ЛАБОРАТОРНАЯ РАБОТА №5

“Шифрование открытого текста на основе эллиптических кривых”

по дисциплине

‘Информационная безопасность’

Вариант 15

Выполнил:
Соболев Иван Александрович
Группа: P34312

Преподаватель:
Маркина Татьяна Анатольевна

Санкт-Петербург, 2024

Цель работы

Зашифровать открытый текст, используя приведенный алфавит на основе кривой $E751(-1, 1) : y2 = x3 - 1x + 1\ (mod\ 751)$ и генерирующей точки $G(0, 1)$.

Программные и аппаратные средства

Для выполнения лабораторной работы был использован компьютер со следующими характеристиками:

- Процессор: Apple M2
- Видеокарта: Apple M2
- Объем оперативной памяти: 8GB
- Использована операционная система: macOS 14.4.1
- Версия Python: 3.13

Задание

№ варианта	**Открытый текст**	**Открытый ключ B**	**Значения случайных чисел k для букв открытого текста**
15	отставной	(286, 136)	5, 3, 3, 2, 4, 19, 2, 4, 10

Листинг разработанной программы

main.py

```
import sys

import alphabet
from curve import Curve
from io_utils import print_red, read_config, print_separator
from point import Point


def main():
    curve = Curve(-1, 1, 751)
    g = Point(0, 1)
    if len(sys.argv) < 2:
        print_red("Укажите название файла с параметрами!")
        return

    input_file = None
    for i in range(1, len(sys.argv)):
        if sys.argv[i] == '-f' and i + 1 < len(sys.argv):
            input_file = sys.argv[i + 1]

    doc = read_config(input_file)
    bx = doc['Bx']
```

```
    by = doc['By']
    pb = Point(bx, by)
    text = doc['T']
    print(f"Pb = {pb}, Сообщение: {text}")

    k = [int(c_k) for c_k in doc['k']]

    res = []
    print_separator()
    for i, c in enumerate(text):
        a_pm = alphabet.ALPHABET[c]
        pm = Point(a_pm.x, a_pm.y)
        c_k = k[i]
        kg = curve.elliptic_mul(g, c_k)
        kpb = curve.elliptic_mul(pb, c_k)
        pmkpb = curve.elliptic_add(kpb, pm)
        print(f"Исходный символ: '{c}'; k = {c_k}; Pm = {pm}; kPb = {kpb}")
        print(f"Cm = (kG, Pm+kPb) = ({kg}, {pmkpb})")
        print_separator()
        res.append(kg)
        res.append(pmkpb)

    print(f"Зашифрованное сообщение: ")
    print(res,  sep='\n')


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

Io_utils.py

```
import yaml


def read_config(input_file):
    """Метод чтения параметров из конфигурационного файла."""
    with open(input_file, 'r') as file:
        file_contents = file.read()

    return yaml.safe_load(file_contents)


def print_green(message: str) -> None:
    """Метод для вывода ключа"""
    print(f"\033[92m{message}\033[0m")


def print_red(message: str) -> None:
    """Метод для вывода расшифрованного текста"""
    print(f"\033[91m{message}\033[0m")
```

```
def print_separator():
    """Метод для вывода разграничителя"""
    print_green("------------------------------------------------------------------")
```

point.py

```
class Point:
    def __init__(self, x: int, y: int):
        self.x = x
        self.y = y

    def __str__(self) -> str:
        return f"({self.x}, {self.y})"

    def __repr__(self) -> str:
        return f"({self.x}, {self.y})"
```

curve.py

```
from point import Point


class Curve:
    def __init__(self, a: int, b: int, p: int):
        self.a = a
        self.b = b
        self.p = p

    def elliptic_mul(self, p: Point, k):
        """Метод эллиптического умножения"""
        R = p
        for i in range(1, k):
            R = self.elliptic_add(R, p)

        return R

    def elliptic_add(self, P1: Point, P2: Point):
        """Метод эллиптического сложения"""
        x1, y1 = P1.x, P1.y
        x2, y2 = P2.x, P2.y

        if P1 != P2:
            slope = (y2 - y1) * pow(x2 - x1, -1, self.p)
        else:
            slope = (3 * x1 ** 2 + self.a) * pow(2 * y1, -1, self.p)

        x3 = pow(slope * slope - x1 - x2, 1, self.p)
        y3 = pow(slope * (x1 - x3) - y1, 1, self.p)

        return Point(x3, y3)
```

alphabet.py

```
from point import Point

ALPHABET = {
    'В': Point(67, 84),
    'е': Point(99, 456),
    'Й': Point(198, 527),
    ' ': Point(33, 355),
    'С': Point(67, 667),
    'f': Point(100, 364),
    'К': Point(200, 30),
    ...
```

config.yaml

```
T: отставной
Bx: 286
By: 136
k:
  - 5
  - 3
  - 3
  - 2
  - 4
  - 19
  - 2
  - 4
  - 10
```

Результаты работы программы

```
/Users/isobolev/Desktop/itmo/itmo-4course/information_sec/2_5/.venv/bin/python /Users/isobolev/Desktop/itmo/itmo-4course/information_sec/2_5/main.py -f config.yaml
Pb = (286, 136), Сообщение: отставной
------------------------------------------------------------
Исходный символ: 'о'; k = 5; Pm = (240, 389); kPb = (750, 1)
Cm = (kG, Pm+kPb) = ((425, 663), (99, 295))
------------------------------------------------------------
Исходный символ: 'т'; k = 3; Pm = (247, 266); kPb = (499, 156)
Cm = (kG, Pm+kPb) = ((56, 419), (606, 147))
------------------------------------------------------------
Исходный символ: 'с'; k = 3; Pm = (243, 664); kPb = (499, 156)
Cm = (kG, Pm+kPb) = ((56, 419), (151, 233))
------------------------------------------------------------
Исходный символ: 'т'; k = 2; Pm = (247, 266); kPb = (327, 108)
Cm = (kG, Pm+kPb) = ((188, 93), (279, 398))
------------------------------------------------------------
Исходный символ: 'а'; k = 4; Pm = (228, 271); kPb = (519, 713)
Cm = (kG, Pm+kPb) = ((16, 416), (218, 601))
------------------------------------------------------------
Исходный символ: 'в'; k = 19; Pm = (229, 151); kPb = (406, 354)
Cm = (kG, Pm+kPb) = ((568, 355), (328, 461))
------------------------------------------------------------
Исходный символ: 'н'; k = 2; Pm = (238, 576); kPb = (327, 108)
Cm = (kG, Pm+kPb) = ((188, 93), (390, 603))
------------------------------------------------------------
Исходный символ: 'о'; k = 4; Pm = (240, 389); kPb = (519, 713)
Cm = (kG, Pm+kPb) = ((16, 416), (585, 540))
------------------------------------------------------------
Исходный символ: 'й'; k = 10; Pm = (236, 712); kPb = (3, 746)
Cm = (kG, Pm+kPb) = ((377, 456), (237, 297))
------------------------------------------------------------
Зашифрованное сообщение:
[(425, 663), (99, 295), (56, 419), (606, 147), (56, 419), (151, 233), (188, 93), (279, 398), (16, 416), (218, 601), (568, 355), (328, 461), (188, 93), (390, 603), (16, 416), (585, 540), (377, 456), (237, 297)]
```