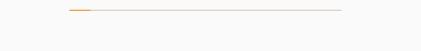
# лементы криптографии. Однократное гаммирование

Лабораторная работа № 7

Абу Сувейлим М. М.

10 января 2003

Российский университет дружбы народов, Москва, Россия



Информация

#### Докладчик

- Абу Сувейлим Мухаммед Мунифович
- Студент
- Российский университет дружбы народов
- · 1032215135@pfur.ru
- https://mukhammed-abu-suveilim.github.io/

# Вводная часть

- · Освоить на практике применение режима однократного гаммирования [@infosec].
- 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.

#### Материалы и методы

1. Kulyabov D., Korolkova A., Gevorkyan M. Информационная безопасность компьютерных сетей: лабораторные работы. 2015.

Выполнение лабораторной работы

### Python: import

Во-первых, нужно импортировать нужные библиотеки командой:

import os

Python: xor\_bytes(text, key)

Далее, для максимально эффективной работы лучше выполнять команды на уровне битов. Для этого мы напиши функцию хог\_bytes наложения гаммы:

```
def xor_bytes(text, key):
    return bytes([a ^ b for a, b in zip(text, key)])
```

### Python: encrypt(text)

Далее, напишем две функции для encoding и decoding:

```
def encrypt(text):
    text_bytes = text.encode('utf-8') # Converet text to bytes
    key = os.urandom(len(text_bytes))
    cipherText = xor_bytes(text_bytes, key)
    return cipherText, key
```

### Python: decrypt(cipherText, key)

```
def decrypt(cipherText, key):
    org_text_bytes = xor_bytes(cipherText, key)
    return org_text_bytes.decode('utf-8')
```

Выполняем пример из учебника:

```
text = "С Новым Годом, друзья!"
cipherText, key = encrypt(text)
print(f"Cipher text: {cipherText}")
print(f"Key: {key}")
decryptText = decrypt(cipherText, key)
print(f"Decrypted text: {decryptText}")
```

#### Результат

#### Получаем такой результат:

```
Cipher text: b'\x1a<]\x0cc\x81\x00\xf5B{"{\xb1\xc3\xf1\x1e\x82\x94RUOZ\xee^\x Key: b'\xca\x9d}\xdc\xfeQ\xbe%\xf0\xaa\xa9\xab\r\xe3!\x8dR*\x82\xe1\x9f\xe4>\ Decrypted text: C Новым Годом, друзья!
```

#### Тест кейс

Если в Кеу поминать значения на другие, например на:

```
key1 = b'\xca\x9d}\xdc\xfeQ\xbe%\xf0\xaa\xa9\xab\r\xe3!\x8dR*\x82\xe1\x9f\xe4
decryptText = decrypt(cipherText, key1)
print(f"Decrypted text: {decryptText}")
```

## Результат тест кейса

получим:

Decrypted text: С Новым Годом, зІузья!

Выводы



Основали на практике применение режима однократного гаммирования.