

УНИВЕРСИТЕТ ИТМО

Факультет программной инженерии и компьютерной техники

Направление подготовки 09.03.04 Программная инженерия

Дисциплина «Информационная безопасность»

Лабораторная работа №2.4

*Атака на алгоритм шифрования RSA, основанная на Китайской
теореме об остатках*

Вариант 22

Студент

Патутин В. М.

P33101

Преподаватель

Маркина Т. А.

Санкт-Петербург, 2022 г.

Цель работы

Изучить атаку на алгоритм шифрования RSA посредством Китайской теоремы об остатках.

Исходные данные:

$$e = 3$$

$$N_1 = 582980801989$$

$$N_2 = 585089367091$$

$$N_3 = 586408807447$$

$$C_1 =$$

428799001102

417746620458

233652090970

425829696584

132807280253

540064099057

191642450251

364237792802

294540030550

287338190886

8030576378

562848664519

$$C_2 =$$

330278110381

413803169370

399528613141

431344022162

133251402314

579394141601

339286468279

235332969532

1036448642

400656499573

47204841232

249621210713

$$C_3 =$$

426468615928

348743875265

261688856582

29957256669

108448874326

23970225383

410917339855

179638698652

282723305676

115801357719

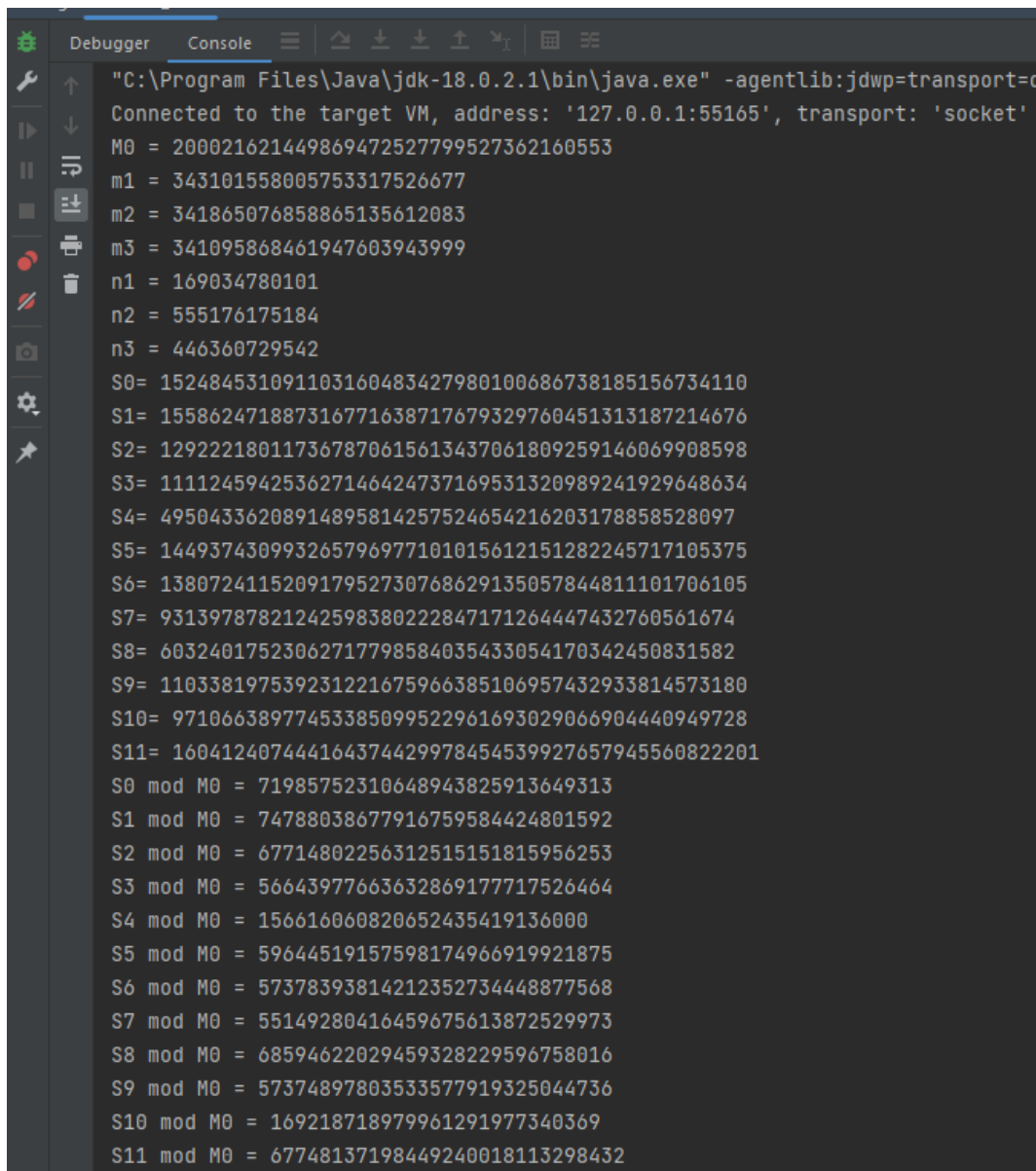
575898855271

528022904569

Алгоритм выполнения

1. Вычисляем $M0 = N1 \cdot N2 \cdot N3$;
2. Вычисляем $m1 = N2 \cdot N3$;
3. Вычисляем $m2 = N1 \cdot N3$;
4. Вычисляем $m3 = N1 \cdot N2$;
5. Вычисляем $n1 = m1^{(-1)} \bmod N1$;
6. Вычисляем $n2 = m2^{(-1)} \bmod N2$;
7. Вычисляем $n3 = m3^{(-1)} \bmod N3$;
8. Вычисляем $S = c1 \cdot n1 \cdot m1 + c2 \cdot n2 \cdot m2 + c3 \cdot n3 \cdot m3$;
9. Вычисляем $S \bmod M0$;
10. Вычисляем $M = (S \bmod M0)^{(1/e)}$;
11. Получаем исходный текст.

Выполнение работы



```
Debugger Console
"C:\Program Files\Java\jdk-18.0.2.1\bin\java.exe" -agentlib:jdwp=transport=
Connected to the target VM, address: '127.0.0.1:55165', transport: 'socket'
M0 = 200021621449869472527799527362160553
m1 = 343101558005753317526677
m2 = 341865076858865135612083
m3 = 341095868461947603943999
n1 = 169034780101
n2 = 555176175184
n3 = 446360729542
S0= 152484531091103160483427980100686738185156734110
S1= 155862471887316771638717679329760451313187214676
S2= 129222180117367870615613437061809259146069908598
S3= 111124594253627146424737169531320989241929648634
S4= 49504336208914895814257524654216203178858528097
S5= 144937430993265796977101015612151282245717105375
S6= 138072411520917952730768629135057844811101706105
S7= 93139787821242598380222847171264447432760561674
S8= 60324017523062717798584035433054170342450831582
S9= 110338197539231221675966385106957432933814573180
S10= 97106638977453385099522961693029066904440949728
S11= 160412407444164374429978454539927657945560822201
S0 mod M0 = 71985752310648943825913649313
S1 mod M0 = 74788038677916759584424801592
S2 mod M0 = 67714802256312515151815956253
S3 mod M0 = 56643977663632869177717526464
S4 mod M0 = 156616060820652435419136000
S5 mod M0 = 59644519157598174966919921875
S6 mod M0 = 57378393814212352734448877568
S7 mod M0 = 55149280416459675613872529973
S8 mod M0 = 68594622029459328229596758016
S9 mod M0 = 57374897803533577919325044736
S10 mod M0 = 169218718979961291977340369
S11 mod M0 = 67748137198449240018113298432
```