Отчёт по лабораторной работе №7

Дисциплина: Основы информационной безопасности

Барсегян Вардан Левонович НПИбд-01-22

Содержание

# 1 Цель работы

Освоить на практике применение режима однократного гаммирования.

# 2 Выполнение лабораторной работы

1. 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:  
 key = generate\_key(length=len(text))  
  
 print(f"Ключ шифрования:", ' '.join(str(s) for s in key))  
 print(f"Исходный текст:", ' '.join(text\_16))  
 encrypted\_text = []  
 for i in range(len(text)):  
 xor\_char = int(text\_16[i], 16) ^ int(key[i], 16)  
 encrypted\_text.append(int2hex(xor\_char))  
  
 encrypted\_text = validate(encrypted\_text)  
 ciphertext = bytes.fromhex(''.join(encrypted\_text)).decode('cp1251')  
 print(f'Шифротекст: {ciphertext}\n\n')  
  
 return {  
 'key': key,  
 'ciphertext': ciphertext  
 }

1. Результат работы функции encrypt() (рис. 1)



Рис. 1: Функция encrypt()

1. 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)):  
 xor\_char = int(ciphertext\_16[i], 16) ^ int(key[i], 16)  
 decrypted\_text.append(int2hex(xor\_char))  
  
 decrypted\_text = validate(decrypted\_text)  
 decrypted\_text = bytes.fromhex(''.join(decrypted\_text)).decode('cp1251')  
 print('Расшифрованный текст: ', decrypted\_text)  
 return {  
 'key': key,  
 'text': decrypted\_text  
 }

1. Результат работы функции decrypt() с тем же ключом, что и в шифровании (рис. 2)

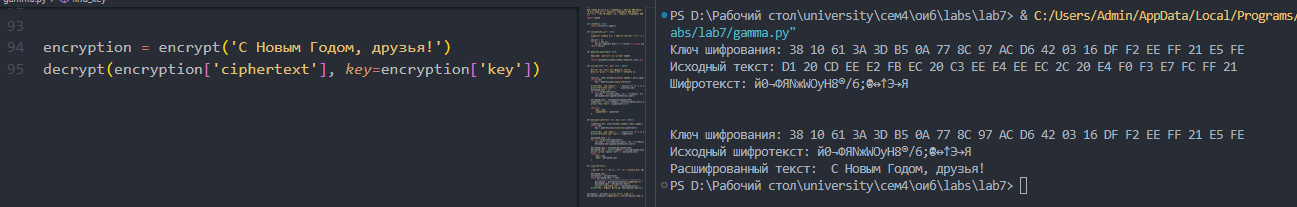


Рис. 2: decrypt() с тем же ключом

1. Результат работы функции decrypt() со случайным ключом (рис. 3)

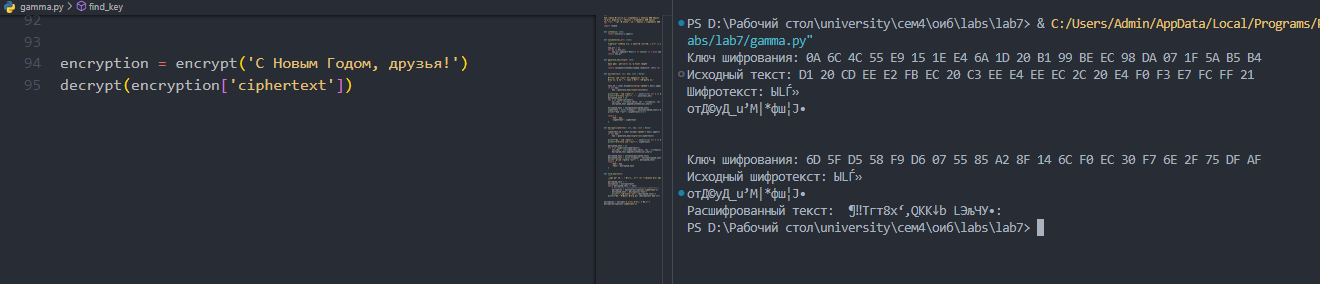


Рис. 3: decrypt() со случайным ключом

1. 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']}")

# 3 Контрольные вопросы

1. Поясните смысл однократного гаммирования

Гаммирование – выполнение операции XOR между элементами гаммы и элементами подлежащего сокрытию текста. Если в методе шифрования используется однократная вероятностная гамма (однократное гаммирование) той же длины, что и подлежащий сокрытию текст, то текст нельзя раскрыть. Даже при раскрытии части последовательности гаммы нельзя получить информацию о всём скрываемом тексте.

1. Перечислите недостатки однократного гаммирования

Шифр абсолютно стойкий только тогда, когда ключ сгенерирован из случайной двоичной последовательности

1. Перечислите преимущества однократного гаммирования

Это симметричный способ шифрования; алгоритм не дает никакой информации об исходном сообщении; шифрование/дешифрование может быть применено одной программой (в обратном порядке)

1. Почему длина открытого текста должна совпадать с длиной ключа?

Если ключ длиннее, то часть текста (разница между длиной ключа и открытого текста) не будет зашифрована. Если же ключ короче, то однозначное дешифрование невозможно

1. Какая операция используется в режиме однократного гаммирования, назовите её особенности?

операция XOR (сложение по модулю 2), ее особенность - симметричность, т.к. если ее применить 2 раза, то вернется исходное значение

1. Как по открытому тексту и ключу получить шифротекст?

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.

1. Как по открытому тексту и шифротексту получить ключ?

Применить операцию XOR для каждого элемента шифротекста и открытого текста: key[i] = crypted[i] XOR text[i]

1. В чем заключаются необходимые и достаточные условия абсолютной стойкости шифра?

Необходимые и достаточные условия абсолютной стойкости шифра:

* полная случайность ключа
* равенство длин ключа и открытого текста
* однократное использование ключа

# 4 Выводы

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