RSA para programa de generación de licencias de software
RSA for software license generation program

Milena Flores https://orcid.org/sites/default/files/images,

[1] Yachay Tech, Urcuquí, Ecuador; milena.flores@yachaytech.edu.ec

Send: 2025/06/03

RESUMEN

Este proyecto se centra en el desarrollo de un programa criptográfico que integra el algoritmo RSA. El algoritmo RSA, un sistema de cifrado asimétrico contemporáneo, basa su seguridad en la factorización de números primos y la aritmética modular. El objetivo principal de este proyecto es utilizar el método de cifrado RSA para generar, verificar y descifrar licencias de software.

Palabras clave: Algoritmo RSA , Encriptación, Licencia de software.

ABSTRACT

This project focuses on the development of a cryptographic program that integrates RSA algorithm. The RSA algorithm, a contemporary asymmetric encryption system, bases its security on prime number factorization and modular arithmetic. The main objetive of this project is use the RSA encryption method for the purpose of generate, verify and decrypt software licenses.

Keywords: RSA algortihm, Encryptation, Software license.

1 INTRODUCTION

In today's digital age, information security has become a fundamental aspect of everyday life. Cryptography is a discipline that studies message encryption techniques. This is responsible for guaranteeing the security of the data through different encryption techniques. Encryption is of great importance, as it helps secure data such as passwords or even bank details.

A mathematical algorithm is a type of encryption based on the use of mathematical methods to encrypt messages or classified information. In this specific project, we will use the RSA method to encript information as name and date to generate a license of software.

We need to introduce about what is RSA algorithm:

The RSA algorithm in cryptography is one of the most used and secured algorithm because is based on factorizing large prime numbers. In general terms this algorithm asks for two prime numbers commonly called p and q, then multiply them and get n=p*q. This n will be a public and private number which will help us to encrypt the message. The way to decrypt the message is according to a number of mathematical procedures that consist of obtaining new numbers that are going to be private. After that, the private number will help to decrypt the number.

Let us see step by step the algorithm and how it works:

1. Chosse to numbers p and q.

2. Calculated

$$n = p * q \tag{1}$$

3. Using the fundamental property of Euler in RSA, calculate

$$\phi(n) = (p-1)*(q-1) \tag{2}$$

4. Chosse an e such that

$$1 < e < \phi(n) \tag{3}$$

and is coprime to phi

$$gdc(e, \phi(n)) = 1 \tag{4}$$

so, they do not share any divisor except 1.

5. Calculated a d such that it is the inverse of e modulo phi

$$d = inv(e, \phi(n)) \tag{5}$$

i.e., we want to search for a number such that

$$(d*e)mod(\phi(n)) = 1 \Leftrightarrow (d*e) = 1(\phi(n)) \tag{6}$$

6. Now, we have the private and public key that will be:
Public key : (e,n) and a private key : (d,n)

7. To encrypt a message we will use the public key like this:

$$c = m^e mod(n) \tag{7}$$

Where,
m : Is the original message represented by a number.
e,n : Are public keys.
c : Is the encrypted message.

8. To decrypt the message we will use the private keys like this:

$$m = c^d mod(n) \tag{8}$$

Where,
m : Is the original message represented by a number.
e,n : Are public keys.
c : Is the encrypted message.
This algorithm works thanks to Euler's and Fermat's theorems applied to encryption.

Now, let us introduce what is an software license: Most software is distributed under a license that serves as a legal agreement between the producer and the user, defining the rights and obligations of each party, this license is a software license that give the permissions to the user to use the software. In this project we will introduce a way to encrypt information of the user to generate a number license.

1.1 Objectives

The objetive of this project is use the RSA encryption method for the purpose of generate, verify, and decrypt software licenses.

2 MATERIALS Y METHODS

2.1 Materials

For this project we use C++ version 11 and standard libraries such us:

- <iostream>
- <string>
- <ctime>
- <limits>

2.2 Methods

For this project we create a console where the following options are shown: generate a license, verify a license, show the public key and decrypt a license. For each of the options we define the following functions:

- First, take in mind that we define the constants: p, q, n, phi, e and d.
- modPow(long base, long exp, long mod): This function is part of the RSA algoritm, this helps to calculate the encrypted and desencryted number.
- calculateAsciiSum(name): This function add all the ASCII values of name.
- showPublicKey(): This function prints on the console the public key of RSA (n,e).
- generateLicense(): This fucntion ask for a username and expiration year, then calculate the variable payload adding the calculateAsciiSum(username) and the expiration year, then with the function modPow generate the license.
- verifyLicense(): This function asks for username, expected year, and a license code, then calculates the expected license and compares with the license given.

- decryptMessage(): This function ask for a license and with modPow decrypt the information.

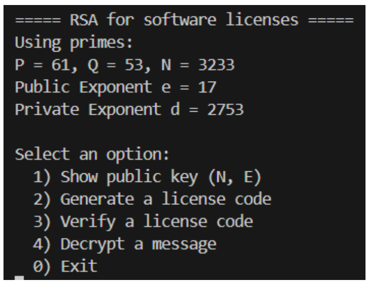
Then using those functions we create a main were we put the console with thw options and each option have one of the above specific functions.

3 RESULTS Y DISCUSSION

As part of the results, the way in which the console acts will be included when in each of its options with an analysis of these results.

3.1 Principal results

First, the console of the program will be:

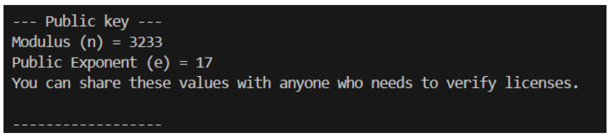


```
===== RSA for software licenses =====
Using primes:
P = 61, Q = 53, N = 3233
Public Exponent e = 17
Private Exponent d = 2753

Select an option:
  1) Show public key (N, E)
  2) Generate a license code
  3) Verify a license code
  4) Decrypt a message
  0) Exit
```

Figura 1*Principal console of the program*

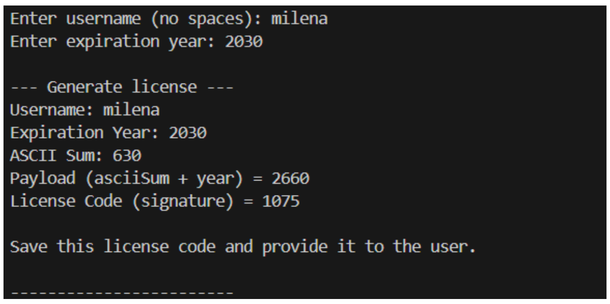
If we select the different option this will be what we have:



```
--- Public key ---
Modulus (n) = 3233
Public Exponent (e) = 17
You can share these values with anyone who needs to verify licenses.

-----------------
```

Figura 2*Option 1. Show public key*



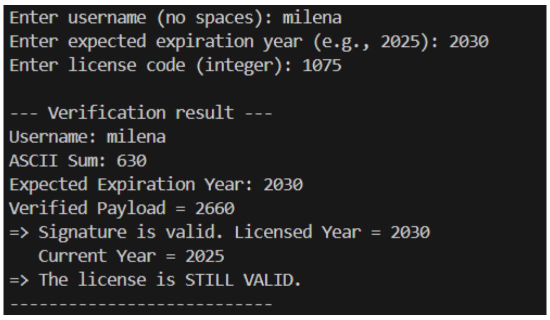
```
Enter username (no spaces): milena
Enter expiration year: 2030

--- Generate license ---
Username: milena
Expiration Year: 2030
ASCII Sum: 630
Payload (asciiSum + year) = 2660
License Code (signature) = 1075

Save this license code and provide it to the user.

------------------------
```

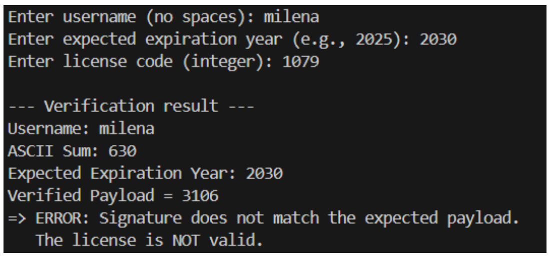
Figura 3*Option 2. generate a license code*



```
Enter username (no spaces): milena
Enter expected expiration year (e.g., 2025): 2030
Enter license code (integer): 1075

--- Verification result ---
Username: milena
ASCII Sum: 630
Expected Expiration Year: 2030
Verified Payload = 2660
=> Signature is valid. Licensed Year = 2030
   Current Year = 2025
=> The license is STILL VALID.
--------------------------
```

(a) *Option 3 with positive result*

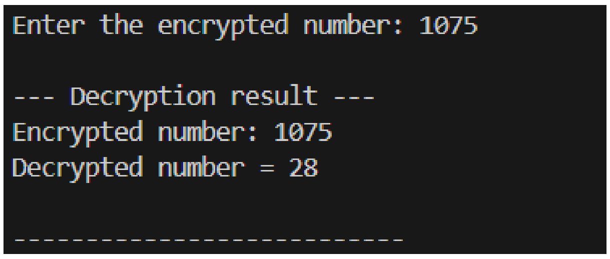


```
Enter username (no spaces): milena
Enter expected expiration year (e.g., 2025): 2030
Enter license code (integer): 1079

--- Verification result ---
Username: milena
ASCII Sum: 630
Expected Expiration Year: 2030
Verified Payload = 3106
=> ERROR: Signature does not match the expected payload.
   The license is NOT valid.
```

(b) *Option 3 with negative result*

Figura 4*Option 3. Verify a license code*



```
Enter the encrypted number: 1075

--- Decryption result ---
Encrypted number: 1075
Decrypted number = 28


--------------------------
```

Figura 5*Option 4. Decrypt a message*

3.2 Discussion

The results obtained meet the proposed objective as it uses the RSA encryption method to generate, verify and decrypt simple license codes. Thus, we can say that the program successfully builds the licenses through functions such as modPow, calculateAsciiSum, generateLicense, etc. Specifically, the modPow function is the implementation of RSA in the code.

Among the strengths of the code that was implemented, it is worth noting that each step and/or option provides the user with a clear understanding and transparency about what is happening. In addition, this code is easy to compile and execute.

However, a detected limitation is that since small prime numbers p and q were used, the code is not truly secure. In real situations, these should be much larger prime numbers in order to be difficult to find. In addition, permanent storage could be implemented to manage licenses.

4 CONCLUSION

In conclusion, we can say that we have achieved our goal of implementing the RSA encryption method to generate software licenses; however, the program is not truly secure given the limitations outlined above.

Future implementations of the program could include incorporating larger numbers, implementing data storage, and having greater control over licenses.

REFERENCES

S. Nadeem. “Cifrado simétrico vs. asimétrico: ¿cuál es la diferencia? - Mailfence Blog”. Mailfence Blog. Accedido el 6 de abril de 2025. [En línea]. Disponible: https://blog.mailfence.com/es/cifrado-simetrico-vs-asimetrico/: :text=La

O. Knill. “Lecture 11: Cryptography”. Harvard Mathematics Department : Home page. Accedido el 6 de abril de 2025. [En línea]. Disponible: https://people.math.harvard.edu/ knill/teaching/mathe320$_2$022/
cryptology.pdf

“Basics of Mathematical Cryptography”. Medium. Accedido el 6 de abril de 2025. [En línea]. Disponible: https://medium.com/intuition/basics-of-mathematical-cryptography-408fdee23826

Dorostkar, Zahra. (2023). Mathematics for Cryptography: A Guide to Mathematical Fundamentals of Different Classes of Cryptography Algorithms.

P. Gupta. "The Mathematical Foundations of Cryptography". Accedido el 6 de abril de 2025. [En línea]. Disponible:https://biolecta.com/articles/mathematics-in-cryptography/