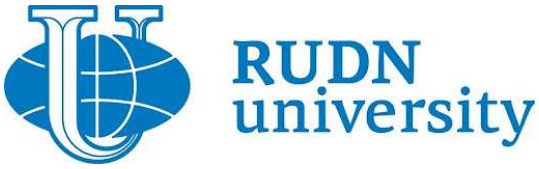
Лабораторная работа №8

Шифрование (кодирование) различных исходных текстов одним ключом

Доборщук В.В., НФИбд-01-18



18 декабря 2021

Содержание

Список иллюстраций

Цель работы

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

Выполнение лабораторной работы

Два текста кодируются одним ключом (однократное гаммирование). 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.

Теоретическое введение

Для выполнения данной лабораторной работы мы использовали данные источники, в виде описания лабораторной работы, а также свободные источники в интернете.

Реализация функционала

Создали дополнительную функцию для генерации случайного ключа:

```
def gen_key(text):
    rn = np.random.randint(0, 255, len(text))
    key = [hex(e)[2:] for e in rn]
    return key
```

Реализована функция для определения вида шифротекста при двух известных экземплярах открытого текста. На вход подается 2 строки равно длины, после чего они переводятся в 16-ричную систему. С их помощью получаем два зашифрованных сообщения, при случайной генерации ключа.

```
def encrypt(p1, p2):
    print(f"P1: {p1}")
    print(f"P2: {p2}")

    hex_p1 = []
    hex_p2 = []

    for i in range(len(p1)):
```

```
        hex_p1.append(p1[i].encode("cp1251").hex())
        hex_p2.append(p2[i].encode("cp1251").hex())

    print("Hex P1: ", hex_p1)
    print("Hex P2: ", hex_p2)

    key = gen_key(p1)
    print("Hex key: ", key)

    hex_c1 = []
    hex_c2 = []

    for i in range(len(hex_p1)):
        hex_c1.append("{:02x}".format(int(key[i], 16) ^ int(hex_p1[i],
↪   16)))
        hex_c2.append("{:02x}".format(int(key[i], 16) ^ int(hex_p2[i],
↪   16)))

    print("Hex C1: ", hex_c1)
    print("Hex C2: ", hex_c2)

    c1 = bytearray.fromhex("".join(hex_c1)).decode("cp1251")
    c2 = bytearray.fromhex("".join(hex_c2)).decode("cp1251")

    print(f"C1: {c1}")
    print(f"C2: {c2}")

    return key, c1, c2
```

Далее, зная два шифротекста и один исходный текст, реализуем функцию для нахождения второго исходного текста без наличия ключа.

```
def decrypt(c1, c2, p1):
    print(f"C1: {c1}")
    print(f"C2: {c2}")
    print(f"P1: {p1}")

    hex_c1 = []
```

```
    hex_c2 = []
    hex_p1 = []

    for i in range(len(p1)):
        hex_c1.append(c1[i].encode("cp1251").hex())
        hex_c2.append(c2[i].encode("cp1251").hex())
        hex_p1.append(p1[i].encode("cp1251").hex())

    print("Hex C1: ", hex_c1)
    print("Hex C2: ", hex_c2)
    print("Hex P1: ", hex_p1)

    hex_p2 = []

    for i in range(len(p1)):
        hex_p2.append("{:02x}".format(int(hex_c1[i], 16) ^ int(hex_c2[i],
↪   16) ^ int(hex_p1[i], 16)))

    print("Hex P2: ", hex_p2)
    p2 = bytearray.fromhex("".join(hex_p2)).decode("cp1251")

    print(f"P2: {p2}")
    return p1, p2
```

Проверка функционала

Создали два текста равной длины.

```
p1 = "возможно будет так"
p2 = "или может быть так"
print(len(p1), len(p2))
```

```
18 18
```

Рис. 1: Исходные P_1 и P_2

Попробовали, используя два исходных текста, получить два шифротекста, при случайной генерации ключа, что у нас успешно получилось.

```
In [11]: key, c1, c2 = encrypt(p1, p2)

P1: возможно будет так
P2: или может быть так
Hex P1:  ['e2', 'ee', 'e7', 'ec', 'ee', 'e6', 'ed', 'ee', '20', 'e1', 'f3', 'e4', 'e5', 'f2', '20', 'f2', 'e
0', 'ea']
Hex P2:  ['e8', 'eb', 'e8', '20', 'ec', 'ee', 'e6', 'e5', 'f2', '20', 'e1', 'fb', 'f2', 'fc', '20', 'f2', 'e
0', 'ea']
Hex key:  ['bd', 'd7', '5c', '9', '5d', 'f9', '35', '3d', 'f', 'fe', 'b8', '49', '38', 'fe', 'a3', 'e7', 'a
4', '0']
Hex C1:  ['5f', '39', 'bb', 'e5', 'b3', '1f', 'd8', 'd3', '2f', '1f', '4b', 'ad', 'dd', '0c', '83', '15', '4
4', 'ea']
Hex C2:  ['55', '3c', 'b4', '29', 'b1', '17', 'd3', 'd8', 'fd', 'de', '59', 'b2', 'ca', '02', '83', '15', '4
4', 'ea']
C1: _9»ei□ШУ/□КЭѓ□Dк
C2: U<r)±□УШэЮYIK□ѓ□Dк
```

Рис. 2: Получение C_1 и C_2

Использовали C_1, C_2 и P_1 для получения P_2. Функция отрабатывает корректно.

```
In [14]: p1_new, p2_new = decrypt(c1, c2, p1)

C1: _9»ei□ШУ/□КЭѓ□Dк
C2: U<r)±□УШэЮYIK□ѓ□Dк
P1: возможно будет так
Hex C1:  ['5f', '39', 'bb', 'e5', 'b3', '1f', 'd8', 'd3', '2f', '1f', '4b', 'ad', 'dd', '0c', '83', '15', '4
4', 'ea']
Hex C2:  ['55', '3c', 'b4', '29', 'b1', '17', 'd3', 'd8', 'fd', 'de', '59', 'b2', 'ca', '02', '83', '15', '4
4', 'ea']
Hex P1:  ['e2', 'ee', 'e7', 'ec', 'ee', 'e6', 'ed', 'ee', '20', 'e1', 'f3', 'e4', 'e5', 'f2', '20', 'f2', 'e
0', 'ea']
Hex P2:  ['e8', 'eb', 'e8', '20', 'ec', 'ee', 'e6', 'e5', 'f2', '20', 'e1', 'fb', 'f2', 'fc', '20', 'f2', 'e
0', 'ea']
P2: или может быть так
```

Рис. 3: Получение P_2 через два шифротекста и P_1

Контрольные вопросы

1. Как, зная один из текстов (P_1 или P_2), определить другой, не зная при этом ключа?

   Для этого надо воспользоваться формулой:

   $$C_1 \oplus C_2 \oplus P_1 = P_1 \oplus P_2 \oplus P_1 = P_2,$$

   где $_1$ и $_2$ – шифротексты.

   Как видно, ключ в данной формуле не используется.

2. Что будет при повторном использовании ключа при шифровании текста?

   В таком случае мы получим исходное сообщение.

3. Как реализуется режим шифрования однократного гаммирования одним ключом двух открытых текстов?

   Он реализуется по следующей формуле:

   $$C_1 = P_1 \oplus KC_2 = P_2 \oplus K,$$

   где $_i$ – шифротексты, P_i – открытые тексты, K – единый ключ шифрования.

4. Перечислите недостатки шифрования одним ключом двух открытых текстов.

   Во-первых, имея на руках одно из сообщений в открытом виде и оба шифротекста, злоумышленник способен расшифровать каждое сообщение, не зная ключа.

   Во-вторых, зная шаблон сообщений, 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. Таким образом, применяя формулу из п. 1, с подстановкой вместо P_1 полученных на предыдущем шаге новых символов сообщения P_2 злоумышленник если не

прочитает оба сообщения, то значительно уменьшит пространство их поиска. Наконец, зная ключ, злоумышленник смоет расшифровать все сообщения, которые были закодированы при его помощи.

5. Перечислите преимущества шифрования одним ключом двух открытых текстов.

   Такой подход помогает упростить процесс шифрования и дешифровки. Также, при отправке сообщений между двумя компьютерами, удобнее пользоваться одним общим ключом для передаваемых данных.

Выводы

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