

Лабораторная работа №8: Презентация.

Элементы криптографии. Шифрование (кодирование) различных исходных текстов одним ключом.

Коне Сирики. Группа - НФИбд-01-20.¹

20 октября, 2023, Москва, Россия

¹Российский Университет Дружбы Народов

Цели и задачи работы

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

1. Рассмотреть особенности и особенности кодирования однократного гаммирования с использованием одного ключа.
2. Создать код который будет показывать принцип работы нескольких шифротекстов с одним ключом и его взлом.
3. изучить способы взлома и декодирование шифротекста без ключа.

Указание к работе

Пример: Исходные данные - две телеграммы Центра: P1 = НаВашисходящийот1204 P2 = ВСеверныйфилиалБанка Ключ Центра длиной 20 байт: K = 05 0C 17 7F 0E 4E 37 D2 94 10 09 2E 22 57 FF C8 0B B2 70 54 Режим шифрования однократного гаммирования одним ключом двух видов открытого текста реализуется в соответствии со схемой (смотреть лабораторную).

Процесс выполнения лабораторной работы

Два текста кодируются одним ключом (однократное гаммирование). Требуется не зная ключа и не стремясь его определить, прочесть оба текста. Необходимо разработать приложение, позволяющее шифровать и дешифровать тексты P_1 и P_2 в режиме однократного гаммирования. Приложение должно определить вид шифротекстов C_1 и C_2 обоих текстов P_1 и P_2 при известном ключе ; Необходимо определить и выразить аналитически способ, при котором злоумышленник может прочесть оба текста, не зная ключа и не стремясь его определить.


```
import random
import string

def keyCreate(s, alf):
    k = ''.join(random.choice(alf) for i in range(s))
    return k

def Hex_coder(cod):
    return ' '.join(hex(ord(i))[2:] for i in cod)
```

```

def string_coder(text, k, iter_numb):
    if iter_numb == 1:
        return ''.join(chr(ord(c) ^ ord(k)) for c, k in zip(text, k))
    else:
        return [''.join(chr(ord(c) ^ ord(k)) for c, k in zip(t, k)) for t in

def find_Key(cypher, texts, s):
    possible_keys = []
    for f in range(len(texts)):
        for i in range(len(cypher[f]) - s + 1):
            key = [chr(ord(c) ^ ord(k)) for c, k in zip(cypher[f][i:i + s], t
            intact_plaintext = string_coder(cypher[f], key, 1)
            if texts[f] in intact_plaintext:
                possible_keys.append(''.join(key))
    return possible_keys

```

```
P1 = input("Текст P1: ")
P2 = input("Текст P2: ")
if len(P1) != len(P2):
    exit(0)
size, char_set = len(P1), string.ascii_lowercase+string.digits
C1, C2 = string_coder([P1, P2], keyCreate(size, char_set), 2)

print(f"Зашифрованный сообщения P1: {C1} | в 16 бит {Hex_coder(C1)}",
      f"Зашифрованный сообщения P2: {C2} | в 16 бит {Hex_coder(C2)}", sep="\n")

possible_keys = find_Key([C1, C2], [P1, P2], size)
print("Возможные ключи для шифротекста:", possible_keys)

D1, D2 = string_coder([C1, C2], possible_keys[-1], 2)
print("Расшифрованный текст:", f"\nC1 => {D1}\nC2 => {D2}")
```

Результаты

```
возможные ключи для шифротекста: [ 'bodt', 'bodt' ]
PS C:\Users\KONE> python -u "C:\Users\KONE\AppData\Local\Temp\tempCodeRunnerFile.python"
Текст P1: time
Текст P2: goal
Зашифрованный сообщения P1: 𐀀 𐀁 | в 16 бит 16 6 9 11
Зашифрованный сообщения P2: 𐀂𐀃 | в 16 бит 5 0 5 18
Возможные ключи для шифротекста: ['bodt', 'bodt']
Расшифрованный текст:
C1 => time
C2 => goal
bodt
PS C:\Users\KONE> 
```

Рис. 1: Результат попытки 1

```
PS C:\Users\KONE> python -u "C:\Users\KONE\AppData\Local\Temp\tempCodeRunnerFile.python"
Текст P1: Rest post
Текст P2: Recols up
Зашифрованный сообщения P1: 6Ue@L в 16 бит 36 55 1b 1d 51 2 e 1c 1b
Зашифрованный сообщения P2: 6U
4e@A4v | в 16 бит 36 55 b 6 1d 1 41 1a 1f
Возможные ключи для шифротекста: ['d0hiqraoo', 'd0hiqraoo']
Расшифрованный текст:
C1 => Rest post
C2 => Recols up
d0hiqraoo
PS C:\Users\KONE> █
```

Рис. 2: Результат попытка 2

Контрольные вопросы

1. Как, зная один из текстов (P_1 или P_2), определить другой, не зная при этом ключа? Ответ: Это возможно сделать только в том случае если текст P_1 и P_2 одной длины и имеют общий ключ.
2. Что будет при повторном использовании ключа при шифровании текста? Ответ: Из-за одинаковости способа кодирования и декодирования после повторного использования слова и ключа даст нам шифротекст.

3. Как реализуется режим шифрования однократного гаммирования одним ключом двух открытых текстов? Ответ: Фактически следуя схеме 8.1 и принципу “шифра XOR” мы просто имеем два параллельных кодирования и декодирования с использованием одного ключа.
4. Перечислите недостатки шифрования одним ключом двух открытых текстов. Ответ: Если вспомнить требования для абсолютной стойкости шифра рассмотренных в предыдущей лабораторной то можно сразу понять по первому пункту что если ключ не будет случайным и каждый раз новым для каждой строки то найдя пересечения или аналоги в шифротекстах можно определить одинаковые символы что может пошатнуть защиту текста даже если у вас нет ни одного исходного кода, а если и есть то определить другие слова легко.

5. Перечислите преимущества шифрования одним ключом двух открытых текстов. Ответ:
На самом деле они есть, но они сомнительны: требуется передать один ключ что
сделать проще и быстрее, при передаче большого количества шифротекста нет шанса
запутаться в их порядке сочетания с ключами.

Выводы по проделанной работе

Освоил на практике применение режима однократного гаммирования и возможных способах взлома при отсутствие ключа и наличие исходных текстов и шифротекстов.