

Правительство Российской Федерации  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ  
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»  
(НИУ ВШЭ)

Московский институт электроники и математики им. А.Н. Тихонова

ОТЧЕТ  
О ПРАКТИЧЕСКОЙ РАБОТЕ № 3  
по дисциплине «Математические основы защиты информации»  
RSA

Студент гр. БИБ 191

\_\_\_\_\_  
И.Г. Тюрин  
" " \_\_\_\_\_ 2021 г.

"

\_\_\_\_\_  
Руководитель  
Заведующий кафедрой информационной  
безопасности киберфизических систем  
канд. техн. наук, доцент

\_\_\_\_\_  
О.О. Евсютин  
" " \_\_\_\_\_ 2021 г.

Москва 2021

## СОДЕРЖАНИЕ

|   |    |
|---|----|
| 1. Задание на практическую работу .....   | 3  |
| 2. Краткая теоретическая часть .....      | 3  |
| 2.1. Описание шифров .....                | 3  |
| 2.2. Методы криптоанализа шифров .....    | 4  |
| 3. Примеры шифрования .....               | 7  |
| 4. Программная реализация шифров .....    | 9  |
| 5. Примеры криптоанализа .....            | 10 |
| 6. Выводы о проделанной работе .....      | 10 |
| 7. Список использованных источников ..... | 12 |

## 1. Задание на практическую работу

Целью данной работы является исследование асимметричной криптосистемы RSA, основанной на проблеме факторизации целых чисел. Задание:

- 1) написать программную реализацию криптосистемы RSA;
- 2) изучить методы криптоанализа криптосистемы RSA;
- 3) реализовать (вручную или программно) не менее одной атаки на криптосистему RSA для случая, когда параметры криптосистемы не являются большими числами;
- 4) подготовить отчет о выполнении работы.

Программа должна обладать следующей функциональностью:

- 1) принимать на вход произвольную последовательность символов, вводимую пользователем в качестве открытого текста или шифртекста;
- 2) принимать на вход ключевую пару (открытый ключ, закрытый ключ);
- 3) давать пользователю возможность сгенерировать ключевую пару;
- 4) осуществлять зашифрование или расшифрование введенного текста по выбору пользователя.

## 2. Краткая теоретическая часть

### 2.1. Криптография с открытым ключом

Концепция криптографии с открытым ключом появилась в середине 1970-х годов как возможное решение проблемы распределения ключей шифрования, актуальной для симметричных алгоритмов шифрования.

Криптосистемы с открытым ключом, также называемые асимметричными криптосистемами, используют два разных ключа: открытый ключ зашифрования и закрытый ключ расшифрования. Открытый ключ в общем случае доступен всем желающим, а закрытый ключ известен только законному владельцу. Оба ключа связаны между собой некоторой математической зависимостью. При этом данная зависимость такова, что, зная один ключ, вычислить другой практически невозможно.

Первые криптографические алгоритмы с открытым ключом основаны на задаче факторизации целых чисел.

### 2.2. Криптосистема RSA

Данная криптосистема является первой криптосистемой с открытым ключом. Она основывается на сложности проблемы факторизации целых чисел, то есть чисел на простые множители.

*Алгоритм генерации ключей.*

1. Пользователь А генерирует два больших простых числа  $p$  и  $q$ , отличных друг от друга. При этом  $|p - q|$  – большое число, хотя  $p$  и  $q$  имеют приблизительно одинаковый битовый размер.

2. Держа  $p$  и  $q$  в секрете, Пользователь А вычисляет их произведение  $n = pq$ , которое называют модулем алгоритма.

3. Пользователь А вычисляет значение функции Эйлера для  $n$  по формуле  $\varphi(n) = (p - 1)(q - 1)$ .

4. Пользователь А выбирает целое число  $e$ , взаимно простое со значением функции  $\varphi(n)$ . Это число называется экспонентой зашифрования.

5. Пользователь А применяет расширенный алгоритм Евклида к паре чисел  $e$  и  $\varphi(n)$  и вычисляет значение  $d$ , удовлетворяющее соотношению  $ed \equiv 1 \pmod{\varphi(n)}$ . Это значение называется экспонентой расшифрования.

6. Пара  $(e, n)$  публикуется в качестве открытого ключа пользователя А,  $d$  является закрытым ключом и держится в секрете.

#### *Алгоритм зашифрования.*

1. Пользователь В получает аутентичную копию открытого ключа пользователя А – пару  $(e, n)$ .

2. Пользователь В представляет сообщение в виде числа  $m$ , меньшего модуля алгоритма. В общем случае сообщение может быть разбито на блоки, каждый из которых представляется своим числом.

3. Пользователь В вычисляет  $c = m^e \pmod{n}$ .

4. Зашифрованное сообщение отправляется пользователю А.

#### *Алгоритм расшифрования.*

Московский институт электроники и математики им. А.М. Тихонова

1. Пользователь А получает криптограмму  $c$  от пользователя В.

2. Пользователь А вычисляет  $m = c^d \pmod{n}$ .

### **2.3. Нахождение обратного элемента по модулю простого числа**

Для нахождения обратного элемента по модулю натурального числа, что необходимо при вычислении экспоненты расшифрования, применяется расширенный алгоритм Евклида.

Вход: целые числа  $a \geq b > 0$ .

Выход:  $d = \text{НОД}(a, b)$  и целые  $x, y$ , такие, что  $ax + by = d$ .

1. Полагаем  $x_2 \leftarrow 1, x_1 \leftarrow 0, y_2 \leftarrow 0, y_1 \leftarrow 1$ .

2. Пока  $b > 0$ , выполнять следующее:

2.1.  $q \leftarrow [a/b], r \leftarrow a - qb, x \leftarrow x_2 - qx_1, y \leftarrow y_2 - qy_1$ ;

2.2.  $a \leftarrow b, b \leftarrow r, x_2 \leftarrow x_1, x_1 \leftarrow x, y_2 \leftarrow y_1, y_1 \leftarrow y$ .

3.  $d \leftarrow a$ ,  $x \leftarrow x^2$ ,  $y \leftarrow y^2$  и возврат  $(d, x, y)$ .

Чтобы найти  $a^{-1} \bmod n$ , необходимо подать на вход алгоритма Евклида пару  $n, a$ , и если  $\text{НОД}(a, n) = 1$ , вернуть в качестве  $a^{-1}$  значение  $y^2$ .

## 2.4. Нахождение обратного элемента по модулю простого числа

Криптографические алгоритмы с открытым ключом при их использовании на практике оперируют числами большой битовой длины (или просто большими числами), когда речь идет о сотнях и тысячах бит. Для некоторых операций над такими числами созданы специальные алгоритмы. В случае криптосистемы RSA необходимо иметь алгоритм, который позволит осуществлять быстрое возведение в степень по модулю.

Данный алгоритм представлен ниже.

Алгоритм возведения в степень по модулю.

Вход:  $a, k \in \mathbb{Z}_n$ ,  $k = \sum_{i=0}^t k_i \cdot 2^i$

Выход:  $a^k \bmod n$ .

1.  $b \leftarrow 1$ . Если  $k = 0$ , то переход к шагу 5.

2.  $A \leftarrow a$ .

3. Если  $k_0 = 1$ , то  $b \leftarrow a$ .

4. Для  $i = (1, t)$  выполняем следующее:

4.1.  $A \leftarrow A^2 \bmod n$ .

4.2. Если  $k_i = 1$ , то  $b \leftarrow (A \cdot b) \bmod n$ .

5. Возврат  $b$ .

Еще одним важным аспектом криптографии с открытым ключом является использование простых чисел, в частности, как было рассмотрено, в криптосистеме RSA элементом открытого ключа является произведение двух больших простых чисел  $n = pq$

Наиболее развитые вероятностные алгоритмы проверки чисел на простоту основаны на малой теореме Ферма.

Малая теорема Ферма.

Пусть  $p$  — простое число,  $a \neq 0$  и  $a \in \mathbb{Z}_p$ . Тогда  $a^{p-1} \equiv 1 \pmod{p}$ .

Соотношение, приведенное в теореме, используется в тесте, проверяющем, является ли заданное число составным. Этот тест называют тестом Ферма.

Тест Ферма.

Вход: нечетное число  $n$ .

Выход: ответ на вопрос «является ли  $n$  простым».

1. Для  $i = (1, t)$  выполняем следующее:

1.1. Выбираем случайное целое число  $a \in [2; n-1]$ .

1.2. Вычисляем  $r = a^{n-1} \bmod n$  с помощью алгоритма возведения в степень по модулю.

1.3. Если  $r \neq 1$ , то возврат « $n$  — составное».

Тест Ферма по основанию  $a$  определяет простоту  $n$  с вероятностью  $1/2$ , после  $t$  итераций вероятность ошибки составляет  $1/2^t$ .

### 3. Примеры шифрования

#### 3.1. Шифрование по буквам

В качестве примера рассмотрим небольшие числа, но на практике используют числа длины 1024 и больше. Будем передавать сообщение: «ARTISTICALLY».

Представим открытый текст в виде последовательности чисел из диапазона 1...26.

Открытый текст: ARTISTICALLY

Закодированный текст: [1, 18, 20, 9, 19, 20, 9, 3, 1, 12, 12, 25]

Пусть  $p = 199$ ,  $q = 167$ . Тогда  $N = 33233$ , а функция Эйлера  $\varphi(n) = (p - 1)(q - 1) = 32868$ . В качестве шифрующей экспоненты выбирают число, взаимно простое с  $\varphi(n)$ . Положим,  $E = 65537$ :  $\text{НОД}(65537, 32868) = 1$ . Тогда  $d = 31877$ , т.к.  $d$  – обратный к 65537 элемент в кольце классов вычетов по модулю  $N$ .

Открытый ключ:  $(N, E) = (33233, 65537)$ .

Секретный ключ:  $(d, p, q) = (31877, 199, 167)$ .

#### Зашифрование

Приступаем к зашифрованию сообщения  $m$  по формуле:  $C_i = m_i^E \pmod{N}$ .

$$C_1 = 1^{65537} \pmod{33233} = 1$$

$$C_2 = 18^{65537} \pmod{33233} = 60$$

$$C_3 = 20^{65537} \pmod{33233} = 6$$

$$C_4 = 9^{65537} \pmod{33233} = 9$$

$$C_5 = 19^{65537} \pmod{33233} = 33$$

$$C_6 = 20^{65537} \pmod{33233} = 6$$

$$C_7 = 9^{65537} \pmod{33233} = 9$$

$$C_8 = 3^{65537} \pmod{33233} = 3$$

$$C_9 = 1^{65537} \pmod{33233} = 1$$

$$C_{10} = 12^{65537} \pmod{33233} = 12$$

$$C_{11} = 12^{65537} \pmod{33233} = 12$$

$$C_{12} = 25^{65537} \pmod{33233} = 2$$

Шифртекст: [1, 60, 9, 9, 33, 6, 9, 3, 1, 12, 12, 25]

## Расшифрование

Расшифровать шифртекст можно, обладая секретным ключом. Расшифрование производится по формуле:  $m_i = C_i^d \pmod{N}$ .

$$m_1 = 1^{31877} \pmod{33233} = 1$$

$$m_2 = 60^{31877} \pmod{33233} = 18$$

$$m_3 = 6^{31877} \pmod{33233} = 20$$

$$m_4 = 9^{31877} \pmod{33233} = 9$$

$$m_5 = 33^{31877} \pmod{33233} = 19$$

$$m_6 = 6^{31877} \pmod{33233} = 20$$

$$m_7 = 9^{31877} \pmod{33233} = 9$$

$$m_8 = 3^{31877} \pmod{33233} = 3$$

$$m_9 = 1^{31877} \pmod{33233} = 1$$

$$m_{10} = 12^{31877} \pmod{33233} = 12$$

$$m_{11} = 12^{31877} \pmod{33233} = 12$$

$$m_{12} = 25^{31877} \pmod{33233} = 25$$

Получим исходный текст:

Полученный текст: [1, 18, 20, 9, 19, 20, 9, 3, 1, 12, 12, 25]

Переведём обратно в текст: «ARTISTICALLY».



## 4. Программная реализация шифров

Результаты работы программы для рассмотренного примера:

### 1) Шифр RSA

Открытый текст: ARTISTICALLY



```
23
24 message = b'ARTISTICALLY'
25
26 encrypted = rsa.encrypt(message, public_key) # Шифрование
27 print(encrypted)
28 message = rsa.decrypt(encrypted, private_key) # Расшифрование
29 print(message)
30
```

RSA x

C:\Users\user-pc\PycharmProjects\pythonProject\venv\Scripts\python.exe C:\Users\user-pc\Desktop\МО3И\MF-of-IS\practic\_4\RSA.py  
b'\x02\xd9\x1a\x11\xfa\x95\x1f\x88(\xe4\xeaV\xfa\x02\xec\xfa\x03\x02.SMI\xe3'\xcb(\xa1\x04\x01[\xc9t-\xc5GtT\x8a\x98~\xa8\xe4\xcf\x9a\xe7\xa1{\x1fV\xbc\xdcz\xfa3\x977\x01h'  
b'ARTISTICALLY'

Рисунок 4.1 – Результаты работы Алгоритма, реализующего шифр RSA

Ссылка на код: [https://github.com/TyurinIvan/MF-of-IS/blob/master/practic\\_4/RSA.py](https://github.com/TyurinIvan/MF-of-IS/blob/master/practic_4/RSA.py)

## 5. Криптоанализ шифра RSA

### Атака на RSA (атака Виенера, Wiener attack)

#### Теоретическая часть

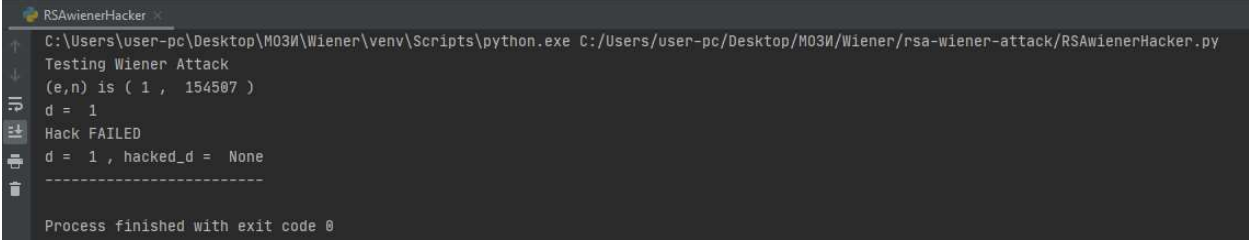
Рассматривается открытый RSA ключ  $(e, N)$ , по которому необходимо определить открытую экспоненту  $d$ . Если известно, что  $d < 1/3 N^{1/4}$ , то это возможно сделать по следующему алгоритму:

1. Разложить дробь  $e/N$  в непрерывную дробь  $[a_1, a_2, \dots]$ .
  2. Для непрерывной дроби  $[a_1, a_2, \dots]$  найти множество всех возможных подходящих дробей  $k_n/d_n$ .
  3. Исследовать подходящую дробь  $k_n/d_n$ .
    - 3.1. Определить возможное значение  $\phi(N)$ , вычислив  $(ed_n - 1)/k_n$ .
    - 3.2. Решив уравнение  $x^2 - ((N - \phi(N) + 1)x + N = 0$ , получить пару корней  $(p_n, q_n)$ .
  4. Если для пары корней  $(p_n, q_n)$  выполняется равенство  $N = p_n * q_n$ , то закрытая экспонента найдена  $d = d_n$ .
- Если условие не выполняется или не удалось найти пару корней, то необходимо исследовать следующую подходящую дробь, вернувшись к шагу 3.

#### Практическая часть

Ссылка на код: [https://github.com/TyurinIvan/MF-of-IS/blob/master/practic\\_4/RSAwiennerHacker.py](https://github.com/TyurinIvan/MF-of-IS/blob/master/practic_4/RSAwiennerHacker.py)

Пример атаки Виенера можно реализовать следующим образом:



```
RSAwiennerHacker
C:\Users\user-pc\Desktop\М03И\Wiener\venv\Scripts\python.exe C:/Users/user-pc/Desktop/М03И/Wiener/rsa-wiener-attack/RSAwiennerHacker.py
Testing Wiener Attack
(e,n) is ( 1 , 154507 )
d = 1
Hack FAILED
d = 1 , hacked_d = None
-----
Process finished with exit code 0
```

Рисунок 5.1 – Атака Виенера на шифр RSA с малыми параметрами (16 бит)

## **6. Выводы о проделанной работе**

В результате выполнения данной работы я могу подвести следующие итоги:

Криптосистема RSA, первая криптосистема асимметричного шифрования, выдержала испытание временем и по сей день активно используются во всем мире. Это можно считать аргументом в пользу криптостойкости данной системы. Хотя аргумент этот слабый, ведь даже если задача RSA эквивалентна проблеме факторизации (а это утверждение не доказано), то с развитием вычислительной техники разумным решением будет отказаться от этой криптосистемы.

Кроме того, даже с настоящим положением вычислительной мощности компьютеров необходимо быть на чеку и тщательно подбирать параметры шифрования. Изучив методы криптоанализа системы RSA, я могу сделать следующие выводы:

- 1) Нельзя использовать общий модуль  $n$  для группы пользователей
- 2) Сообщения перед шифрованием необходимо дополнять случайными числами
- 3) Показатель дешифрования должен быть большим.

## **7. Список использованных источников**

1. Традиционные шифры с симметричным ключом [электронный ресурс] – URL: [https://ru.wikipedia.org/wiki/Атака\\_Винера](https://ru.wikipedia.org/wiki/Атака_Винера)
2. Cryptanalysis of the Hill Cipher [электронный ресурс] – URL: <http://practicalcryptography.com/cryptanalysis/stochastic-searching/cryptanalysis-hill-cipher/>
3. Хилл шифр – Hill cipher [электронный ресурс] – URL: [https://ru.qaz.wiki/wiki/Hill\\_cipher](https://ru.qaz.wiki/wiki/Hill_cipher)