Элементы криптографии. Кодирование различных исходных текстов одним ключом

Лабораторная работа № 8

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

Цели и задачи

- Освоить на практике применение режима однократного гаммирования на примере кодирования различных исходных текстов одним ключом [@infosec].
- 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 P1 и P2 в режиме однократного гаммирования. Приложение должно определить вид шифротекстов C1 и C2 обоих текстов P1 и P2 при известном ключе; Необходимо определить и выразить аналитически способ, при котором злоумышленник может прочитать оба текста, не зная ключа и не стремясь его определить.

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 xor_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')
    cipher_text = xor_bytes(text_bytes, key)
    return cipher_text
```

Python: decrypt(cipherText, key)

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

Тесты P1 и P2

Выполняем пример из учебника:

```
# Given texts (P1 and P2)
P1 = "НаВашисходящийот1204"
P2 = "ВСеверныйфилиалБанка"

# Display the original texts and the key
print(f"Text P1: {P1}")
print(f"Text P2: {P2}")

# Convert texts to bytes
P1_bytes = P1.encode('utf-8')
P2_bytes = P2.encode('utf-8')
```

Ключ

Создадим ключ дленной текста P1:

```
key = os.urandom(len(P1_bytes))
```

Выполняем пример

```
# Encrypt P1 and P2 with the same key
C1 = encrypt(P1, key)
C2 = encrypt(P2, key)

# Display the encrypted texts and the key
print(f"C1 (Cipher text for P1): {C1}")
print(f"C2 (Cipher text for P2): {C2}")
print(f"Key: {key}")
```

Выполняем пример

```
# Decrypt the texts back to verify correctness
P1_decrypted = decrypt(C1, key)
P2_decrypted = decrypt(C2, key)
print(f"Decrypted P1: {P1_decrypted}")
print(f"Decrypted P2: {P2_decrypted}")
```

Выполняем пример

$$C_1 \oplus C_2 = (P_1 \oplus K) \oplus (P_2 \oplus K) = P_1 \oplus P_2$$

$$P_1 \oplus P_2 \oplus P_1 = P_2$$

```
P1_xor_P2 = xor_bytes(C1, C2)
print(f"P1 XOR P2: {P1_xor_P2}")


P2_recovered_bytes = xor_bytes(P1_xor_P2, P1_bytes)
P2_recovered = P2_recovered_bytes.decode('utf-8')
print(f"Recovered P2 using P1 and C1 XOR C2: {P2_recovered}")
```

Результат

Получаем такой результат:

```
Text P1: НаВашисходящийот1204
Text P2: ВСеверныйфилиалБанка
C1 (Cipher text for P1): b"\x95\x0e4\x99\x8d\xef.\xff\x11\x8dp\x8e\xde\xcc\xc
C2 (Cipher text for P2): b"\x95\x014\x88\x8d\xc8.\xfd\x10\xb0q\xb6\xdf\xf0\xc
Key: b'E\x93\xe4)]}\xfeO\xc0\x05\xa06\x0fM\x1c\xd9\xfc\x9e\x89\xd0\xdcc\r\x02
Decrypted P1: НаВашисходящийот1204
Decrypted P2: ВСеверныйфилиалБан
P1 XOR P2: b"\x00\x0f\x00\x11\x00'\x00\x02\x01=\x018\x01<\x00\x0e\x00\x07\x01
Recovered P2 using P1 and C1 XOR C2: ВСеверныйфилиалБан
```

Выводы

Выводы

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