УНИВЕРСИТЕТ ИТМО

Факультет программной инженерии и компьютерной техники

Направление подготовки 09.03.04 Программная инженерия

Лабораторная работа №2.3

Дисциплина «Информационная безопасность»

Вариант 13

Выполнил: студент группы P34131

Кузнецов Максим Александрович

Преподаватель:

Маркина Татьяна Анатольевна

Санкт-Петербург, 2023 г.

Цель работы

Изучить атаку на алгоритм шифрования RSA посредством метода бесключевого чтения.

Задание

Вариант	Модуль, N	Экспоненты		Блоки зашифрованного текста	
		e_1	e_2	C_1	C_2
13	518587807081	293177	1209781	373852443734 447989059513 140756140384 207791711792 252160015422 151272799305 431450717984 252882800366 112417596471 301753741810 480461056512 334158277030 368394150653	22286870422 343015689591 281801228231 360270382562 264253306719 128520421967 399665129411 448878989738 70913527757 295285211952 247990966487 202711954425 201121363025

Ход работы

1. Решаем уравнение $e_1 * r - e_2 * s = \pm 1$
2. Построчно производим дешифрацию: возводим c_1 в степень r, а c_2в степень s по модулю N.
3. Перемножаем полученные числа и берем модуль по N.
4. Преобразуем результат в текст.
5. Повторяем шаги 2–4 для каждой строки и получаем итоговый текст.

Для решения задачи была разработана программа на Python.

Листинг разработанной программы

```
import math

N = 518587807081
e1 = 293177
e2 = 1209781

C1 = '''
373852443734
447989059513
140756140384
207791711792
252160015422
151272799305
431450717984
252882800366
112417596471
301753741810
480461056512
334158277030
368394150653
'''

C2 = '''
22286870422
343015689591
281801228231
360270382562
264253306719
128520421967
399665129411
448878989738
70913527757
295285211952
247990966487
202711954425
201121363025
'''


def gcd_extended(num1, num2):
    if num1 == 0:
        return num2, 0, 1
```

```
    else:
        div, x, y = gcd_extended(num2 % num1, num1)
    return div, y - (num2 // num1) * x, x


c_1 = list(map(int, C1.split()))
c_2 = list(map(int, C2.split()))

a, r, s = gcd_extended(e1, e2)

print(f"1. Шаг первый: r = {r}, s = {s}")

output = ""

for i in range(len(c_1)):
        c_1_pow_r = pow(c_1[i], r, N)
        c_2_pow_s = pow(c_2[i], s, N)
        print(f"2. Шаг второй: c_1_pow_r = {c_1_pow_r}, c_2_pow_s =
{c_2_pow_s}")
        res = (c_1_pow_r * c_2_pow_s) % N
        print(f"3. Шаг третий: (c_1_pow * c_2_pow_s) mod N = {res}")
        msg = res.to_bytes(4, byteorder='big').decode('cp1251')
        print(f"4. Шаг четвертый: text(part) = {msg}")
        output += msg

print(f"5. Шаг пятый: итоговый текст -->{output}")
```

Ссылка на программу: ссылка

Результат работы программы:

```
1. Шаг первый: r = 559972, s = -135703
2. Шаг второй: c_1_pow_r = 182329854436, c_2_pow_s = 69595587711
3. Шаг третий: (c_1_pow * c_2_pow_s) mod N = 3488671776
4. Шаг четвертый: text(part) = При
2. Шаг второй: c_1_pow_r = 377770072921, c_2_pow_s = 190293051609
3. Шаг третий: (c_1_pow * c_2_pow_s) mod N = 4058965988
4. Шаг четвертый: text(part) = созд
2. Шаг второй: c_1_pow_r = 432156597280, c_2_pow_s = 385846773469
3. Шаг третий: (c_1_pow * c_2_pow_s) mod N = 3773688040
4. Шаг четвертый: text(part) = ании
2. Шаг второй: c_1_pow_r = 438370720662, c_2_pow_s = 203901545441
3. Шаг третий: (c_1_pow * c_2_pow_s) mod N = 552726245
4. Шаг четвертый: text(part) =  сое
2. Шаг второй: c_1_pow_r = 187538453050, c_2_pow_s = 235946809363
3. Шаг третий: (c_1_pow * c_2_pow_s) mod N = 3840470501
4. Шаг четвертый: text(part) = дине
2. Шаг второй: c_1_pow_r = 486184940203, c_2_pow_s = 222117792391
3. Шаг третий: (c_1_pow * c_2_pow_s) mod N = 3991469856
4. Шаг четвертый: text(part) = ния
2. Шаг второй: c_1_pow_r = 473555064772, c_2_pow_s = 375710700894
3. Шаг третий: (c_1_pow * c_2_pow_s) mod N = 4059229936
4. Шаг четвертый: text(part) = стор
2. Шаг второй: c_1_pow_r = 221036696750, c_2_pow_s = 7032409519
3. Шаг третий: (c_1_pow * c_2_pow_s) mod N = 4008573728
4. Шаг четвертый: text(part) = оны
2. Шаг второй: c_1_pow_r = 180661034032, c_2_pow_s = 218945970929
3. Шаг третий: (c_1_pow * c_2_pow_s) mod N = 4007783653
4. Шаг четвертый: text(part) = обме
2. Шаг второй: c_1_pow_r = 49881527346, c_2_pow_s = 193573491959
3. Шаг третий: (c_1_pow * c_2_pow_s) mod N = 3991462624
4. Шаг четвертый: text(part) = нива
2. Шаг второй: c_1_pow_r = 343264156798, c_2_pow_s = 44234210562
3. Шаг третий: (c_1_pow * c_2_pow_s) mod N = 4277334527
4. Шаг четвертый: text(part) = ются
2. Шаг второй: c_1_pow_r = 97511061110, c_2_pow_s = 288865829588
3. Шаг третий: (c_1_pow * c_2_pow_s) mod N = 552461809
4. Шаг четвертый: text(part) =  нес
2. Шаг второй: c_1_pow_r = 453498220685, c_2_pow_s = 369887064049
3. Шаг третий: (c_1_pow * c_2_pow_s) mod N = 3941474336
4. Шаг четвертый: text(part) = ко
5. Шаг пятый: итоговый текст -->При создании соединения стороны обмениваются неско
```

Итоговый текст: При создании соединения стороны обмениваются неско

Вывод

В ходе выполнения данной лабораторной работы я:

- ознакомился с методом бесключевого чтения для атаки на алгоритм шифрования RSA
- Реализовал данный метод на языке Python.