

Отчёт по лабораторной работе №8

Дисциплина: Основы информационной безопасности

Мишина Анастасия Алексеевна

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Контрольные вопросы	9
4	Выводы	10
	Список литературы	11

Список иллюстраций

2.1	Исходный код программы	7
2.2	Работа программы	8

Список таблиц

1 Цель работы

Освоить на практике применение режима однократного гаммирования на примере кодирования различных исходных текстов одним ключом [1].

2 Выполнение лабораторной работы

Рассмотрим файл `main2.py`, в котором написана наша программа (рис. 2.1). Создаем функцию `encrypt()`, которая будет шифровать заданный текст с помощью однократного гаммирования. Также есть возможность подавать на вход определенный ключ шифрования. Если ключа нет, то он генерируется случайно. Сначала исходный текст и ключ шифрования преобразуются в 16-ую систему счисления, затем, применяется операция XOR для каждого элемента ключа и текста. Полученный шифротекст декодируется из 16-ой СС и получается набор из символов. В функции `generate_key()` происходит генерация ключа (если он не задан) из `ascii`-символов и цифр.

Работа программы: сначала создается случайный ключ и с этим ключом шифруются два текста `p1` и `p2` (в переменные `s1` и `s2`). Далее шифротекст `s1` шифруется по ключу `s2`. Полученный шифротекст `s1_s2` шифруется по ключу открытого текста, в результате мы получаем второй открытый текст, ранее неизвестный.

```

7 import string
8
9
10 def generate_key(length: int):
11     '''
12     Генерация случайного ключа длины length
13     '''
14     return random.sample(string.ascii_letters + string.digits, length)
15
16
17 def encrypt(text: str, key: list = None):
18     '''
19     Выводит шифротекст для заданного текста.
20     Если ключа нет, то генерируется случайный ключ
21     '''
22     if not key:
23         key = generate_key(length=len(text))
24
25     text_16 = [ord(char) for char in text]
26     key = [ord(el) for el in key]
27
28     print(f"Ключ шифрования:", ' '.join(str(s) for s in key))
29     print(f"Исходный текст:", text)
30
31     encrypted_text = []
32     for i in range(len(text)):
33         encrypted_text.append(text_16[i] ^ key[i])
34
35     ciphertext = ''.join([chr(i) for i in encrypted_text])
36     print(f'Шифротекст: {ciphertext}\n\n')
37
38     return ciphertext
39
40
41 p1 = 'НаВашисходящийот1204'
42 p2 = 'ВСеверныйфилиалБанка'
43 key = generate_key(20)
44
45 c1 = encrypt(p1, key=key)
46 c2 = encrypt(p2, key=key)
47
48 c1_c2 = encrypt(c1, key=c2)
49
50 encrypt(c1_c2, p1)
51 encrypt(c1_c2, p2)

```

Рис. 2.1: Исходный код программы

Запустим файл main2.py и посмотрим результат работы программы (рис. 2.2).

```

[aamishina@aamishina Documents]$ python main2.py
Ключ шифрования: 57 106 69 83 75 48 51 76 122 66 85 103 89 78 97 101 107 70 108 109
Исходный текст: НаВашисходящийот1204
Шифротекст: Ф%тЪГ'30&фVк00uуЧZt\Y

Ключ шифрования: 57 106 69 83 75 48 51 76 122 66 85 103 89 78 97 101 107 70 108 109
Исходный текст: ВСеверныйфилиалБанка
Шифротекст: ЫV00VYIyIж00Vh0iY

Ключ шифрования: 1067 1099 1136 1121 1150 1136 1038 1031 1091 1030 1133 1116 1121 1150 1114 1140 1115 1147 1110 1117
Исходный текст: Ф%тЪГ'30&фVк00uуЧZt\Y
Шифротекст: '}x|pwr SEU&E

Ключ шифрования: 1053 1072 1042 1072 1096 1080 1089 1093 1086 1076 1103 1097 1080 1081 1086 1090 49 50 48 52
Исходный текст: '}x|pwr SEU&E
Шифротекст: ВСеверныйфилиалБанка

Ключ шифрования: 1042 1057 1077 1074 1077 1088 1085 1099 1081 1092 1080 1083 1080 1072 1083 1041 1072 1085 1082 1072
Исходный текст: '}x|pwr SEU&E
Шифротекст: НаВашисходящийот1204

[aamishina@aamishina Documents]$ █

```

Рис. 2.2: Работа программы

3 Контрольные вопросы

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

Нужно применить XOR для двух шифротекстов, а к полученному результату применить XOR с ключом, равным известному открытому тексту. Тогда результатом будет второй открытый текст

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

Шифрование будет небезопасным, т.к. с помощью шифротекстов и одного открытого текста можно дешифровать другой текст

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

Каждый текст шифруется однократным гаммированием отдельно с использованием этого ключа

4. Перечислите недостатки шифрования одним ключом двух открытых текстов

Главный недостаток - можно дешифровать открытый текст без знания ключа

5. Перечислите преимущества шифрования одним ключом двух открытых текстов.

Т.к. ключей используется меньше, то тратится меньше памяти на хранение и передачу ключей

4 Выводы

В ходе выполнения данной лабораторной работы я освоила на практике применение режима однократного гаммирования на примере кодирования различных исходных текстов одним ключом.

Список литературы

1. Кулябов Д.С., Королькова А.В., Геворкян М.Н. Информационная безопасность компьютерных сетей. Лабораторные работы, учебное пособие. Москва: РУДН, 2015. 64 с.