Отчёт по лабораторной работе №7

по дисциплине: Информационная безопасность

Тема: Элементы криптографии. Однократное гаммирование

Олейников Артём Игоревич

Группа НБИбд-01-23

Дата выполнения: 16.05.2025

1. Цель работы

Освоить на практике применение режима однократного гаммирования на примере шифра Вернама.

2. Ход выполнения

2.1 Генерация ключа

Для генерации ключа длиной, равной длине сообщения, была использована команда:

python vernam_lab7.py

Программа сгенерировала случайный ключ в формате HEX.

Ключ (HEX): 5F3A47A9D1B29338EE83F516589BDDAA6DC65181A6F1885BA60DA2

2.2 Шифрование сообщения

Открытый текст: "С Новым Годом, друзья!"

Открытый текст был закодирован в байты и зашифрован по алгоритму XOR с ключом.

Результат (шифртекст в HEX):
303F26C291D4E24B8796A6F53EDAD8C04AAB317CC4D9A2EED52B8B

2.3 Расшифровка

Выполнено обратное наложение XOR между шифртекстом и ключом.

Полученный результат успешно совпал с оригинальным сообщением.

2.4 Восстановление ключа

Была выполнена операция XOR между шифртекстом и открытым текстом.

Полученный ключ совпал с изначально сгенерированным.

```python
import os


def xor_bytes(a: bytes, b: bytes) -> bytes:
    return bytes([x^y for x, y in zip(a,b)])

def encrypt(plaintext: str, key: bytes) -> bytes:
    plaintext_bytes = plaintext.encode("utf-8")
    if len(plaintext_bytes) != len(key):
        raise ValueError("Длина ключа должна совпадать с длиной текста в
            байтах")
    return xor_bytes(plaintext_bytes, key)

def decrypt(ciphertext: bytes, key: bytes) -> str:
    if len(ciphertext) != len(key):
        raise ValueError("Длины шифртекста и ключа должны совпадать")
    decrypted_bytes = xor_bytes(ciphertext, key)
    return decrypted_bytes.decode("utf-8", errors = "replace")

def find_key(ciphertext: bytes, plaintext: str) -> bytes:
    plaintext_bytes = plaintext.encode("utf-8")
    if len(ciphertext) != len(plaintext_bytes):
        raise ValueError("Длины текста и шифртекста должны совпадать")
    return xor_bytes(ciphertext, plaintext_bytes)

def main():
    plaintext = "С Новым Годом, друзья!"
    print(f"Открытый текст: {plaintext}")

    plaintext_bytes = plaintext.encode("utf-8")
    print(f"Длина в байтах: {len(plaintext_bytes)}")

    key = os.urandom(len(plaintext_bytes))
    print(f"Случайный ключ (hex): {key.hex().upper()}")

    ciphertext = encrypt(plaintext, key)
    print(f"Зашифрованный текст (hex): {ciphertext.hex().upper()}")

    recovered_key = find_key(ciphertext, plaintext)
    print(f"Восстановленый ключ (hex): {recovered_key.hex().upper()}")

    decrypted = decrypt(ciphertext, key)
    print(f"Восстановленный ключ (hex): {recovered_key.hex().upper()}")

    print("Ключи совпадают:", key == recovered_key)

```

```
Открытый текст: С Новым Годом, друзья!
Длина в байтах: 39
Случайный ключ (hex):
C2C8B4C2FC880D48ECA70A04040DFBA787FDDBBC9D5AFFC8503C58BF410CC943F4E45C15190B66
Зашифрованный текст (hex):
1269941261588B3985E7681D4B82D2B3457430B084DE42F747C1C880B908C18C024538D99C88447
Восстановленый ключ (hex):
C2C8B4C2FC880D48ECA70A04040DFBA787FDDBBC9D5AFFC8503C58BF410CC943F4E45C15190B66
Восстановленный ключ (hex):
C2C8B4C2FC880D48ECA70A04040DFBA787FDDBBC9D5AFFC8503C58BF410CC943F4E45C15190B66
Ключи совпадают: True
```

3. Выводы

В ходе выполнения лабораторной работы было реализовано приложение на языке Python, выполняющее шифрование и расшифровку текста по алгоритму однократного гаммирования (XOR). Результаты подтвердили абсолютную криптостойкость метода при соблюдении условий: длина ключа равна длине сообщения, ключ используется один раз, ключ является случайным.