Национальный исследовательский университет ИТМО

Факультет программной инженерии и компьютерной техники

Направление программная инженерия
Образовательная программа системное и прикладное программное обеспечение
Специализация системное программное обеспечение

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1.3

курса «Информационная безопасность»

по теме: «Криптографические системы с секретным ключом»

Вариант № 3

Выполнил студент:
Тюрин Иван Николаевич
группа: P33102

Преподаватель:
Маркина Т.А.,
Рыбаков С.Д.

Санкт-Петербург, 2024 г.

Содержание

Лабораторная работа № 1.3 Криптографические системы с секретным ключом

1. Описание

Лабораторная работа № 1 «Основы шифрования данных».

Цель работы. Изучение структуры и основных принципов работы современных алгоритмов поточного симметричного шифрования, приобретение навыков программной реализации поточных симметричных шифров.

Задание варианта № 3: *изучение структуры и основных принципов работы современных алгоритмов поточного симметричного шифрования, приобретение навыков программной реализации поточных симметричных шифров.*

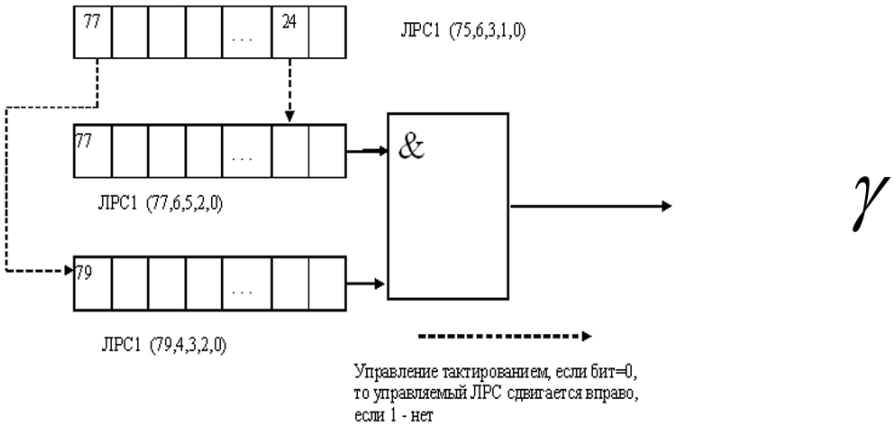

Рис. 1.1: Задание для варианта № 3

2. Выполнение задания

Из-за использования необщепринятых названий и своеобразного иллюстрирования схемы было сложно понять, что требуется сделать, но после некоторого времени разбирательств было выяснено, что задание понимать можно так: имеется несколько регистров сдвига с линейной обратной связью (ЛРС), которые почему-то обозначены одинаковым названием, но имеют разную функцию обратной связи от бит с номерми в скобках; верхний регистр управляет сдвигом двух других регистров, тем самым воплощая схему “Чередующийся генератор «стоп-пошёл»„ для ключей потокового шифра.

В соответствии с условием задания разработан скрипт на языке Python выполняющий шифрование текста из файла по заданию. Исходный код скрипта можно видеть на листинге 1.2. Справку по работе с программой можно видеть на листинге 1.1.

```
usage: main.py [-h] [--seed SEED] --mode {encrypt,decrypt} --
    input INPUT --output OUTPUT

options:
  -h, --help            show this help message and exit
  --seed SEED, -s SEED  Начальное состояние для генератора ключ
    ей
  --mode {encrypt,decrypt}, -m {encrypt,decrypt}
                        Режим работы
  --input INPUT, -i INPUT
                        Входной файл
  --output OUTPUT, -o OUTPUT
                        Выходной файл
```

Листинг 1.1: справка по использованию разработанной программы

Пример работы утилиты по шифрованию и дешифрованию фрагмента текста из трактата «Lorem ipsum» можно видеть на изображении 1.2.

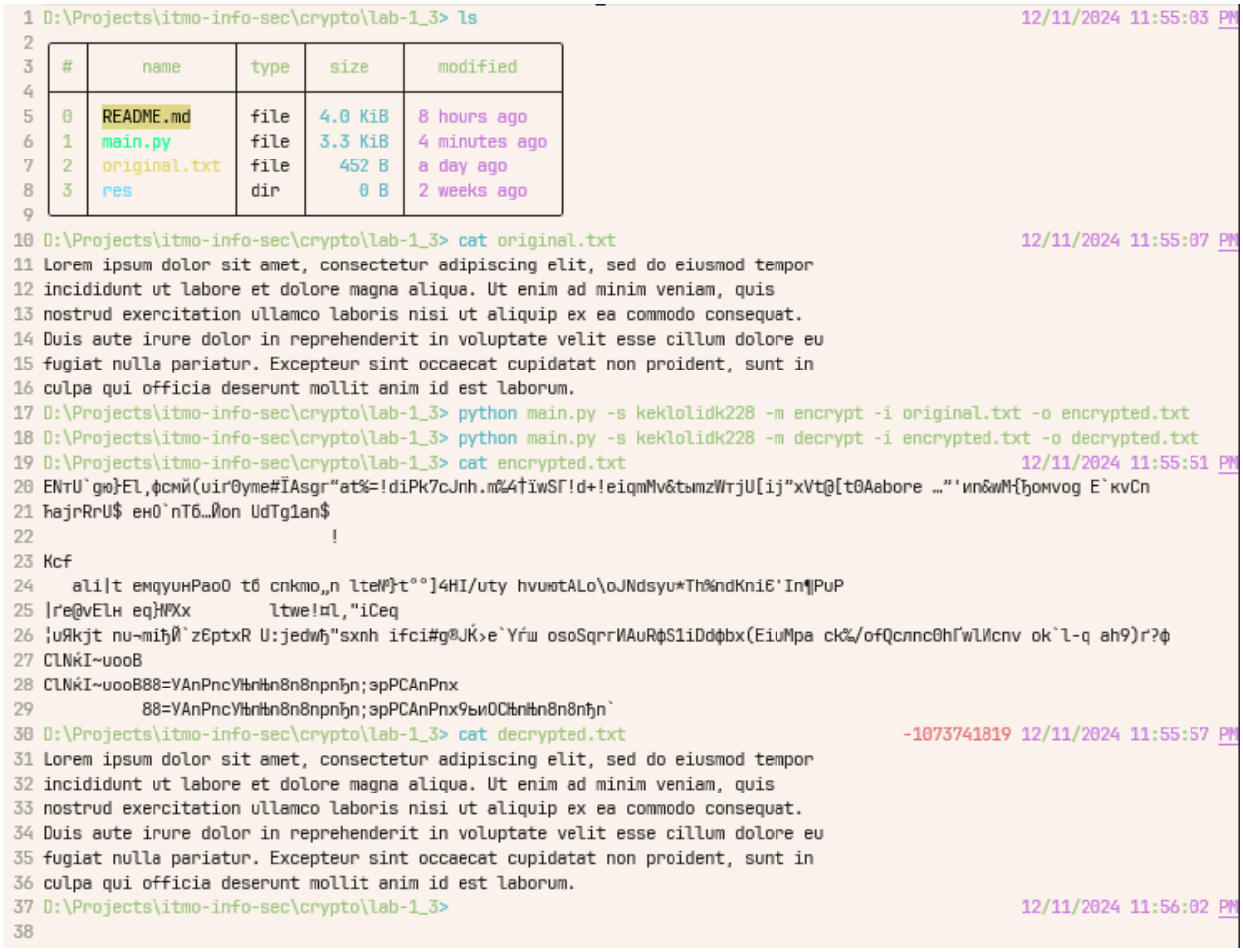

Рис. 1.2: Пример работы утилиты для шифрования фрагмента трактата Lorem ipsum.

```python
#!/usr/bin/env python

from argparse import ArgumentParser
from functools import reduce
from typing import Literal


type Mode = Literal["encrypt", "decrypt"]


class LSR:
    def __init__(self, size: int, init_state: int, fb_ids: list[int]):
        self.size: int = size
        self.state: int = init_state & ((1 << size) - 1)
        self.fb_ids: list[int] = fb_ids
        for i in fb_ids:
            assert (i < size), f"bit {i} must be in reg with size {size}"

    def shift(self) -> int:
        shift_bit = self.state & 1
```

```
        feedback_bit = reduce(
            lambda a, b: a ^ b, [(self.state & (1 << i)) >> i for i in self.fb_ids], 0
        )
        self.state = (feedback_bit << (self.size - 1)) | (self.state >> 1) & ((1 << self.size) - 1)
        return shift_bit

    def out(self) -> int:
        return self.state & 1

def setup(seed: int,debug=False):
    lsr0 = LSR(78, seed, [77, 6, 5, 2, 0])
    lsr1 = LSR(80, seed, [79, 4, 3, 2, 0])
    lsrc = LSR(78, seed, [75, 6, 3, 1, 0])

    def y_gen() -> int:
        cbit = lsrc.shift()
        match cbit:
            case 0:
                sbit = lsr0.shift()
            case 1:
                sbit = lsr1.shift()
            case _:
                raise ValueError(f"control bit is out of bounds: {cbit}")
        if debug:
            print(f"cbit={cbit} sbit={sbit} lsr0={lsr0.state} lsr1={lsr1.state} lsrc={lsrc.state}")
        out = lsr0.out() & lsr1.out()
        assert out == 0 or out == 1, "output value is out of bounds {out}"
        return out

    def y_byte_gen() -> int:
        y_byte = 0
        for i in range(8):
            y_byte |= y_gen() << i
        return y_byte

    return (y_gen, y_byte_gen)


def process_file(
    seed: int,
    mode: Mode,
    input_file: str,
    output_file: str,
):
    with open(input_file, "rb") as f:
        data = bytearray(f.read())

```

```
    _, y_byte_gen = setup(seed, debug=False)
    result = bytearray()

    match mode:
        case "encrypt":
            for b in data:
                y = y_byte_gen()
                nb = b ^ y
                result.append(nb)
        case "decrypt":
            for b in data:
                y = y_byte_gen()
                nb = b ^ y
                result.append(nb)

    with open(output_file, "wb") as f:
        f.write(result)


def main():
    parser = ArgumentParser(description="")
    parser.add_argument("--seed", "-s", required=False,
                        help="Начальное состояние для генератора ключей")
    parser.add_argument("--mode", "-m", choices=["encrypt", "decrypt"],
                        required=True, help="Режим работы",)
    parser.add_argument("--input", "-i", required=True, help="Входной файл")
    parser.add_argument("--output", "-o", required=True, help="Выходной файл")
    args = parser.parse_args()

    seed: int = (
        int.from_bytes(args.seed.encode(encoding="utf-8"), byteorder="little")
        if args.seed
        else 0xABCDEF123456790ABCDE
    )
    mode: Mode = args.mode

    ifile: str = args.input
    ofile: str = args.output
    process_file(seed, mode, ifile, ofile)

if __name__ == "__main__":
    main()
```

Листинг 1.2: Код скрипта выполняющего зашифрование и расшифрование текста поточным методом в соответсвии с заданной схемой

3. Вывод

Изучили структуру и основные принципы работы современных алгоритмов поточного симметричного шифрования, приобретение навыков программной реализации поточных симметричных шифров на языке программирования Python.