Отчет по лабораторной работе 7

Основы информационной безопасности

Нджову Нелиа

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

Освоить на практике применение режима однократного гаммирования

2 Задание

Нужно подобрать ключ, чтобы получить сообщение «С Новым Годом, друзья!». Требуется разработать приложение, позволяющее шифровать и дешифровать данные в режиме однократного гаммирования. Приложение должно:

1. Определить вид шифротекста при известном ключе и известном открытом тексте.
2. Определить ключ, с помощью которого шифротекст может быть преобразован в некоторый фрагмент текста, представляющий собой один из возможных вариантов прочтения открытого текста

3 Теоретическое введение

Гаммирование (схема Вернама) — это надёжный и простой способ шифрования. Он основан на **однократном наложении** специальной последовательности (называемой **гаммой**) на открытые данные. Это наложение выполняется с помощью операции **XOR (по модулю 2)** между каждым символом открытого текста и соответствующим символом гаммы (ключа).

Операция XOR работает так:

- 0 ⊠ 0 = 0
- 0 ⊠ 1 = 1
- 1 ⊠ 0 = 1
- 1 ⊠ 1 = 0

Шифрование и расшифровка производятся одинаково: `Ci = Pi ⊠ Ki` где

- `Ci` — зашифрованный символ,
- `Pi` — символ открытого текста,
- `Ki` — символ ключа.

Если известен шифротекст и открытый текст, ключ можно найти: `Ki = Ci ⊠ Pi`

Для **абсолютной стойкости** такого шифра нужно:

- чтобы ключ был **полностью случайным**,

- чтобы его длина была равна длине текста,
- и чтобы он использовался **только один раз**.

Пример:

Сообщение Центра (в hex):

```
D8 F2 E8 F0 EB E8 F6 20 2D 20 C2 FB 20 C3 E5 F0 EE E9 21 21
```

(= *Штирлиц – Вы Герой!!*)

Ключ Центра:

```
05 0C 17 7F 0E 4E 37 D2 94 10 09 2E 22 57 FF C8 0B B2 70 54
```

Шифротекст у Мюллера:

```
DD FE FF 8F E5 A6 C1 F2 B9 30 CB D5 02 94 1A 38 E5 5B 51 75
```

Если злоумышленник попробует другой ключ, он получит другой осмысленный текст, например:

Штирлиц - Вы Болван!

что показывает, что множество возможных ключей может дать множество разных сообщений одинаковой длины.

4 Выполнение лабораторной работы

Я выполнала лабораторную работа на языке программирования Python, листинг программы и результаты выполнения приведены в отчете.

Требуется разработать программу, позволяющее шифровать и дешифровать данные в режиме однократного гаммирования. Начнем с создания функции для генерации случайного ключа(рис.1).

```
import random
import string

def generate_key_hex(text):
    key = ''
    for i in range(len(text)):
        key += random.choice(string.ascii_letters + string.digits)
    return key
```

Рис. 4.1: Функция генерации ключа

Необходимо определить вид шифротекста при известном ключе и известном открытом тексте. Так как операция исключающего или отменяет сама себя, делаю одну функцю и для шифрования и для дешифрования текста(рис.2).

```
def en_de_crypt(text, key):
    new_text = ''
    for i in range(len(text)): #проход по каждому символу в тексте
        new_text += chr(ord(text[i]) ^ ord(key[i % len(key)]))
    return new_text
```

Рис. 4.2: Функция для шифрования текста

Нужно определить ключ, с помощью которого шифротекст может быть преобразован в некоторый фрагмент текста, представляющий собой один из возможных

вариантов прочтения открытого текста. Для этого создаю функцию для нахождения возможных ключей для фрагмента текста(рис.3).

```
def find_possible_key(text, fragment):
    possible_keys = []
    for i in range(len(text) - len(fragment) + 1):
        possible_key = ""
        for j in range(len(fragment)):
            possible_key += chr(ord(text[i + j]) ^ ord(fragment[j]))
        possible_keys.append(possible_key)
    return possible_keys
```

Рис. 4.3: Подбор возможных ключей для фрагмента

Проверка работы всех функций. Шифрование и дешифрование происходит верно, как и нахождение ключей, с помощью которых можно расшифровать верно только кусок текста(рис.4).

```
t = 'С Новым Годом, друзья!'
key = generate_key_hex(t)
en_t = en_de_crypt(t, key)
de_t = en_de_crypt(en_t, key)
keys_t_f = find_possible_key(en_t, 'С Новым')
fragment = "С Новым"
print('Открытый текст: ', t, "\nКлюч: ", key, '\nШифротекст: ', en_t, '\nИсходный текст: ', de

print('Возможные ключи: ', keys_t_f)
print('Расшифрованный фрагмент: ', en_de_crypt(en_t, keys_t_f[0]))
```

```
Открытый текст:  С Новым Годом, друзья!
Ключ:  v99hWo919aN5sYDPpqjTnE
Шифротекст:  ЇФіѥФSЪџОЋяudЄавйИСd
Исходный текст:  С Новым Годом, друзья!
Возможные ключи:  ['v99hWo9', 'иЄК[\x16NЭ', '\x05Ўx\x1a7њ\x16', 'wx9;Уас', 'DЄ\x18Я\x18\x14
F', '\x05XЌ\x14m17', '$17aH@s', 'аЊBD9\x04щ', '\x0bўg5}oj', '~њ\x16qчЯX', '[ЫRыі/\x0c', '*ӟЏ
њV{\x0e', 'nUџZ\x02ya', 'єDy\x0e\x00\x16$', 'хф-\x0coS\x1d', 'EA/c*jj']
Расшифрованный фрагмент:  С НовымgГАВќPLЙЉњЂvИ
```

Рис. 4.4: Результат работы программы

Листинг программы 1:

```
import random
import string


def generate_key_hex(text):
    key = ''
    for i in range(len(text)):
```

```
        key += random.choice(string.ascii_letters + string.digits)
    return key

def en_de_crypt(text, key):
    new_text = ''
    for i in range(len(text)): #проход по каждому символу в тексте
        new_text += chr(ord(text[i]) ^ ord(key[i % len(key)]))
    return new_text

def find_possible_key(text, fragment):
    possible_keys = []
    for i in range(len(text) - len(fragment) + 1):
        possible_key = ""
        for j in range(len(fragment)):
            possible_key += chr(ord(text[i + j]) ^ ord(fragment[j]))
        possible_keys.append(possible_key)
    return possible_keys

t = 'С Новым Годом, друзья!'
key = generate_key_hex(t)
en_t = en_de_crypt(t, key)
de_t = en_de_crypt(en_t, key)
keys_t_f = find_possible_key(en_t, 'С Новым')
fragment = "С Новым"
print('Открытый текст: ', t, "\nКлюч: ", key, '\nШифротекст: ', en_t, '\nИсходный

print('Возможные ключи: ', keys_t_f)
print('Расшифрованный фрагмент: ', en_de_crypt(en_t, keys_t_f[0]))
```

5 Ответы на контрольные вопросы

1. Поясните смысл однократного гаммирования. - Однократное гаммирование - это метод шифрования, при котором каждый символ открытого текста гаммируется с соответствующим символом ключа только один раз.
2. Перечислите недостатки однократного гаммирования. - Недостатки однократного гаммирования:

- Уязвимость к частотному анализу из-за сохранения частоты символов открытого текста в шифротексте.
- Необходимость использования одноразового ключа, который должен быть длиннее самого открытого текста.
- Нет возможности использовать один ключ для шифрования разных сообщений.

3. Перечислите преимущества однократного гаммирования. - Преимущества однократного гаммирования:

- Высокая стойкость при правильном использовании случайного ключа.
- Простота реализации алгоритма.
- Возможность использования случайного ключа.

4. Почему длина открытого текста должна совпадать с длиной ключа? - Длина открытого текста должна совпадать с длиной ключа, чтобы каждый символ открытого текста гаммировался с соответствующим символом ключа.

5. Какая операция используется в режиме однократного гаммирования, назовите её особенности? - В режиме однократного гаммирования используется операция XOR (исключающее ИЛИ), которая объединяет двоичные значения символов открытого текста и ключа для получения шифротекста. Особенность XOR - если один из битов равен 1, то результат будет 1, иначе 0.

6. Как по открытому тексту и ключу получить шифротекст? - Для получения шифротекста по открытому тексту и ключу каждый символ открытого текста гаммируется с соответствующим символом ключа с помощью операции XOR.

7. Как по открытому тексту и шифротексту получить ключ? - По открытому тексту и шифротексту невозможно восстановить действительный ключ, так как для этого нужна информация о каждом символе ключа.

8. В чем заключаются необходимые и достаточные условия абсолютной стойкости шифра - Необходимые и достаточные условия абсолютной стойкости шифра:

- Ключи должны быть случайными и использоваться только один раз.
- Длина ключа должна быть не менее длины самого открытого текста.
- Ключи должны быть храниться и передаваться безопасным способом.

6 Выводы

В ходе выполнения данной лабораторной работы мной было освоено на практике применение режима однократного гаммирования.

Список литературы

Кулябов Д. С. Г.М.Н. Королькова А. В. Лабораторная работа № 7. Элементы криптографии. Однократное гаммирование [Электронный ресурс]. 2023. URL: https://esystem.rudn.ru/pluginfile.php/2293722/mod_resource/content/2/007-lab_crypto-gamma.pdf.