VIETNAM NATIONAL UNIVERSITY-HO CHI MINH CITY

UNIVERSITY OF SCIENCE

Faculty of Information Technology

REPORT PROJECT

Nhập môn mã hóa mật mã

20MMT

Mã hóa thông điệp với RSA

20127245- Hồ Bá Nam

Lectures:

Nguyễn Văn Quang Huy – Ngô Đình Hy – Lê Phúc Lữ - Nguyễn Đình Thúc

Ho Chi Minh - 2022

Mục lục

1. Thông tin 3
1.1. Thành viên 3
1.2. Đồ án 3
2. Mã hóa thông điệp sử dụng thuật toán RSA 3
2.1. Vấn đề: 3
2.2. Giải quyết 4
2.2.1. Tạo khóa 5
2.2.2. Mã hóa 7
2.2.3. Giải mã 7
3. Tài liệu tham khảo 8

1. Thông tin

1.1. Thành viên

- **Tên:** Hồ Bá Nam
- **MSSV:** 20127245
- **Mail:** honambsn@gmail.com

1.2. Đồ án

- **Tên:** Mã hóa thông điệp sử dụng thuật toán RSA
- **Nội dung:** Thông điệp được gửi qua lại giữ server và client bằng socket của Python được mã hóa bằng cách sử dụng thuật toán RSA.
- **Ngôn ngữ lập trình:** Python

2. Mã hóa thông điệp sử dụng thuật toán RSA

2.1. Vấn đề:

A và B liên lạc cho nhau một cách bí mật. Với thuật toán RSA, A sẽ gửi cho B khóa công khai của mình (A_public_key) và giữ lại cặp khóa bí mật (A_private_key).

2.2. Giải quyết

Phía B:

Lúc này, khi B muốn gửi thông điệp **M** của B cho A. M sẽ được chuyển đổi thành **m** sao cho **m** < **n** thông qua hàm có thể đảo ngược, tức là chuyển đổi từ **m** sang **M** và ngược lại, từ **M** sang **m** bằng hàm nghịch đảo/ 2 chiều. Quá trình mã hóa diễn ra:

$$c = m^e \ mod \ n$$

Với:

c là bản mã hóa **của** m theo công thức trên

m là thông điệp **M** đã được chuyển đổi bằng hàm 2 chiều

n, e là khóa công khai mà A đã cho B biết

Sau đó, B sẽ gửi **c** cho A

Phía A:

A nhận được **c** từ B và có khóa bí mật. A giải mã thông điệp c theo công thức:

$$m = c^d \ mod \ n$$

Chứng minh:

$$c^d \equiv (m^e)^d \ \equiv \ m^{ed} \ mod \ n$$

Theo định lý Fermat nhỏ:

$ed \equiv 1 \ (\text{mod } p\text{-}1)$

$ed \equiv 1 \ (\text{mod } q\text{-}1)$

Nên:

$$m^{ed} \ \equiv m \ (mod \ p)$$

Và:

$$m^{ed} \ \equiv m \ (mod \ q)$$

Mà **p**, **q** là 2 số nguyên tố cùng nhau, nên theo định lý đồng dư Trung Quốc:

$$m^{ed} \ \equiv m \ (mod \ pq)$$

Hoặc:

$$c^d \equiv m \ (mod \ n)$$

2.2.1. Tạo khóa

1) Chọn $\boldsymbol{p}$ và $\boldsymbol{q}$ ngẫu nhiên và độc lập. Sao cho p,q là số nguyên tố và $\boldsymbol{p \neq q}$
2) Tính ***mô đun-n*** = **pq**
3) Tính giá trị của phi Euler $\boldsymbol{\phi n} = (p-1)(q-1)$
4) Chọn ***e-số mũ công khai*** sao cho $1 < \boldsymbol{e} < \boldsymbol{\phi n}$ và $\boldsymbol{e}$ là số nguyên tố cùng nhau với $\boldsymbol{\phi n}$. Thực hiện bằng Euler mở rộng
5) Tính $\boldsymbol{d}$ sao cho $\mathbf{de} \equiv 1 \pmod{\boldsymbol{\phi n}}$

- Khóa công khai: **n,e**
- Khóa bí mật: **n,d**

```
def generate_keypair(p, q, keysize):
    nMin = 1 << (keysize - 1)
    nMax = (1 << keysize) - 1
    primes = [2]
    start = 1 << (keysize // 2 - 1)
    stop = 1 << (keysize // 2 + 1)

    if start >= stop:
        return []

    for i in range(3, stop + 1, 2):
        for p in primes:
            if i % p == 0:
                break
        else:
            primes.append(i)

    while (primes and primes[0] < start):
        del primes[0]

    while primes:
        p = random.choice(primes)
        primes.remove(p)
        q_values = [q for q in primes if nMin <= p * q <= nMax]
        if q_values:
            q = random.choice(q_values)
            break
    print(p, q)
    n = p * q
    phi = (p - 1) * (q - 1)

    e = random.randrange(1, phi)
    g = gcd(e, phi)

    while True:
        e = random.randrange(1, phi)
        g = gcd(e, phi)
        #generate private key
        d = mod_inverse(e, phi)
        if g == 1 and e != d:
            break


    return ((e, n), (d, n))
```

2.2.2.Mã hóa

$$c = m^e \bmod n$$

Ở đây, m và M được chuyển đổi thông qua hàm **chr()** và **ord()** của Python

- **Hàm chr():** trả về một kí tự (một chuỗi) từ một số nguyên mà số nguyên ấy là đại diện cho mã unicode của ký tự được trả về.
- **Hàm ord():** trả về số nguyên đại diện cho mã Unicode của ký tự được chỉ định.

```
def encrypt(msg_plaintext, package):
    e, n = package
    msg_ciphertext = [pow(ord(c), e, n) for c in msg_plaintext]
    return msg_ciphertext
```

2.2.3.Giải mã

$$m = c^d \bmod n$$

Quá trình chuyển đổi từ m sang M cũng diễn ra tại đây

```
def decrypt(msg_ciphertext, package):
    d, n = package
    msg_plaintext = [chr(pow(c, d, n)) for c in msg_ciphertext]
    # No need to use ord() since c is now a number
    # After decryption, we cast it back to character
    # to be joined in a string for the final result
    return (''.join(msg_plaintext))
```

3. Tài liệu tham khảo

- Mã hóa RSA hoạt động thế nào? (viblo.asia)
- Hệ mã hóa RSA và chữ ký số (viblo.asia)
- RSA (mã hóa) – Wikipedia tiếng Việt
- Định lý nhỏ Fermat – Wikipedia tiếng Việt
- Số học 4.5 - Nghịch đảo modulo (vnoi.info)
- Giải thuật Euclid mở rộng – Wikipedia tiếng Việt
- Thuật toán Euclid mở rộng, Nghịch đảo Modulo, và Định lý số dư Trung Quốc | Thien Hoang (tvhoang.com)
- 7.5 Implementing RSA in Python (toronto.edu)
- Mã hóa RSA - w3seo tìm hiểu kiến thức của mã hóa công khai RSA (websitehcm.com)
- RSA là gì? Cách thức hoạt động của mã hóa RSA (vietnix.vn)
- Số học 4 - Phi hàm Euler (vnoi.info)
- Thuật toán RSA - Tính đúng đắn, Tính an toàn và Ví dụ tính toán cụ thể - YouTube
- Hàm phi Euler – Wikipedia tiếng Việt
- Hàm chr() trong Python - HKT SOFT
- Hàm chr() trong Python - QuanTriMang.com
- What is the difference between an Ord () and a CHR () function? - Quora
- Python Socket Programming - Server, Client Example | DigitalOcean