Лабораторная работа №8

Шифрование (кодирование) различных исходных текстов одним ключом

Доборщук В.В., НФИбд-01-18

18 декабря 2021

Цель работы

Цель работы

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

Выполнение лабораторной работы

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.

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
```

Реализация функционала

```python
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
```

Реализация функционала

```python
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
```

Рис. 2: Функция нахождения второго исходного текста

Проверка функционала

Создали два текста равной длины.

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

```
18 18
```

Рис. 3: Исходные P_1 и P_2

Проверка функционала

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

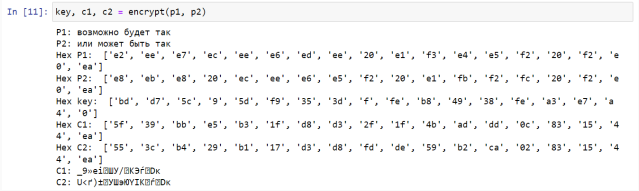

Рис. 4: Получение C_1 и C_2

Проверка функционала

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<ґ)±▯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. 5: Получение P_2 через два шифротекста и P_1

Заключение

Заключение

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