
Front matter

title: "Лабораторная работа №8"
subtitle: "Отчёт по лабораторной работе"
author: "Зайцева Анна Дмитриевна, НПМбд-02-21"

Generic options

lang: ru-RU

Bibliography

bibliography: bib/cite.bib
csl: pandoc/csl/gost-r-7-0-5-2008-numeric.csl

Pdf output format

toc: true # Table of contents
toc-depth: 2
lof: true # List of figures
lot: true # List of tables
fontsize: 12pt
linestretch: 1.5
papersize: a4
documentclass: scrreprt

Fonts

mainfont: PT Serif
romanfont: PT Serif
sansfont: PT Sans
monofont: PT Mono
mainfontoptions: Ligatures=TeX
romanfontoptions: Ligatures=TeX
sansfontoptions: Ligatures=TeX,Scale=MatchLowercase
monofontoptions: Scale=MatchLowercase,Scale=0.9

Pandoc-crossref LaTeX customization

figureTitle: "Рис."
tableTitle: "Таблица"
listingTitle: "Листинг"
lofTitle: "Список иллюстраций"
lotTitle: "Список таблиц"
lolTitle: "Листинги"

Misc options

indent: true
header-includes:

- \usepackage{indentfirst}
- \usepackage{float} # keep figures where there are in the text

- \floatplacement{figure}{H} # keep figures where there are in the text

---

Цель работы

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

Задание

Два текста кодируются одним ключом (однократное гаммирование).
Требуется не зная ключа и не стремясь его определить, прочитать 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 прочитать оба текста, не
зная ключа и не стремясь его определить.открытого текста

Теоретическое введение

Исходные данные.

Две телеграммы Центра:

P_1 = НаВашисходящийот1204

P_2 = ВСеверныйфилиалБанка

Ключ Центра длиной 20 байт:
K = 05 0C 17 7F 0E 4E 37 D2 94 10 09 2E 22 57 FF C8 OB B2 70 54

Шифротексты обеих телеграмм можно получить по формулам режима
однократного гаммирования:

$C_1 = P_1 ⊕ K$,

$C_2 = P_2 ⊕ K$. (8.1)

Открытый текст можно найти, зная шифротекст двух телеграмм, зашифрованных одним ключом. Для это оба равенства (8.1) складываются по модулю 2. Тогда с учётом свойства операции XOR

$$1 ⊕ 1 = 0, 1 ⊕ 0 = 1 (8.2)$$

получаем:

$$C_1 ⊕ C_2 = P_1 ⊕ K ⊕ P_2 ⊕ K = P1 ⊕ P_2.$$

Предположим, что одна из телеграмм является шаблоном — т.е. имеет текст фиксированный формат, в который вписываются значения полей.
Допустим, что злоумышленнику этот формат известен. Тогда он получает
достаточно много пар $C_1 ⊕ C_2$ (известен вид обеих шифровок). Тогда зная
P_1 и учитывая (8.2), имеем:

$C_1 ⊕ C_2 ⊕ P_1 = P_1 ⊕ P_2 ⊕ P_1 = P_2. (8.3)$

Таким образом, злоумышленник получает возможность определить те символы сообщения P_2, которые находятся на позициях известного шаблона сообщения P_1. В соответствии с логикой сообщения P_2, злоумышленник имеет реальный шанс узнать ещё некоторое количество символов сообщения P_2. Затем вновь используется (8.3) с подстановкой вместо P1 полученных на предыдущем шаге новых символов сообщения P_2. И так далее. Действуя подобным образом, злоумышленник даже если не прочитает оба сообщения, то значительно уменьшит пространство их поиска.

Выполнение лабораторной работы

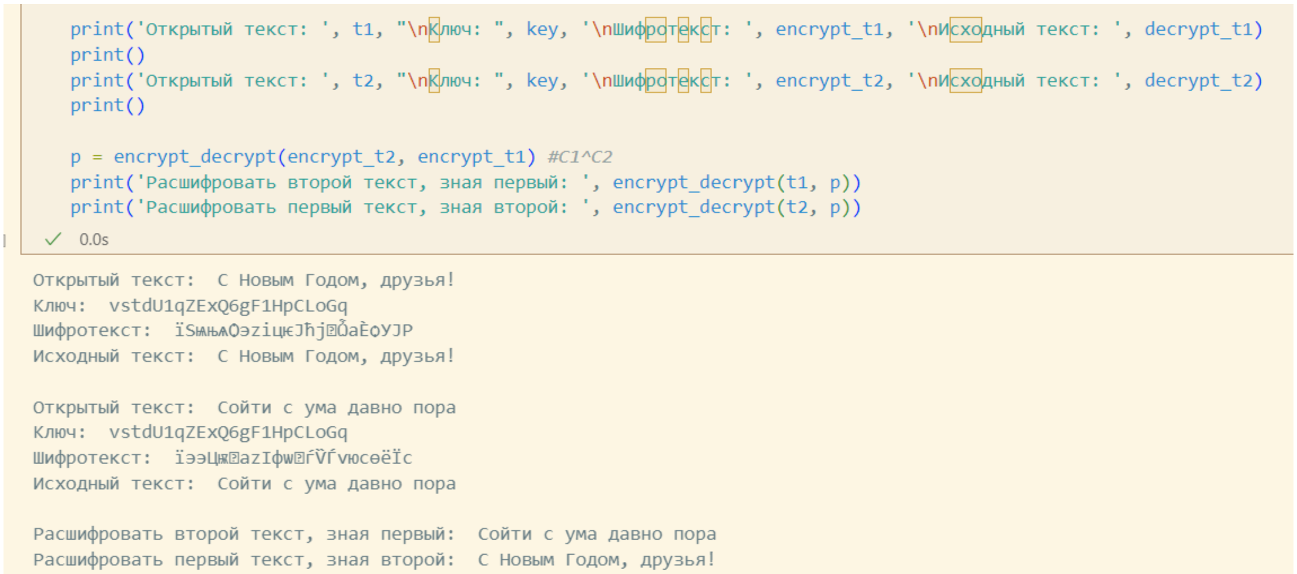
1) Я выполнила лабораторную работа на языке программирования Python, используя функции, написанные в своей предыдущей работе.

Сперва я использовала функцию для генерации случайного ключа, а потом зашифровала с его помощью два разных текста (Рис. [-@fig:001]):

```
import random
import string

def hex_key_generator(text):
    key = ''
    for i in range(len(text)):
        key += random.choice(string.ascii_letters + string.digits) # generation of a number for each character in the text
    return key

def encrypt_decrypt(text, key):
    new_text = ''
    for i in range(len(text)):
        new_text += chr(ord(text[i]) ^ ord(key[i % len(key)]))
    return new_text

t1 = 'С Новым Годом, друзья!'
key = hex_key_generator(t1)
encrypt_t1 = encrypt_decrypt(t1, key)
decrypt_t1 = encrypt_decrypt(encrypt_t1, key)

t2 = 'Сойти с ума давно пора'
encrypt_t2 = encrypt_decrypt(t2, key)
decrypt_t2 = encrypt_decrypt(encrypt_t2, key)
```

{ #fig:001 width=70% }

2) Расшифровала оба текста сначала с помощью одного ключа, затем предположила, что мне неизвестен ключ, но известен один из текстов, и уже расшифровала второй, зная шифротексты и первый текст (Рис. [-@fig:002]):



{ #fig:002 width=70% }

3) Листинг программы:

```
import random
import string

def hex_key_generator(text):
    key = ''
    for i in range(len(text)):
        key += random.choice(string.ascii_letters + string.digits) # generation of a
number for each character in the text
    return key

def encrypt_decrypt(text, key):
    new_text = ''
    for i in range(len(text)):
        new_text += chr(ord(text[i]) ^ ord(key[i % len(key)]))
    return new_text

t1 = 'С Новым Годом, друзья!'
key = hex_key_generator(t1)
encrypt_t1 = encrypt_decrypt(t1, key)
decrypt_t1 = encrypt_decrypt(encrypt_t1, key)

t2 = 'Сойти с ума давно пора'
encrypt_t2 = encrypt_decrypt(t2, key)
decrypt_t2 = encrypt_decrypt(encrypt_t2, key)

print('Открытый текст: ', t1, "\nКлюч: ", key, '\nШифротекст: ', encrypt_t1, '\nИсходный
текст: ', decrypt_t1)
print()
print('Открытый текст: ', t2, "\nКлюч: ", key, '\nШифротекст: ', encrypt_t2, '\nИсходный
текст: ', decrypt_t2)
print()

p = encrypt_decrypt(encrypt_t2, encrypt_t1) #C1^C2
print('Расшифровать второй текст, зная первый: ', encrypt_decrypt(t1, p))
```

```
print('Расшифровать первый текст, зная второй: ', encrypt_decrypt(t2, p))
```

Ответы на контрольные вопросы

1. Для определения другого текста (P_2) можно просто взять зашифрованные тексты $C_1 ⊕ C_2$, далее применить XOR к ним и к известному тексту: $C_1 ⊕ C_2 ⊕ P_1 = P_2$.
2. При повторном использовании ключа мы получим дешифрованный текст.
3. Режим шифрования однократного гаммирования одним ключом двух открытых текстов осуществляется путем XOR-ирования каждого бита первого текста с соответствующим битом ключа или второго текста.
4. Недостатки шифрования одним ключом двух открытых текстов включают возможность раскрытия ключа или текстов при известном открытом тексте.
5. Преимущества шифрования одним ключом двух открытых текстов включают использование одного ключа для зашифрования нескольких сообщений без необходимости создания нового ключа и выделения на него памяти.

Вывод

Приобрела навыки применения режима однократного гаммирования на примере кодирования различных исходных текстов одним ключом.

Библиография

* https://www.youtube.com/watch?v=tAjBULW_OjQ
* https://bugtraq.ru/library/books/crypto/chapter7/
* https://xakep.ru/2019/07/18/crypto-xor/