

Федеральное государственное автономное образовательное учреждение высшего
образования
«Национальный исследовательский университет ИТМО»
Факультет программной инженерии и компьютерной техники

Лабораторная работа 5

«Атака на алгоритм шифрования RSA методом бесключевого чтения»

Вариант № 12

Группа: Р34102

Выполнил: Лапин А.А.

Проверил:
Рыбаков С.Д.

Санкт-Петербург
2024г.

Оглавление

Введение	3
Ход работы	4
Программная реализация	4
Результаты работы программы	6
Заключение	8

Введение

Цель работы: изучить атаку на алгоритм шифрования RSA посредством метода бесключевого чтения.

Текст задания

Вариант	Модуль, N	Экспоненты		Блок зашифрованного текста	
		e1	e2	C1	C2
12	385751370271	365797	1109663	58541562205	78032032470
				167003685579	13064174635
				381877628242	326727914830
				256218527098	364066420370
				164244249864	177576861402
				6588741823	65863828523
				180308234660	111437045566

Ход работы

Будем строить последовательность: $c_1 = c$, $c_i = c_{i-1}^e \bmod N, i > 1$.

$N = 385751370271$	\Rightarrow	$c_1 = m^{e_1} \bmod N$
$e_1 = 365797$		$c_2 = m^{e_2} \bmod N$
$e_2 = 1109663$		Используя расширенный алгоритм Евклида, находим r и s , такие что $re_1 + se_2 = 1$
$c_1 = 58541562205...$		Тогда $m = c_1^r c_2^s \bmod N$
$c_2 = 78032032470...$		

Программная реализация

Listing 1: main.py

```
1 import math
2 from omegaconf import DictConfig
3 import hydra
4 from tqdm import tqdm
5 import logging
6
7 logger = logging.getLogger(__name__)
8
9
10 def int_to_bytes(m):
11     """
12     Convert an integer to bytes.
13     """
14     hex_str = hex(m)[2:]
15     if len(hex_str) % 2:
16         hex_str = '0' + hex_str
17     return bytes.fromhex(hex_str)
18
19
20 def extended_gcd(a, b):
21     """
22     Extended Euclidean Algorithm.
23     Returns a tuple of (gcd, x, y), where gcd is the gcd of a and b,
24     and x, y satisfy the equation: a*x + b*y = gcd
25     """
26     if a == 0:
27         return (b, 0, 1)
28     else:
29         gcd, x1, y1 = extended_gcd(b % a, a)
30         x = y1 - (b // a) * x1
31         y = x1
32         return (gcd, x, y)
33
34
35 def reading_attack(y1, e1, y2, e2, N):
```

```

36 """
37 Perform RSA cryptanalysis using the Reading Attack (Бесключевое чтение).
38 Given two ciphertexts  $y_1 = x^{e_1} \bmod N$  and  $y_2 = x^{e_2} \bmod N$ ,
39 recover the original plaintext  $x$ .
40 """
41 gcd, r, s = extended_gcd(e1, e2)
42 if gcd != 1:
43     logger.error(
44         "Exponents e1 and e2 are not coprime. Attack cannot be performed.")
45     return None
46
47 # Compute  $x = (y_1^r * y_2^s) \bmod N$ 
48 x = (pow(y1, r, N) * pow(y2, s, N)) % N
49
50 logger.debug(f"Recovered plaintext x: {x}")
51 return x
52
53
54 @hydra.main(version_base=None, config_path=".", config_name="config")
55 def main(cfg: DictConfig):
56     N = cfg.N
57     e1 = cfg.e1
58     e2 = cfg.e2
59     c1 = cfg.c1
60     c2 = cfg.c2
61
62     print(f"N = {N}")
63     print(f"e1 = {e1}")
64     print(f"e2 = {e2}")
65     print(f"c1 = {c1}")
66     print(f"c2 = {c2}")
67
68     # Perform Reading Attack on each ciphertext pair
69     print("Performing Reading Attack on ciphertext pairs...")
70     decrypted_bytes = b''
71
72     for idx, (y1, y2) in tqdm(enumerate(zip(c1, c2), start=1), total=len(c1), desc="
73         Decrypting Ciphertext Pairs"):
74         logger.debug(f"Decrypting ciphertext pair {idx}: y1={y1}, y2={y2}")
75         plaintext_int = reading_attack(y1, e1, y2, e2, N)
76         if plaintext_int is None:
77             print(f"Failed to decrypt ciphertext pair {idx}.")
78             continue
79         decrypted_bytes += int_to_bytes(plaintext_int)
80
81     try:
82         plaintext = decrypted_bytes.decode('cp1251')
83         print(f"Plaintext: {plaintext}")
84     except UnicodeDecodeError:
85         print("Decrypted bytes could not be decoded to cp1251. Raw bytes:")
86         print(decrypted_bytes)
87
88 if __name__ == "__main__":
89     main()

```

```

1 # RSA Configuration Parameters
2
3 # The modulus N, which is the product of two primes p and q.
4 N: 385751370271
5
6 # The public exponent e.
7 e1: 365797
8 e2: 1109663
9
10 # The list of ciphertext blocks to be decrypted.
11 c1:
12 - 58541562205
13 - 167003685579
14 - 381877628242
15 - 256218527098
16 - 164244249864
17 - 6588741823
18 - 180308234660
19 - 174572441677
20 - 259951955034
21 - 378589342820
22 - 319378579620
23 - 21405495597
24 - 226860843155
25 c2:
26 - 78032032470
27 - 13064174635
28 - 326727914830
29 - 364066420370
30 - 177576861402
31 - 65863828523
32 - 111437045566
33 - 124743274954
34 - 119577259869
35 - 85769669875
36 - 4688914942
37 - 261002397567
38 - 341722428571

```

Результаты работы программы

Listing 3: Вывод в консоль

```
1 > python main.py
2 N = 385751370271
3 e1 = 365797
4 e2 = 1109663
5 c1 = [58541562205, 167003685579, 381877628242, 256218527098, 164244249864,
        6588741823, 180308234660, 174572441677, 259951955034, 378589342820,
        319378579620, 21405495597, 226860843155]
6 c2 = [78032032470, 13064174635, 326727914830, 364066420370, 177576861402,
        65863828523, 111437045566, 124743274954, 119577259869, 85769669875, 4688914942,
        261002397567, 341722428571]
7 Performing Infinite Reading Attack on ciphertext pairs...
8 Decrypting Ciphertext Pairs: ████████████████████████████████████████100%|| 13/13
   [00:00<00:00, 53667.28it/s]
```

9 Plaintext: из исходного пакета, в котором переданы эти 900 б.

Заключение

В ходе выполнения лабораторной работы была реализована атака на алгоритм шифрования RSA методом бесключевого чтения.