Лабораторная Работа №7. Элементы криптографии. Однократное гаммирование

Основы информационной безопасности

Барсегян В.Л.

Российский университет дружбы народов им. Патриса Лумумбы, Москва, Россия

Докладчик

- Барсегян Вардан Левонович
- НПИбд-01-22
- Российский университет дружбы народов
- [1132222005@pfur.ru]
- https://github.com/VARdamn/oib

Вводная часть

Цели и задачи

Освоить на практике применение режима однократного гаммирования.

Выполнение лабораторной работы

Создаю функцию encrypt(), которая будет шифровать заданный текст с помощью гаммирования. Также можно подать на вход определенный ключ шифрования. Если ключа нет, то он генерируется рандомно. Сначала исходный текст и ключ шифрования преобразуются в 16-ную СС, затем, применяется операция XOR для каждого элемента ключа и текста. Полученный шифротекст декодируется из 16-ной СС и получается набор из символов.

```
def encrypt(text: str, key: list = None):
    '''
    Выводит шифротекст для заданного текста.
    Если ключа нет, то генерируется случайный ключ
    '''

    text_16 = [char.encode(encoding='cp1251').hex().upper() for char in text]
    if not key:
```

Результат работы функции encrypt()

Figure 1: Функция encrypt()

Далее, создаю функцию decrypt(), которая по заданному шифротексту выводит исходный текст. Также можно опционально задать ключ дешифровки, или же он будет сгенерирован автоматически. Функция преобразует шифротекст в 16-ную СС и применяет XOR для шифротекста и ключа

```
def decrypt(ciphertext: str, key: list = None):
    """
    ciphertext_16 = [char.encode('cp1251').hex().upper() for char in ciphertext]
    if not key:
        key = generate_key(length=len(ciphertext))

    print(f"Ключ шифрования:", ' '.join(str(s) for s in key))
    print(f"Исходный шифротекст:", ciphertext)

    decrypted_text = []
    for i in range(len(ciphertext)):
```

Результат работы функции decrypt() с тем же ключом, что и в шифровании

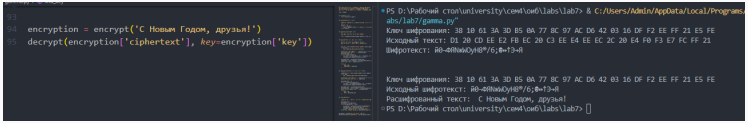

Figure 2: decrypt() с тем же ключом

Результат работы функции decrypt() со случайным ключом

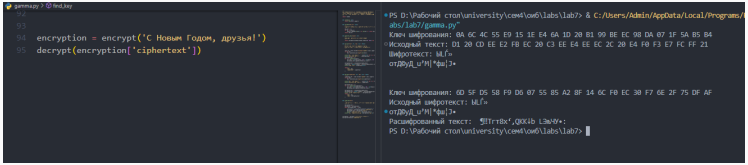

Figure 3: decrypt() со случайным ключом

Функция find_key() вызывает функцию decrypt() до тех пор, пока расшифрованный и исходный текст не совпадут, т.е пытается подобрать ключ для расшифровки

```
def find_key(text):
    '''
    Подбирает ключ, с помощью которого сообщение было зашифровано
    '''
    decrypted_text = ''
    encryption = encrypt(text)
    while decrypted_text != text:
        decryption = decrypt(encryption['ciphertext'])
        decrypted_text = decryption['text']
        print(f'Полученный текст: {decrypted_text}')
    print(f"Ключ успешно подобран! {decryption['key']}")
```

Вывод

Я узнал о схеме однократного гаммирования и научился ее применять на практике