

Лабораторная работа №8

Дисциплина: информационная безопасность

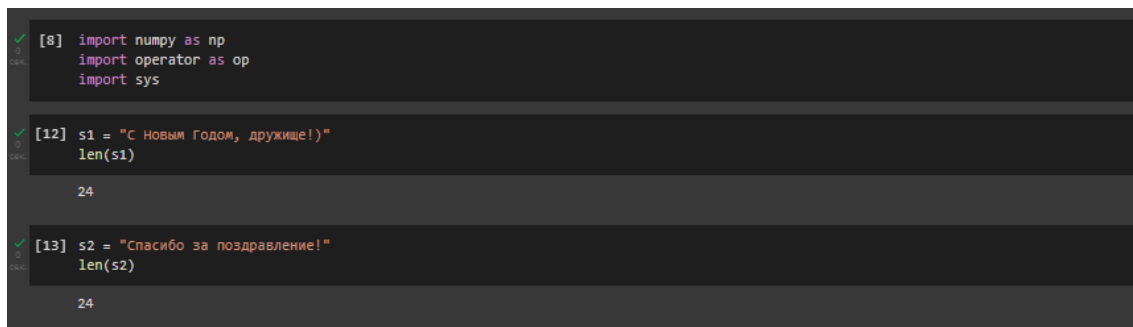
Студент: Подорога Виктор Александрович

Цель работы

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

Выполнение лабораторной работы

1. Импортируем необходимые библиотеки и введем исходные текстовые сообщения:



```
[8] import numpy as np
import operator as op
import sys

[12] s1 = "С Новым Годом, дружище!"
len(s1)

24

[13] s2 = "Спасибо за поздравление!"
len(s2)

24
```

Рис. 1. Импорт библиотек и исходные данные

2. Опишем функцию шифрования:

```

[14] # функция шифрования
# Получает на вход две символьные строки, которые затем переводятся в 16-ую систему.
# далее генерируется случайный ключ, при помощи которого определяются соответствующие шифротексты в 16-й системе.
# Затем шифротекст переводится в строковый формат. Функция возвращает ключ, оба шифротекста в 16-ой системе и строковом формате
def encryption(text1, text2):
    # Работа с первым текстом
    print("Открытый текст 1: ", text1)
    new_text1 = []
    for i in text1:
        new_text1.append(i.encode("cp1251").hex())
    print("\nОткрытый текст 1 в 16-ой системе: ", new_text1)

    # Работа со вторым текстом
    print("\nОткрытый текст 2: ", text2)
    new_text2 = []
    for i in text2:
        new_text2.append(i.encode("cp1251").hex())
    print("\nОткрытый текст 2 в 16-ой системе: ", new_text2)

    # Генерация ключа
    r = np.random.randint(0, 255, len(text1))
    key = [hex(i)[2:] for i in r]
    new_key = []
    for i in key:
        new_key.append(i.encode("cp1251").hex().upper())
    print("\nКлюч в 16-ой системе: ", key)

    # Получение зашифрованного сообщения из 1 текста
    xor_text1 = []
    for i in range(len(new_text1)):
        xor_text1.append("{:02x}".format(int(key[i], 16) ^ int(new_text1[i], 16)))
    print("\nШифротекст 1 в 16-ой системе: ", xor_text1)
    # Переведем зашифрованное сообщение 1 в строку
    en_text1 = bytearray.fromhex("".join(xor_text1)).decode("cp1251")
    print("\nШифротекст 1: ", en_text1)

    # Получение зашифрованного сообщения из 2 текста
    xor_text2 = []
    for i in range(len(new_text2)):
        xor_text2.append("{:02x}".format(int(key[i], 16) ^ int(new_text2[i], 16)))
    print("\nШифротекст 2 в 16-ой системе: ", xor_text2)
    # Переведем зашифрованное сообщение 2 в строку
    en_text2 = bytearray.fromhex("".join(xor_text2)).decode("cp1251")
    print("\nШифротекст 2: ", en_text2)

    return key, xor_text1, en_text1, xor_text2, en_text2

```

Рис. 2. Код функции шифрования

3. Выведем результат работы функции шифрования:

```

[15] k, t1, et1, t2, et2 = encryption(s1, s2)

Открытый текст 1: С Новым Годом, дружище!)
Открытый текст 1 в 16-ой системе: ['d1', '20', 'cd', 'ee', 'e2', 'fb', 'ec', '20', 'c3', 'ee', 'e4', 'ee', 'ec', '2c', '20', 'e4', 'f0', 'f3', 'e6', 'e8', 'f9', 'e5', '21', '29']
Открытый текст 2: спасибо за поздравление!
Открытый текст 2 в 16-ой системе: ['d1', 'ef', 'e0', 'f1', 'e8', 'e1', 'ee', '20', 'e7', 'e0', '20', 'ef', 'ee', 'e7', 'e4', 'f0', 'e0', 'e2', 'eb', 'e5', 'ed', 'e0', 'e5', '21']
Ключ в 16-ой системе: ['d4', '85', '2a', 'd8', '62', 'c6', '29', 'd5', '71', '30', 'aa', '31', 'e1', 'c5', '97', '83', 'cd', '20', '4c', '2e', 'e1', '1f', '79', '95']
Шифротекст 1 в 16-ой системе: ['05', 'a5', 'e7', '36', 'b0', '3d', 'c5', 'f5', 'b2', 'de', '4c', 'df', '0d', 'e9', 'b7', '67', '3d', 'd3', 'aa', 'c6', '18', 'fa', '58', 'bc']
й-g-УХВъхJ
Шифротекст 2 в 16-ой системе: ['05', '6a', 'ca', '29', '8a', '27', 'c7', 'f5', '96', 'd0', '8a', 'de', '0f', '22', '73', '73', '2d', 'c2', 'a7', 'cb', '0c', 'f7', '9c', 'b4']
Шифротекст 2: ВJc)A'3x-PM0W'ss-B5лчмг

```

Рис. 3. Результат работы функции шифрования

4. Опишем функцию дешифровки:

```
[16] # c1, c2 - шифротексты
# p1 - открытый текст сообщения
# функция расшифровки
# функция получает на вход два шифротекста и один открытый в строковом формате, затем переводит их в 16-ю систему.
# Затем применяя принцип одноразового гаммирования, находит вид второго открытого сообщения без использования ключа шифрования.
# Возвращает функция второе расшифрованное сообщение в строковом формате и 16-ой системе.

def decryption(c1, c2, p1):
    # Работа с первым шифротекстом
    print("Шифротекст 1: ", c1)
    new_c1 = []
    for i in c1:
        new_c1.append(i.encode("cp1251").hex())
    print("\nШифротекст 1 в 16-ой системе: ", new_c1)

    # Работа со вторым шифротекстом
    print("\nШифротекст 2: ", c2)
    new_c2 = []
    for i in c2:
        new_c2.append(i.encode("cp1251").hex())
    print("\nШифротекст 2 в 16-ой системе: ", new_c2)

    # Работа с открытым текстом
    print("\nОткрытый текст 1: ", p1)
    new_p1 = []
    for i in p1:
        new_p1.append(i.encode("cp1251").hex())
    print("\nОткрытый текст 1 в 16-ой системе: ", new_p1)

    print("\nНахожу второй открытый текст...")

    # Получение расшифрованного сообщения p2
    xor_tmp = []
    sp2 = []
    for i in range(len(p1)):
        xor_tmp.append("{:02x}".format(int(new_c1[i], 16) ^ int(new_c2[i], 16)))
        sp2.append("{:02x}".format(int(xor_tmp[i], 16) ^ int(new_p1[i], 16)))
    print("\nОткрытый текст 2 в 16-ой системе: ", sp2)

    # Переведем расшифрованное сообщение 2 в строку
    p2 = bytearray.fromhex("".join(sp2)).decode("cp1251")
    print("\nОткрытый текст 2: ", p2)
    return p2, sp2
```

Рис. 4. Код функции дешифровки

5. Выведем результат работы функции дешифровки:

```
[17] s3 = decryption(et1, et2, s1)

й-г-уежхъкј
Шифротекст 1 в 16-ой системе: ['65', 'a5', 'e7', '36', '80', '3d', 'c5', 'f5', 'b2', 'de', '4e', 'df', '0d', 'e9', 'b7', '67', '3d', 'd3', 'aa', 'c6', '18', 'fa', '58', 'bc']
Шифротекст 2: 0jK)A'Эх-РВМВ"ss-B5лчгг
Шифротекст 2 в 16-ой системе: ['05', '6a', 'ca', '29', '8a', '27', 'c7', 'f5', '96', 'd0', '8a', 'de', '0f', '22', '73', '73', '2d', 'c2', 'a7', 'cb', '0c', 'f7', '9c', 'b4']
Открытый текст 1: С Новым Годом, дружище!
Открытый текст 1 в 16-ой системе: ['d1', '20', 'cd', 'ee', 'e2', 'fb', 'ec', '20', 'c3', 'ee', 'e4', 'ee', 'ec', '2c', '20', 'e4', 'f0', 'f3', 'e5', 'e8', 'f9', 'e5', '21', '29']
Нахожу второй открытый текст...
Открытый текст 2 в 16-ой системе: ['d1', 'ef', 'e0', 'f1', 'e8', 'e1', 'ee', '20', 'e7', 'e0', '20', 'ef', 'ee', 'e7', 'e4', 'f0', 'e0', 'e2', 'eb', 'e5', 'ed', 'e8', 'e5', '21']
Открытый текст 2: Спасибо за поздравление!
```

Рис. 5. Результат работы функции дешифровки

Как, зная один из текстов (P1 или P2), определить другой, не зная при этом ключа?

Для этого надо воспользоваться формулой:

$C1 \oplus C2 \oplus P1 = P1 \oplus P2 \oplus P1 = P2$, где $C1$ и $C2$ – шифротексты. Как видно, ключ в данной формуле не используется.

Что будет при повторном использовании ключа при шифровании текста?

В таком случае мы получим исходное сообщение.

Как реализуется режим шифрования одноразового гаммирования одним ключом двух открытых текстов?

Он реализуется по следующей формуле:

$C1 = P1 \oplus K$, $C2 = P2 \oplus K$, где Ci – шифротексты, Pi – открытые тексты, K – единый ключ шифрования.

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 пункта 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 сможет расшифровать все сообщения, которые были закодированы при его помощи.

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.

Вывод

В ходе лабораторной работы я получил практические навыки применения режима однократного гаммирования на примере кодирования различных исходных текстов одним ключом.