

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
KHOA AN TOÀN THÔNG TIN



Môn học: Thực Tập Cơ Sở
Báo Cáo Bài Thực Hành 16
Lập trình thuật toán mật mã học

Họ và tên: Trần Thị Thu Phương

Mã sinh viên: B21DCAT151

Nhóm môn học: 04

Giảng viên: Đinh Trường Duy

Hà Nội, 4/2024

Mục lục

| | | |
|-------------|--|-----------|
| 1. | Mục đích..... | 3 |
| 2. | Nội dung thực hành | 3 |
| 2.1. | Cơ sở lý thuyết..... | 3 |
| 2.1.1. | Tìm hiểu về lập trình số lớn với các phép toán cơ bản..... | 3 |
| 2.1.2. | Tìm hiểu về giải thuật mật mã khóa công khai RSA..... | 4 |
| 2.2. | Các bước thực hiện | 6 |
| 2.2.1. | Chuẩn bị môi trường..... | 6 |
| 2.2.2. | Thực hiện | 6 |
| 3. | Kết luận..... | 10 |
| 4. | Tài liệu tham khảo | 10 |

Danh mục hình ảnh

| | |
|--|---|
| Hàm tính $ab \bmod m$ | 7 |
| Hàm tìm số nghịch đảo d sao cho $d.e \bmod m = 1$ | 8 |
| Hàm sinh khóa, để đơn giản ta chọn luôn $e = 65537$ | 8 |
| Hàm mã hóa và giải mã | 9 |
| Thử nghiệm code với mã hóa và giải mã | 9 |
| Kết quả, do kết quả rất dài nên em xin phép chỉ chụp 1 phần..... | 9 |

1. Mục đích

Sinh viên tìm hiểu một giải thuật mã hóa phổ biến và lập trình được chương trình mã hóa và giải mã sử dụng ngôn ngữ lập trình phổ biến như C/C++/Python/Java, đáp ứng chạy được với số lớn.

2. Nội dung thực hành

2.1. Cơ sở lý thuyết

2.1.1. Tìm hiểu về lập trình số lớn với các phép toán cơ bản

Một số lý thuyết cơ bản về lập trình số nguyên lớn:

- **Biểu diễn số nguyên lớn:** Trong lập trình số nguyên lớn, số này thường được biểu diễn dưới dạng mảng hoặc danh sách các chữ số. Mỗi phần tử trong mảng hoặc danh sách này thường lưu trữ một chữ số của số nguyên.
- **Cộng và Trừ:**
 - + Các phép toán cộng và trừ có thể được thực hiện bằng cách duyệt qua từng chữ số của hai số và thực hiện phép toán tương ứng từng cặp chữ số, kèm theo việc xử lý nhớ và mượn nếu cần.
 - + Bắt đầu từ chữ số đơn vị và tiếp tục lần lượt đến chữ số hàng đơn vị, thực hiện phép toán cộng hoặc trừ tương ứng.
- **Nhân:**
 - + Phép nhân hai số nguyên lớn có thể thực hiện bằng thuật toán nhân dài (long multiplication) hoặc các thuật toán nhân nhanh hơn như nhân Karatsuba hoặc nhân Toom-Cook.
 - + Thuật toán nhân dài thực hiện phép nhân tương tự như cách thực hiện phép nhân trong toán học cơ bản: duyệt qua từng chữ số của một số và nhân với từng chữ số của số kia, sau đó cộng các kết quả lại.
- **Chia:**
 - + Phép chia hai số nguyên lớn có thể thực hiện bằng thuật toán chia dài (long division) hoặc các thuật toán chia hiệu quả hơn như thuật toán chia như thuật toán chia nhân bán (divide-and-conquer division) hoặc thuật toán chia nhưng tốt hơn (Newton-Raphson division).
 - + Tương tự như phép nhân, thuật toán chia dài duyệt qua từng chữ số của số chia và thực hiện phép chia tương ứng.
- **Tính Lũy Thừa:**
 - + Phép tính lũy thừa của một số nguyên lớn có thể được thực hiện bằng thuật toán lũy thừa nhanh (exponentiation by squaring). Thuật toán này giảm thiểu số lượng phép toán bằng cách phân tích số mũ thành các phần nhỏ và tính lũy thừa của từng phần.
- **Tìm Ước Chung Lớn Nhất (GCD):**

- + Phép tìm ước chung lớn nhất của hai số nguyên lớn có thể được thực hiện bằng thuật toán Euclid mở rộng (extended Euclidean algorithm) hoặc thuật toán nhị phân (binary GCD algorithm). Cả hai thuật toán này đều dựa trên việc lặp đi lặp lại một số phép chia và tính toán dựa trên phần dư.

Trong lập trình số nguyên lớn, việc tối ưu hóa và cải thiện hiệu suất là rất quan trọng. Điều này bao gồm việc sử dụng các thuật toán hiệu quả nhất để thực hiện các phép toán cơ bản và giải quyết các vấn đề liên quan đến hiệu năng và bộ nhớ.

Trong Python, số nguyên không bị giới hạn về giá trị. Tuy nhiên, số nguyên có thể lớn đến mức nào phụ thuộc vào bộ nhớ có sẵn trên hệ thống mà bạn đang sử dụng. Chính vì vậy, ta có thể xử lý số nguyên lớn dễ dàng với Python

2.1.2. Tìm hiểu về giải thuật mật mã khóa công khai RSA

a. Giải thuật mã hóa khóa công khai

Mã hóa khóa bất đối xứng (Hay còn gọi là mã hóa khóa công khai): là phương pháp mã hóa mà khóa sử dụng để mã hóa sẽ khác với khóa dùng để giải mã. Khóa dùng để mã hóa gọi là khóa công khai và khóa dùng để giải mã gọi là khóa bí mật. Tất cả mọi người đều có thể biết được khóa công khai (kể cả hacker), và có thể dùng khóa công khai để mã hóa thông tin. Nhưng chỉ có người nhận mới nắm giữ khóa bí mật, nên chỉ có người nhận mới có thể giải mã được thông tin.

Đặc điểm nổi bật của các hệ mã hóa khóa bất đối xứng là kích thước khóa lớn, lên đến hàng ngàn bit. Do vậy, các hệ mã hóa dạng này thường có tốc độ thực thi chậm hơn nhiều lần so với các hệ mã hóa khóa đối xứng với độ an toàn tương đương. Mặc dù vậy, các hệ mã hóa khóa bất đối xứng có khả năng đạt độ an toàn cao và ưu điểm nổi bật nhất là việc quản lý và phân phối khóa đơn giản hơn do khóa công khai có thể phân phối rộng rãi.

Hệ thống mật mã hóa khóa công khai có thể sử dụng với các mục đích:

- Mã hóa: giữ bí mật thông tin và chỉ có người có khóa bí mật mới giải mã được.
- Tạo chữ ký số: cho phép kiểm tra văn bản có phải đã được tạo với một khóa bí mật nào đó hay không.
- Thỏa thuận khóa: cho phép thiết lập khóa dùng để trao đổi thông tin giữa hai bên.

Các giải thuật mã hóa khóa bất đối xứng điển hình bao gồm: RSA, Rabin, ElGamal, McEliece và Knapsack. Trong bài tiểu luận này, chúng tôi tìm hiểu và trình bày về giải thuật mã hóa RSA – một trong các giải thuật mã hóa khóa đối xứng được sử dụng rộng rãi nhất trên thực tế.

b. Giải thuật mã hóa khóa công khai RSA

Thuật toán mã hóa RSA là một thuật toán mã hóa khóa công khai. Thuật toán RSA được xây dựng bởi các tác giả Ron Rivest, Adi Shamir và Len Adleman tại học viện MIT vào năm 1977, và ngày nay đang được sử dụng rộng rãi. Về mặt tổng quát thuật toán RSA là một phương pháp mã hóa theo khối. Trong đó thông điệp M và bản mã C là các số nguyên từ 0 đến 2^i với i số bit của khối. Kích thước thường dùng của i là 1024 bit. Thuật toán RSA sử dụng hàm một chiều là vấn đề phân tích một số thành thừa số nguyên tố và bài toán Logarith rời rạc.

❖ Thuật toán

Để thực hiện mã hóa và giải mã, thuật toán RSA dùng phép lũy thừa modulo của lý thuyết số.

Quá trình sinh khóa

- Chọn hai số nguyên tố lớn p và q và tính $N = pq$. Cần chọn p và q sao cho:
 $M < 2^{i-1} < N < 2^i$ với i là chiều dài bản rõ.
- Tính $\Phi(n) = (p - 1)(q - 1)$
- Tìm một số e sao cho: $\{e \text{ và } \Phi(n) \text{ là 2 số nguyên tố cùng nhau và } 0 < e < \Phi(n)\}$
- Tìm một số d sao cho: $e \cdot d \equiv 1 \pmod{\Phi(n)}$ (hay: $d = e^{-1} \pmod{\Phi(n)}$)
(d là nghịch đảo của e trong phép modulo $\Phi(n)$)
- Chọn khóa công khai K_u là cặp (e, N) , khóa riêng K_r là cặp (d, N)

Quá trình mã hóa:

- Thông điệp m được chuyển thành số: $m < n$
- Bản mã $c = m^e \pmod n$

Quá trình giải mã:

- Bản mã c , $c < n$
- Bản rõ $m = c^d \pmod n$

Lưu ý: Khi chọn hai số nguyên tố p và q để sử dụng trong hệ mã hoá RSA, quan trọng nhất là chúng không nên quá gần nhau hoặc quá lệch nhau.

❖ Đánh giá:

Giải thuật RSA là một trong những giải thuật mã hoá công khai (asymmetric cryptography) quan trọng và phổ biến nhất được sử dụng trong lĩnh vực bảo mật thông tin. Dưới đây là một số điểm đánh giá về giải thuật RSA:

- An toàn và Bảo mật: RSA cung cấp một cơ chế mạnh mẽ để bảo vệ thông tin truyền trực tuyến bằng cách sử dụng các khóa công khai và bí mật. Việc phân tích khóa bí mật từ khóa công khai được cho là một bài toán rất khó đối với các thuật toán hiện đại nhưng chưa có bằng chứng toàn diện chứng minh rằng nó không thể được thực hiện.

- Khả năng chống lại các cuộc tấn công: RSA có thể chống lại nhiều loại cuộc tấn công như tấn công brute force (tấn công dò mò), tấn công miền trung gian (man-in-the-middle), và tấn công factorization (phân tích nhân tử). Tuy nhiên, việc sử dụng các cặp khóa ngắn có thể dễ dàng bị tấn công bằng các thuật toán factorization mạnh.
- Hiệu suất: RSA có thể tính toán chậm đối với các khóa dài. Các phép toán như phép nhân số nguyên lớn và tính modulo có thể tốn nhiều thời gian, đặc biệt là với các số nguyên lớn có hàng trăm hoặc hàng nghìn bit.
- Sử dụng rộng rãi: RSA được sử dụng rộng rãi trong các ứng dụng bảo mật như trao đổi khóa, ký số, và mã hoá dữ liệu truyền trực tuyến. Nó được tích hợp vào nhiều giao thức mạng như HTTPS, SSH, và SSL/TLS.
- Quản lý khóa: RSA đòi hỏi quản lý khóa cẩn thận. Quản lý và bảo vệ khóa bí mật là một thách thức đối với việc triển khai và vận hành các hệ thống sử dụng RSA.
- Chi phí tính toán: RSA có thể tốn kém về chi phí tính toán so với các giải thuật mã hoá khác như AES (Advanced Encryption Standard) trong một số tình huống.

Tóm lại, RSA là một giải thuật mã hoá công khai mạnh mẽ và phổ biến, nhưng cũng cần cân nhắc về hiệu suất tính toán và quản lý khóa khi triển khai trong các ứng dụng thực tế.

2.2. Các bước thực hiện

2.2.1. Chuẩn bị môi trường

- Python 3.11.7

2.2.2. Thực hiện

- Lập trình thư viện số lớn với các phép toán cơ bản để sử dụng trong giải thuật mã hóa/giải mã RSA
- Thử nghiệm chứng minh thư viện hoạt động tốt với các ví dụ phép toán cho số lớn
- Lập trình giải thuật mã hóa và giải mã
- Thử nghiệm mã hóa và giải mã chuỗi ký tự: “I am B21DCAT151” (thay bằng mã sinh viên của mình vào)

Bài 16: Lập trình thuật toán mật mã học

The screenshot displays a Windows IDE with a Python script and a Windows PowerShell terminal window.

Python Script (pykt087_SoDacBiet.py):

```
1 from math import gcd
2 import sympy
3 import random
4 import math
5
6 #Hàm tính lũy thừa: a^b mod m
7 def bipow(a, b, m):
8     if b == 0:
9         return 1
10    x = bipow(a, b//2, m) %m
11    if b%2==0:
12        return (x*x)%m
13    else:
14        return (x*x*a)%m
15 print(bipow(9999999999999999, 99999999999999999999999999999999, 4))
16 # def generate_keypair(p, q):
17 #     n = p * q
18 #     phi = (p-1) * (q-1)
19
20 #     # Chọn một số nguyên e sao cho 1 < e < phi, và e là số nguyên tố cùng
21 #     e = random.randint(1, phi)
22 #     while math.gcd(e, phi) != 1:
```

Windows PowerShell Window:

```
B21DCAT151"
Trần Thị Thu Phương B21DCAT151
PS C:\Users\Admin> date

Wednesday, May 1, 2024 12:36:13 AM
```

Terminal Window:

```
PS D:\HocTap\code\python> & C:/Users/Admin/anaconda3/python.exe "d:/HocTap/đại học/năm 3/kỳ 2/Thực tập cơ sở/lập trình với mã hóa.py"
3
PS D:\HocTap\code\python>
```

Hàm tính $a^b \bmod m$

Bài 16: Lập trình thuật toán mật mã học

The screenshot shows a VS Code editor with a Python file named `lập trình với mã hóa.py`. The code defines a function `multiplicative_inverse(e, phi)` that finds the modular inverse of e modulo ϕ using the extended Euclidean algorithm. The function returns $d + \phi$ if $\gcd(e, \phi) = 1$, otherwise it returns `-1`. The script also prints the result of `multiplicative_inverse(3233, 3120)` and the value of $3233 \times d \pmod{\phi}$.

A PowerShell terminal window is open, showing the command `date` being executed, which displays the date and time: `Wednesday, May 1, 2024 12:58:34 AM`.

Hàm tìm số nghịch đảo d sao cho $d.e \pmod{m} = 1$

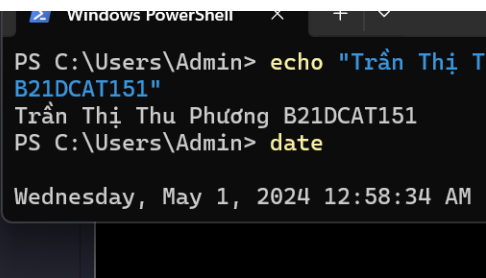
The screenshot shows a VS Code editor with a Python file named `lập trình với mã hóa.py`. The code defines a function `generate_keypair(p, q)` that generates a key pair (e, n) and (d, n) . It chooses a random number e such that $1 < e < \phi$ and $\gcd(e, \phi) = 1$. It then uses the extended Euclidean algorithm to find d such that $d.e \pmod{\phi} = 1$. The function returns $((e, n), (d, n))$.

A PowerShell terminal window is open, showing the command `echo "Trần Thị Thu Phương B21DCAT151"` being executed, which displays the output: `Trần Thị Thu Phương B21DCAT151`.

Hàm sinh khóa, để đơn giản ta chọn luôn $e = 65537$

Bài 16: Lập trình thuật toán mật mã học

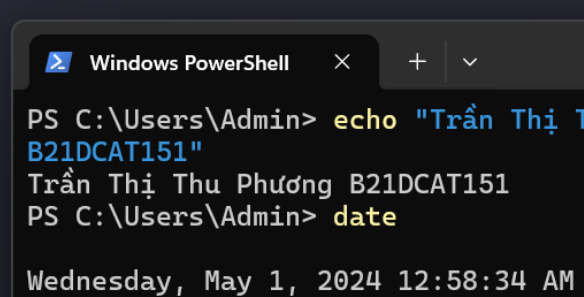
```
57 def encrypt(public_key, plaintext):
58     e, n = public_key
59     #Hàm ord: nhận 1 kí tự làm đầu vào và trả về mã ASCII của kí tự đó
60     cipher = [bipow(ord(char), e, n) for char in plaintext]
61     return cipher
62
63 def decrypt(private_key, ciphertext):
64     d, n = private_key
65     #Hàm chr: nhận 1 số nguyên làm đầu vào và trả về kí tự ứng với mã ASCII
66     plain = [chr(bipow(char, d, n)) for char in ciphertext]
67     return ''.join(plain)
68
```



```
PS C:\Users\Admin> echo "Trần Thị Thu Phương B21DCAT151"
Trần Thị Thu Phương B21DCAT151
PS C:\Users\Admin> date
Wednesday, May 1, 2024 12:58:34 AM
```

Hàm mã hóa và giải mã

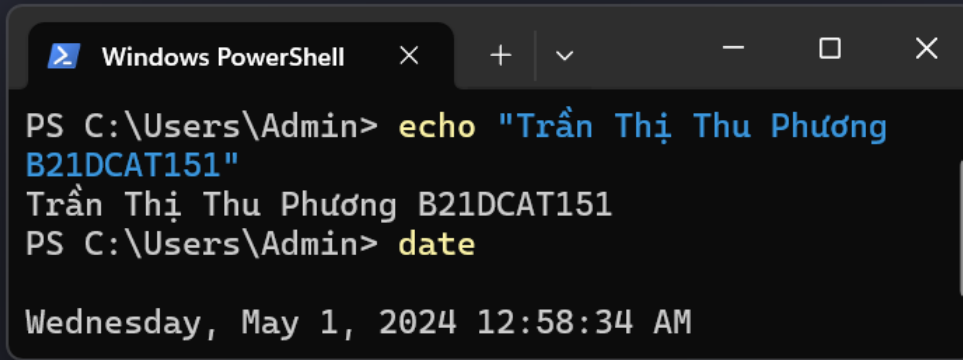
```
70
71 # Thử nghiệm mã hóa và giải mã
72 # p = 61
73 # q = 53
74 p = sympy.randprime(2**1023, 2**1024)
75 q = sympy.randprime(2**1023, 2**1024)
76
77 public_key, private_key = generate_keypair(p, q)
78 print("Khóa công khai:", public_key)
79 print("Khóa bí mật:", private_key)
80
81 message = "I am B21DCAT151"
82 print("Plaintext:", message)
83
84 encrypted_msg = encrypt(public_key, message)
85 out_encrypted = ''.join(str(x) for x in encrypted_msg)
86 # print("Encrypted:", encrypted_msg)
87 print("Encrypted:", out_encrypted)
88
89 decrypted_msg = decrypt(private_key, encrypted_msg)
90 print("Decrypted:", decrypted_msg)
91
```



```
PS C:\Users\Admin> echo "Trần Thị Thu Phương B21DCAT151"
Trần Thị Thu Phương B21DCAT151
PS C:\Users\Admin> date
Wednesday, May 1, 2024 12:58:34 AM
```

Thử nghiệm code với mã hóa và giải mã

```
Khóa công khai: (65537, 1948878860727409217137565743299127205682294263193475001018597519
Khóa bí mật: (28442325329681520665019700792466594551884998870470590650689652925370361262
Plaintext: I am B21DCAT151
Encrypted: 18138237740313946509047936936247935427799803590152412485393726078066711241411
Decrypted: I am B21DCAT151
```



```
PS C:\Users\Admin> echo "Trần Thị Thu Phương B21DCAT151"
Trần Thị Thu Phương B21DCAT151
PS C:\Users\Admin> date
Wednesday, May 1, 2024 12:58:34 AM
```

Kết quả, do kết quả rất dài nên em xin phép chỉ chụp 1 phần

3. Kết luận

- Lý thuyết về mã hóa khóa công khai
- Chạy giải thuật mã hóa RSA thành công

4. Tài liệu tham khảo

- [1]. Đỗ Xuân Chợt, Bài giảng Mật mã học cơ sở, Học viện Công Nghệ Bưu Chính Viễn Thông, 2021.
- [2]. Đỗ Xuân Chợt, Bài giảng Mật mã học nâng cao, Học viện Công Nghệ Bưu Chính Viễn Thông, 2021.