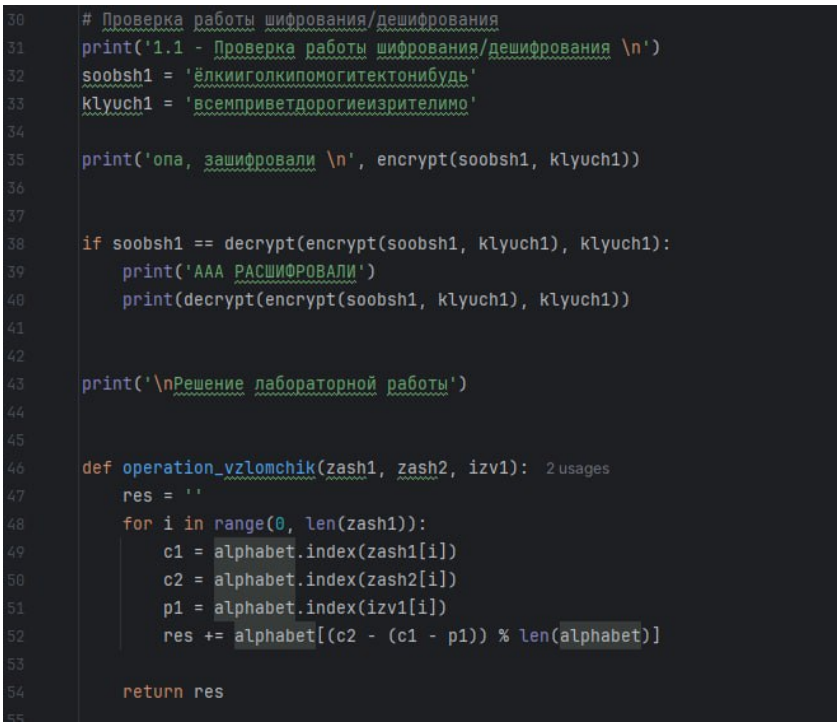
Требуется написать программу, которая сможет получить расшифрованное сообщение 2, зная только оба зашифрованных сообщения и расшифрованное сообщение 1, без наглядного поиска ключа - на первом фото: функции шифрования и дешифрования текстов. Принцип такой: у нас есть алфавит из русских больших, маленьких букв и цифр. У каждого символа есть индекс. Также индексы есть и у сообщений. Складываем индексы сообщения и ключа, так получаем новый индекс и достаём из алфавита другую букву, соответствующую новому индексу. Дешифрование реализуется также, но вместо сложения, происходит вычитание (рис. 2.1).

```python
alphabet = [
    "а", "б", "в", "г", "д", "е", "ё", "ж", "з", "и", "й", "к", "л", "м", "н", "о",
    "п", "р", "с", "т", "у", "ф", "х", "ц", "ч", "ш", "щ", "ъ", "ы", "ь", "э", "ю", "я",
    "А", "Б", "В", "Г", "Д", "Е", "Ё", "Ж", "З", "И", "Й", "К", "Л", "М", "Н", "О",
    "П", "Р", "С", "Т", "У", "Ф", "Х", "Ц", "Ч", "Ш", "Щ", "Ъ", "Ы", "Ь", "Э", "Ю", "Я",
    "0", "1", "2", "3", "4", "5", "6", "7", "8", "9"
]

# Шифрование
def encrypt(text, key):  5 usages
    res = ''
    for i in range(0, len(text)):
        i1 = alphabet.index(text[i])
        i2 = alphabet.index(key[i])
        res += alphabet[(i1 + i2) % len(alphabet)]
    return res


# Дешифрование
def decrypt(text, key):  2 usages
    res = ''
    for i in range(0, len(text)):
        i1 = alphabet.index(text[i])
        i2 = alphabet.index(key[i])
        res += alphabet[(i1 - i2) % len(alphabet)]
    return res
```

Рис. 2.1: Функции шифрования и дешифрования

На втором фото: функция получения расшифрованного сообщения 2, зная сооб-

щение 1 и оба зашифрованных сообщений. Принцип такой, т.к. чтобы получить ключ, нам требовалось вычесть из индекса буквы сообщения индекс буквы ключа, то теперь, чтобы по сути получить ключ мы вычитаем из индекса буквы зашифрованного сообщения индекс буквы расшифрованного сообщения, так мы по сути получили индекс буквы ключа, который можем вычесть из зашифрованного сообщения (рис. 2.1).



```
# Проверка работы шифрования/дешифрования
print('1.1 - Проверка работы шифрования/дешифрования \n')
soobsh1 = 'ёлкииголкипомогитектонибудь'
klyuch1 = 'всемприветдорогиеизрителимо'

print('она, зашифровали \n', encrypt(soobsh1, klyuch1))


if soobsh1 == decrypt(encrypt(soobsh1, klyuch1), klyuch1):
    print('ААА РАСШИФРОВАЛИ')
    print(decrypt(encrypt(soobsh1, klyuch1), klyuch1))


print('\nРешение лабораторной работы')


def operation_vzlomchik(zash1, zash2, izv1):  2 usages
    res = ''
    for i in range(0, len(zash1)):
        c1 = alphabet.index(zash1[i])
        c2 = alphabet.index(zash2[i])
        p1 = alphabet.index(izv1[i])
        res += alphabet[(c2 - (c1 - p1)) % len(alphabet)]

    return res
```

Рис. 2.2: Функция взлома

На третьем фото: результат работы всей программы

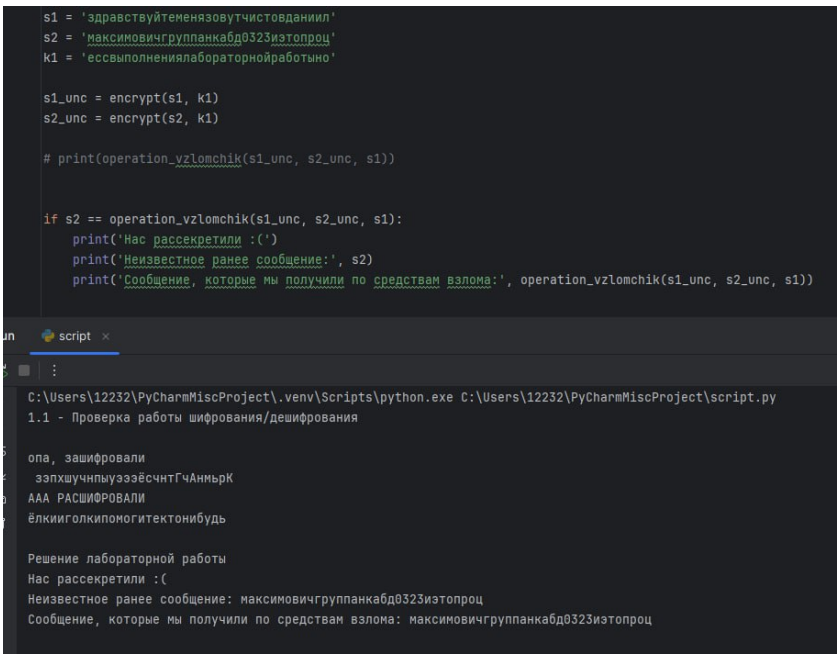


Рис. 2.3: Результат взлома

Сам код:

```
alphabet = [
    "а", "б", "в", "г", "д", "е", "ё", "ж", "з", "и", "й", "к", "л", "м", "н", "о",
    "п", "р", "с", "т", "у", "ф", "х", "ц", "ч", "ш", "щ", "ъ", "ы", "ь", "э", "ю", "я
    "А", "Б", "В", "Г", "Д", "Е", "Ё", "Ж", "З", "И", "Й", "К", "Л", "М", "Н", "О",
    "П", "Р", "С", "Т", "У", "Ф", "Х", "Ц", "Ч", "Ш", "Щ", "Ъ", "Ы", "Ь", "Э", "Ю", "Я
    "0", "1", "2", "3", "4", "5", "6", "7", "8", "9"
]

# Шифрование
def encrypt(text, key):
    res = ''
    for i in range(0, len(text)):
        i1 = alphabet.index(text[i])
        i2 = alphabet.index(key[i])
        res += alphabet[(i1 + i2) % len(alphabet)]
```

```
    return res


# Дешифрование
def decrypt(text, key):
    res = ''
    for i in range(0, len(text)):
        i1 = alphabet.index(text[i])
        i2 = alphabet.index(key[i])
        res += alphabet[(i1 - i2) % len(alphabet)]
    return res



# Проверка работы шифрования/дешифрования
print('1.1 - Проверка работы шифрования/дешифрования \n')
soobsh1 = 'ёлкииголкипомогитектонибудь'
klyuch1 = 'всемприветдорогиеизрителимо'

print('опа, зашифровали \n', encrypt(soobsh1, klyuch1))


if soobsh1 == decrypt(encrypt(soobsh1, klyuch1), klyuch1):
    print('ААА РАСШИФРОВАЛИ')
    print(decrypt(encrypt(soobsh1, klyuch1), klyuch1))


print('\nРешение лабораторной работы')
```

```
def operation_vzlomchik(zash1, zash2, izv1):
    res = ''
    for i in range(0, len(zash1)):
        c1 = alphabet.index(zash1[i])
        c2 = alphabet.index(zash2[i])
        p1 = alphabet.index(izv1[i])
        res += alphabet[(c2 - (c1 - p1)) % len(alphabet)]

    return res


s1 = 'здравствуйтеменязовутчистовданиил'
s2 = 'максимовичгруппанкабд0323иэтопроц'
k1 = 'ессвыполнениялабораторнойработыно'

s1_unc = encrypt(s1, k1)
s2_unc = encrypt(s2, k1)

# print(operation_vzlomchik(s1_unc, s2_unc, s1))


if s2 == operation_vzlomchik(s1_unc, s2_unc, s1):
    print('Нас рассекретили :(')
    print('Неизвестное ранее сообщение:', s2)
    print('Сообщение, которые мы получили по средствам взлома:', operation_vzlomchik(s
```

3 Выводы

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

4 Список литературы

Лабораторная работа №8