Лабораторная работа №7

Элементы криптографии. Однократное гаммирование

Буллет Т. А.

15 февраля 2025

Российский университет дружбы народов, Москва, Россия

Информация

Докладчик

- Буллер Татьяна Александровна
- студент направления Бизнес-информатика
- Российский университет дружбы народов

Вводная часть

Объект и предмет исследования

- Метод шифрования XOR
- Среда виртуализации VirtualBox

Цели и задачи

- Освоить на практике применение режима однократного гаммирования.

Материалы и методы

- Процессор pandoc для входного формата Markdown
- Среда виртуализации VirtualBox
- Язык программирования Python

Выполнение лабораторной работы

Определение кодировки шифротекста

Сообщение “D8 F2 E8 F0 EB E8 F6 20 2D 20 C2 FB 20 C3 E5 F0 EE E9 21 21” написано на русском языке, однако при переводе его из hex в текст стандартных кодировок ASCII/UTF-8 результат не совпадал с тем, какой был задан условием задания. Путем перебора кодировок было выяснено, что сообщение было написано в кодировке Windows-1251.

Определение кодировки шифротекста

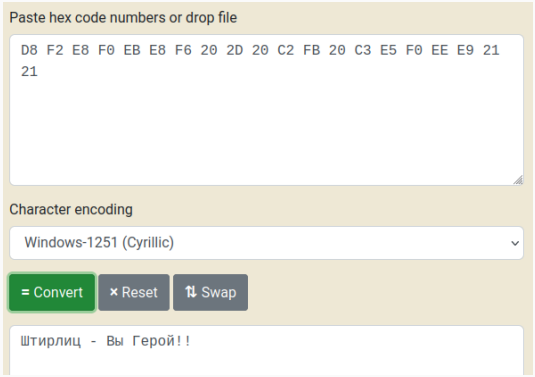

Рис. 1: Кодировка файла

Программа дешифрования по известному открытому тексту

Далее необходимо было написать код, с помощью которого можно было бы дешифровать сообщение. Метод шифрования XOR крайне уязвим к атакам по известному открытому тексту, поэтому, зная сообщение на выходе, мы без труда можем получить ключ, которым нужно было зашифровать строку, чтобы его получить. Для этого нужно обернуть операцию.

Программа дешифрования по известному открытому тексту

Переведем известные нам строки шифра и открытого текста (“С Новым Годом, друзья!” в кодировке Windows-1251) в hex. Для того, чтобы далее провести с ними операции XOR, нужно перевести эти строки далее в бинарный формат. После сравним строки посимвольно и запишем результат: если символ в позиции n строки А совпадает с символом в той же позиции в строке Б, то в результат дописывается 0, иначе - 1. Необходимо отметить, что для корректной работы кода и получения полного ключа строки должны совпадать по длине. В случае, если какая-то из них короче другой, она повторяется до того, как длины совпадут.

Программа дешифрования по известному открытому тексту

```python
# само за себя говорит
def hex_to_bin(ints):
    scale = 16
    res = bin(int(ints, scale)).zfill(8)
    return res

# xor для двух строк (должны быть одинаковой длины, чтобы получить полный ключ/шифротекст. в случае, если длины строк не совпадают, ту, что короче, необходимо повторить до соответствия длине второй строки)
def xor(a, b, n):
    ans = ""
    for i in range(n):
        if (a[i] == b[i]):
            ans += "0"
        else:
            ans += "1"
    return ans

# main code
if __name__ == "__main__":
    a = hex_to_bin('d8f2e8f0ebe8f6202d20c2fb20c3e5f0eee92121d8f2')
    b = hex_to_bin('d120cdeee2fbec20c3eee4eeec2c20e4f0f3e7fcff21')

    n = len(a)
    c = xor(a, b, n)
    print(c)

```

Рис. 2: Код программы

Программа дешифрования по известному открытому тексту

Программа выдает ключ в формате двоичного числа, который при необходимости далее можно перевести в шестнадцатиричный формат. Использовав полученный ключ вместо шифротекста, мы получим другой двоичный вывод. Переведя его в текст и расшифровав в кодировке Windows-1251 получим сообщение, ключ к которому и хотели найти, что говорит о том, что код сработал корректно.

Программа дешифрования по известному открытому тексту

Рис. 3: Расшифровка полученного с новым ключом сообщения

Выводы

Выводы

Было освоено на практике применение режима однократного гаммирования, написана программа, переводящая строки из шестнадцатиричного формата в двочиный и проводящая между ними XOR-операцию посимвольно.