TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÀI TẬP LỚN**

**MÔN CẤU TRÚC RỜI RẠC**

*Người hướng dẫn*: **GV.TRẦN LƯƠNG QUỐC ĐẠI**

*Người thực hiện*: **TRẦN THỊ VẸN - 52100674**

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**

**MÔN CẤU TRÚC RỜI RẠC**

*Người hướng dẫn*: **GV.TRẦN LƯƠNG QUỐC ĐẠI**

*Người thực hiện*: **TRẦN THỊ VẸN - 52100674**

Lớp **: 21050301**

Khoá  **: 25**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023**

LỜI CẢM ƠN

Trong suốt quá trình học tập và rèn luyện, chúng em đã nhận được rất nhiều sự giúp đỡ tận tình, sự quan tâm, chăm sóc của GV. Ngoài ra, chúng em còn được GV truyền đạt những kiến thức, phương pháp mới về toán hay ho và thú vị, thầy cô còn giúp sinh viên có được nhiều niềm vui trong việc học và cảm thấy thoải mái, … Chúng em xin chân thành cảm ơn các thầy cô rất nhiều trong suốt quá trình học tập này!

Bởi lượng kiến thức của chúng em còn hạn hẹp và gặp nhiều vấn đề trong quá trình học nên báo cáo này sẽ còn nhiều thiếu sót và cần được học hỏi thêm. Chúng em rất mong em sẽ nhận được sự góp ý của quý thầy cô về bài báo cáo này để chúng em rút kinh nghiệm trong những môn học sắp tới. Cuối cùng, chúng em xin chân thành cảm ơn quý thầy cô.

TP Hồ Chí Minh, ngày 04 tháng 04 năm 2023

Sinh viên:

Trần Thị Vẹn – 52100674

**ĐỒ ÁN ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là sản phẩm đồ án của chúng tôi và được sự hướng dẫn của GV Doãn Xuân Thanh;. 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, ngày 04 tháng 04 năm 2023*

*Tác giả*

*(ký tên và ghi rõ họ tên)*

*Trần Thị Vẹn*

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

Sau khi hoàn thành bài báo cáo với nội dung liên quan đến tìm kiếm nghịch đảo modulo và hệ mã hóa RSA, thì chúng ta có thể rút ra được những vấn đề như:

* Tìm kiếm nghịch đảo modulo là một kỹ thuật quan trọng trong toán học, được sử dụng rộng rãi trong các thuật toán mã hóa và giải mã. Khi sử dụng nó, cần phải hiểu rõ các khái niệm như modulo, phép nhân, phép chia,…
* Hệ mã hóa RSA là một trong những hệ mã hóa phổ biến nhất hiện nay, được sử dụng rộng rãi trong các ứng dụng bảo mật. Để hiểu rõ về hệ thống này, cần phải hiểu rõ các khái niệm như phép nhân modulo, khóa công khai và khóa bí mật,…
* Khi triển khai hệ thống mã hóa RSA, cần phải đảm bảo tính hiệu quả và độ bảo mật của hệ thống. Điều này đòi hỏi phải có sự am hiểu sâu sắc về các thuật toán liên quan và các kỹ thuật bảo mật.
* Có những mối đe dọa bảo mật tiềm ẩn khi sử dụng hệ thống mã hóa RSA, như việc tấn công Brute Force, tấn công dựa trên tính toán lượng tử, v.v. Do đó, cần phải có những biện pháp phòng ngừa và cập nhật kỹ thuật bảo mật để đảm bảo tính an toàn của hệ thống.

MỤC LỤC

TÓM TẮT iv

MỤC LỤC v

DANH MỤC HÌNH VẼ vi

CHƯƠNG 1: NGHỊCH ĐẢO MODULO CÙNG THUẬT TOÁN 1

1.1 Tổng quan về modulo và định nghĩa liên quan đến nghịch đảo modulo 1

1.1.1 Định nghĩa và tính chất của modulo 1

1.1.2 Các định nghĩa liên quan đến thuật toán nghịch đảo 2

1.2 Thuật toán Euclid và Euclid mở rộng: 3

1.2.1 Thuật toán Euclid 3

1.2.2 Thuật toán Euclid mở rộng 4

1.3 Nghịch đảo modulo 5

1.4 Thực thi, thảo luận về code cũng như kết quả nghịch đảo module: 6

1.4.1 Giải thích code và demo kết quả 6

1.4.2 Thảo luận về thuật toán 7

CHƯƠNG 2: HỆ MẬT MÃ KHÓA RSA 7

2.1 Lý thuyết về hệ mật mã khóa RSA 7

2.1.1 Khái niệm về hệ mật mã khóa RSA 7

2.2.2 Sự che dấu thông tin trong hệ thống RSA 8

2.2 Cơ sở toán học của hệ mật mã khóa RSA 8

2.2.1 Thuật toán Euclid 8

2.2.2 Thuật toán Euclid mở rộng 8

2.2.3 Thuật toán bình phương và nhân 9

2.3 Cách hoạt động của hệ mật RSA 9

2.4 Thuật toán và ví dụ hệ mật mã khóa RSA 10

2.5 Thực thi, thảo luận về code cũng như kết quả hệ mật mã khóa RSA 12

2.5.1 Giải thích code và demo kết quả 12

2.5.2 Mối đe dọa và giới hạn của hệ thống RSA code đang thực thi: 15

2.5.3 Các mối đe dọa và giới hạn của hệ thống RSA 15

2.5.4 Đề xuất để cải thiện triển khai hệ thống RSA đã triển khai 16

TÀI LIỆU THAM KHẢO I

DANH MỤC HÌNH VẼ

Hình 1: Code thuật toán Euclid 4

Hình 2: Code của thuật toán Euclid mở rộng 5

Hình 3: Cơ chế hoạt động của RSA 10

Hình 4: Bảng tính toán từng bước theo giá trị của các bít của 3 11

Hình 5: Bảng tính toán từng bước theo giá trị của các bit của 7 11

Hình 6: Bảng tính toán từng bước theo giá trị của các bít của 7 12

Hình 7: Bảng tính toán từng bước theo giá trị của các bít của 3 12

Hình 8: Chạy thử chuỗi trong RSA thực thi 14

Hình 9: Chạy thử số trong RSA thực thi 15

CHƯƠNG 1: NGHỊCH ĐẢO MODULO CÙNG THUẬT TOÁN

* 1. Tổng quan về modulo và định nghĩa liên quan đến nghịch đảo modulo
     1. Định nghĩa và tính chất của modulo

Trong điện toán, phép toán modulo là phép toán tìm số dư của phép chia 2 số (đôi khi được gọi là modulus). Kí hiệu của phép toán modulo là %.

Ví dụ: “5 mod 2″ hay 5 % 2=1 vì 5 chia cho 2 có thương số là 2 là số dư là 1.

Ta có a ≡ b (mod n) nếu a = b + kn, trong đó k là một số nguyên. Nếu a và b dương và a nhỏ hơn n, chúng ta có thể gọi a là phần dư của b khi chia cho n.

Vậy a và b đều là phần dư khi chia cho n. Người ta gọi b là thặng dư của a theo modulo n, và a là đồng dư của b theo modulo n. Phép so sánh đồng dư được ký hiệu bằng dấu ≡.

a ≡ b (mod n) hay viết thành a ≡ b mod n

Ví dụ: 42 = 6+4\*9, vậy 42 ≡ 6 (mod 9)

Modulo số học cũng như số học bình thường, bao gồm các phép giao hoán, kết hợp và phân phối. Mặt khác giảm mỗi giá trị trung gian trong suốt quá trình tính toán.

Cho a, b và n là các số nguyên, phép modulo có các tính chất:

* (a+b) mod n = ((a mod n) + (b mod n)) mod n
* (a-b) mod n = ((a mod n) – (b mod n)) mod n
* (a\*b) mod n = ((a mod n) \* (b mod n)) mod n
* (a\* (b + c)) mod n = (((a \* b) mod n) + ((a \* c) mod n)) mod n

Vì quan hệ modulo là quan hệ tương đương, nên ta có các tính chất từ quan hệ tương đương:

* Phản xạ: a ≡ a (mod n)
* Đối xứng: a ≡ b (mod n) khi và chỉ khi b ≡ a (mod n) với mọi a, b
* Bắc cầu: nếu a ≡ b (mod n) và b ≡ c (mod n) thì a ≡ c (mod n)

Nếu a1 ≡ b1 (mod n) và a2 ≡ b2 (mod n), hoặc a ≡ b (mod n), thì:

* a + k ≡ b + k (mod n) với mọi số nguyên k
* k a ≡ k b (mod n) với mọi số nguyên k
* a1 + a2 ≡ b1 + b2 (mod n) (bảo toàn phép cộng)
* a1 - a2 ≡ b1 - b2 (mod n) (bảo toàn phép trừ)
* a1\*a2 ≡ b1\*b2 (mod n) (bảo toàn phép nhân)
* ak ≡ bk (mod n) với mọi số nguyên không âm k (bảo toàn phép mũ)
* p(a) ≡ p(b) (mod n), với mọi đa thức p(x) có hệ số nguyên (bảo toàn với đa thức)

Nếu a ≡ b (mod n), ta thường dễ nhầm cho rằng ka ≡ kb (mod n).

Nếu c ≡ d (mod φ(n)), với φ is Hàm phi euler, thì ac ≡ ad (mod n) — nếu như a nguyên tố cùng nhau với n. Đối với việc loại bỏ phần tử ở hai bên, ta có:

* Nếu a + k ≡ b + k (mod n), với k là số nguyên bất kì, thì a ≡ b (mod n)
* Nếu k a ≡ k b (mod n) và k nguyên tố cùng nhau với n, thì a ≡ b (mod n)
* Nếu k a ≡ k b (mod kn) , thì a ≡ b (mod n)

Các phép tính trong các hệ mật mã hầu hết đều tính theo một modulo N nào đó. Ví dụ: Áp dụng các tính chất của Modulo số học, tính:

(14 + 7) mod 15 → (21) mod 15 = 6

(7 − 11) mod 13 → (−4) mod 13 = 9

(7 \* 11) mod 20 → (77) mod 20 = 17

(12 + 18) mod 7 = (12 mod 7 + 18 mod 7) mod 7 = 2

* + 1. Các định nghĩa liên quan đến thuật toán nghịch đảo
* **Ước số**

Số nguyên b không âm được gọi là ước số của a, nếu có số m sao cho: a = mb trong đó a, b, m đều nguyên. Khi a chia hết cho b, ta ký hiệu là b|a.

Ví dụ: 1, 2, 3, 4, 6, 8, 12, 24 là các ước số của 24.

* **Ước số chung lớn nhất**

Hai số nguyên dương a và b. Bài toán tìm ước chung lớn nhất của hai số nguyên dương là bài toán chung của lý thuyết số. Ta ký hiệu GCD(a,b) là ước số chung dương lớn nhất của a và b, tức là số nguyên dương vừa là ước của a vừa là ước của b và là số nguyên dương lớn nhất có tính chất đó.

Ví dụ: GCD (60,24) = 12; GCD (6, 15) = 3; GCD (8, 21) = 1.

* **Số nguyên tố**

Số nguyên tố là số tự nhiên lớn hơn một không phải là tích của hai số tự nhiên nhỏ hơn. Nói cách khác, số nguyên tố là những số chỉ có đúng hai ước số là một và chính nó.

Ví dụ: 2, 3, 5, 7, 11, 13, 17, 23, …

* **Nguyên tố cùng nhau**

Ta thấy 1 bao giờ cũng là ước số chung của hai số nguyên dương bất kỳ. Nếu GCD (a, b) = 1, thì a, b được gọi là hai số nguyên tố cùng nhau.

Ví dụ: GCD (8,15) = 1; do đó 8 và 15 là hai số nguyên tố cùng nhau.

* 1. Thuật toán Euclid và Euclid mở rộng:
     1. Thuật toán Euclid

Trong toán học, giải thuật Euclid (hay thuật toán Euclidean) là một giải thuật để tính ước chung lớn nhất (UCLN) của hai số nguyên, là số lớn nhất có thể chia được bởi hai số nguyên đó với số dư bằng không. Thuật toán Euclidean là một thuật toán đệ quy. Độ phức tạp của thuật toán: O(logmax(a,b)). Muốn tìm ước chung lớn nhất của hai số bất kỳ bằng thuật toán Euclid, ta cần:

* Bước 1: Trong hai số đã cho, ta lấy số lớn hơn chia cho số nhỏ hơn.
* Bước 2: Nếu phép tính chia vừa thực hiện ở Bước 1 còn dư thì ta tiếp tục lấy số chia đem chia cho số dư. Ta cứ tiếp tục thực hiện như vậy cho tới khi ta nhận được số dư bằng 0 thì ta dừng lại.
* Bước 3: Số chia trong phép chia hết cuối cùng nhận được chính là ước chung lớn nhất mà ta cần tìm.

Ví dụ: tính gcd(1970,1066)

1970 = 1 × 1066 + 904 gcd(1066, 904)

1066 = 1 × 904 + 162 gcd(904, 162)

904 = 5 × 162 + 94 gcd(162, 94)

162 = 1 × 94 + 68 gcd(94, 68)

94 = 1 × 68 + 26 gcd(68, 26)

68 = 2 × 26 + 16 gcd(26, 16)

26 = 1 × 16 + 10 gcd(16, 10)

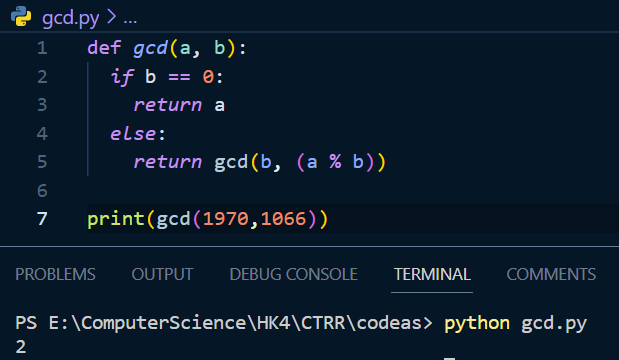
16 = 1 × 10 + 6 gcd(10, 6)

10 = 1 × 6 + 4 gcd(6, 4)

6 = 1 × 4 + 2 gcd(4, 2)

4 = 2 × 2 + 0

Vậy gcd(1970, 1066) = 2



Hình 1: Code thuật toán Euclid

* + 1. Thuật toán Euclid mở rộng

Extended Euclidean là thuật toán mở rộng thuật toán Euclid ở chỗ:

Khi a và b là 2 số nguyên tố cùng nhau, tức là UCLN(a, b) =1 với a≥ 0 và b >0

thì thuật toán cho biết thêm giá trị b-1 mod a (nghịch đảo của b trong phép chia modulo a), Độ phức tạp của thuật toán Euclid mở rộng là O(logmax(a,b)).

Ta có công thức: **b. b-1 = 1 mod a**

Thuật toán Euclid mở rộng trả về 2 giá trị:

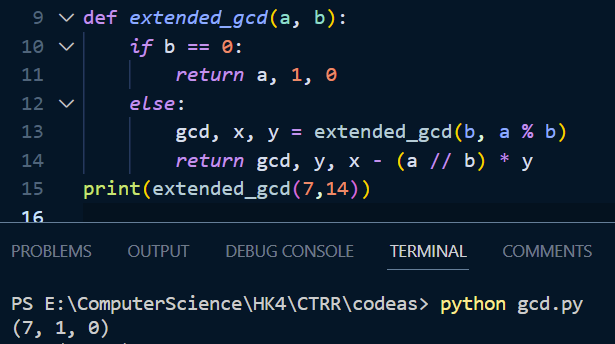
* UCLN (a,b)
* Nghịch đảo của b theo modulo a: b-1 mod a

Ví dụ: thuật toán Euclid mở rộng với a=7, b=14 như sau:

Bước 1: gcd(7, 14) = gcd(7, 14 % 7) = gcd(7, 0) = 7. Ta có x = 1, y = 0.

Bước 2: Từ bước 1, ta có gcd(7, 14) = 7 và x = 1, y = 0.

Tiến hành lùi về các bước trước khi đó, gcd = 7, x = 0, y = 1.



Hình 2: Code của thuật toán Euclid mở rộng

1.3 Nghịch đảo modulo

Nghịch đảo module hay Module Inverse của một số nguyên a trên miền module m ([0…m−1]) là một số nguyên a−1 thoả mãn: **a∗a-1 ≡ 1(mod m).** Có thể chứng minh được rằng a−1 tồn tại nếu và chỉ nếu a, m **nguyên tố cùng nhau**, tức gcd(a, m)=1.

**Lưu ý:** không phải lúc nào a−1 cũng tồn tại.

Ví dụ: m=4, a=2, sẽ không tồn tại a−1 thoả mãn do nó không phải là số nguyên tố cùng nhau. Các bước tìm nghịch đảo modulo bằng thuật toán Euclid mở rộng:

* Tìm ước số chung lớn nhất (GCD) của hai số a và b bằng thuật toán Euclid mở rộng.
* Để tìm nghịch đảo của a modulo m, chúng ta cần đảm bảo rằng a và n là hai số nguyên tố cùng nhau (UCLN(a,m) = 1). Nếu không thì nghịch đảo của a modulo m không tồn tại.
* Sau đó, chúng ta sử dụng thuật toán Euclidean mở rộng để giải phương trình ax + my = 1.
* Kết quả của phương trình này sẽ cho chúng ta giá trị của x, tức là nghịch đảo của a modulo m. Đây cũng chính là giá trị cần tìm.

Ví dụ: Tìm số nguyên dương x sao cho 7x ≡ 1 (mod 12)

Bước 1: Tìm ước số chung lớn nhất (GCD) của hai số 7 và 12 bằng thuật toán Euclid mở rộng.

12 = 1×7 + 5

7 = 1×5 + 2

5 = 2×2 + 1

2 = 2×1 + 0

GCD(7, 12) = 1

Bước 2: Vì 7 và 12 là 2 số nguyên tố cùng nhau. Áp dụng thuật toán Euclid mở rộng để tìm số nguyên dương x sao cho 7x + 12y = 1.

Bước 2.1: Sử dụng phép tính như sau: r = a - qb

5 = 12 - 1×7

2 = 7 - 1×5

1 = 5 - 2×2

Bước 2.2: Tính ngược các hệ số bằng cách lần lượt thay thế các giá trị tương ứng vào công thức sau: ri = ri-2 - qi × ri-1

1 = 5 - 2×2 = 5 - 2(7 - 5) = 3×5 - 2×7= 3(12 - 7) - 2×7 = 3×12 - 5×7

Bước 3: Lấy kết quả của bước 2.3 với m = 12 là kết quả tìm được. Vì ta quan tâm đến hệ số của 7, nên ta lấy hệ số x = 3. Do đó, x = 3 là số nguyên dương cần tìm sao cho 7x ≡ 1 (mod 12).

1.4 Thực thi, thảo luận về code cũng như kết quả nghịch đảo module:

1.4.1 Giải thích code và demo kết quả

* Giải thích code:

**Hàm extended\_gcd(a, b)** được dùng để tìm GCD và hệ số mở rộng x, y của hai số nguyên a, b.

**Hàm inverse\_modulo(a, m)** được dùng để tìm nghịch đảo của a trong modulo m.

**gcd, x, y = extended\_gcd(a, m)**

Nếu gcd(a, m) khác 1, tức là a và m không nguyên tố cùng nhau, hàm sẽ trả về None. Ngược lại, nó sẽ trả về nghịch đảo của a trong modulo m.

if gcd != 1:

return None

else:

return x % m

* Kết quả chạy code:

**print("Nghich dao cua {} mod {} = {}".format(12, 7, inverse\_modulo(12,7)))**

**print("Nghich dao cua {} mod {} = {}".format(4, 8, inverse\_modulo(4,8)))**

Kết quả trên terminal:

**Nghich dao cua 12 mod 7 = 3**

**Nghich dao cua 4 mod 8 = None**

1.4.2 Thảo luận về thuật toán

Code sử dụng thuật toán mở rộng Euclid để tính GCD và hệ số mở rộng x, y, đảm bảo tính chính xác và hiệu quả của thuật toán. Nếu m không là số nguyên tố cùng nhau với a, hàm sẽ trả về None thay vì trả về một giá trị sai. Do đó, trước khi sử dụng hàm, cần kiểm tra m và a có nguyên tố cùng nhau hay không.

Một số giá trị không thể sử dụng để tính nghịch đảo modulo, như là các giá trị không phải số nguyên tố hoặc các giá trị mà GCD không bằng 1. Nếu a hoặc m quá lớn, thuật toán mở rộng Euclid sẽ tốn nhiều thời gian và tài nguyên tính toán. Trong trường hợp này, cần sử dụng các thuật toán khác như sàng Eratosthenes để tìm số nguyên tố cùng nhau với a, hoặc sử dụng các phương pháp tính toán nhanh hơn như sử dụng định lý Fermat.

CHƯƠNG 2: HỆ MẬT MÃ KHÓA RSA

2.1 Lý thuyết về hệ mật mã khóa RSA

RSA là một trong những hệ thống mã hoá bất đối xứng được sử dụng rộng rãi. Nó được đặt theo tên của 3 nhà khoa học MIT thiết kế ra nó là: Ron Rivest, Adi Shamir, và Leonard Adleman. Ý tưởng then chốt để đảm bảo tính an toàn của RSA là dựa trên sự khó khăn trong việc phân tích nhân tử của 2 số nguyên tố lớn. (a x b = c, tìm ngược lại a, b từ c là phân tích nhân tử).

2.1.1 Khái niệm về hệ mật mã khóa RSA

Giả sử p, q là các số nguyên tố lớn, pq, N=p\*q

Hệ mật RSA là hệ mật trong đó M C Zn và tập các khóa K = {k = (N,p,g,d,e): e\*d= 1 (mod (N)) } Với mỗi k = (N,p,g,d,e) ta xác định:

+ Hàm mã hóa: ek(x) = xe mod N

+ Hàm giải mã: dk(y) = yd mod N

với mọi x,y Zn. Các giá trị N, e được công khai, các giá trị p, q, d được giữ kín.

2.2.2 Sự che dấu thông tin trong hệ thống RSA

Hệ thống RSA có một đặc điểm đặc trưng là thông tin không phải luôn luôn được che dấu. Giả sử người gửi có e=17, n = 35.

Nếu muốn gửi bất cứ data nào thuộc tập sau:{1, 6, 7, 8, 13, 14, 15, 20, 21, 22, 27, 28, 29, 34} thì mọi mật mã cũng chính là data ban đầu. Để xác định chính xác số message không được che dấu (không bị thayđổi sau khi mã hóa) ta sử dụng định lý sau: "Nếu các message được mã hóa trong hệ thống RSA được xác địnhbởi số modul n = pq (p, q là số nguyên tố) và khóa công khai e thì có:

**m = [ 1 + UCLN(e-1, p-1)][1 + UCLN(e-1, q-1)]**

message không bị che dấu".

2.2 Cơ sở toán học của hệ mật mã khóa RSA

2.2.1 Thuật toán Euclid

Thuật toán Euclid dùng để tìm ước số chung lớn nhất của hai số nguyên a và b, kí hiệu là UCLN(a,b). Thuật toán này dựa trên định lý: Với mọi số nguyên a≥0 và b >0 thì UCLN(a,b)=UCLN(b, a mod b)

**Tham khảo ở mục 1.2.1**

2.2.2 Thuật toán Euclid mở rộng

Extended Euclidean algorithm là thuật toán mở rộng thuật toán Euclid ở chỗ:

Khi a và b là 2 số nguyên tố cùng nhau, tức là UCLN(a, b) =1 với a≥ 0 và b >0

thì thuật toán cho biết thêm giá trị b-1 mod a (nghịch đảo của b trong phép chia modulo a), tức là ta có: **b. b-1 = 1 mod a**

Thuật toán Euclid mở rộng trả về 2 giá trị:

* UCLN (a,b)
* Nghịch đảo của b theo modulo a: b-1 mod a

**Tham khảo ở mục 1.2.2**

2.2.3 Thuật toán bình phương và nhân

Là thuật toán tính nhanh lũy thừa tự nhiên của một số thực hoặc một số nguyên trong trường hợp số nguyên có thể được rút gọn theo một modun nào đó. Phép nâng lên lũy thừa tự nhiên bậc n của số x (x được gọi là cơ số) xn =x\*x\*x…\*x (n thừa số x) với n lớn thì số phép nhân là rất lớn. Việc tính xn dựa vào công thức đệ quy:

Với n=0 thì xn =1 còn với n>0 ta có công thức:

2.3 Cách hoạt động của hệ mật RSA

* **Tạo khóa**

Mỗi đầu cần tạo một khóa công khai và một khóa riêng tương ứng các bước:

(1) Tạo 2 số nguyên tố lớn ngẫu nhiên và khác nhau p và q, p và q có độlớn xấp xỉ nhau.

(2) Tính n = p.q và Φ(n)= (p-1).(q-1).

(3) Chọn một số ngẫu nhiên e, 1<e < Φ(n), sao cho (e, Φ(n)) = 1.

(4) Sử dụng thuật toán Euclidean mở rộng để tính một số nguyên duynhất 1<d< Φ(n) thỏa mãn e.d ≡ 1 (mod Φ(n)) (d là nghịch đảo của e đốivới Φ(n)).

(5) Khóa công khai là cặp số (n, e). Khóa bí mật là d.

Các số nguyên e và d ở trên được gọi là số mũ mã hóa và số mũ giải mã.

* **Mã hóa**

A mã hóa một thông báo m để gửi cho B bản mã cần giải. A phải thực hiện:

(1) Thu nhận khóa công khai (n, e) của B.

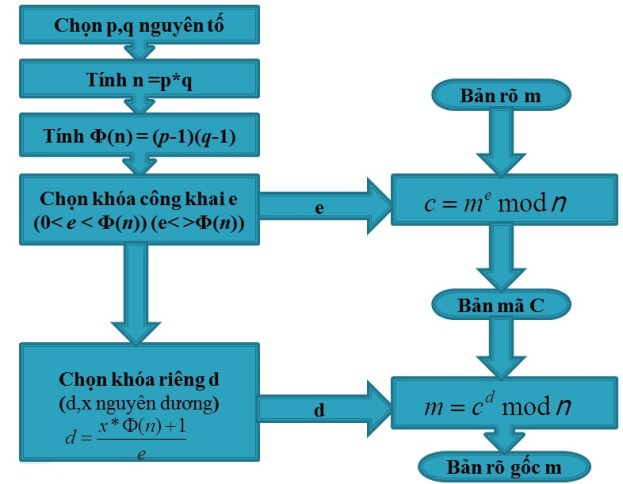
(2) Biểu diễn bản tin dưới dạng một số nguyên m trong dải [0, n -1].

(3) Tính c = me mod n.

(4) Gửi bản mã c cho B.

* **Giải mã**

Khôi phục bản rõ m từ c. B phải thực hiện phép tính m = cd mod n bằngcách dùng khóa riêng.



Hình 3: Cơ chế hoạt động của RSA

2.4 Thuật toán và ví dụ hệ mật mã khóa RSA

Các bước của thuật toán:

1) Chọn 2 số nguyên tố lớn p và q, tính N=pq. Cần chọn p và q sao cho M<2i-1<N<2i, với i=1024 thị N là một số nguyên dài khoảng 309 chữ số.

2) Tính n=(N) =(p+q) = (p-1)/(q-1)

3) Tìm 1 số e sao cho e nguyên tố cùng nhau với n.

4) Tìm 1 số d sao cho d.e=1 (mod n)

5) Hủy bỏ n, p, q.

Khóa công khai KU —e, (N); khóa riêng KR = (d, N);

6) Việc mã hóa thực hiện theo công thức:

Theo phương án 1: Mã hóa bảo mật, ta có: C=E(M, KU )=Me mod N

Theo phương án 2: Mã hóa chứng thực, ta có: C=E(M, KR )=Md mod N

7) Việc giải mã thực hiện theo công thức:

Theo phương án 1: Mã hóa bảo mật, ta có: M=D(C, KR )=Cd mod N

Theo phương án 2: Mã hóa chứng thực, ta có: M=D(C, KU )=Ce mod N

**Ví dụ: Bản rõ M=15 với p=11, q=3, e=3**

1) Có 2 nguyên tố p = 11; q = 3 nên N=p.q=11.3=33

2) n=(p-1) \* (q-1) = (11-1)\*(3-1)=10\*2=20

3) Chọn 1 số e nguyên tố cùng nhau với n=20 thì e=3 thỏa mãn.

4) Tính d là nghịch đảo của e trong modulo n tức là: d.e 1 (mod 20)

Ta có: d.3 = 1 (mod 20) thì d = 7 (tỉnh d=7 theo thuật toán Euclid mở rộng)

Vì 7.3 = 21 mà 21 – 20.1 + 1 tức là 21 = 1 mod 20

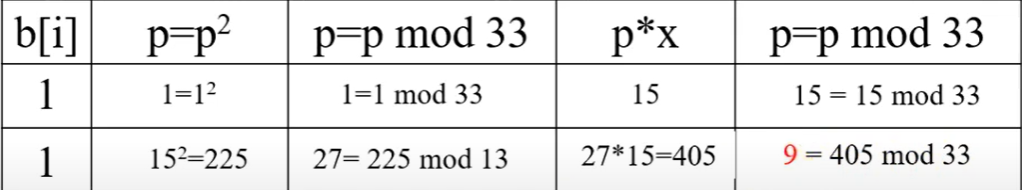
5) Khóa công khai KU=(e,N)=(3,33)

**Thực hiện mã hóa với bản rõ M = 15**

**Theo phương án 1**: Mã hóa bảo mật, ta có: C= E(M, KR)=Me mod N

Với M: 15, c = 3 và M = 33 thì C=153 mod 33.

Tính 153 mod 33 => x=15, n=3, m=33. Đổi n=3 ra số nhị phân ta được: 3(10)=11(2). Mảng b[1...k]=b[1,1].



Hình 4: Bảng tính toán từng bước theo giá trị của các bít của 3

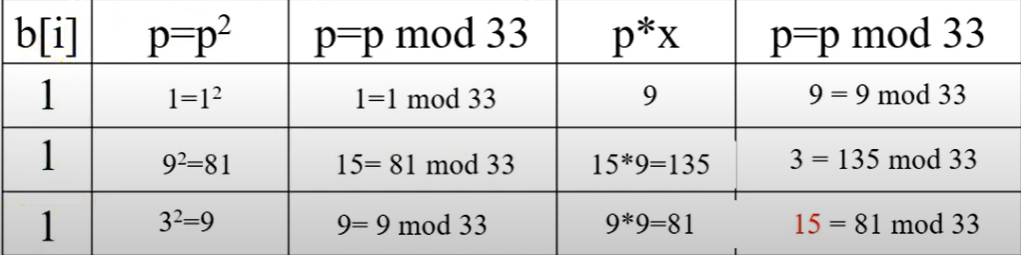
**Vậy C = 153 mod 33 =9**

**Thực hiện giải mã theo mã bảo mật C=9**

Hàm giải mã: M=D(C,KR) =Cd mod N = 97 mod 33

Tính 97 mod 33 dùng thuật toán bình phương phương và nhân =>x=9, n = 7, m =3

Đổi n=7 ra số nhị phân ta được: 70(10)=111(2). Mảng b[1...k] b[1,1,1];



Hình 5: Bảng tính toán từng bước theo giá trị của các bit của 7

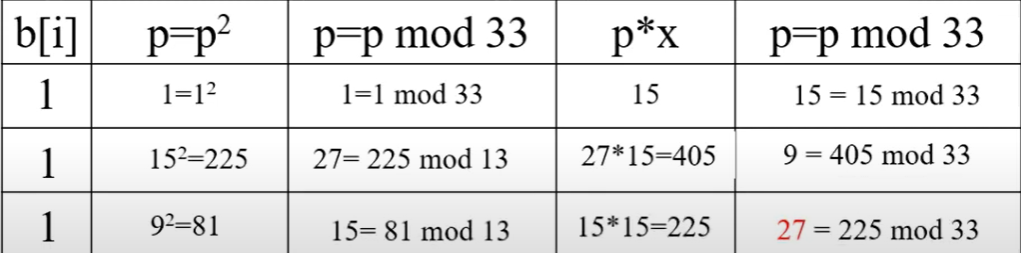
* **Theo phương án 2**: Mã hóa chứng thực, ta có: C= E(M, KR )=Md mod N

Với M=15, d= 7, M= 33 thì C=157 mod 33

Thuật toán bình phương và nhân tính xn mod m.

Tính 157 mod 33 => x=15, n=7, m=33 .Đổi n=7 ra số nhị phân ta được: 7(10)= 111(2).

Mảng b[1,...k]=b[1,1,1]



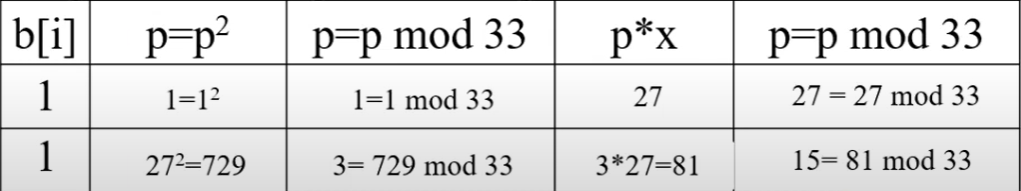
Hình 6: Bảng tính toán từng bước theo giá trị của các bít của 7

Thực hiện giải mã theo mã hóa chứng thực C = 27

Hàm giải mã: M=D(C, KU )=Ce mod N= 273 mod 33

Tính 273 mod 33 dùng thuật toán bình phương và nhân ⇒x=27, n=3, m=33

Đổi n=3 ra số nhị phân ta được: 3(10)=11(2). Mảng b[1,...k] b[1,1];



Hình 7: Bảng tính toán từng bước theo giá trị của các bít của 3

2.5 Thực thi, thảo luận về code cũng như kết quả hệ mật mã khóa RSA

2.5.1 Giải thích code và demo kết quả

* **Giải thích code:**

**Hàm generate\_keys():** hàm này sử dụng thư viện RSA để tạo ra cặp khóa công khai và khóa bí mật mới. Để tạo cặp khóa, hàm này sử dụng hàm rsa.newkeys() với độ dài khóa 2048 bit. Sau khi tạo cặp khóa mới, hàm lưu khóa công khai và khóa bí mật vào hai file tương ứng.

**Hàm load\_keys():** hàm này đọc khóa công khai và khóa bí mật từ hai file tương ứng và trả về chúng dưới dạng đối tượng khóa RSA.

**Hàm encrypt(message, key):** hàm này mã hóa tin nhắn văn bản đầu vào sử dụng khóa công khai RSA đã được cung cấp, sau đó trả về dữ liệu mã hóa. Để mã hóa, hàm sử dụng hàm rsa.encrypt() với tin nhắn đầu vào được chuyển đổi sang dạng byte bằng phương thức encode().

**Hàm decrypt(ciphertext, key):** hàm này giải mã dữ liệu mã hóa đầu vào sử dụng khóa bí mật RSA đã được cung cấp, sau đó trả về tin nhắn gốc. Để giải mã, hàm sử dụng hàm rsa.decrypt() với dữ liệu mã hóa đầu vào. Nếu giải mã không thành công do khóa không đúng, hàm sẽ trả về chuỗi "Decryption error: Incorrect key".

**Hàm sign(message, key):** hàm này tạo chữ ký số cho tin nhắn đầu vào bằng cách sử dụng khóa bí mật RSA đã được cung cấp và thuật toán băm SHA-256. Hàm sử dụng hàm rsa.sign() để tạo chữ ký số, sau đó trả về dữ liệu chữ ký số.

**Hàm verify(message, signature, key):** hàm này xác minh tính hợp lệ của chữ ký số đầu vào sử dụng khóa công khai RSA đã được cung cấp và thuật toán băm SHA-256. Để xác minh tính hợp lệ, hàm sử dụng hàm rsa.verify(). Nếu chữ ký số hợp lệ, hàm sẽ trả về giá trị True, ngược lại sẽ trả về False.

* **Kết quả chạy demo:**

**Enter message:** Lop 210301 khoa CNTT DH Ton Duc Thang

**Ciphertext:**

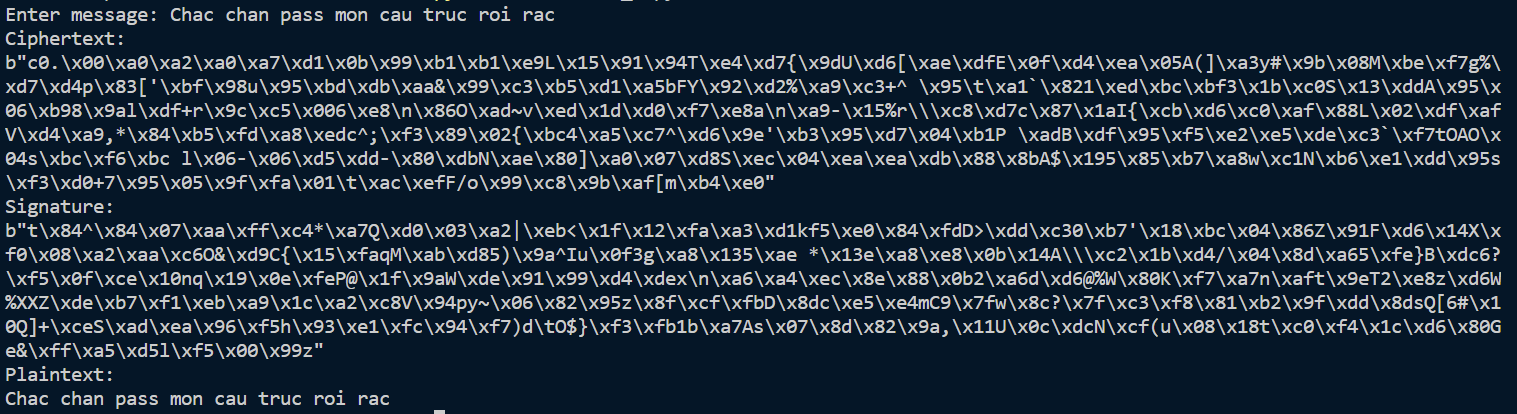
b"z\xca\xde\xfb\xfd\xeey?Iq\x9fS]\r\xb6\x19\xc3\xdd\xde\xdb\xffRU\x06\x07\xcc\x8a\x1e\x99;\xde\n%\xd2\xa4\xb7\xa6b\xb7\xeb\xd9\xfb\xe3\xf1\xe5\xd6\xec\x84\xe7\x18`\xeb\x83\xcc\xf9\xe2=\xabSS]\xf1H\xa6\xae\x89D'\xff3t\_w\xa8\xe6K\xe27M(\x98TxabcO\xfc~vaL\x1e\t;\xf9\x92\x86\x8b\xf9f\xf6.r\x92\*zHgA\x92\xb1\x01\x87\x18\x1f4\xdf\xf3;\xe8\xe3\x9a\x85\x15\x11#\xc4\xb0\x82\xf9t\xe5W\xbc\x9a\xe2NpW\x87y\xc2L\x10\xbd\xe7\x83\x1f\xf8\x13\xca\xd4bl2\xb7\xf09w\xe2.\xa2\x1f>\x8al\x1aZ;\x86\xa9\xeek/1\x12\x9b\xb5\xa7\xcd~\xdds\x14~\xa5\xaf\xed\x951\x9c\x11\xca\x0cs\x16k3\xdb\xecI\xc2t(\x99\x0b\x15\x9aBxp\xd4\xb5F\x9e\x1e\xf8\xb7\xc5\xa0\x0b\x06\xb2\x92\x91\xca\xc8`d\xb6\xf6%\xd6W\xda\xd2s\xae\xf6\xe8\xc9\xc2f\xe1'\x05\xa9K\xaar\xda}J\xdan"

**Signature:**

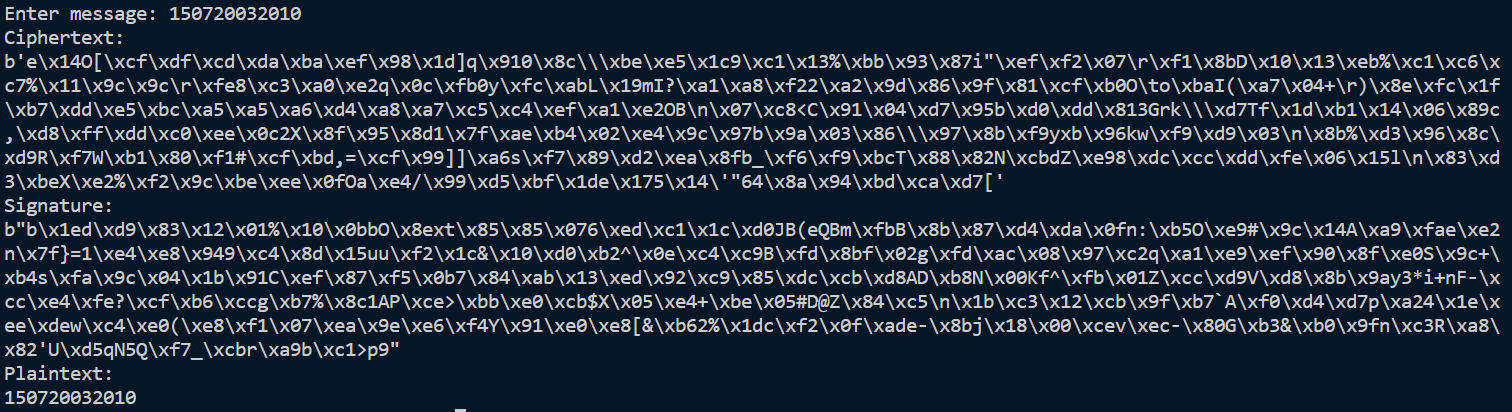
b"H\xd0\xea\xef>\xbe\xc2\x16\xe6'\xbag\x12\x9d\x85\r\xe6\xbdj6\x87\x88'\x15\xa3\xee!ic\xa4\xe3\x8f/\x0c\x11\x07\xda\xf8\xfa<s\xb2{\xcc\xd7\xf1I\xac\xf8\xe2\xfbr\x82f\xe1\x1a\xd6\xaeW\x07\xaf\xe8\r\x9e6\x10 \xd1(\xceS\xdd\x85\rmLj\x86T\x00w\xf6e\xae\xe8\x00Q\xdb<!\xf7V\x9c\xb8\x96\xa6\xd5S\xb0\xd0\xa1<\xc7\xeb\xfc\xc9\xa8\x06\x98\xf4\xfd\xe5\xbd\xea\x87U\xb1\x98\x06\x8a5#X]\xa6\xb9\xda\xe8|m@\x92Z\x9aFW\x87\xa0\x1d\x80\x02\x04\xda\x97\xe4\x04\xfb\xfc\*\x12p\xc8\xa4\r`x\xbd\x94\x95m\xcb\x7fQ\xba\xd7f\x95\xec\xc0\xd8d;9\xfd\xdc\xdf[\x1e.\xd1\xa1:\xd8\x8b\xd3\x05Oy&\xf9o\xbaI-\xecW5)\x01E\xe6`\xa8\xec\x9b2d.y\xd6\xe0\xa2\x7f\xb95%\xcf9\xb3\x85\xdd\x1e\xce\xefCE\xa7\x03o\xebqa\x10\x1c\x882>T\xc7\xccZ\xae\x1a\xf2\\\xa1\xf6\xf9C\xda\x96\xe18w\x9e\x82"

**Plaintext:**

Lop 210301 khoa CNTT DH Ton Duc Thang



Hình 8: Chạy thử chuỗi trong RSA thực thi



Hình 9: Chạy thử số trong RSA thực thi

2.5.2 Mối đe dọa và giới hạn của hệ thống RSA code đang thực thi:

Hệ thống RSA được triển khai có những cái vấn đề như:

* **Về hiệu quả**: hệ thống này sử dụng khóa có độ dài 2048 bit, là một mức độ bảo mật tương đối cao, đủ để chống lại các tấn công từ các máy tính hiện đại. Tuy nhiên, sử dụng các khóa có độ dài lớn hơn (ví dụ: 4096 bit) có thể cải thiện đáng kể độ bảo mật. (Nhưng nếu dùng 4096-bit thì hệ thống này khởi động rất chậm).
* **Về bảo mật**: sử dụng thuật toán băm SHA-256 cho việc ký và xác minh chữ ký. Tuy nhiên, SHA-256 đã bị phát hiện ra các lỗ hổng bảo mật trong quá khứ và có thể bị tấn công bằng các phương pháp tìm kiếm và tấn công định danh của khóa. Do đó, để cải thiện độ bảo mật, hệ thống cần sử dụng các thuật toán băm tốt hơn và thường xuyên thay đổi thuật toán để đối phó với các tấn công.
* **Về tốc độ**: mã này sử dụng mã hóa đối xứng RSA, có thể chạy chậm hơn so với các thuật toán mã hóa khác như AES. Tuy nhiên, hiệu quả của hệ thống này có thể được cải thiện bằng cách sử dụng mã hóa đối xứng khác để mã hóa thông điệp, và sử dụng RSA chỉ để mã hóa khóa đối xứng.

2.5.3 Các mối đe dọa và giới hạn của hệ thống RSA

* **Đe dọa**
* ***Tấn công theo phương pháp brute force:*** Khi độ dài của khóa RSA nhỏ, tấn công brute force có thể được sử dụng để phá vỡ hệ thống RSA bằng cách thử tất cả các khóa có thể có.
* ***Tấn công theo phương pháp đường giữa:*** Tấn công đường giữa (man-in-the-middle attack) là khi kẻ tấn công can thiệp vào quá trình truyền thông giữa hai bên, giả mạo thông tin và thực hiện các hành động độc hại.
* ***Tấn công theo phương pháp xâm nhập phần mềm độc hại:*** Nếu phần mềm chạy trên máy tính của người sử dụng bị xâm nhập bởi phần mềm độc hại.
* **Giới hạn của hệ thống RSA**
* ***Hiệu suất tính toán:*** RSA đòi hỏi tính toán số học đặc biệt phức tạp, đặc biệt là việc tính toán mũ trên các số nguyên lớn. Việc tính toán này tốn nhiều thời gian, đặc biệt là đối với các khóa lớn.
* ***Kích thước khóa:*** Để đảm bảo tính bảo mật của hệ thống RSA, kích thước của khóa phải đủ lớn. Tuy nhiên, kích thước khóa lớn có thể ảnh hưởng đến hiệu suất tính toán và cần phải được xử lý đặc biệt để tăng hiệu suất.
* ***Sự cố về toán học:*** Khi RSA được áp dụng trong thực tế, những sự cố toán học có thể xảy ra, ví dụ như tìm ra hai số nguyên tố giống nhau bằng cách tạo ra các khóa sai hoặc sự khác biệt giữa các hệ thống số trong các thiết bị khác nhau.

2.5.4 Đề xuất để cải thiện triển khai hệ thống RSA đã triển khai

* *Sử dụng các thư viện mã hóa đáng tin cậy:* Sử dụng các thư viện RSA đáng tin cậy và được kiểm chứng để đảm bảo tính bảo mật và độ tin cậy của hệ thống mã hóa.
* *Tăng kích thước khóa:* Sử dụng kích thước khóa lớn hơn để tăng tính bảo mật của hệ thống. Ví dụ, sử dụng khóa 4096-bit thay vì 2048-bit.
* *Sử dụng các thuật toán băm mạnh hơn:* Sử dụng các thuật toán băm mạnh hơn để tăng độ bảo mật của hệ thống. Ví dụ, sử dụng SHA-3 thay vì SHA-256.
* *Điều chỉnh thời gian sống của khóa:* Thay đổi thời gian sống của khóa để đảm bảo tính bảo mật của hệ thống. Ví dụ, đổi khóa sau một khoảng thời gian nhất định.
* *Sử dụng hệ thống mã hóa kết hợp:* Sử dụng các hệ thống mã hóa kết hợp để tăng cường tính bảo mật của hệ thống. Ví dụ, sử dụng RSA kết hợp với AES để tăng cường tính bảo mật của hệ thống.

TÀI LIỆU THAM KHẢO

**Tiếng Anh**

[1] "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems" by Ron Rivest, Adi Shamir and Leonard Adleman.

[2] "Handbook of Applied Cryptography" by Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone, 1996.

[3] "Cryptography Engineering: Design Principles and Practical Applications" by Niels Ferguson, Bruce Schneier, and Tadayoshi Kohn, 2010.

[4] "RSA and Public-Key Cryptography" by Richard A. Mollin, 2002.

[5] "Introduction to Modern Cryptography" by Jonathan Katz and Yehuda Lindell, 2008.