

Лабораторная работа №8

Основы информационной безопасности

Серёгина Ирина Андреевна

Содержание

1	Цель работы	3
2	Задание	4
3	Теоретическое введение	5
4	Выполнение лабораторной работы	6
5	Выводы	9
	Список литературы	10

1 Цель работы

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

2 Задание

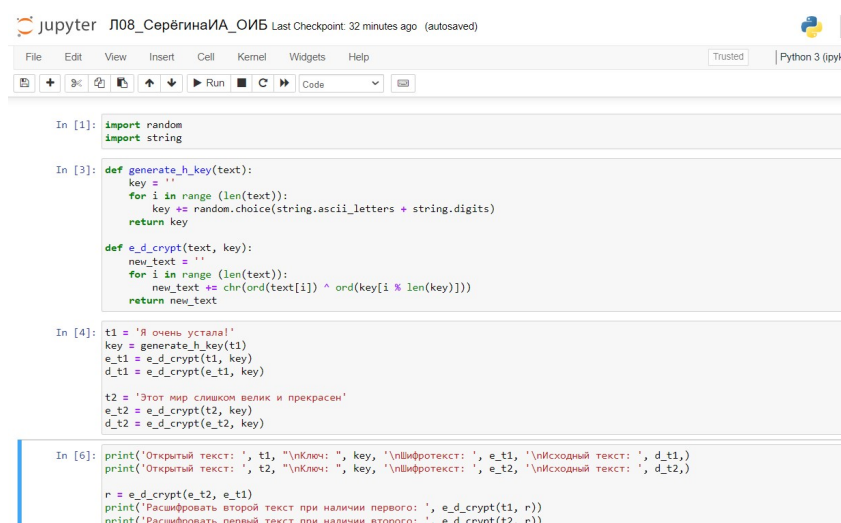
Требуется не зная ключа и не стремясь его определить, прочитать оба текста. Необходимо разработать приложение, позволяющее шифровать и дешифровать тексты P_1 и P_2 в режиме однократного гаммирования. Приложение должно определить вид шифротекстов C_1 и C_2 обоих текстов P_1 и P_2 при известном ключе; Необходимо определить и выразить аналитически способ, при котором злоумышленник может прочитать оба текста, не зная ключа и не стремясь его определить.

3 Теоретическое введение

Исходные данные. Две телеграммы Центра: $P_1 = \text{НаВашиходящийот1204}$ $P_2 = \text{ВСеверныйфилиалБанка}$ Ключ Центра длиной 20 байт: $K = 05\ 0C\ 17\ 7F\ 0E\ 4E\ 37\ D2\ 94\ 10\ 09\ 2E\ 22\ 57\ FF\ C8\ 0B\ B2\ 70\ 54$ Режим шифрования однократного гаммирования одним ключом двух видов открытого текста реализуется в соответствии со схемой, приведённой на рис. 8.1. Шифротексты обеих телеграмм можно получить по формулам режима однократного гаммирования: $C_1 = P_1 \boxtimes K$, $C_2 = P_2 \boxtimes K$. (8.1) Открытый текст можно найти в соответствии с (8.1), зная шифротекст двух телеграмм, зашифрованных одним ключом. Для это оба равенства (8.1) складываются по модулю 2. Тогда с учётом свойства операции XOR $1 \boxtimes 1 = 0$, $1 \boxtimes 0 = 1$ (8.2) получаем: $C_1 \boxtimes C_2 = P_1 \boxtimes K \boxtimes P_2 \boxtimes K = P_1 \boxtimes P_2$. 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 $C_1 \boxtimes C_2$ (известен вид обеих шифровок). Тогда зная P_1 и учитывая (8.2), имеем: $C_1 \boxtimes C_2 \boxtimes P_1 = P_1 \boxtimes P_2 \boxtimes P_1 = P_2$. (8.3) Таким образом, злоумышленник получает возможность определить те символы сообщения P_2 , которые находятся на позициях известного шаблона сообщения P_1 . В соответствии с логикой сообщения P_2 , злоумышленник имеет реальный шанс узнать ещё некоторое количество символов сообщения P_2 . Затем вновь используется (8.3) с подстановкой вместо P_1 полученных на предыдущем шаге новых символов сообщения P_2 . И так далее. Действуя подобным образом, злоумышленник даже если не прочитает оба сообщения, то значительно уменьшит пространство их поиска.

4 Выполнение лабораторной работы

1. Реализую две функции, как и в прошлой лабораторной работе, ввожу необходимые данные (рис. 4.1).



```
jupyter Л08_СерёгинаИА_ОИБ Last Checkpoint: 32 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipyk

In [1]: import random
import string

In [3]: def generate_h_key(text):
key = ''
for i in range (len(text)):
key += random.choice(string.ascii_letters + string.digits)
return key

def e_d_crypt(text, key):
new_text = ''
for i in range (len(text)):
new_text += chr(ord(text[i]) ^ ord(key[i % len(key)]))
return new_text

In [4]: t1 = 'Я очень устала!'
key = generate_h_key(t1)
e_t1 = e_d_crypt(t1, key)
d_t1 = e_d_crypt(e_t1, key)

t2 = 'Этот мир слишком велик и прекрасен'
e_t2 = e_d_crypt(t2, key)
d_t2 = e_d_crypt(e_t2, key)

In [6]: print('Открытый текст: ', t1, "\nКлюч: ", key, "\nШифротекст: ', e_t1, '\nИсходный текст: ', d_t1,)
print('Открытый текст: ', t2, "\nКлюч: ", key, "\nШифротекст: ', e_t2, '\nИсходный текст: ', d_t2,)

r = e_d_crypt(e_t2, e_t1)
print('Расшифровать второй текст при наличии первого: ', e_d_crypt(t1, r))
print('Расшифровать первый текст при наличии второго: ', e_d_crypt(t2, r))
```

Рис. 4.1: код программы

2. Получаю необходимый результат (рис. 4.2).

```
r = e_d_crypt(e_t2, e_t1)
print('Расшифровать второй текст при наличии первого: ', e_d_crypt(t1, r))
print('Расшифровать первый текст при наличии второго: ', e_d_crypt(t2, r))

Открытый текст: Я очень устала!
Ключ: yIMIAuZZEe4szUJ
Шифротекст: ine9VvshKzi0Vusiek
Исходный текст: Я очень устала!
Открытый текст: Этот мир слишком велик и прекрасен
Ключ: yIMIAuZZEe4szUJ
Шифротекст: еКёґащбКеФҫығзVхпѡ00а0z0ёЕґгяҗщU0yV
Исходный текст: Этот мир слишком велик и прекрасен
Расшифровать второй текст при наличии первого: Этот мир слишком
Расшифровать первый текст при наличии второго: Я очень устала!Я очень устала!Я оч
```

Рис. 4.2: результат работы программы

#Листинг

```

import random
import string

def generate_h_key(text):
    key = ''
    for i in range (len(text)):
        key += random.choice(string.ascii_letters + string.digits)
    return key

def e_d_crypt(text, key):
    new_text = ''
    for i in range (len(text)):
        new_text += chr(ord(text[i]) ^ ord(key[i % len(key)]))
    return new_text

t1 = 'Я очень устала!'
key = generate_h_key(t1)
e_t1 = e_d_crypt(t1, key)
d_t1 = e_d_crypt(e_t1, key)

t2 = 'Этот мир слишком велик и прекрасен'
e_t2 = e_d_crypt(t2, key)
d_t2 = e_d_crypt(e_t2, key)

print('Открытый текст: ', t1, "\nКлюч: ", key, '\nШифротекст: ', e_t1, '\nИсходный текст: ', t2, "\nКлюч: ", key, '\nШифротекст: ', e_t2, '\nИсходный текст: ', d_t2)

r = e_d_crypt(e_t2, e_t1)
print('Расшифровать второй текст при наличии первого: ', e_d_crypt(t1, r))

```

```
print('Расшифровать первый текст при наличии второго: ', e_d_crypt(t2, r))
```


5 Выводы

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

Список литературы