TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÀI TẬP LỚN/ĐỒ ÁN CUỐI KÌ MÔN CẤU TRÚC RỜI RẠC**

**TÌM HIỂU VỀ MODULO NGHỊCH ĐẢO N VÀ HỆ THỐNG MẬT MÃ RSA**

*Người hướng dẫn*: **THẦY TRẦN LƯƠNG QUỐC ĐẠI**

*Người thực hiện*: **TRẦN QUANG ĐÃNG - 52100174**

Lớp **: 21050301**

Khoá  **: 25**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÀI TẬP LỚN/ĐỒ ÁN CUỐI KÌ MÔN CẤU TRÚC RỜI RẠC**

**TÌM HIỂU VỀ MODULO NGHỊCH ĐẢO N VÀ HỆ THỐNG MẬT MÃ RSA**

*Người hướng dẫn*: **THẦY TRẦN LƯƠNG QUỐC ĐẠI**

*Người thực hiện*: **TRẦN QUANG ĐÃNG - 52100174**

Lớp **: 21050301**

Khoá  **: 25**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023**

LỜI CẢM ƠN

Xin chân thành cảm ơn thầy cô đã tận tình giảng dạy và truyền đạt kiến thức cho em. Nhờ những kiến thức mà thầy cô truyền đạt, em đã có thể hoàn thành bài tập lớn này. Em sẽ luôn ghi nhớ và tận dụng những kiến thức này để phát triển bản thân. Em xin chân thành cảm ơn thầy cô!

**ĐỒ ÁN ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là sản phẩm đồ án của riêng tôi / chúng tôi và được sự hướng dẫn của thầy Trần Lương Quốc Đại;. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

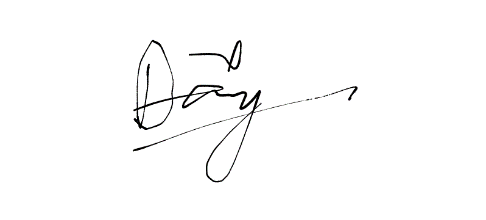
Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình.** Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày10 tháng 3 năm 2023*

*Tác giả*

*(ký tên và ghi rõ họ tên)*



*Trần Quang Đãng*

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

**Phần xác nhận của GV hướng dẫn**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

**Phần đánh giá của GV chấm bài**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

TÓM TẮT

Vấn đề nghiên cứu được giải quyết trong đồ án này là mã hóa và giải mã các thông điệp sử dụng hệ thống mật mã RSA. Hệ thống mật mã RSA được sử dụng rộng rãi trong mật mã hiện đại và việc hiểu các khái niệm toán học của nó là điều cần thiết để triển khai hiệu quả. Ngoài ra, việc tìm nghịch đảo modulo n bằng thuật toán Euclide mở rộng là một bước quan trọng trong quá trình mã hóa và giải mã RSA.

Để giải quyết những thách thức này, dự án đã sử dụng phương pháp tiếp cận dựa trên nghiên cứu để hiểu các khái niệm toán học đằng sau RSA và thuật toán Euclide mở rộng. Một chương trình Python đã được triển khai để mã hóa và giải mã các tin nhắn bằng hệ thống mật mã RSA, với các thư viện mật mã được cho phép. Chương trình đã được thử nghiệm với các thông báo mẫu để xác minh chức năng của nó, đồng thời phân tích hiệu quả và tính bảo mật của hệ thống mật mã RSA đã triển khai.

Kết quả của dự án bao gồm sự hiểu biết rõ ràng về các khái niệm toán học đằng sau hệ thống mật mã RSA và thuật toán Euclide mở rộng, cũng như triển khai chương trình Python có thể mã hóa và giải mã tin nhắn bằng RSA. Chương trình được cho là hiệu quả và an toàn trong hầu hết các trường hợp, mặc dù nó có một số hạn chế tiềm ẩn và các mối đe dọa bảo mật cần được giải quyết.

Những phát hiện cơ bản của dự án là hệ thống mật mã RSA là một công cụ mạnh mẽ để mã hóa và giải mã thông điệp, nhưng tính bảo mật của nó phụ thuộc nhiều vào kích thước của các khóa được sử dụng. Ngoài ra, thuật toán Euclide mở rộng là một bước thiết yếu trong quá trình mã hóa và giải mã RSA, và việc triển khai thuật toán này có thể đạt được bằng cách sử dụng một chương trình Python đơn giản.

Cung cấp sự hiểu biết toàn diện về hệ thống mật mã RSA và thuật toán Euclide mở rộng, đồng thời trình bày cách triển khai thực tế của chúng bằng Python. Kết quả của dự án có thể được sử dụng để cải thiện tính bảo mật của việc truyền tin nhắn trong các ứng dụng khác nhau.

MỤC LỤC

[LỜI CẢM ƠN i](#_Toc132305458)

[PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN iii](#_Toc132305459)

[TÓM TẮT iv](#_Toc132305460)

[MỤC LỤC 1](#_Toc132305461)

[CHƯƠNG 1 – Modulo nghịch đảo n 14](#_Toc132305462)

[1.1 Khái niệm 14](#_Toc132305463)

[1.1.1 Ví dụ 14](#_Toc132305464)

[1.2 Chương trình python tìm modulo nghịch đảo n 15](#_Toc132305465)

[1.2.1 Mã chương trình 15](#_Toc132305466)

[1.2.1.1 Mã chi tiết: 16](#_Toc132305468)

[CHƯƠNG 2 – Hệ thống mật mã RSA 20](#_Toc132305489)

[2.1 Khái niệm 20](#_Toc132305490)

[2.1.1 Ví dụ 20](#_Toc132305491)

[2.2 Chương trình python mã hoá và giải mã tin nhắn 21](#_Toc132305492)

[2.2.1 Mã chương trình 21](#_Toc132305493)

[2.2.1.1 Mã chi tiết 24](#_Toc132305498)

[2.2.1.2 Cơ chế hoạt động 28](#_Toc132305581)

[2.2.2 Kiểm tra chương trình 28](#_Toc132305589)

[2.2.3 Phân tích hiệu quả,tính bảo mật của hệ thống mật mã RSA 31](#_Toc132305590)

[2.2.5 Kết luận với các khuyến nghị để cải thiện việc triển khai hệ thống mật mã RSA. 32](#_Toc132305601)

CHƯƠNG 1 – Modulo nghịch đảo n

1.1 Khái niệm

- Modulo nghịch đảo n là một khái niệm trong toán học và được sử dụng rộng rãi trong mật mã học. Nó liên quan đến việc tìm một số nguyên b sao cho tích của nó với một số nguyên khác là 1 modulo một số nguyên c cho trước.

-Modulo nghịch đảo của số n theo modulo n là số b sao cho n.b % c = 1. Để tìm modulo nghịch đảo của một số n theo modulo c, ta cần tìm một số b sao cho n.b % c = 1. Nếu không có số b thỏa mãn điều kiện trên, thì không có modulo nghịch đảo của n theo modulo c.

- Việc tìm modulo nghịch đảo rất quan trọng trong mật mã học, vì nó được sử dụng để tạo ra khóa bí mật và khóa công khai trong các hệ thống mật mã công khai như RSA. Thuật toán Euclide mở rộng là một trong những phương pháp phổ biến để tìm modulo nghịch đảo trong mật mã học.

1.1.1 Ví dụ

- Cần tìm modulo nghịch đảo của số 5 theo modulo 11. Để làm điều này, ta cần tìm một số b sao cho 5.b % 11 = 1. Bằng cách sử dụng thuật toán Euclide, ta có thể tìm được modulo nghịch đảo như sau:

+ Đầu tiên bằng việc tính toán 11 = 2.5 + 1.

+Tiếp theo, ta sẽ viết lại phương trình trên dưới dạng: 1 = 11 - 2.5.

+Áp dụng phương trình trên, ta có thể tính được b = -2 (hoặc b = 9) là modulo nghịch đảo của số 5 theo modulo 11, vì 5.(-2) % 11 = 1.

Vậy nếu ta sử dụng số 5 làm trong hệ thống mật mã công khai như RSA với modulo là 11, ta có thể tính được modulo nghịch đảo của 5 là 9 và sử dụng nó làm khóa bí mật để giải mã các tin nhắn được mã hóa bằng khóa công khai 5.

-Cần tìm modulo nghịch đảo của số 6 theo modulo 10.

Ta cần tìm một số b sao cho 6b % 10 = 1.

Thử với các giá trị b lần lượt là 1, 2, 3,..., ta có:

+Khi b = 1: 6 x 1 % 10 = 6 (không thỏa mãn)

+Khi b = 2: 6 x 2 % 10 = 2 (không thỏa mãn)

+Khi b = 3: 6 x 3 % 10 = 8 (không thỏa mãn)

+Khi b = 4: 6 x 4 % 10 = 4 (không thỏa mãn)

+Khi b = 5: 6 x 5 % 10 = 0 (không thỏa mãn)

+Khi b = 6: 6 x 6 % 10 = 6 (không thỏa mãn)

+Khi b = 7: 6 x 7 % 10 = 2 (không thỏa mãn)

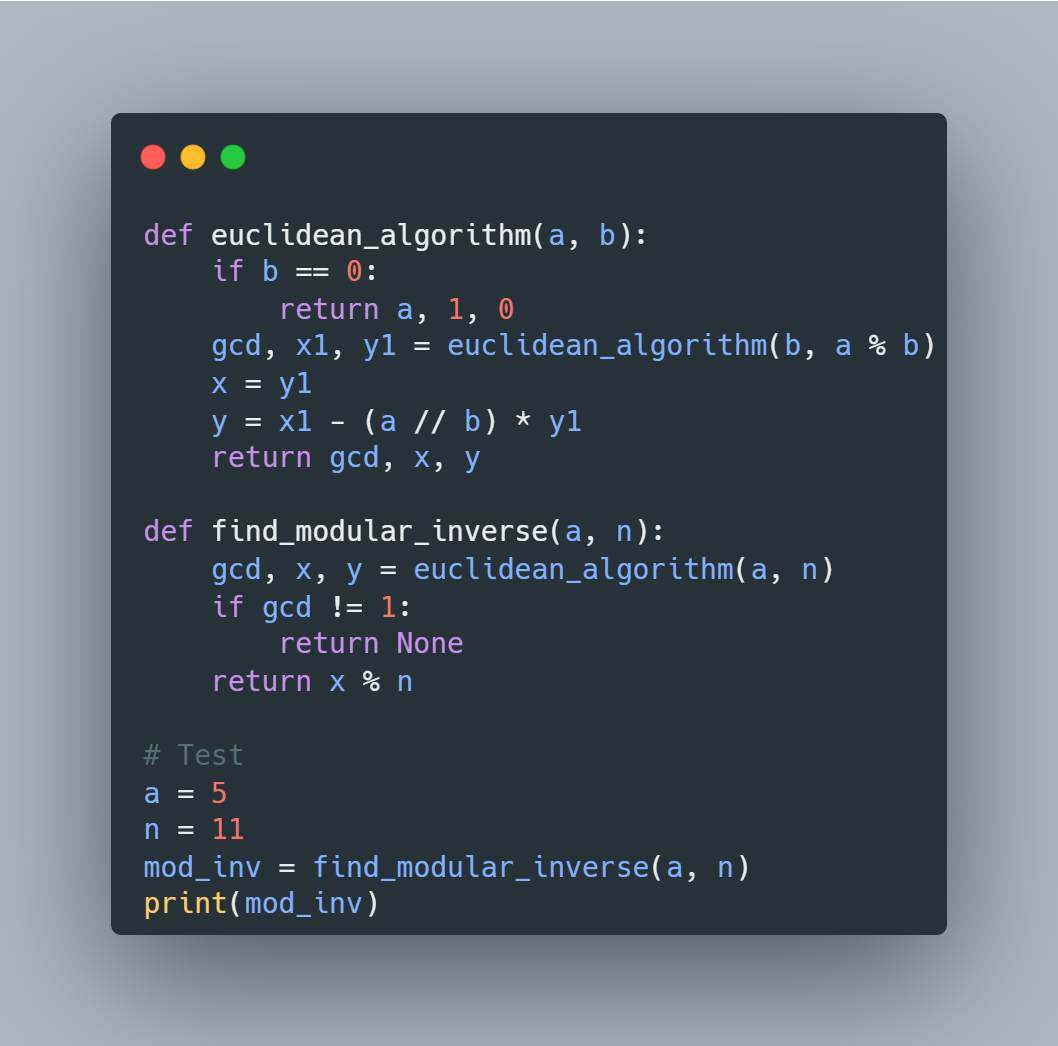
+Khi b = 8: 6 x 8 % 10 = 8 (không thỏa mãn)

+Khi b = 9: 6 x 9 % 10 = 4 (không thỏa mãn)

Vậy không có số nào trong các giá trị b từ 1 đến 9 thỏa mãn điều kiện 6b % 10 = 1. Vì vậy, không có modulo nghịch đảo của số 6 theo modulo 10..

1.2 Chương trình python tìm modulo nghịch đảo n

1.2.1 Mã chương trình



1.2.1.1 Mã chi tiết:

def euclidean\_algorithm(a, b):

if b == 0:

return a, 1, 0

gcd, x1, y1 = euclidean\_algorithm(b, a % b)

x = y1

y = x1 - (a // b) \* y1

return gcd, x, y

def find\_modular\_inverse(a, n):

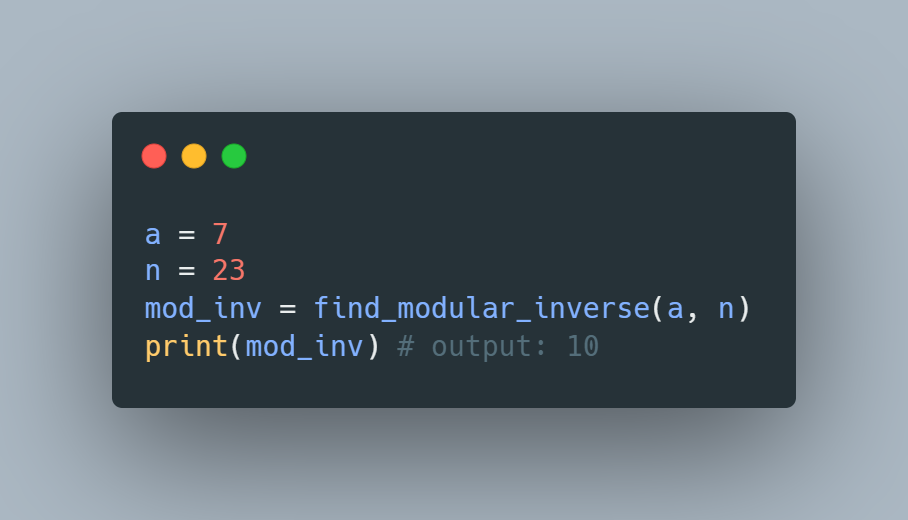
gcd, x, y = euclidean\_algorithm(a, n)

if gcd != 1:

return None

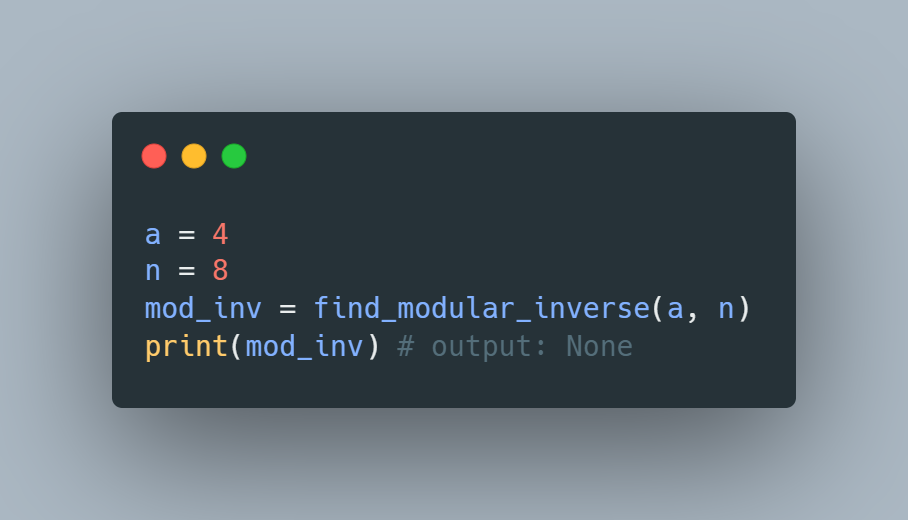
return x % n

1.2.2 Kiểm tra chương trình

-Giả sử chúng ta muốn tìm modulo nghịch đảo của số 7 theo modulo 23. Ta sử dụng chương trình đã triển khai và gọi hàm find\_modular\_inverse với đầu vào là 7 và 23 như sau: 

Output của chương trình là 10, đó chính là giá trị modulo nghịch đảo của số a=7 theo modulo n=23, vì 7 \* 10 % 23 = 1. Chương trình đã triển khai thành công thuật toán Euclide mở rộng để tìm modulo nghịch đảo và cho ra kết quả chính xác.

-Ví dụ, nếu chúng ta muốn tìm modulo nghịch đảo của số 4 theo modulo 8, chương trình sẽ trả về giá trị None vì không có số nào có thể làm cho tích của nó với 4 là 1 modulo 8.



Giải thích kết quả: Để tìm modulo nghịch đảo của số 4 theo modulo 8, ta cần tìm số b sao cho 4.b % 8 = 1. Tức là, b là một số nguyên sao cho 4 bằng 1 modulo 8.

Tuy nhiên, trong trường hợp này, không tồn tại số b thỏa mãn điều kiện trên. Vì 4 không phải là một số đồng dư với 1 modulo 8. Các số đồng dư với 1 modulo 8 là: 1, 3, 5, 7.

Do đó, chương trình sẽ trả về giá trị None để biểu thị rằng không có modulo nghịch đảo của số 4 theo modulo 8.

CHƯƠNG 2 – Hệ thống mật mã RSA

*Tổng quan*: Hệ thống mật mã RSA là một trong những hệ thống mật mã khóa công khai phổ biến nhất hiện nay. Nó được sử dụng rộng rãi trong các ứng dụng mật mã và bảo mật thông tin trực tuyến. Hệ thống RSA dựa trên sự khó giải các bài toán toán học phức tạp như phân tích thừa số nguyên tố và thuật toán Euclide mở rộng để tạo ra khóa mã hóa và giải mã. Việc triển khai đúng cách của hệ thống RSA là rất quan trọng để đảm bảo tính bảo mật của thông tin được mã hóa và giải mã. Tuy nhiên, hệ thống RSA cũng có những hạn chế và mối đe dọa bảo mật tiềm ẩn, vì vậy việc nghiên cứu và cải thiện hệ thống RSA vẫn là một chủ đề nghiên cứu quan trọng trong lĩnh vực mật mã và bảo mật thông tin.

2.1 Khái niệm

Hệ thống mật mã RSA (tên lấy từ tên ba nhà toán học: Rivest, Shamir và Adleman) được xem là một trong những hệ thống mật mã công khai (public-key cryptography) phổ biến nhất hiện nay.

Để hiểu về RSA, ta cần phải biết về tạo số nguyên tố và phân tích thừa số nguyên tố. Tạo số nguyên tố là quá trình tìm số nguyên tố ngẫu nhiên với độ dài cần thiết cho việc mã hóa thông điệp. Phân tích thừa số nguyên tố là quá trình tìm ra các thừa số nguyên tố của một số nguyên tự nhiên.

RSA còn sử dụng các khái niệm toán học khác như số học mô-đun và thuật toán Euclide mở rộng để tính toán nghịch đảo của một số nguyên trong một mô-đun cho trước

2.1.1 Ví dụ

Tạo số nguyên tố là khi ta chọn ngẫu nhiên hai số nguyên tố lớn p=17 và q=11, và tính n=pq=187. Tiếp đó, ta tính hàm Euler của n, được định nghĩa là số các số tự nhiên từ 1 đến n-1 có thể tương đối nguyên tố với n. Trong trường hợp này, Euler(n) = 160.

Tiếp theo, ta chọn số nguyên e sao cho 1 < e < Euler(n) và e không có ước chung với Euler(n). Trong trường hợp này, chúng ta chọn e=7. Ta tính nghịch đảo của e trong mô-đun Euler(n) bằng thuật toán Euclide mở rộng. Điều này cho ta d = 23.

Vậy, khóa công khai (public key) của hệ thống RSA là (e, n) = (7, 187) và khóa bí mật (private key) là (d, n) = (23, 187).

Sau đó, ta có thể sử dụng khóa công khai để mã hóa thông điệp và khóa bí mật để giải mã thông điệp đó.

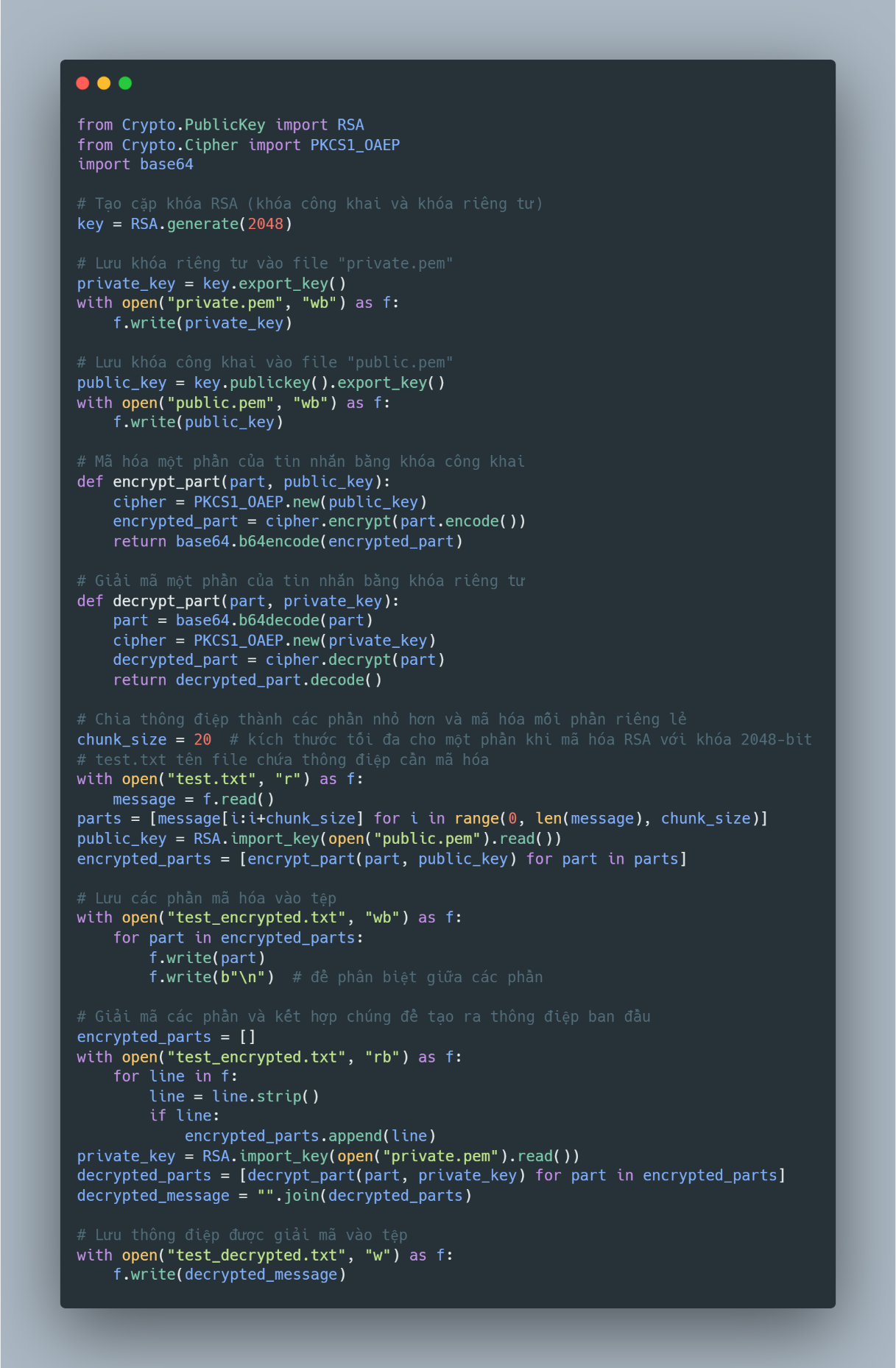
2.2 Chương trình python mã hoá và giải mã tin nhắn

2.2.1 Mã chương trình

-Code mã hoá tin nhắn văn bản



-Code mã hoá văn bản



2.2.1.1 Mã chi tiết

-Code mã hoá tin nhắn văn bản

from Crypto.PublicKey import RSA

from Crypto.Cipher import PKCS1\_OAEP

# Tạo cặp khóa RSA (khóa công khai và khóa riêng tư)

key = RSA.generate(2048)

# Lưu khóa riêng tư vào file "private.pem"

private\_key = key.export\_key()

with open("private.pem", "wb") as f:

f.write(private\_key)

# Lưu khóa công khai vào file "public.pem"

public\_key = key.publickey().export\_key()

with open("public.pem", "wb") as f:

f.write(public\_key)

# Mã hóa tin nhắn hoặc file văn bản

def encrypt\_message(message, public\_key\_file):

# Đọc khóa công khai từ file

with open(public\_key\_file, "rb") as f:

public\_key = RSA.import\_key(f.read())

# Sử dụng khóa công khai để mã hóa tin nhắn

cipher = PKCS1\_OAEP.new(public\_key)

encrypted\_message = cipher.encrypt(message.encode())

return encrypted\_message

# Giải mã tin nhắn hoặc file văn bản

def decrypt\_message(encrypted\_message, private\_key\_file):

# Đọc khóa riêng tư từ file

with open(private\_key\_file, "rb") as f:

private\_key = RSA.import\_key(f.read())

# Sử dụng khóa riêng tư để giải mã tin nhắn

cipher = PKCS1\_OAEP.new(private\_key)

decrypted\_message = cipher.decrypt(encrypted\_message)

return decrypted\_message.decode()

-Code mã hoá văn bản

from Crypto.PublicKey import RSA

from Crypto.Cipher import PKCS1\_OAEP

import base64

# Tạo cặp khóa RSA (khóa công khai và khóa riêng tư)

key = RSA.generate(2048)

# Lưu khóa riêng tư vào file "private.pem"

private\_key = key.export\_key()

with open("private.pem", "wb") as f:

f.write(private\_key)

# Lưu khóa công khai vào file "public.pem"

public\_key = key.publickey().export\_key()

with open("public.pem", "wb") as f:

f.write(public\_key)

# Mã hóa một phần của tin nhắn bằng khóa công khai

def encrypt\_part(part, public\_key):

cipher = PKCS1\_OAEP.new(public\_key)

encrypted\_part = cipher.encrypt(part.encode())

return base64.b64encode(encrypted\_part)

# Giải mã một phần của tin nhắn bằng khóa riêng tư

def decrypt\_part(part, private\_key):

part = base64.b64decode(part)

cipher = PKCS1\_OAEP.new(private\_key)

decrypted\_part = cipher.decrypt(part)

return decrypted\_part.decode()

# Chia thông điệp thành các phần nhỏ hơn và mã hóa mỗi phần riêng lẻ

chunk\_size = 20 # kích thước tối đa cho một phần khi mã hóa RSA với khóa 2048-bit

# test.txt tên file chứa thông điệp cần mã hóa

with open("test.txt", "r") as f:

message = f.read()

parts = [message[i:i+chunk\_size] for i in range(0, len(message), chunk\_size)]

public\_key = RSA.import\_key(open("public.pem").read())

encrypted\_parts = [encrypt\_part(part, public\_key) for part in parts]

# Lưu các phần mã hóa vào tệp

with open("test\_encrypted.txt", "wb") as f:

for part in encrypted\_parts:

f.write(part)

f.write(b"\n") # để phân biệt giữa các phần

# Giải mã các phần và kết hợp chúng để tạo ra thông điệp ban đầu

encrypted\_parts = []

with open("test\_encrypted.txt", "rb") as f:

for line in f:

line = line.strip()

if line:

encrypted\_parts.append(line)

private\_key = RSA.import\_key(open("private.pem").read())

decrypted\_parts = [decrypt\_part(part, private\_key) for part in encrypted\_parts]

decrypted\_message = "".join(decrypted\_parts)

# Lưu thông điệp được giải mã vào tệp

with open("test\_decrypted.txt", "w") as f:

f.write(decrypted\_message)

2.2.1.2 Cơ chế hoạt động

-Mã hoá tin nhắn

Tạo một cặp khóa RSA, gồm khóa công khai và khóa riêng tư. Sau đó, nó lưu khóa riêng tư vào file "private.pem" và khóa công khai vào file "public.pem".

Tiếp theo, nó cung cấp hai hàm để mã hóa và giải mã tin nhắn hoặc file văn bản sử dụng khóa công khai và khóa riêng tư. Hàm encrypt\_message nhận một thông điệp và đường dẫn tới file chứa khóa công khai, sau đó sử dụng PKCS1\_OAEP để mã hóa thông điệp. Hàm decrypt\_message nhận thông điệp đã được mã hóa và đường dẫn tới file chứa khóa riêng tư, sau đó sử dụng PKCS1\_OAEP để giải mã thông điệp.

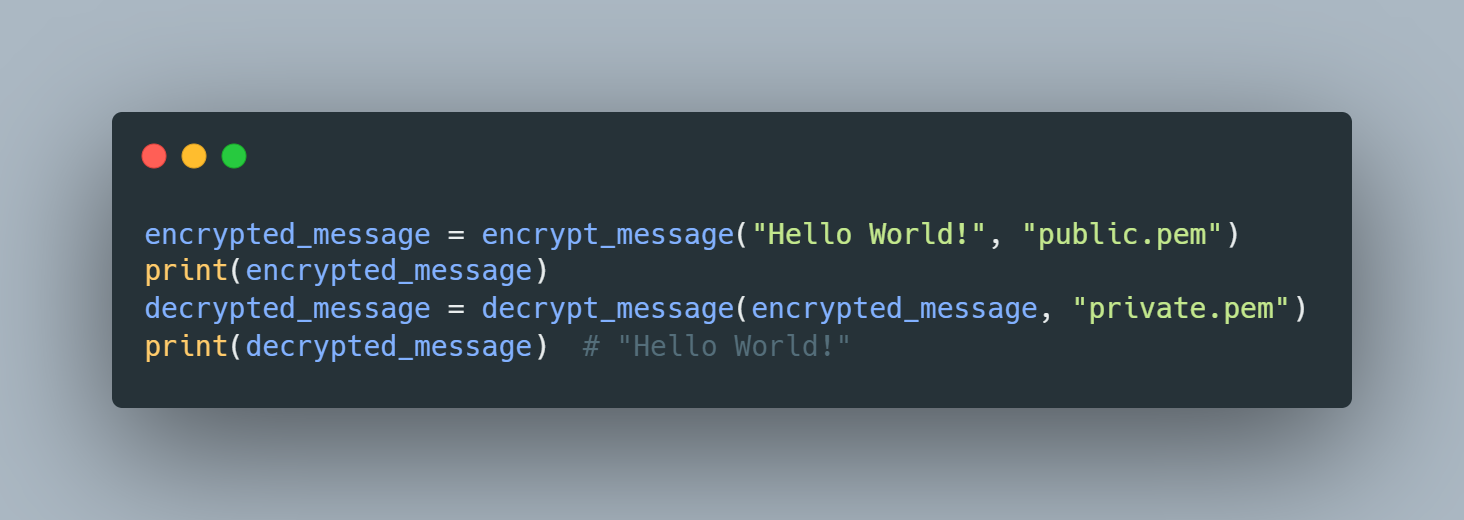
-Mã hoá file

Tạo một cặp khóa RSA (khóa công khai và khóa riêng tư) với độ dài 2048 bit và lưu khóa riêng tư và khóa công khai vào các tệp "private.pem" và "public.pem" tương ứng.

Sau đó, đoạn code chia thông điệp cần mã hóa thành các phần có kích thước không quá 20 ký tự và mã hóa mỗi phần riêng lẻ bằng khóa công khai, sử dụng hàm encrypt\_part và lưu các phần đã mã hóa vào tệp "test\_encrypted.txt".

Để giải mã, đoạn code đọc các phần mã hóa từ tệp "test\_encrypted.txt", giải mã từng phần bằng khóa riêng tư và kết hợp chúng để tạo ra thông điệp ban đầu. Cuối cùng, thông điệp được giải mã được lưu vào tệp "test\_decrypted.txt".

2.2.2 Kiểm tra chương trình

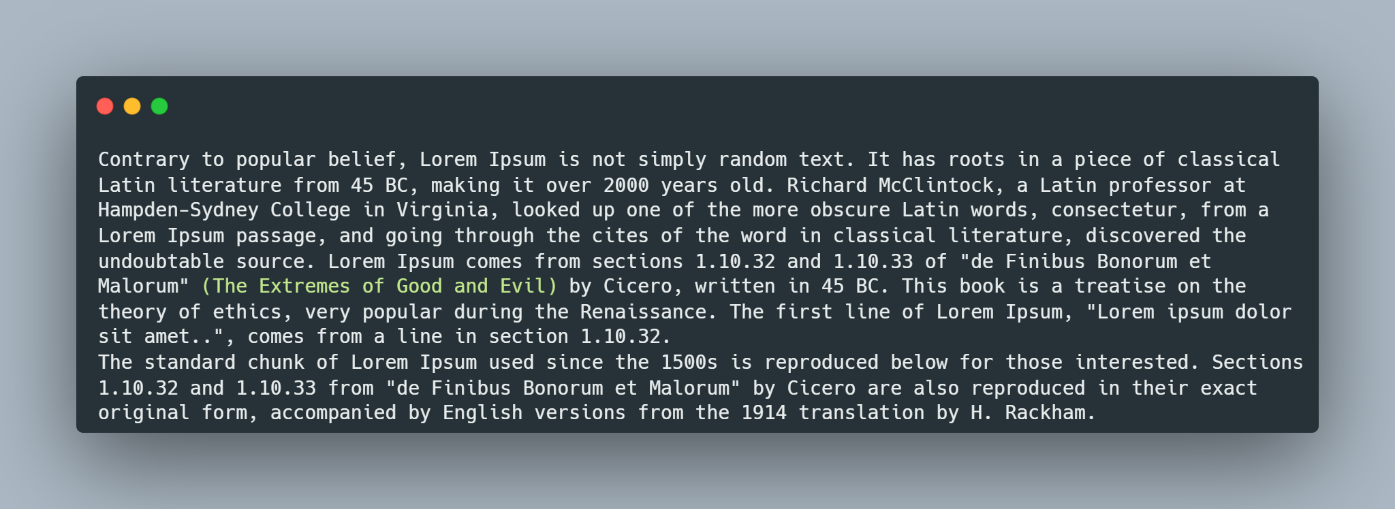
-Kiểm tra với tin nhắn:

Kết quả:

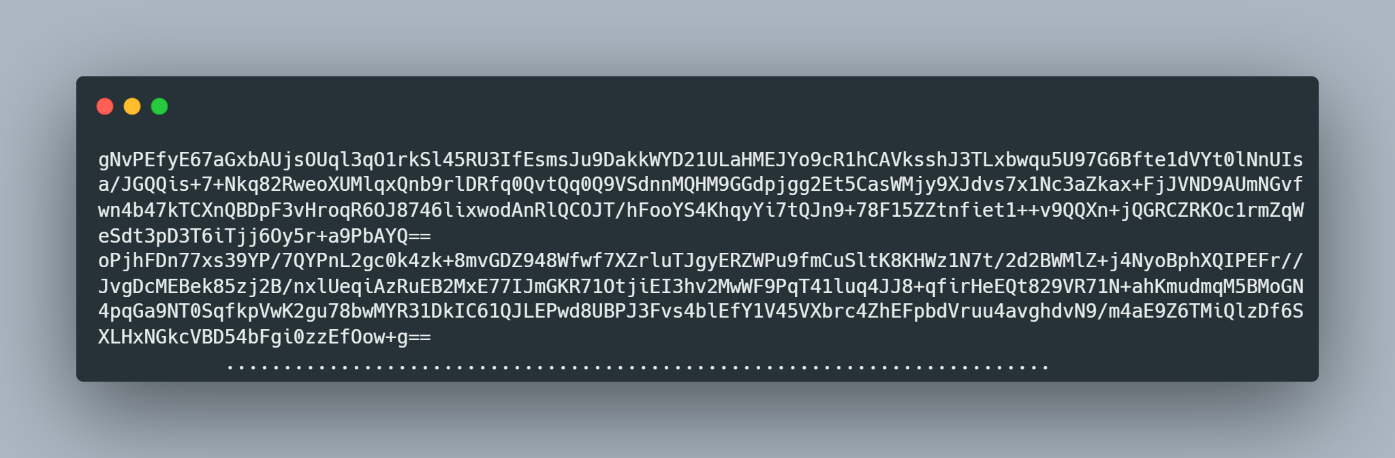


-Kiểm tra với tệp văn bản

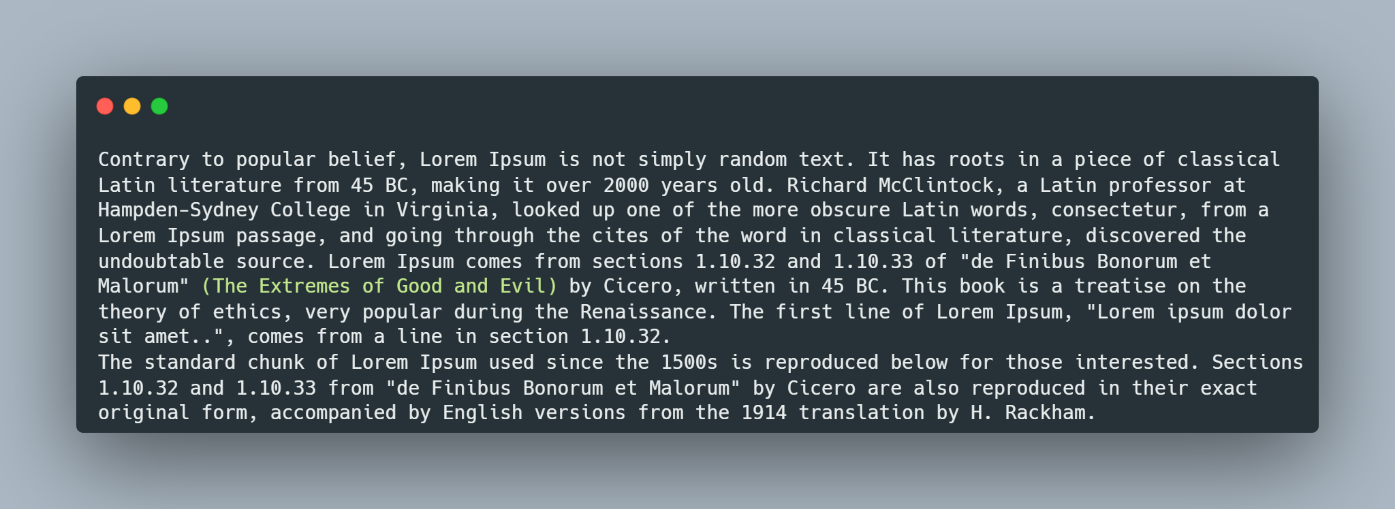
File test.txt



Kết quả file test\_encrypted.txt



File test\_decrypted.txt



2.2.3 Phân tích hiệu quả,tính bảo mật của hệ thống mật mã RSA

Hệ thống mật mã RSA là một trong những hệ thống mật mã công khai được sử dụng rộng rãi nhất hiện nay. Nó được sử dụng để mã hóa và giải mã thông điệp bằng cách sử dụng cặp khóa công khai và khóa bí mật. Hệ thống mật mã RSA là một hệ thống mật mã an toàn và hiệu quả với khả năng chống lại các cuộc tấn công mã hóa bằng phương pháp phân tích thử và lỗi, dò tìm các cặp khóa giả và phân tích theo số lượng lớn.

Tuy nhiên, một số tấn công còn lại trên hệ thống RSA, như tấn công theo phương pháp đo thời gian, tấn công man-in-the-middle và tấn công theo phương pháp lưu trữ khóa.

Đối với tính hiệu quả, hệ thống mật mã RSA có thể được sử dụng để mã hóa và giải mã thông điệp nhanh chóng và dễ dàng bằng cách sử dụng các phần mềm mã hóa. Tuy nhiên, với các kích thước khóa lớn, việc mã hóa và giải mã thông điệp sẽ mất nhiều thời gian và tài nguyên tính toán hơn.

Tóm lại, hệ thống mật mã RSA là một hệ thống mật mã công khai an toàn và hiệu quả với nhiều ứng dụng thực tế, tuy nhiên nó cũng có thể bị tấn công và cần được sử dụng đúng cách để đảm bảo tính bảo mật cao nhất.

2.2.4 Thảo luận về các mối đe dọa bảo mật tiềm ẩn và những hạn chế của hệ thống mật mã RSA

Hệ thống mật mã RSA đã được chứng minh là rất mạnh và an toàn trong nhiều năm. Tuy nhiên, vẫn có một số mối đe dọa bảo mật tiềm ẩn và hạn chế của hệ thống này.

Một trong những mối đe dọa bảo mật của RSA là tấn công brute force, trong đó kẻ tấn công cố gắng tìm ra khóa bí mật bằng cách thử các giá trị ngẫu nhiên cho p và q. Để ngăn chặn tấn công này, độ dài của khóa bí mật phải đủ lớn để không thể dò được bằng brute force.

Ngoài ra, còn có một số tấn công khác như tấn công đường giữa (man-in-the-middle), tấn công phân tích bằng công cụ phần cứng đặc biệt, và tấn công theo thời gian (timing attack).

Một trong những hạn chế của RSA là nó yêu cầu tính toán số học lớn, đặc biệt là phép tính mũ lớn. Điều này có thể làm chậm hệ thống và làm cho quá trình mã hóa và giải mã chậm hơn. Ngoài ra, RSA cũng không phù hợp với việc mã hóa các tệp tin lớn vì nó sử dụng phép tính mũ lớn trên các số nguyên lớn.

Trong tổng thể, RSA là một hệ thống mật mã mạnh và được sử dụng rộng rãi. Tuy nhiên, như với bất kỳ hệ thống mật mã nào khác, nó cũng có những mối đe dọa và hạn chế, và điều quan trọng là cần phải được sử dụng một cách thận trọng và hiểu biết để đảm bảo tính bảo mật và hiệu quả của hệ thống.

2.2.5 Kết luận với các khuyến nghị để cải thiện việc triển khai hệ thống mật mã RSA.

Tóm lại, hệ thống mật mã RSA là một phương pháp mã hóa đối xứng mạnh mẽ được sử dụng rộng rãi. Tuy nhiên, nó cũng có những hạn chế và mối đe dọa bảo mật tiềm ẩn cần được lưu ý. Để cải thiện việc triển khai hệ thống mật mã RSA, tôi đưa ra một số khuyến nghị như sau:

1. Sử dụng độ dài khóa an toàn hơn: Tăng độ dài khóa RSA sẽ làm cho việc tấn công bằng cách phân tích các giá trị khóa trở nên khó khăn hơn.
2. Sử dụng cơ chế bảo vệ khóa: Lưu trữ khóa mật tại nơi an toàn và sử dụng các phương pháp bảo vệ khóa khác như mã hóa tuyến tính để giảm thiểu nguy cơ bị lộ khóa.
3. Thực hiện các phương pháp kiểm tra lỗi: Tất cả các ứng dụng mã hóa RSA phải được thực hiện và kiểm tra lỗi để đảm bảo rằng chúng hoạt động đúng cách và không có lỗ hổng nào.
4. Sử dụng các giải pháp bảo mật mã hóa khác: RSA là một trong những thuật toán mã hóa phổ biến nhất, tuy nhiên có rất nhiều giải pháp mã hóa khác nhau được sử dụng trong các ứng dụng khác nhau. Nên sử dụng các giải pháp khác nhau để tăng tính đa dạng của hệ thống bảo mật.
5. Cập nhật thường xuyên: Các hệ thống bảo mật và các thư viện mã hóa RSA cần được cập nhật thường xuyên để đảm bảo tính bảo mật và hiệu quả của chúng.
6. Với các khuyến nghị trên, việc triển khai hệ thống mật mã RSA sẽ được cải thiện và bảo mật hơn. Tuy nhiên, chúng ta cần luôn cập nhật các kiến thức về bảo mật và theo dõi các mối đe dọa mới để đảm bảo rằng hệ thống của chúng ta luôn được bảo vệ tốt nhất.

**TÀI LIỆU THAM KHẢO**

**Tiếng Anh**

1. RSA and Public-Key Cryptography (Discrete Mathematics and Its Applications) 1st Edition
2. https://en.wikipedia.org/wiki/Modular\_multiplicative\_inverse