Федеральное государственное автономное образовательное учреждение высшего образования

Университет ИТМО

Дисциплина: Информационная безопасность (Криптография)

Лабораторная работа 2.3

Вариант 12

Работу выполнил студент группы P34111:
Кривоносов Егор Дмитриевич

Преподаватель:
Маркина Татьяна Анатольевна

2022 г.

г. Санкт-Петербург

Оглавление

Цель работы

Изучить атаку на алгоритм шифрования RSA посредством метода бесключевого чтения.

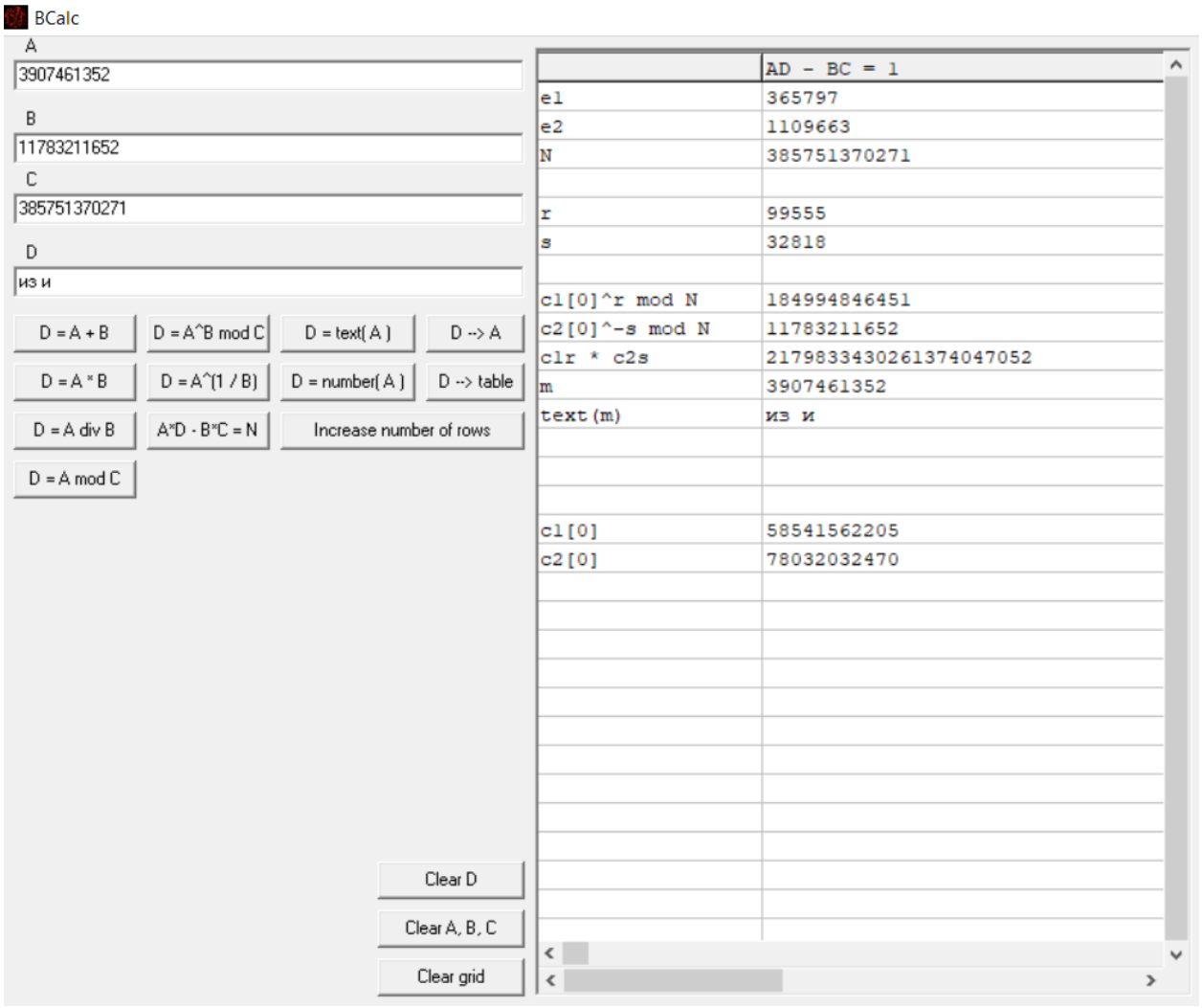
Задание

Вариант	**Модуль, N**	**Экспоненты**		**Блоки зашифрованного текста**	
		e_1	e_2	C_1	C_2
12	385751370271	365797	1109663	58541562205 167003685579 381877628242 256218527098 164244249864 6588741823 180308234660 174572441677 259951955034 378589342820 319378579620 21405495597 226860843155	78032032470 13064174635 326727914830 364066420370 177576861402 65863828523 111437045566 124743274954 119577259869 85769669875 4688914942 261002397567 341722428571

Ход работы

1. Решаем уравнение $e_1 \times r - e_2 \times s = \pm 1$. Получаем $r = 99555$ и $s = -32818$.
2. Построчно производим дешифрацию: c_1 возводим в степень r, а c_2 в степень $-s$ по модулю N.
3. Перемножаем полученные на предыдущем шаге числа и берем модуль по N из полученного числа.
4. Преобразуем результат в текст.
5. Повторяем шаги 2-5 для каждой строки.

Скриншот работы программы Bcalc.exe



Листинг разработанной программы

```
N = 385751370271
e1 = 365797
e2 = 1109663

C1 = '''
58541562205
167003685579
381877628242
256218527098
164244249864
```

```
6588741823
180308234660
174572441677
259951955034
378589342820
319378579620
21405495597
226860843155
'''

C2 = '''
78032032470
13064174635
326727914830
364066420370
177576861402
65863828523
111437045566
124743274954
119577259869
85769669875
4688914942
261002397567
341722428571
'''


# Алгоритм Евклида
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


def solver(N, e1, e2, C1, C2):
    print("==========================")
    print("========== DATA ==========")
    print("==========================")
    print(f"N = {N}")
    print(f"e1 = {e1}")
    print(f"e2 = {e2}")
    print(f"C1 = {C1}")
    print(f"C2 = {C2}")

    print("==========================")
    print("== SOLVE - keyless read ==")
    print("==========================")

    a, r, s = gcd_extended(e1, e2)
    print(f"(e1 * r) + (e2 * s) = ±1")
    print(f" r = {r},\n s = {s}", "\n")

    c1 = list(map(int, C1.split()))
    c2 = list(map(int, C2.split()))
```

```
    message = ""
    for i in range(len(c1)):
        c1r = pow(c1[i], r, N)
        c2s = pow(c2[i], s, N)
        m = (c1r * c2s) % N
        part = m.to_bytes(4, byteorder='big').decode('cp1251')
        message += part
        print(f"(C1[{i}]^r) mod N = {c1r}")
        print(f"(C2[{i}]^s) mod N = {c2s}")
        print(f"m{i} = ({c1r} * {c2s}) mod {N} = {m} => text({m}) = {part}", "\n")

    print(f"message = {message}")


solver(N, e1, e2, C1, C2)
```

Скриншоты работы программы Python

```
==========================
========== DATA ==========
==========================
N = 385751370271
e1 = 365797
e2 = 1109663
C1 =
58541562205
167003685579
381877628242
256218527098
164244249864
6588741823
180308234660
174572441677
259951955034
378589342820
319378579620
21405495597
226860843155
```

```
C2 =
78032032470
13064174635
326727914830
364066420370
177576861402
65863828523
111437045566
124743274954
119577259869
85769669875
4688914942
261002397567
341722428571
```

```
 ==========================
 == SOLVE - keyless read ==
 ==========================
 (e1 * r) + (e2 * s) = ±1
  r = 99555,
  s = -32818

 (C1[0]^r) mod N = 184994846451
 (C2[0]^s) mod N = 11783211652
 m0 = (184994846451 * 11783211652) mod 385751370271 = 3907461352 => text(3907461352) = из и

 (C1[1]^r) mod N = 186056688973
 (C2[1]^s) mod N = 319756402469
 m1 = (186056688973 * 319756402469) mod 385751370271 = 4059426532 => text(4059426532) = сход

 (C1[2]^r) mod N = 339720290897
 (C2[2]^s) mod N = 178045441404
 m2 = (339720290897 * 178045441404) mod 385751370271 = 3991856110 => text(3991856110) = ного

 (C1[3]^r) mod N = 19229721971
 (C2[3]^s) mod N = 258929268603
 m3 = (19229721971 * 258929268603) mod 385751370271 = 552591594 => text(552591594) =  пак

 (C1[4]^r) mod N = 269160964140
 (C2[4]^s) mod N = 236318738639
 m4 = (269160964140 * 236318738639) mod 385751370271 = 3857899564 => text(3857899564) = ета,
```

```
(C1[5]^r) mod N = 224970171140
(C2[5]^s) mod N = 10819702523
m5 = (224970171140 * 10819702523) mod 385751370271 = 551690474 => text(551690474) =  в к

(C1[6]^r) mod N = 7626941261
(C2[6]^s) mod N = 52830577429
m6 = (7626941261 * 52830577429) mod 385751370271 = 4008898288 => text(4008898288) = отор

(C1[7]^r) mod N = 356403904642
(C2[7]^s) mod N = 89027639917
m7 = (356403904642 * 89027639917) mod 385751370271 = 4008452335 => text(4008452335) = ом п

(C1[8]^r) mod N = 131921104344
(C2[8]^s) mod N = 293176270874
m8 = (131921104344 * 293176270874) mod 385751370271 = 3857769956 => text(3857769956) = еред

(C1[9]^r) mod N = 19836462054
(C2[9]^s) mod N = 310613432190
m9 = (19836462054 * 310613432190) mod 385751370271 = 3773692704 => text(3773692704) = аны

(C1[10]^r) mod N = 287942708736
(C2[10]^s) mod N = 305943536572
m10 = (287942708736 * 305943536572) mod 385751370271 = 4260554784 => text(4260554784) = эти

(C1[11]^r) mod N = 258450658366
(C2[11]^s) mod N = 116113919033
m11 = (258450658366 * 116113919033) mod 385751370271 = 959459360 => text(959459360) = 900
```

```
(C1[12]^r) mod N = 229456042062
(C2[12]^s) mod N = 322548736593
m12 = (229456042062 * 322548736593) mod 385751370271 = 3777896480 => text(3777896480) = 6.

message = из исходного пакета, в котором переданы эти 900 б.
```

Полученный результат: “из исходного пакета, в котором переданы эти 900 б.”

Вывод

В ходе выполнения данной лабораторной работы я ознакомился с методом бесключевого чтения для атаки на алгоритм шифрования RSA и реализовал его работу на языке Python.

Полезные ссылки

Алгоритм Евклида Python