ĐẠI HỌC THÁI NGUYÊN
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

ĐINH THỊ HẢI YẾN

TÌM HIỂU KHẢ NĂNG AN TOÀN CỦA HỆ MẬT MÃ RSA

LUẬN VĂN THẠC SĨ KHOA HỌC MÁY TÍNH

THÁI NGUYÊN, 2017

ĐẠI HỌC THÁI NGUYÊN
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

ĐINH THỊ HẢI YẾN

TÌM HIỂU KHẢ NĂNG AN TOÀN CỦA HỆ MẬT MÃ RSA

Chuyên ngành: Khoa học máy tính
Mã số: 60 48 01 01

LUẬN VĂN THẠC SĨ KHOA HỌC MÁY TÍNH

Người hướng dẫn khoa học: TS. HỒ VĂN CANH

THÁI NGUYÊN, 2017

LỜI CAM ĐOAN

Tôi xin cam đoan kết quả nghiên cứu trong luận văn là sản phẩm của riêng cá nhân tôi, không sao chép lại của người khác. Trong toàn bộ nội dung của luận văn, những điều đã trình bày là của cá nhân tôi hoặc là được tôi tổng hợp từ nhiều nguồn tài liệu. Tất cả các nguồn tài liệu tham khảo có xuất xứ rõ ràng và được trích dẫn hợp pháp.

Tôi xin chịu toàn bộ trách nhiệm và chịu mọi hình thức kỷ luật theo quy định cho lời cam đoan của tôi.

Thái Nguyên, tháng 6 năm 2017

Đinh Thị Hải Yến

LỜI CÁM ƠN

Để hoàn thành luận văn "Tìm khả năng an toàn của hệ mật mã RSA" em đã nhận được sự hướng dẫn và giúp đỡ nhiệt tình của nhiều tập thể và cá nhân.

Trước hết, em xin bày tỏ lòng biết ơn chân thành đến ban lãnh đạo cùng quý thầy cô trong khoa Công nghệ thông tin – Trường Đại học Công nghệ và truyền thông, Đại học Thái Nguyên đã tạo tình dạy dỗ, truyền đạt kiến thức, kinh nghiệm và tạo điều kiện thuận lợi cho em trong suốt thời gian học tập và thực hiện đề tài.

Đặc biệt, em xin bày tỏ lòng biết ơn sâu sắc đến thầy hướng dẫn TS. Hồ Văn Canh, người đã gợi cho em những ý tưởng về đề tài, đã tận tình hướng dẫn và giúp đỡ để đề tài được thực hiện và hoàn thành.

Xin chân trọng gửi đến gia đình, bạn bè và người thân những tình cảm tốt đẹp nhất đã giúp đỡ động viên trong suốt khóa học và hoàn thành luận văn.

Thái Nguyên, tháng 6 năm 2017

Tác giả

Đinh Thị Hải Yến

MỤC LỤC

DANH MỤC HÌNH

DANH MỤC CÁC KÝ HIỆU, CÁC CHỮ VIẾT TẮT

Ký hiệu	Tiếng anh	Tiếng việt
N hoặc Z^+	Set of natural numbers or positive integers N = Z^+={1,2,3,…}	Tập hợp các số tự nhiên N hoặc các số nguyên dương Z^+
Q	Set of rational numbers: $Q = \left\{\frac{a}{b}, a, b \in Z \text{ and } b \neq 0\right\}$	Tập hợp các phân số: $Q = \left\{\frac{a}{b}, a, b \in Z \text{ và } b \neq 0\right\}$
Z_n hoặc $\frac{Z}{nZ}$	Residue classes modulo n: $Z_n = \frac{Z}{nZ} = \{0,1,2,\dots,n-1\}$	
Z_n^*	Multiplicative group: $Z_n^* = \{a \in Z_n, \gcd(a,n) = 1\}$	
kP	$kP = P \oplus P \oplus \dots \oplus P$, where P is a point (x,y) on an elliptic curve E: $y^2 = x^3 + ax + b$	$kP = P \oplus P \oplus \dots \oplus P$, trong đó P là một điểm có tọa độ (x,y) trên đường cong Elliptic E: $y^2 = x^3 + ax + b$
O_E	Point at infinity on an elliptic curve E	O là điểm tại vô cực trên đường cong Elliptic E
gcd(a,b)	Greatest common divisor of (a,b)	
lcm(a,b)	Least common multiple of (a,b)	
$\lfloor x \rfloor or [x]$	Greatest integer less than or equal to x	Lấy cận trên của x
$\lceil x \rceil$	Least integer greater than or equal to x	Lấy cận dưới của x
$(\frac{a}{n})$	Jacobi symbol, where n is composit	Ký hiệu Jacobi
J_n	$J_n = \{a \in Z_n^* : \left(\frac{a}{n}\right) = 1\}$	
ECM	Elliptic Curve Method (for factoring)	Đường cong elliptic

LLL	Lenstra- Lenstra-Lovasz lattice reducation algorithm	Giải thuật Lenstra- Lenstra-Lovaszlattice
P	Class of problems solvable in polynomial –time by a deterministic Turing machine	
$A \overset{P}{\Leftrightarrow} B$	A and B are deterministic polynomial-time equivalent	

MỞ ĐẦU

1. Lý do chọn đề tài

Ngày nay, các ứng dụng của công nghệ thông tin ngày càng phổ biến rộng rãi và đã ảnh hưởng rất lớn đến diện mạo của đời sống, kinh tế, xã hội. Mọi công việc hằng ngày của chúng ta đều có thể thực hiện được từ xa nhờ sự hỗ trợ của máy tính và mạng internet. Tất cả thông tin liên quan đến những công việc này đều do máy vi tính quản lý và truyền đi trên hệ thống mạng. Đối với những thông tin bình thường thì không có ai chú ý đến nhưng đối với những thông tin mang tính chất sống còn đối với một cá nhân hay tổ chức thì vấn đề bảo mật rất quan trọng.

Mật mã học ra đời là một ngành quan trọng và có nhiều ý nghĩa trong đời sống. Các ứng dụng mã hóa và bảo mật thông tin đang được sử dụng ngày càng phổ biến hơn trong các lĩnh vực khác nhau trên thế giới. Cùng với sự phát triển của tin học, ngành mật mã ngày càng trở nên quan trọng. Có thể nói rằng "sự ra đời của các hệ mật mã khóa công khai (Public Key Cryptography) là một cuộc cách mạng trong lĩnh vực mật mã". Hệ mật mã RSA thường được sử dụng trong các ứng dụng mà vấn đề bảo mật được ưu tiên hàng đầu. Bên cạnh đó RSA cũng được các nhóm phân tích nhằm tìm ra các mức không an toàn của nó. Các phân tích này chủ yếu là minh họa cho các mối nguy hiểm của việc sử dụng RSA không đúng cách. Do đó an toàn khi sử dụng RSA là một nhiệm vụ không hề tầm thường.

Với mong muốn hiểu rõ cách thức gửi công khai mà vẫn giữ được tính bảo mật của thông tin và khả năng an toàn như thế nào, luận văn sẽ nghiên cứu, phân tích khả năng an toàn của Hệ mật mã RSA.

2. Những đóng góp của luận văn

- Trong luận văn này tôi sẽ trình bày hệ mật mã RSA, phân tích các phương pháp tấn công vào hệ mật RSA. Sau đó xây dựng và cài đặt thuật toán thử nghiệm một phương pháp tấn công vào RSA.

3. Bố cục của luận văn

Nội dung của luận văn gồm có: Phần mở đầu, ba chương chính, kết luận, mục lục và tài liệu tham khảo. Nội dung cơ bản của luận văn được trình bày như sau:

Chương 1. Tổng quan về lý thuyết mật mã.

Ở chương này luận văn sẽ đi tìm hiểu về khái niệm mật mã, cơ sở toán học của mật mã.

Chương 2. Tổng quan về hệ mã hóa khóa công khai RSA

Ở chương này luận văn sẽ tìm hiểu và nghiên cứu hệ mật RSA.

Chương 3. Các phương pháp tấn công vào hệ mã hóa RSA

Ở chương này luận văn sẽ tìm hiểu các khả năng tấn công hệ mật RSA và cài đặt thuật toán thử nghiệm một phương pháp tấn công vào RSA.

NỘI DUNG

CHƯƠNG 1. TỔNG QUAN VỀ LÝ THUYẾT MẬT MÃ

1.1. CÁC KHÁI NIỆM CƠ BẢN

Trong lý thuyết mật mã có một số thuật ngữ cơ bản sau:

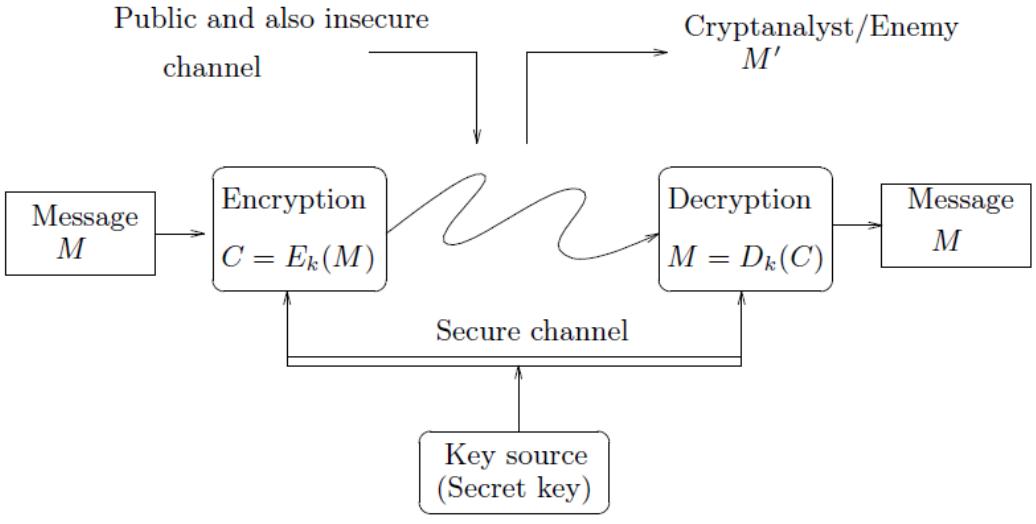
Bản rõ (PlainText): là các thông điệp cần chuyển đi và cần được bảo vệ an toàn.

Bản mã (CipherText): là thông điệp đã được mã hóa.

Mã hóa (Encryption): quá trình chuyển đổi thông tin từ bản rõ sang bản mã. Trong quá trình này thông tin trong bản rõ sẽ được ẩn đi do đó bất kì người nào đọc được thông điệp này cũng không hiểu được trừ trường hợp người có thể giải mã (*Plain Text* $\rightarrow$ *CipherText*)

Giải mã (Decryption): là quá tình giải mã để lấy lại thông tin ban đầu, ngược với quá trình mã hóa *(CipherText* $\rightarrow$ *PlainText)*.

Mật mã là nghiên cứu về quá trình mã hóa thông tin (quá trình thay đổi hình dạng thông tin gốc – bản rõ, và người khác "khó" nhận ra – bản mã, bằng việc sử dụng các khóa mã hóa), và giải mã (nghịch đảo các bản mã trở lại bản rõ, sử dụng các khóa giải mã tương ứng) để người nhận dự định có thể giải mã và đọc được thông tin ban đầu.



Hình 1.1 Lược đồ Mã hóa và giải mã thông tin

Hệ mã hóa được định nghĩa là bộ năm (M, C, K, E, D), trong đó:

- M là tập hữu hạn các bản rõ có thể
- C là tập hữu hạn các bản mã có thể
- K là tập hữu hạn các khóa có thể
- E là tập các hàm lập mã
- D là tập các hàm giải mã

Đối với mỗi khóa lập mã $ke \in K$, có hàm lập mã $e_{ke} \in E$, $e_{ke} : M \rightarrow C$

Người nhận được bản mã, họ giải mã bằng khoá giải mã $kd \in K$, có hàm giải mã $d_{kd} \in D$, $d_{kd} : C \rightarrow M$ sao cho $d_{kd}(e_{ke}(x)) = x$, $\forall x \in M$ (x- là bản rõ, e_{ke} – là bản mã)

Hệ mật mã hiện đại cần phải đáp ứng được những yêu cầu sau:

- *Tính bảo mật (Confidentiality)*: đảm bảo dữ liệu được truyền đi một cách an toàn và không bị lộ nếu như ai đó cố tình muốn có được thông điệp gốc ban đầu. Chỉ những người được phép mới có khả năng đọc được nội dung thông tin ban đầu.

- *Tính xác thực (Authentication)*: giúp cho người nhận thông điệp các định được chắc chắn thông điệp mà họ nhận là thông điệp gốc ban đầu. Kẻ giả mạo không thể giả dạng một người khác hay nói cách khác không thể mạo danh để gửi thông điệp. Người nhận có khả năng kiểm tra nguồn gốc thông điệp mà họ nhận được.

- *Tính toàn vẹn (Integrity)*: người nhận thông điệp có thể kiểm tra thông điệp không bị thay đổi trong quá trình truyền đi. Kẻ giả mạo không thể có khả năng thay thế dữ liệu ban đầu bằng dữ liệu giả mạo.

- *Tính không thể chối bỏ (Non – repudation)*: người gửi, người nhận không thể chối bỏ sau khi đã gửi hoặc nhận thông điệp.

1.2. PHÂN LOẠI CÁC HỆ MẬT MÃ

Công nghệ thông tin phát triển, việc sử dụng máy tính gia tăng cùng với tốc độ phát triển mạnh mẽ của Internet càng làm tăng nguy cơ bị đánh cắp các thông tin độc quyền. Với mối đe dọa đó có nhiều biện pháp để đối phó song mã hóa là một phương pháp chính để có thể bảo vệ các giá trị của thông tin điện tử. Có thể nói mã hóa là công cụ tự động, quan trọng nhất cho an ninh mạng và truyền thông.

Có hai hình thức mã hóa được sử dụng phổ biến là mã hóa đối xứng (*symmetric – key cryptography*) và mã hóa bất đối xứng (*asymmetric key cryptography*).

1.2.1. Mã hoá đối xứng

Là phương thức mã hóa mà trong đó cả người gửi và người nhận đều sử dụng chung một khóa để mã hóa và giải mã thông điệp hoặc, ít phổ biến hơn, người gửi và người nhận sử dụng các khóa khác nhau nhưng mối liên hệ giữa chúng dễ dàng tính toán được.

1.2.2. Mã hoá bất đối xứng

Định nghĩa mã hóa bất đối xứng: là hệ mật mã bao gồm một tập hợp các phép biến đổi mã hóa {Ee} và một tập hợp các phép biến đổi giải mã {Dd} được gọi là mật mã khóa công khai hoặc mật mã bất đối xứng nếu với mỗi cặp khóa (e, d) trong đó khóa mã hóa e được gọi là khóa công khai (có giá trị mà ai cũng biết), khóa giải mã d được gọi là khóa riêng hay khóa bí mật. Hệ mật mã này phải đảm bảo an toàn để không có khả năng tính được d từ e.

Nguyên tắc hoạt động: Người nhận B sinh ra cặp khóa gồm: khóa công khai K_P và khóa bí mật K_R. Sau đó B sẽ gửi K_P cho A và khóa này được công khai ai cũng có thể biết. A sẽ dùng K_P để mã hóa thông điệp và gửi thông điệp đã mã hóa cho B. Lúc này, B sẽ cùng K_R để giải mã thông điệp mà A gửi.

1.3. MỘT SỐ KHÁI NIỆM TOÁN HỌC

1.3.1. Ước chung lớn nhất

Ước số chung lớn nhất của các số nguyên dương a, b được kí hiệu *gcd(a, b),* là số nguyên lớn nhất mà cả a, b đều chia hết cho nó.

Và gcd (a,0) = gcd(0,a) = a.

gcd(a, b) = gcd $(|a|, |b|)$.

Ví dụ: gcd (21, 9) = 9

1.3.2. Số nguyên tố và số nguyên tố cùng nhau

Số nguyên tố là số nguyên lớn hơn 1 và chỉ chia hết cho 1 và chính nó.

Ví dụ: 2, 3, 5, 7, 11, 17, …

Hệ mật mã khóa thường sử dụng các số nguyên tố ít nhất là lớn hơn 10^{150}.

Hai số a và b được gọi là *nguyên tố cùng nhau,* nếu ước số chung lớn nhất của chúng bằng 1 và được ký hiệu là *gcd(a, b) = 1* hoặc có khi để đơn giản, người ta ký hiệu (a, b) =1.

Ví dụ: 6 và 17 là hai số nguyên tố cùng nhau.

1.4. ĐỒNG DƯ THỨC

1.4.1. Định nghĩa đồng dư thức

Cho số nguyên dương n, hai số nguyên a, b được gọi là đồng dư modulo n nếu chúng cho cùng số dư khi chia cho n (hay a - b chia hết cho n).

Kí hiệu là: $a \equiv b \ (mod\ n)$

Ví dụ: 5 ≡ 17 (mod 6) vì 5 mod 6 = 5 và 17 mod 6 = 5

1.4.2. Tính chất đồng dư thức

Cho a, a_1, b, b_1, c ∈ Z. Ta có các tính chất sau:

- ✓ Tính phản xạ: $a \equiv a \ (mod\ n)$
- ✓ Tính đối xứng: Nếu $a \equiv b \ (mod\ n)$ thì $b \equiv a (mod\ n)$
- ✓ Tính giao hoán: Nếu $a \equiv b \ (mod\ n)$ và $b \equiv c \ (mod\ n)$ thì $a \equiv c \ (mod\ n)$
- ✓ Các phép toán trên đồng dư thức

Nếu ta có:

$a \equiv a_1 (mod\ n)$

$b \equiv b_1 \ (mod\ n$

Thì ta có:

- $(a + a_1) \equiv (b + b_1) \ (mod\ n)$
- $(a - a_1) \equiv (b - b_1) \ (mod\ n)$
- $a * b \equiv a_1 * b_1 \ (mod\ n)$
- $a^k \equiv a_1^k \ (mod\ n)$ với k nguyên dương

✓ Luật giản ước

Nếu *(a * b) = (a_1 * b)(mod n)* và (b, n) =1 (b, n là nguyên tố cùng nhau) thì *a ≡ a_1 (mod n)*

✓ Hệ thặng dư đầy đủ

- Định nghĩa:

Tập hợp $\{a_1, a_2, .., a_n\}$ được gọi là hệ thặng dư đầy đủ modulo n nếu với mọi số nguyên i, $0 \leq i \leq n-1$, tồn tại duy nhất chỉ số j sao cho: $a_j \equiv i \ (mod\ n)$

- Tính chất

- Nếu $\{a_1,a_2,..,a_n\}$ là một hệ thặng dư đầy đủ mô-đun *n* thì $\{a_1 + a, a_2 + a, .., a_n + a\}$ là một hệ thặng dư đầy đủ mô-đun *n* với mọi số nguyên *a*.

- Nếu $\{a_1,a_2,..,a_n\}$ là một hệ thặng dư đầy đủ mô-đun *n* thì $\{a_1 a, a_2 a, .., a_n a\}$ là một hệ thặng dư đầy đủ mô-đun *n* với mọi số nguyên *a*.

1.5. KHÔNG GIAN Z_n VÀ Z_n^*

Không gian các số nguyên theo modulo n:

1.5.1. Không gian Z_n

Z_n là tập hợp các số nguyên không âm nhỏ hơn n. Tức là: $\boldsymbol{Z_n} = \{0,1,..,n-1\}$. Tất cả các phép toán trong Z_n đều được thực hiện theo modulo n.

Ví dụ: $Z_{11} = \{0,1,..,10\}$. Trong Z_{11}: $6 + 7 \equiv 2(mod\ 11)$

1.5.2. Không gian Z_n^*

Không gian Z_n^* là tập các số nguyên $p \in Z_n$ sao cho (p, n) = 1.

Tức là: $Z_n^* = \{p \in Z_n \mid \gcd(p,n) = 1\}$.

Nếu n là một số nguyên tố thì: $Z_n^* = \{ p \in Z_n \mid 1 \leq p \leq n-1\}$

Ví dụ: $Z_2 = \{0,1\}$ *và* $Z_2^* = |1|$*vì* $\gcd(1,2) = 1$

1.6. PHẦN TỬ NGHỊCH ĐẢO

1.6.1. Định nghĩa

Cho $a \in Z_n$. Nghịch đảo nhân của a theo modulo n là số nguyên $x \in Z_n$ sao cho: $a * x \equiv 1\ (mod\ n)$. Nếu x tồn tại thì đó là giá trị duy nhất, và a được gọi là khả nghịch, nghịch đảo của a ký hiệu là a^{-1}.

1.6.2. Tính chất

- ✓ Cho $a, b \in Z_n$. Phép chia của a cho b theo modulo n là tích của a và b^{-1} theo modulo n, và chỉ được xác định khi b có nghịch đảo theo modulo n.
- ✓ Cho $a \in Z_n$, a là khả nghịch khi và chỉ khi gcd (a, n) = 1.

✓ Giả sử d = gcd(a,n). Phương trình đồng dư ax = b mod n có nghiệm x nếu và chỉ nếu b chia hết cho d, trong trường hợp các nghiệm d nằm trong khoảng 0 đến n-1 thì các nghiệm đồng dư theo modulo n/d.

Ví dụ: $3^{-1} = 11\ (mod\ 8) vì\ 3 * 11 \equiv 1\ (mod\ 8)$

1.7. KHÁI NIỆM NHÓM, NHÓM CON VÀ NHÓM CYCLIC

1.7.1. Khái niệm nhóm

Nhóm là bộ phần tử (G,*) thỏa mãn các tính chất sau:

- Tính kết hợp: $(x * y) *\ z = x * (y * z)$
- Tính tồn tại phần tử trung gian $e\ \in G: e * x = x * e, \forall x\ \in G$
- Tính tồn tại phần tử nghịch đảo $x' \in G: x' *\ x = x * x' =\ e$

1.7.2. Khái niệm nhóm con

Nhóm con là bộ các phần tử (S,*) là nhóm thỏa mãn các tính chất sau:

- $S\ \in G$, phần tử trung gian $e\ \in S, S \subset G$
- $x, y\ \in S \Rightarrow x * y \in S$

1.7.3. Khái niệm nhóm Cyclic

Nhóm Cyclic là nhóm mà mọi phần tử x của nó được sinh ra từ một phần tử đặc biệt $g \in G$. Phần tử này được gọi là phần tử nguyên thủy, tức là:

$$\forall x \in G: \exists\ n\ \in N\ mà\ g^n = x$$

Ví dụ: (Z^+,*) là một nhóm cyclic có phần tử sinh là 1.

1.8. HÀM PHI EULER Φ(n)

1.8.1. Định nghĩa

Cho n là số nguyên dương, $n \geq 1$. Hàm phi Euler $\emptyset(n)$ được định nghĩa là số các số nguyên trong khoảng từ [1,n] nguyên tố cùng nhau với n.

Ví dụ: $\emptyset(7) =\ 6$ vì trong khoảng từ [1,6] có 6 số = {1, 2, 3, 4, 5, 6} là nguyên tố cùng nhau với 7.

1.8.2. Tính chất

✓ Nếu n là số nguyên tố thì $\emptyset(n) = n - 1$

✓ Nếu (m,n) =1 thì $\emptyset(m * n) =\ \emptyset(m) * \emptyset(n)$

✓ Nếu n được phân tích thành tích các thừa số nguyên tố n = $p_1^{e1}p_2^{e2}\ldots p_k^{ek}$ thì ta có:

$$\emptyset(n) = n\left(1 - \frac{1}{p_1}\right)\left(1 - \frac{1}{p_2}\right)\ldots(1 - \frac{1}{p_n})$$

✓ Nếu $n = p^k$, trong đó p là số nguyên tố và k ≥ 1 thì ta có:

$$\emptyset(n) = \emptyset(p^k) = p^{k-1} * (p-1)$$

1.9.3. Định lý Euler

Giả sử a, n là các số tự nhiên sao cho (a, n)=1. Khi đó ta có: $a^{\emptyset(n)} \equiv 1\ (mod\ n)$

Ví dụ: Cho a = 2, n = 35. Ta có (2, 35) =1.Phân tích n thành tích các thừa số nguyên tố, ta có: n = 35 = 5*7

Ta tính được hàm $\emptyset(n)$ dựa trên tính chất của hàm phi Euler như sau:

$\emptyset(n) = \emptyset(35) = \emptyset(5 * 7) = \emptyset(5) * \emptyset(7)$ (1.1)

Mặt khác, vì 5, 7 là các số nguyên tố nên ta có:

$$\emptyset(5) = 4\ và\ \emptyset(7) = 6 \quad (1.2)$$

Thay (1.2) vào (1.1) ta tính được $\emptyset(N) = \emptyset(35) = 4 * 6 = 24$

Áp dụng định lý Euler ta tính được: $2^{\emptyset(35)} = 2^{24}$. Do đó, $2^{24} \equiv 1(mod\ 35)$

✓ Mệnh đề 1: $\sum_{d/n} \emptyset(d) = n$

✓ Mệnh đề 2:

Cho d =(m; n) và a>1 là số nguyên. Khi đó: $(a^m - 1, a^n - 1) = a^d - 1$

1.9. CÁC PHÉP TOÁN CƠ BẢN TRONG MODULO

1.9.1. Thuật toán Euclid

Thuật toán Euclid là thuật toán xác định ước số chung lớn nhất (gcd) của 2 phần tử thuộc vùng Euclid (ví dụ: gcd của các số nguyên).

1/. Mô tả thuật toán:

Cho 2 số nguyên a, b. Ta xét các trường hợp sau:

- Nếu b là ước của a thì gcd(a,b) = b
- Nếu b không là ước của a, ta có:

  a = b*q + r khi đó ta có gcd(a,b) = gcd (b,r)

Thuật toán được mô phỏng như sau:

$$a = b * q + r_1;\ \ 0 < r_1 < b$$

$$b = r_1 * q_1 + r_2;\ \ 0 < r_2 < r_1$$

$$r_1 = r_2 * q_2 + r_3;\ \ 0 < r_3 < r_2$$
$$r_2 = r_3 * q_3 + r_4;\ \ 0 < r_4 < r_3$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$
$$r_{n-2} = r_{n-1} * q_{n-1} + r_n;\ \ 0 < r_n < r_{n-1}$$
$$r_{n-1} = r_n * q_n + 0;$$

Thuật toán kết thúc khi số dư bằng 0. Trong trường hợp 2 này ta có:

gcd(a,b) = gcd(b,r_1) = gcd(r_1,r_2)=…= gcd(r_{n-1},r_n) = r_n

2/. Giải thuật Euclid:

Input: a,b: integer, a>b>0

Output: gcd(a,b)

- Bước 1: khai báo và khởi tạo giá trị cho các biến
  - r_{-1} = a;
  - r_0=b;
  - i=0;
- Bước 2: nếu r_i=0, chuyển tới bước 4
- Bước 3: gán lại giá trị cho các biến
  - q_i= [r_{i-1}/r_i];
  - r_{i+1} =r_{i-1} – q_i*r_i;
  - i++;
  - quay lại bước 2 để kiểm tra
- Bước 4: return r_{i-1}

3/. Ví dụ:

Cho a = 287 và b = 91, dựa vào giải thuật Euclid để tìm ước chung lớn nhất của a,b. Ta có bảng kết quả như sau:

q_{i-1}	r_{i-1}
3	91
6	14
2	7
	D

1.9.2. Thuật toán Euclid mở rộng

1/. Bài toán:

Cho trước m,n (giả sử m>n). Hãy tìm x, y và k sao cho mx + ny = k. Trong đó, gcd(m,n)=k; Ta có thuật toán sau (thuật toán Euclid mở rộng):

Cho (a1,a2,a3), (b1,b2, b3), (c1, c2, c3) là ba vecto.

Bước 1: (a1, a2, a3) = (1, 0, m) ; (b1, b2, b3) = (0, 1, n);

Bước 2: Nếu b3=0 thì thuật toán dừng và nội dung trong (a1, a2, a3) là đáp số bài toán

Bước 3: Đặt q = [a3/b3]; và (c1, c2, c3) = (a1, a2, a3) – q(b1, b2, b3);

(a1, a2, a3) = (b1, b2, b3);

(b1, b2, b3) = (c1, c2, c3); và trở lại bước 2

Ví dụ: a1=x, a2=y, a3=k=(m, n). Cho m = 37, n =2. Ta có các bước trên được biểu diễn trên bảng sau:

q	a1	a2	a3	b1	b2	b3	c1	c2	c3
18	1	0	37	0	1	2	1	-18	1
2	0	1	2	1	-18	1	-2	37	0
	1	-18	1	-2	37	0			
Vậy x=1, y=-18 và k=1=(37, 2)									

2/. Ứng dụng giải thuật Euclid mở rộng tìm số nghịch đảo trong không gian Z_n

$\boldsymbol{Z_n}$ là tập hợp các số nguyên không âm nhỏ hơn n. Khi đó $\boldsymbol{Z_n = \{0, 1, 2, .., n-1\}}$

Nhận xét rằng: nếu a, b $\in$ Z_n thì:

(a + b) mod n = $\begin{cases} a+b & if \quad a+b<n \\ a+b-n & if \quad a+b \geq n \end{cases}$

Vì vậy, phép cộng modulo (và phép trừ modulo) có thể được thực hiện mà không cần thực hiện các phép chia dài.

Phép nhân modulo của a và b được thực hiện bằng phép nhân thông thường a với b như các số nguyên bình thường, sau đó lấy phần dư của kết quả sau khi chia cho n.

$$a + b = (a + b)\ (mod\ n)$$

$$a * b = (a * b)\ (mod\ n)$$

Phần tử a của Z_n được gọi là khả nghịch theo modulo n nếu tồn tại phần tử x trong Z_n với gcd(a,x) =1 thì $ax \equiv 1(mod\ n)$. Khi đó x được gọi là nghịch đảo của a theo modulo n. Khi đó tồn tại các số nguyên x, y sao cho: $a * x + b * y = 1$. Áp dụng thuật toán Euclid mở rộng ta có thể tính được phần tử nghịch đảo của a theo modulo n.

Ví dụ: Tìm x của phương trình: $5033464705x \equiv 1(mod\ 12347)$. Từ phương trình ban đầu ta có:

$5033464705x \equiv 1(mod\ 12347) \leftrightarrow 5033464705x - 12347y = 1$

Dựa vào thuật toán Euclid mở rộng trên để tính các giá trị x, y của phương trình

q	x_1	y_1	r_1
-	1	0	5033464705
407667	0	1	12347
48	1	-407667	256
4	-48	19568017	59
2	193	-78679735	20
1	-434	176927487	19
19	627	-255607222	1
	X		d

Vậy, x = 627 là nghịch đảo của $a = 5033464705$theo modulo $n = 12347$

Trong trường hợp nếu gcd(a,n) > 1, để giải phương trình $ax \equiv b(mod\ n)$ được chia ra 2 trường hợp sau: d = gcd(a,n)

- Nếu d không là ước của b thì không có giải pháp để đồng dư.
- Nếu d là ước của b, khi đó ta có phương trình mới như sau:

  $\left({}^{a}/_{d}\right)x \equiv {}^{b}/_{d}\,(mod\ {}^{n}/_{d})$.

Ví dụ: cho phương trình $803x \equiv 22(mod\ 154) \leftrightarrow 803x - 154\,y = 22(*)$

Sử dụng thuật toán Euclid mở rộng trên ta có bảng dữ liệu sau:

q	u_1	u_2	u_3
-	1	0	803
5	0	1	154
4	1	-5	33
1	-4	21	22
2	5	-26	11
	X		d

Vậy d = 11 và x =5. Nghiệm của phương trình (*) trên là tập các giá trị x thỏa mãn đẳng thức: $x + i * {}^{n}/_{d}$. Vậy ta có:

$$x = \{5, 19, 33, 47, 61, 75, 89, 103, 117, 131, 145\}$$

1.9.3. Định lý đồng dư Trung Hoa

Ta xét phương trình đồng dư tuyến tính có dạng

$$ax \equiv b(mod\ n) \qquad (1.1)$$

Trong đó a, b, n là các số nguyên, n >0 và x là ẩn số. Phương trình (1.1) có nghiệm khi và chỉ khi d =gcd(a,n)|b, và khi đó nó có đúng d nghiệm theo modulo n. Thực vậy, đặt a'=a/d, b' = b/d, n'=n/d ta thấy phương trình đồng dư (1.1) tương đương với phương trình sau:

$$a'x \equiv b'(mod\ n') \qquad (1.2)$$

Vì gcd(a',n')=1, nên phương trình (1.2) có nghiệm theo modulo n':

$$x = x_0 \equiv b'.a'^{-1}(mod\ n')$$

Và do đó, phương trình (1.1) có d nghiệm theo modulo n là:

$$x = x_0, x_0 + n', \dots, x_0 + (d-1)n' (mod\ n)$$

Tất cả d nghiệm đó khác nhau theo modulo n, nhưng cùng đồng dư với nhau theo modulo n.

Bây giờ ta xét hệ thống các phương trình đồng dư tuyến tính sau:

$$\begin{cases} x_1 \equiv a_1\ (mod\ n_1) \\ x_2 \equiv a_2\ (mod\ n_2\ ) \\ \dots\dots\dots\dots\dots\dots\dots\dots \\ x_k \equiv a_k\ (mod\ n_k) \end{cases} \qquad (1.3)$$

Ta ký hiệu: $n = n_1 n_2 \dots n_k, n_i = n/n_i$. Ta có định lý số dư CRT sau đây

Định lý

Giả sử các số nguyên $n_1, n_2, n_{3,\dots\dots\dots,}\ n_k$ là từng cặp nguyên tố với nhau. Khi đó hệ phương trình đồng dư tuyến tính (1.3) có một nghiệm duy nhất theo modulo n được cho bởi công thức sau:

$$x = \sum_{i=1}^{k} a_i . n_i . M_i\ mod\ n$$

Trong đó, $M_i = n_i^{-1}\ mod\ n_i$ (có M_i vì N_i và n_i nguyên tố cùng nhau).

Ví dụ: Cặp phương trình $x \equiv 3(mod\ 7) v\grave{a}\ x \equiv 7(mod\ 13)$ có nghiệm duy nhất $x \equiv 59(mod\ 91)$

1.10. HÀM MỘT PHÍA VÀ HÀM MỘT PHÍA CÓ CỬA SẬP

1.10.1. Hàm một phía

Một hàm một phía là hàm mà dễ dàng tính toán ra quan hệ một chiều nhưng rất khó để tính ngược lại. Ví như : Biết giả thiết x thì có thể dễ dàng tính ra f(x), nhưng nếu biết f(x) thì rất khó tính ra được x. Trong trường hợp này "khó" có nghĩa là để tính ra được kết quả thì phải mất rất nhiều thời gian để tính toán.

Ví dụ:

Tính $y = f(x) = \alpha^x \bmod p$ là dễ nhưng tính ngược lại $x = \log_\alpha y$ là bài toán "khó" (bài toán logarit rời rạc)

1.10.2. Hàm một phía có cửa sập

F(x) được gọi là hàm một phía có cửa sập nếu tính xuôi y = f (x) thì dễ nhưng tính ngược x = f - 1(y) thì khó tuy nhiên nếu có "cửa sập" thì vấn đề tính ngược trở nên dễ dàng. Cửa sập ở đây là một điều kiện nào đó giúp chúng ta dễ dàng tính ngược.

Ví dụ: $y = f(x) = x^b \bmod n$ tính xuôi thì dễ nhưng tính ngược $x = y^a \bmod n$ thì khó vì phải biết a với $a * b \equiv 1 \pmod{\emptyset(n)}$ trong đó $\emptyset(n) = (p-1)(q-1)$. Nhưng nếu biết cửa sập p, q thì việc tính $n = p * q$ và tính a trở nên dễ dàng.

Hộp thư là một ví dụ khác về hàm một phía có cửa sập. Bất kỳ ai cũng có thể bỏ thư vào thùng. Bỏ thư vào thùng là một hành động công cộng. Mở thùng thư không phải là hành động công cộng. Nó là khó khăn, bạn sẽ cần đến mỏ hàn để phá hoặc những công cụ khác. Tuy nhiên, nếu bạn có "cửa sập" (trong trường hợp này là chìa khóa của hòm thư) thì công việc mở hòm thư thật dễ dàng.

1.11. ĐỘ PHỨC TẠP TÍNH TOÁN

1.11.1. Độ phức tạp tính toán

Mục đích chính của lý thuyết độ phức tạp là cung cấp cho cúng ta kỹ thuật để phân lớp những bài toán số học theo các tài nguyên cần thiết để giải chúng. Việc phân lớp này không cần thiết phải phụ thuộc vào mô hình tính toán cụ thể nhưng cần thiết phải "đo" được độ khó của từng bài toán để giải chúng.

Nguồn tài nguyên được đo bao gồm: thời gian, không gian bộ ss, các bít ngẫu nhiên, số các bộ vi xử lý … Tuy nhiên, nguồn tài nguyên chính là thời gian và không gian bộ nhớ.

1/. Định nghĩa 1: Một thuật toán là một thủ tục tính toán được sao cho nó nhận các biến đầu vào và cho kết quả đầu ra.

2/. Định nghĩa 2: Kích cỡ đầu vào là số toàn bộ các bít cần thiết để biểu diễn đầu vào theo các ký hiệu nhị phân thông thường.

Ví dụ:

Số các bít trong biểu diễn nhị phân của một số nguyên dương n là $1 + \log_2 n$ bít. Chẳng hạn n =16 thì số bít để biểu diễn nhị phấn số n là 5. Thật vậy ta có $16_2 = 10000_2$

3/. Định nghĩa 3: Thời gian chạy của một thuật toán trên đầu vào cụ thể là số các phép toán sơ cấp hoặc số các bước mà máy tính thực hiện để cho kết quả đầu ra.

4/. Định nghĩa 4: Cho f, g là hai hàm số không âm

(i)/. Cho f(n) =O(g(n)) nếu có tồn tại một hằng số dương c và một số nguyên dương n_0 sao cho: $0 \leq f(n) \leq c.g(n)$ đối với mọi $n \geq n_0$

(ii)/. Về cận dưới xấp xỉ: $f(n) = \Omega(g(n))$ nếu tồn tại một hằng số dương c và một số nguyên n_0 sao cho: $0 \leq c.g(n) \leq f(n)$ đối với mọi $n \geq n_0$

(iii)/. Bị chặn xấp xỉ chặt: $f(n) = \theta(g(n))$ nếu có tồn tại hai hằng số c_1, c_2 dương và một số nguyên dương n_0 sao cho: $c_1 g(n) \leq f(n) \leq c_2 g(n) \quad \forall n \geq n_0$

(iv)/. $f(n) = O(g(n))$ nếu với bất kỳ hằng số dương c có tồn tại một số nguyên dương n_0 sao cho $0 \leq f(n) < c.g(n)$ đối với mọi $n \geq n_0$.

Chẳng hạn, khi viết $f(n) = O(g(n))$ có nghĩa là f không lớn hơn g với một hàng số nhân c. Một số khẳng định:

(i)/. Nếu f(n) là một đa thức bậc k với các hệ số dương khi đó, $f(n) = \theta(g(n))$

(ii)/. Với mọi hằng số c >0 ta đều có $\log_c n = \theta(logn)$

(iii)/. Với mọi số nguyên dương $n \geq 1$, ta có:$\sqrt{2\pi n}(\frac{n}{e})^n \leq n! \leq \sqrt{2\pi n}(\frac{n}{e})^{n+(\frac{1}{2n})}$

Do đó: $n! = \sqrt{2\pi n}(\frac{n}{e})^n(1 + \theta\left(\frac{1}{n}\right))$ hay $n1 = O(n^n)$ và $n! = \Omega(2^n)$

(iv)/. $\log(n!) = \theta(nlgn)$

1.11.2. Các lớp độ phức tạp

1/. Định nghĩa 1:

Một thuật toán thời gian đa thức là một thuật toán mà hàm thời gian tồi nhất của nó có dạng $O(n^k)$, trong đó n là kích cỡ đầu vào còn k là một hằng số. Một thuật toán mà thời gian chạy của nó không bị chặn được gọi là thuật toán có độ phức tạp là hàm mũ.

2/. Định nghĩa 2:

Lớp độ phức tạp P là tập hợp tất cả những bài toán quyết định có thể giải được trong khoảng thời gian đa thức.

3/. Định nghĩa 3:

Lớp có độ phức tạp NP là tập hợp tất cả những bài toán quyết định mà đối với chúng, câu trả lời "yes" có thể được kiểm tra trong thời gian đa thức nếu cho trước một thông tin phụ nào đó.

4/. Định nghĩa 4:

Giả sử cho trước L_1 và L_2 là 2 bài toán quyết định. Khi đó, nếu $L_1 \leq pL_2$ và $L_2 \leq pL_1$. Khi đó L_1 và L_2 được gọi là hai bài toán tương đương nhau về mặt tính toán.

5/. Định nghĩa 5:

Bài toán quyết định L được gọi là thuộc lớp NP- đầy đủ (NP-complete) nếu thỏa mãn:

(i) $L \in NP$

(ii) $L_1 \leq PL$ đối với mọi $L_1 \in NP$

Lớp tất cả những bài toán thuộc lớp NP- đầy đủ được ký hiệu là NPC.

Bài toán thuộc lớp NPC là những bài toán khó nhất trong lớp NP theo nghĩa ít nhất cũng khó bằng mọi bài toán trong NP. Hàng ngàn bài toán khác nhau như các bài toán tổ hợp, lý thuyết số và logic đều thuộc về lớp NPC.

Ví dụ: Cho trước tập hợp các số nguyên dương $\{a_1, a_2, \dots, a_n\}$ và một số nguyên dương s. Hãy xác định có hay không một tập con $\{a_i\} i = \overline{1,k}$ sao cho có tổng $\sum_{i=1}^{k} a_i = s$. Đây là bài toán thuộc lớp NPC.

CHƯƠNG 2. TỔNG QUAN VỀ HỆ MÃ HÓA KHÓA CÔNG KHAI RSA

2.1. MÃ HÓA KHÓA CÔNG KHAI

Hệ mã hóa công khai sử dụng 2 khóa: khóa bí mật (private key) và khóa công khai (public key)

- Khóa công khai (public key): được sử dụng để mã hóa những thông tin mà ta muốn chia sẻ với bất cứ ai. Chính vì vậy, ta có thể tự do phân phát nó cho bất cứ ai mà ta cần chia sẻ thông tin ở dạng mã hóa.
- Khóa bí mật (private key): khóa này thuộc sở hữu riêng tư của bạn và nó được sử dụng để giải mã thông tin. Chỉ mình bạn sở hữu nó, khóa này không được phép và không nên phân phát cho bất kỳ ai.

Nghĩa là mỗi người sẽ giữ hai khóa, một khóa dùng để mã hóa thông tin và công bố rộng rãi, một khóa dùng để giải mã và khóa này được giữ kín.

Hình 2.1 Sơ đồ mã hóa khóa công khai

2.2. MÃ HÓA KHÓA CÔNG KHAI RSA

2.2.1. Định nghĩa hệ mã hóa RSA

Hệ mã hóa RSA gồm một bộ các thành phần (M, C, K, n, e, d, E, D) trong đó:

- M là tập các bản rõ
- C là tập các bản mã
- K là tập các khóa, được gọi là không gian khóa
- $n = p * q$ là module với p,q là hai số nguyên tố, thường có ít nhất 10, 0 chữ số
- $\{(e, n), (d, n)\} \in K$ với $e \neq d$ là khóa mã hóa và khoá giải mã tương ứng, thỏa mãn biểu thức sau:

$$ed \equiv 1 \big(mod\ \emptyset(n)\big) \tag{2.1}$$

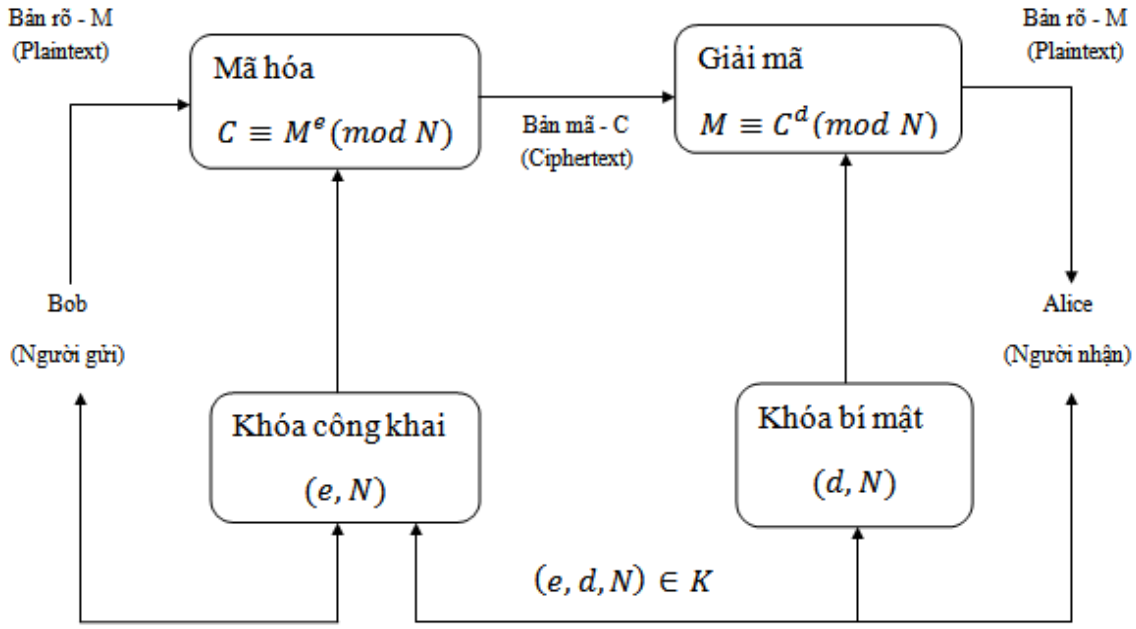
Với $\emptyset(n) = (p - 1) * (q - 1)$ là hàm Euler

- E là hàm lập mã (mã hóa): $E_{e,n}: M \rightarrow C$ (2.2)

  Sử dụng khóa công khai (e,n) để mã hóa bản rõ M, kết quả thu được là bản mã C theo phương trình sau: $C \equiv M^e (mod\ n)$ (2.3)

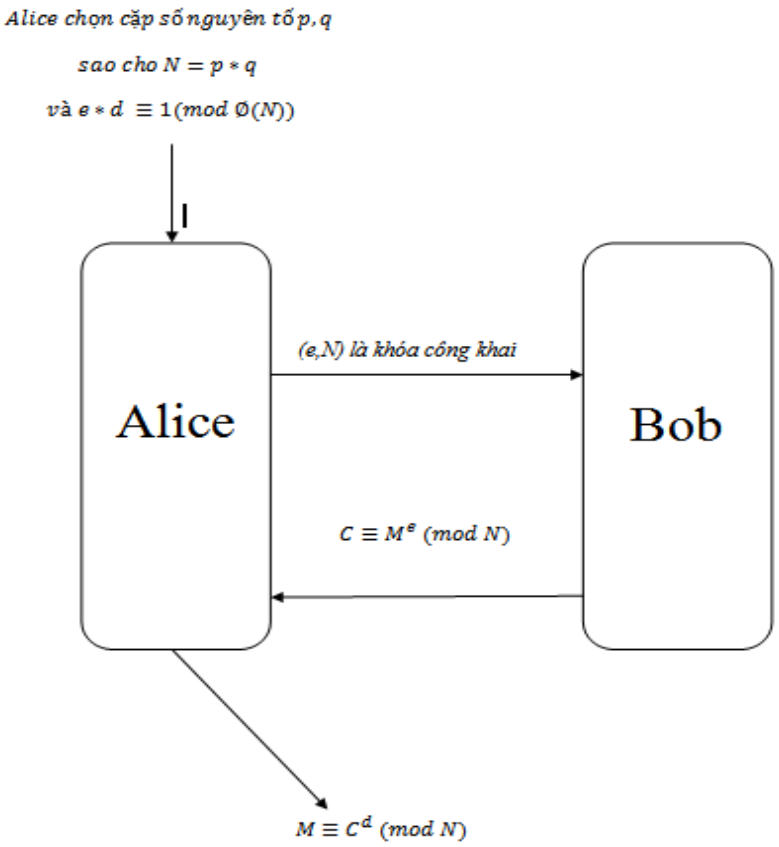
- D là hàm giải mã: $D_{d,n}: C \rightarrow M$ (2.4)

  Sử dụng khóa bí mật (d,n) để giải mã bản mã C, kết quả thu được là bản rõ M ban đầu theo phương trình sau: $M \equiv C^d \equiv (M^e)^d (mod\ n)$ (2.5)



Hình 2.2 Sơ đồ thuật toán mã hóa RSA

Ý tưởng của thuật toán mã hóa RSA có thể được mô tả theo hình bên dưới:



Hình 2.3 Sơ đồ thuật toán RSA

2.2.2. Định lý (Tính đúng đắn của RSA)

Cho M, C, n, e, d lần lượt là bản rõ, bản mã, module n, số mũ mã hóa, số mũ giải mã. Ta có: $(M^e)^d \equiv M(mod\ n)$ (2.6)

Chứng minh định lý:

Theo (2.3) ta có:

$C^d \equiv (M^e)^d\ (mod\ n)$ $\quad (theo\ C \equiv M^e (mod\ n))$

$\equiv M^{1+k\emptyset(n)}\ (mod\ n)$ $\quad (theo\ ed \equiv 1\ (mod\ \emptyset(n))$

$\equiv M.M^{k\emptyset(n)}\ (mod\ n)$

$\equiv M.\left(M^{\emptyset(n)}\right)^k\ (mod\ n)$

$\equiv M.(1)^k\ (mod\ n)$ $\quad \left(theo\ \text{định lý } Euler: a^{\emptyset(n)} \equiv 1\ (mod\ n)\right)$

$\equiv M$. Đó là điều cần chứng minh.

Từ định lý nêu trên, ta có thuật toán thực hiện mã hóa RSA như sau:

1/. Giải thuật 2.2.2:

Cho biết (e, M, n), giải thuật mã hóa kết quả trả về là C – bản mã, $C \equiv M^e \ (mod\ n)$; hoặc cho biết (d,C,n), giải thuật giải mã kết quả trả về là M- bản rõ ban đầu với $M \equiv C^d \ (mod\ \ n)$ trong thời gian đa thức $\log d$ hoặc $\log e$ tương ứng.

- Thuật toán mã hóa:

```
Input:      e, M, n
Output:     C
begin
set C=1;
while e ≥ 1 do{
      if e mod 2 =1
            then C = C.M mod n;
      M =   M^2 mod n;
      e = e/2;
}
print C
end
```

- Thuật toán giải mã:

```
Input:
      d, C, n
Output:
      M
set M=1;
while d ≥ 1 do{
      if d mod 2 =1
            then M = M.C mod n
      C =   C^2 mod n
      d = d/2;}
print M
```

2.2.3. Một số nhận xét

Nhận xét 1:

Đối với quá trình giải mã trong RSA, người dùng được biết d. Vậy nên thay vì trực tiếp thực thi đồng dư thức $M \equiv C^d (mod\ n)$, người dùng có thể tăng tốc độ xử lý tính toán bằng cách thực thi trên hai đồng dư thức sau:

$$M_p \equiv C^d \equiv C^{d\ mod\ p-1}\ (mod\ p) \quad (2.7)$$

$$Mq\ \equiv C^d \equiv C^{d\ mod\ q-1}\ (mod\ q) \quad (2.8)$$

Và được dùng trong định lý số dư Trung Hoa (CRT) để có được đồng dư thức sau:

$$M\ \equiv\ M_p.q.q^{-1}\ mod\ p +\ M_q.p.p^{-1}\ mod\ p\ (mod\ N) \quad (2.9)$$

Việc sử dụng định lý Trung Hoa là một con dao hai lưỡi. Một mặt nó cung cấp thuật toán để tăng tốc độ tính toán trong quá trình giải mã RSA, và thực hiện bởi một chip bảo mật với chi phí thấp. Mặt khác, nó có một số vấn đề an ninh nghiêm trọng dễ bị tấn công bởi một số các cuộc tấn công kênh bên (side-chanel), đặc biệt là các cuộc tấn công lỗi ngẫu nhiên.

Ví dụ:

Bảng mã ký tự được quy định như sau: Space=00, A=01, B=02, …,Z=26. Cho biết:

$$\begin{aligned} e &= 9007, \\ M &= 2008050013010709030023151804190001180500191721050113 09\_ \\ &\quad 190800151919090618010705, \\ N &= 1143816257578888676692357799761466120102182967212423 62\_ \\ &\quad 562561842935706935245733897830597123563958705058989075\_ \\ &\quad 147599290026879543541. \end{aligned}$$

Để mã hóa có thể sử dụng giải thuật 2.2.2:

$$\begin{aligned} C &\equiv M^e \\ &\equiv 9686961375462206147714092225435588290575999112457431 98\_ \\ &\quad 746951209308162982251457083569314766228839896280133919\_ \\ &\quad 90551829945157815154\ (\mathrm{mod}\ N). \end{aligned}$$

Để giải mã, từ hai số nguyên tố p, q của N biết được, người có thẩm quyền thực hiện giải mã:

$$p = 3490529510847650949147849619903898133417764638493387 8\_43990820577,$$

$$q = 3276913299326670954996198819083446141317764296799294 2\_539798288533,$$

khi đó,

$d \equiv \frac{1}{e} mod\ \emptyset(n)$

$\equiv 1066986143685780224428687713289201548070990663378 62\_80122622449663106312591177440873340168597462306553968\_$

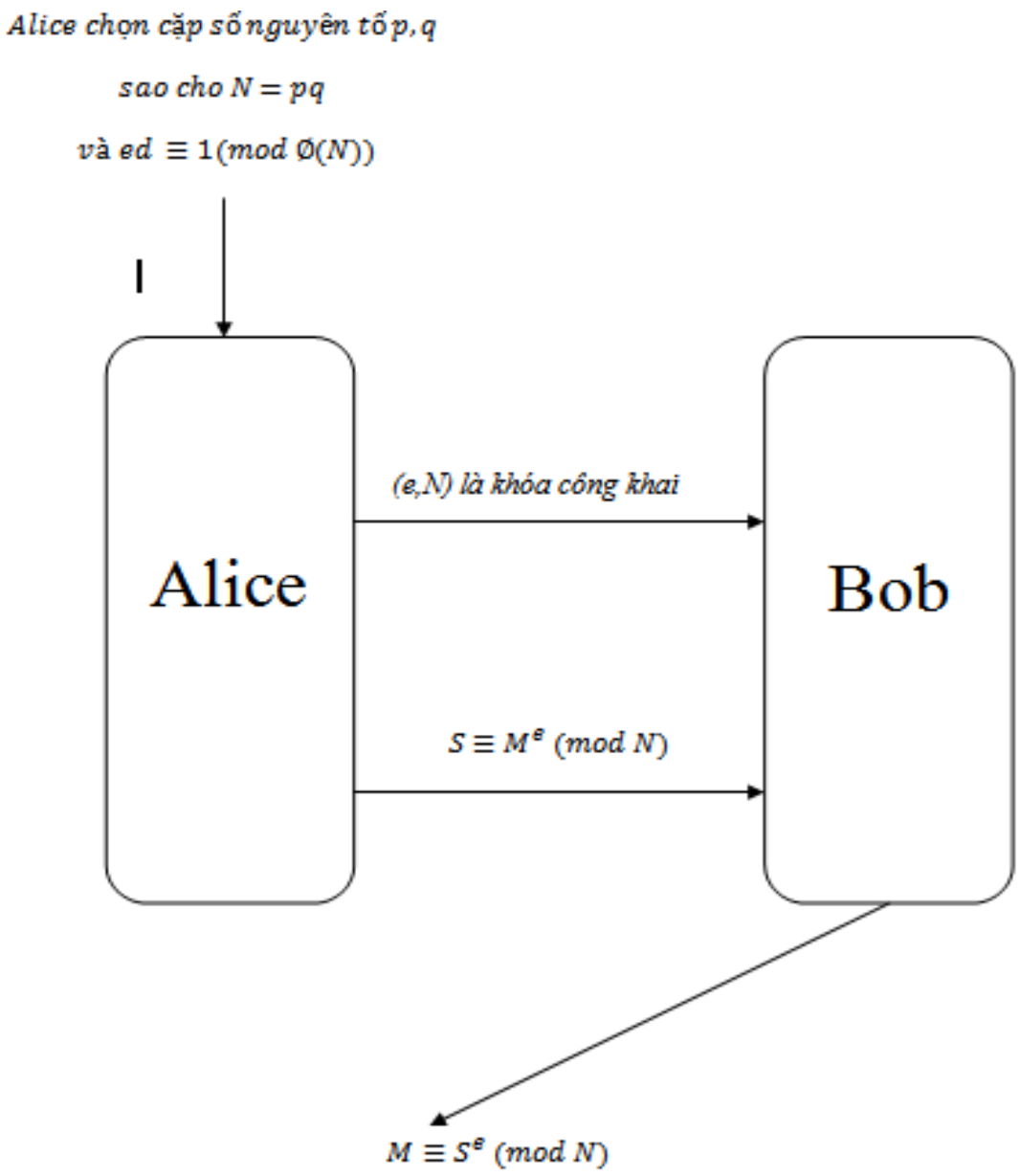
5445132771090536060 95 (mod(p-1)(q-1)).

Như vậy, bản rõ M ban đầu có thể được phục hồi một cách trực tiếp bằng cách sử dụng thuật toán 2.2.2, hoặc gián tiếp bằng cách sử dụng kết hợp các thuật toán 2.2.2 và định lý số dư Trung Hoa (CRT).

$$M \equiv C^d = 2008050013010709030023151804190001180500191721050113 09\_19080015191909061801 0705 \pmod{N}$$

Nhận xét 2:

Một trong những tính năng quan trọng nhất của mã hóa RSA là nó được sử dụng cho thuật toán chữ ký số. Cho M là một tài liệu được ký, và n=pq với p; q là các số nguyên tố, (e; d) lần lượt là số mũ khóa công khai và số mũ khóa bí mật như trong chương trình mã hóa RSA. Khi đó, quá trình ký số RSA và xác nhận chữ ký giống như quá trình mã hóa và giải mã; sử dụng d cho quá trình tạo chữ ký số và e cho quá trình xác minh chữ ký số. Ta có sơ đồ ký số RSA như hình dưới:



Hình 2.4 Sơ đồ chữ ký số RSA

- Thực hiện ký số : $S \equiv M^d (mod\ N)$ (2.10)
- Xác thực chữ ký số: $M \equiv S^e (mod\ N)$ (2.11)

2.3. CÁC VẤN ĐỀ AN TOÀN HỆ MÃ HÓA RSA

Như đã trình bày ở phần trước, toàn bộ ý tưởng của mã hóa và giải mã RSA như sau:

$$\left.\begin{array}{l} C \equiv M^e (mod\ n), \\ M \equiv C^d (mod\ n) \end{array}\right\} \quad (2.12)$$

Trong đó:

$$\left.\begin{array}{l} ed \equiv 1 (mod\ \emptyset(n)), \\ n = pq\ với\ p, q\ là\ số\ nguyên\ tố \end{array}\right\} \quad (2.13)$$

Như vậy, hàm RSA có thể được định nghĩa bởi:

$$f_{RSA}: M \to M^e\ mod\ n \quad (2.14)$$

Nghịch đảo của hàm RSA được định nghĩa bởi:

$$f_{RSA}^{-1}: M^e \to M\ mod\ N \quad (2.15)$$

Rõ ràng là hàm RSA là hàm cửa sập một chiều, với: $\{d, p, q, \emptyset(n)\}$ (2.16)

Đối với mục đích an ninh, việc thiết lập thông tin này phải được giữ bí mật và không được tiết lộ dưới bất kỳ hình thức nào thậm chí chỉ một phần.

Bây giờ giả sử Bob gửi C cho Alice, nhưng Eva muốn ngăn chặn và đọc thông tin trong C. Kể cả khi Eva biết được (e; n; C) và không biết bất kỳ thành phần nào của hàm cửa sập một chiều trong (2.16). Sau đó, Eva thực hiện giải mã để khôi phục M từ C nhưng quá trình giải mã khó có thể thực hiện được.

$$\{e, N, C \equiv M^e\ (mod\ n)\} \overset{khó}{\longrightarrow} \{M \equiv C^d\ (mod\ n)\} \quad (2.17)$$

Mặt khác, Alice biết d có nghĩa là Alice biết tất cả các thành phần trong hàm cửa sập một chiều trong (2.16), vì

$$\{d\} \overset{C}{\Leftrightarrow} \{p\} \overset{C}{\Leftrightarrow} \{q\} \overset{C}{\Leftrightarrow} \{\emptyset(n)\} \quad (2.18)$$

Chúng ta sẽ bàn tiếp (2.18) trong chương 3. Vì vậy, Alice dễ dàng khôi phục được M từ C:

$$\{N, C \equiv M^e\ (mod\ n)\} \xrightarrow[dễ]{\{d, p, q, \emptyset(n)\}} \{M \equiv C^d\ (mod\ n)\} \quad (2.19)$$

Vậy tại sao Eva khó phục hồi M từ C? Bởi vì, Eva đang đối mặt với một vấn đề tính toán khó, cụ thể là vấn đề RSA như sau:

Với khóa công khai (e; n) và bản mã C, tìm tương ứng bản rõ M trong RSA đó là:

$$\{e, n, C\} \rightarrow \{M\} \quad (2.20)$$

Điều đó là phỏng đoán mặc dù chưa được chứng minh hay bác bỏ.

Phỏng đoán RSA: với khóa công khai (e; n) và bản mã RSA, rất khó tìm thấy bản rõ M tương ứng. Đó là: $\{e, n, C\} \overset{khó}{\longrightarrow} \{M\}$ (2.21)

Với giả định RSA trên thì để thực hiện việc tìm kiếm bản rõ M tương ứng, chương trình phải thực hiện trong khoảng thời gian đa thức sau:

$$O\left(\exp\left(c(\log n)^{\frac{1}{3}}(\log\log n)^{\frac{2}{3}}\right)\right) \quad (2.22)$$

Vì vậy RSA được giả định là khó có thể phá vỡ trong thời gian đa thức và hệ thống RSA được phỏng đoán là an toàn.

2.4. CÁC BÀI TOÁN LIÊN QUAN TỚI HỆ MÃ HÓA RSA

Sự ra đời của khái niệm hệ mã bất đối xứng là một tiến bộ có tính chất bước ngoặt trong lịch sử mật mã nói chung, gắn liền với sự phát triển của khoa học tính toán hiện đại. Mã hóa bất đối xứng là một dạng mật mã hóa cho phép người sử dụng trao đổi các thông tin mật mà không cần phải trao đổi các khóa chung bí mật trước đó. Điều này được thực hiện bằng cách sử dụng một cặp khóa có quan hệ toán học với nhau là khóa công khai và khóa cá nhân (hay khóa bí mật). Trong mã bất đối xứng, khóa cá nhân phải được giữ bí mật trong khi khóa công khai được phổ biến công khai. Trong 2 khóa, một dùng để mã hóa và khóa còn lại dùng để giải mã. Điều quan trọng đối với hệ thống là khó thể tìm ra khóa bí mật nếu chỉ biết khóa công khai. Hệ thống mã bất đối xứng có thể sử dụng với các mục đích như:

- 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 một 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 mật giữa 2 bên.
- Trong thực tế, người ta thường sử dụng cả hệ mật khóa bí mật với hệ mật khóa công khai trong việc truyền tin mật.

Các kỹ thuật mã bất đối xứng đòi hỏi khối lượng tính toán nhiều hơn các kỹ thuật mã hóa khóa đối xứng nhưng những lợi điểm mà chúng mang lại khiến cho chúng được áp dụng trong nhiều ứng dụng. Các hệ mã bất đối xứng dựa trên tính chất của các bài toán cơ bản như:

2.4.1. Bài toán phân tích số nguyên thành tích các thừa số nguyên tố

Cho số nguyên dương n , tìm tất cả các ước số nguyên tố của nó, hay là tìm dạng phân tích chính tắc của $N = p_1^{\alpha_1}.p_2^{\alpha_2}...p_k^{\alpha_k}$, trong đó p_i là các số nguyên tố từng cặp khác nhau và các $\alpha_i \geq 1$.

Bài toán này có liên hệ mật thiết với các bài toán thử tính nguyên tố hay thử tính hợp số của một số nguyên, nhưng với những gì mà ta biết đến nay, nó dường như khó hơn nhiều so với hai bài toán thử tính nguyên tố và tính hợp số.

Vấn đề phân tích một hợp số thành tích các số nguyên tố có nhiều bài toán trong thực tế đòi hỏi muốn hạn chế chúng trong khong gian hẹp hơn. Trong mục này, chúng ta sẽ tìm hiểu một số mệnh đề quan trọng.

Trước hết, theo định lý nhỏ của Fermat, chúng ta biết rằng nếu n là số nguyên tố thì với bất kỳ b số nguyên dương nào sao cho (b,n)=1, ta có : $B^{n-1} \equiv 1 \bmod n$ (*)

Tuy nhiên, nếu n không phải là số nguyên tố thì hệ thức (*) có thể vẫn đúng. Ta có định nghĩa sau:

1/. Định nghĩa 1*:*

Nếu n là một tập hợp số lẻ và b là số nguyên sao cho (b,n)=1 và hệ thức (*) được duy trì. Khi đó, n được gọi là giả nguyên tố (pseudoprime) đối với cơ số b.

Ví dụ:

Cho n =91, b=3 vì (b,n)=(3,91) =1 và $3^{91} \equiv 1 \bmod 91$ (tức là thỏa mãn hệ thức (1)). Do đó, 91 là giả nguyên tố cơ số 3. Tuy nhiên số 91 không phải là giả nguyên tố đối cơ số b =2. Thật vậy, $2^{90} \equiv 64 \bmod 91 \neq 1 \ mod \ 91$.

Ta có các mệnh đề sau:

2/. Mệnh đề 1:

Cho n là một hợp số nguyên lẻ. Khi đó,

a) n là giả nguyên tố cơ số b, trong đó (b,n)=1 nếu và chỉ nếu cấp của b là ước của n -1 (trong Z_n^*).

b) Nếu n là một số giả nguyên tố đối với cơ số b1 và b2 (Trong đó (n,b1) = (n,b2) =1). Khi đó n là giả nguyên tố cơ số b1b2 và vì vậy nó cũng là giả nguyên tố cơ số $b1b2^{-1}$.

3/. Định nghĩa 2:

Giả sử n là số nguyên dương lẻ, và $(\frac{b}{n})$ ký hiệu là Jacobi. Khi đó, nếu n là số nguyên tố thì $b^{\frac{n-1}{2}} \equiv \left(\frac{b}{n}\right) mod\ n$ (**)

Một hợp số n, b là nguyên sao cho (n,b)=1 và thỏa mãn (**) thì n được gọi là giả nguyên tố Euler đối với cơ số b.

4/. Định nghĩa 3:

Cho N là một hợp số lẻ và được viết dưới dạng n-1 = 2^st với t lẻ và $b \in Z_n^*$. Khi đó nếu b và n thỏa mãn hoặc là:ư

(i) $b^t \equiv 1\ mod\ n$ hoặc là

(ii) tồn tại r: $0 \leq r \leq s$ sao cho $b^{2^r * t} \equiv -1\ mod\ n$ thì n được gọi là giả nguyên tố mạnh đối với cơ số b.

Mệnh đề 1:

Xét một số $b^n - 1$, n nguyên dương tùy ý. Nếu p là một nhân tử nguyên tố của $2^n - 1$. Khi đó, hoặc là

(i) $b^d - 1$ chia hết cho p đối với một nhân tử thực sự nào đó của n hoặc là

(ii) $p \equiv 1 mod\ n$. Trường hợp $p > 2$ và n là số tự nhiên lẻ thì trong trường hợp (ii) ta có $p \equiv 1\ mod\ 2n$.

Ví dụ:

Xét $2^{11} - 1 = 2047$. Vì 11 là số nguyên tố do đó, ta có (ii):

$p \equiv 1\ mod\ 22$. Ta kiểm tra ra:

$p = 23,67,89,\ldots\ [\sqrt{2047}]$ (thực tế chỉ cần đến $[\sqrt{2047}] = 45$ thôi.)

Ta có p =23 hoặc p =89. Tức là 2047 = 23 * 89.

Mệnh đề 2:

Xét số dạng $b^n + 1$. Khi đó nếu p là môt số nguyên tố sao cho $p| b^n + 1$ thì hoặc là:

(i) $p| b^n + 1$ với một ước thực sự của n đối với nó, $\frac{n}{a}$ là số lẻ hoặc là:

(ii) $p \equiv 1 \, mod \, 2n$

Ví dụ:

Xét m = $2^{24} + 1 = 16777217 = 97 * 172961 = 97 * 257 * 673$. Hãy tìm một nhân tử nguyên tố của m. $p \in \{97,193,289,481,577,673,\dots,4093\}$.

Ta có p = 97 hoặc p = 673. Vậy $2^{24} + 1 = 16777217 = 257 * 97 * 673$

Mệnh đề 3:

Cho n là một số nguyên dương lẻ. Khi đó có tồn tại phép tương ứng 1 đối 1 giữa các nhân tử của n có dạng n = a*b. Trong đó $0 < b \leq a$ và các biểu diễn của số n dưới dạng t^2-s^2. Ở đây, s, t là không âm và $t \geq s$. Sự tương ứng đó được cho bởi phương trình sau đây:

$$t = \frac{a+b}{2}, s = \frac{a-b}{2}, a = s + t, b = t - s$$

2.4.2. Bài toán tìm căn bậc hai module n

Cho một số nguyên lẻ n là hợp số Blum, và một số $a \in Qn$, tức a là một thặng dư bậc hai theo mod n . Hãy tìm một căn bậc hai của a theo mod n, tức tìm x sao cho $x^2 \equiv a$ (mod n).

Nếu biết phân tích n thành thừa số nguyên tố, n =$p.q$, thì bằng cách giải các phương trình $x^2 \equiv a$ theo các modp và mod q, rồi sau đó kết hợp các nghiệm của chúng lại theo định lý số dư Trung quốc ta sẽ được nghiệm theo mod n , tức là căn bậc hai của a theo mod n cần tìm. Vì mỗi phương trình $x^2 \equiv a$ theo mod p và mod q có hai nghiệm (tương ứng theo mod p và mod q), nên kết hợp lại ta được bốn nghiệm, tức bốn căn bậc hai của a theo mod n. Người ta đã tìm được một số thuật toán tương đối đơn giản (trong thời gian đa thức) giải phương trình $x^2 \equiv a$ (mod p) với p là số nguyên tố. Như vậy, bài toán tìm căn bậc hai mod n có thể qui dẫn trong thời gian đa thức về bài toán phân tích số nguyên.

Ngược lại, nếu có thuật toán A giải bài toán tìm căn bậc hai mod n thì cũng có thể xây dựng một thuật toán giải bài toán phân tích số nguyên như sau: Chọn ngẫu

nhiên một số x với gcd(x,n) =1, và tính $a = x^2 \bmod n$. Dùng thuật toán A cho a để tìm một căn bậc hai mod n của a. Gọi căn bậc hai tìm được đó là y. Nếu $y \equiv \pm x \pmod n$, thì phép thử coi như thất bại, và ta phải chọn tiếp một số x khác. còn nếu $y \not\equiv \pm x \pmod n$, thì gcd($x$-$y$, n) chắc chắn là một ước số không tầm thường của n, cụ thể là p hay là q. Vì n có 4 căn bậc hai mod n nên xác suất của thành công ở mỗi lần thử là 1/2, và do đó số trung bình (kỳ vọng toán học) các phép thử để thu được một thừa số p hay q của n là 2, từ đó ta thu được một thuật toán giải bài toán phân tích số nguyên (Blum) với thời gian trung bình đa thức.

Tóm lại, theo một nghĩa không chặt chẽ lắm, ta có thể xem hai bài toán phân tích số nguyên và tìm căn bậc hai mod n là khó tương đương nhau.

CHƯƠNG 3. CÁC PHƯƠNG PHÁP TẤN CÔNG VÀO HỆ MÃ HÓA RSA

3.1. PHÂN TÍCH NHÂN TỬ SỐ NGUYÊN LỚN

Vấn đề phân tích một số nguyên tố lớn thành tích các số nguyên tố khác nhau là bài toán rất hấp dẫn và đã được nhiều nhà toán học quan tâm nghiên cứu; Trong phạm vi của một luận văn cao học, em chỉ tập trung nghiên cứu trong trường hợp n là tích của hai số nguyên tố phân biệt. Sau đây là một số mệnh đề quan trọng phục vụ việc tấn công cơ bản:

3.1.1. Mệnh đề 1

Cho trước n = p*q, trong đó p, q là hai số nguyên tổ lẻ riêng biệt (ta giả thiết p < q). Khi đó, $\Phi(n) < n - 2*\sqrt{n} + 1$.

Chứng minh: Thật vậy, theo định nghĩa và tính chất của hàm $\Phi(n)$ ta có

$= (p - 1) * (q - 1) = p * q - (p + q) + 1 \leq n - 2 * \sqrt{n} + 1.$

(Vì $\frac{p+q}{2} \geq \sqrt{p * q} = \sqrt{n}$ hay $p + q \geq 2\sqrt{p * q} = 2\sqrt{n}$)

3.1.2. Mệnh đề 2

Giả sử n là một số tự nhiên không chính phương (perfect square), tức n không phải là bình phương đúng của một số nguyên tố, thỏa mãn điều kiện:

$$n - 1 > \emptyset(n) > n - n^{2/3}$$

Khi đó n là tích của 2 số nguyên tố phân biệt.

Chứng minh:

Thật vậy, rõ ràng n không phải là số nguyên tố vì nếu n là số nguyên tố thì $\emptyset(n) = n - 1$, trái với giả thiết. Do giả thiết n không phải là bình phương của một số nguyên tố. Như vậy nếu n không phải là tích của 2 số nguyên tố phân biệt thì nó phải là tích của nhiều hơn 2 số nguyên tố (không cần phân biệt). Giả sử p là số nguyên tố nhỏ nhất của tích; Khi đó $p \leq n^{1/3}$ do đó chúng ta có:

$$\emptyset(n) \leq n\left(1 - \frac{1}{p}\right) \leq n(1 - n^{1/3}) = n - n^{2/3}$$

Điều này mâu thuẫn với giả thiết. Vậy n = p.q trong đó p,q là 2 số nguyên tố lẻ, phân biệt. Chú ý: Mệnh đề đảo của mệnh đề 1 cũng đúng.

3.1.3. Mệnh đề 3

Với (n,e) là khóa công khai của RSA. Cho trước khóa riêng d, người ta có thể phân tích thành nhân tử modul n=pq một cách hiệu quả. Ngược lại cho các thừa số của n, người ta có thể khôi phục được d một cách có hiệu quả. Từ các mệnh đề ở trên người ta đã đưa ra một số tấn công vào RSA sau đây.

3.1.4. Thuật toán

3.1.4.1/. Thuật toán 3.1.4.1/.

1/. Bài toán đặt ra:

Cho n là số nguyên dương lẻ, $d \geq 2\sqrt[3]{n} + 1$. Nội dung bải toán này là tìm nhân tử bé nhất f của N sao cho $d < f \leq \sqrt{n}$ hoặc khẳng định rằng không tồn tại nhân tử nào như vậy.

2/. Giải thuật được thực hiện như sau:

Đầu vào: Cho số nguyên n,

Đầu ra: n= $(\frac{x-y}{2}) * (\frac{x+y-2}{2})$

Bước 1:

Đặt $x = 2 * \sqrt{n} + 1;\ y = 1; r = \lceil\sqrt{n}\rceil^2 - n;$

Bước 2:

Nếu $r \leq 0$ $goto$ bước 4;

Bước 3:

Đặt $r = r - y; y = y + 2;$

Goto bước 2

Bước 4:

Nếu $r = 0$ thuật toán kết thúc và ta có

n= $(\frac{x-y}{2}) * (\frac{x+y-2}{2})$

Bước 5:

Đặt $r = r + x;\ x = x + 2;$ Goto bước 3;

3.1.4.2/. Thuật toán 3.1.4.2/.

1/. Bài toán đặt ra:

Cho trước n= $p * q$(2< p < q) sao cho độ dài $l(p) \approx l(q) = \left[\frac{l(n)}{2}\right]$. Theo mệnh đề 1 và mệnh đề 2, ta cần tìm $\Phi'(n) = \frac{\Phi(n)}{2} = \frac{(p-1)*(q-1)}{2}$ sao cho $\Delta = (n' - \Phi'(n))^2$ - n là số chính phương khi: $\Phi'(n) \in \left[\frac{n-n^{2/3}}{2}, \frac{n-2\sqrt{n}+1}{2}\right]$. Đây là khoảng chấp nhận được.

<u>Chú ý:</u>

Trong biểu thức $\frac{n-n^{2/3}}{2}$, có chứa căn bậc 3 mà việc tính căn bậc 3 tương đối phức tạp. Do đó, chúng ta có thể tiến hành như sau:

Đặt $x = n^{2/3}$. Lấy $\log_a$ cả 2 vế ta nhận được $\log_a x = \frac{2}{3}\log_a n = [c]$.

Vậy:

$x = a^c$. Ta có thể chọn a = e hoặc a =10, chẳng hạn chọn a = 10, ta nhận được:

$x = 10^{[c]}$ trong đó [c] là số nguyên lớn nhất nhưng bé hơn hoặc bằng c.

2/. Thuật toán

Bước1:

$y = \frac{n-2\sqrt{n}+1}{2}$;

n' = (n+1)/2;

Bước 2:

While (y $\geq \frac{n-n^{2/3}}{2}$) {

$delta_i = (n' - y)^2 - n$;

Nếu $delta_i$ chính phương goto bước 3;

y=2*y-2;

lặp lại bước 2;

}

Bước 3:

$x = \sqrt{delta_i}$;

Nếu $\big((y-x),(y+x)\big) = 1$ *thì*

$p = (y-x); q = (y+x)$;

Bước 4: Nếu $delta_i$ không phải là số chính phương và y $\notin \left[\frac{n-n^{2/3}}{2}, \frac{n-2\sqrt{n}+1}{2}\right]$ thì thuật toán dừng và không tìm được $\Phi'(n)$.

3.2. TẤN CÔNG DỰA TRÊN VIỆC PHÂN TÍCH SỐ NGUYÊN n THÀNH TÍCH THỪA SỐ NGUYÊN TỐ

3.2.1. Phương pháp phân tích n thành tích thừa số nguyên tố của Fermat (Fermat Factoring Attack)

Hai phương pháp này phân tích một số bằng cách biểu diễn chúng dưới dạng hiệu của hai số chính phương. Những phân tích này sẽ thành công khi khoảng cách giữa hai số nguyên tố tạo nên nó là rất nhỏ, hoặc khi tỷ lệ của chúng gần với tỷ lệ của hai số nguyên nhỏ. Cho N là một thừa số đủ lớn hoặc cho RSA là khó có thể phá vỡ, thì p; q là 2 số nguyên tố được chọn nên có cùng số bit. Tuy nhiên, nếu chọn p; q quá gần nhau thì người ta có thể sử dụng phương pháp phân tích thừa số của Fermat để tìm thấy chúng.

Cho n =pq, trong đó $p \le q$ là các số lẻ, khi đó $x = \frac{1}{2}(p+q) và\ y = \frac{1}{2}(q-p)$ là các số chúng ta cần tìm thỏa mãn n= $x^2 - y^2 = (x+y)(x-y)$, hoặc $y^2 = x^2 - n$. Ta có thuật toán phân tích thừa số của Fermat được cài đặt như sau:

1/. Thuật toán 3.2.1 (Fermat Factoring Attack):

Input: số nguyên lẻ n >1

Output: Xác định được các thừa số lớn nhất $\le \sqrt{n}\ của\ n$

Begin

[1] Nhập n, gán $k = \lfloor \sqrt{n} \rfloor + 1, y = k.k - n, d = 1$

[2] If $\lfloor \sqrt{y} \rfloor = \sqrt{y}$ goto step [4];

Else {

$y = y + 2.k + d;$

$d = d + 2;$}

[3] If $\lfloor \sqrt{y} \rfloor \le n/2$ goto step [2]

Else{

Print “Không tìm thấy thừa số”;

goto step [5];}

[4] $x = \sqrt{n+y},\ \ y = \sqrt{y};$

print x-y và x+y; //đó là các thừa số của n

[5] $Exit;$

2/. Định lý 3.2.1

Cho n =pq với p > q và $\Delta = p - q$. Khi $\Delta < n^{1/4}$, thì giải thuật 3.2.1 có thể phân tích n thành thừa số.

3.2.2. Phương pháp phân tích $p \pm 1$ và đường cong Elliptic

Các số nguyên tố p,q của RSA được chọn có dạng $p \pm 1$ và $q \pm 1$, và có ít nhất một số nguyên tố có giá trị lớn hơn 10^{20}. Nói một cách khác, số nguyên tố p có thể được tìm thấy dựa trên thuật toán phân tích "*p-1*" của Pollard hoặc dựa trên thuật toán phân tích "*p+1*" của Williams.

1/. Giải thuật 3.2.2.1 ("p-1" factoring)

Input: cho số nguyên lẻ n >1

Output: tìm được các thừa số nguyên tố của n

Begin

[1] (khởi tạo) chọn ngẫu nhiên $a \in Z_n$
Chọn B mịn, k = lcm(1,2,..B)
[2] Tính $a_k \equiv a^k \ (mod\ n)$;
[3] Tính $f = \gcd(a_k - 1, n)$;
[4] If $1 < f < n$ thì f là một thừa số nguyên tố của n, output f và goto [6];
[5] If f không là một thừa số nguyên tố của n, muốn tiếp tục tìm thì goto [2];
Else goto [6];
[6] exit

end

Phương pháp này hiệu quả khi số n cần phân tích có các thừa số nguyên tố p có dạng $p-1$ là mịn, nghĩa là $p-1$ chỉ chứa các thừa số nhỏ. Độ phức tạp của giải thuật là $O(B \log B (\log n)^2)$, giải thuật này hiệu quả hơn khi chọn B là thừa số nguyên tố lớn nhất của p-1.

Ví dụ 3.2.2.1:

Cho biết n = 540143. Chọn B = 8, do đó $k = lcm(1,2,3,4,5,6,7,8) = 840$. Chọn a =2. Khi đó:

$$f = \gcd(2^{840} - 1 \; mod \; 540143, 540143) = \gcd(53046, 540143) = 421$$

Thực vậy, 540143 = 421*1283. Chú ý rằng tất cả các thừa số nguyên tố của 421-1 là rất nhỏ: 421 – 1 = 2^2.3.5.7

2/. Giải thuật 3.2.2.2 (Lentra's Elliptic Curve Method)

Các phương pháp trên mất rất nhiều thời gian tăng theo cấp số mũ của chiều dài theo bit của p, các thừa số mà chúng tìm thấy rất chậm. Phương pháp này cao cấp hơn chúng, độ phức tạp của phương pháp này là $O(e^{\sqrt{2(\ln p)(\ln \ln p)}})$. Phương pháp này thường hiệu quả khi thừa số bé của n chỉ có khoảng từ 13 đến 47 chữ số còn thừa số lớn thì lại có thể lớn hơn rất nhiều. Thừa số lớn nhất (có 67 chữ số) được tìm thấy bằng ECM vào 24/8/2006 bởi B. Dodson.

Input: n > 1 là một hợp số với gcd(n,6) = 1

Output: tìm ra thừa số nguyên tố của n sử dụng giải thuật đường cong elliptic kết hợp với thuật toán phân tích thừa số "p-1" của Pollard

begin

[1] Chọn một cặp số ngẫu nhiên (E,P)
Trong đó, E là đường cong elliptic thỏa mãn biểu thức $y^2 = x^3 + ax + b$ trên Z_n, và $P(x, y) \in E\ (Z_n)$ là 1 điểm trên E.
Chọn ngẫu nhiên $a, x, y \in Z_n$, gán $b = y^2 - x^3 - ax$
If $\gcd(4a^3 + 27b^2) \neq 1$, thì E không phải là 1 đường cong elliptic, bắt đầu lại từ đầu và chọn cặp số ngẫu nhiên (E,P) khác.
[2] Trong thuật toán "p-1" của Pollard. Chọn B mịn, k = lcm(1,2,..B)hoặc k = B!
[3] Tính điểm $kP \in E\ (Z/nZ)$. Sử dụng công thức sau để tính:$P_3(x_3, y_3) = P_1(x_1, y_1) + P_2(x_2, y_2) \; mod \; N$:

$$(x_3, y_3) = (\lambda^2 - x_1 - x_2 \; mod \; n, \lambda(x_1 - x_3) - y_1 \; mod \; n)$$

Trong đó:

$$\lambda = \begin{cases} \frac{m_1}{m_2} \equiv \frac{3x_1^2 + a}{2y_1} (mod\ n) \;\; if \; P_1 = P_2 \\ \frac{m_1}{m_2} \equiv \frac{y_2 - y_1}{x_2 - x_1} (mod\ n) \end{cases}$$

[4] If $kP \equiv O_E (mod\ n)$ thì gán $m_2 = z$ và tính $d = \gcd(z, n)$
Else goto [1] để chọn một giá trị a mới hoặc chọn lại cặp số (E,P)
[5] If $1 < d < N$, khi đó d là số cần tìm, output d và goto [7]
[6] Nếu d không phải là số cần tìm, goto [1] để lặp lại việc tìm kiếm ngược lại goto [7]
[7] Exit
[8] end

Ví dụ 3.2.2.2:

Cho n = 187:

[1] Chọn B=3, k = lcm(1,2,3) = 6. Gán P=(0,5) là 1 điểm trên đường cong Elliptic E: $y^2 = x^3 + x + 25$ với $gcd(n, 4a^3 + 27b^2) = gdc(187,16879) = 1$ với a =1 và b =25

[2] $k = 6 = 110_2$, ta có 6P=2(P+2P) sẽ được tính theo dưới đây:

a/. $2P = P + P = (0,5) + (0,5)$

$$\begin{cases} \lambda = \dfrac{m_1}{m_2} = \dfrac{1}{10} \equiv 131\ (mod\ 187) \\ x_3 = 144\ (mod\ 187) \\ y_3 = 18\ (mod\ 187) \end{cases}$$

Vậy 2P=(144,18) với m_2 = 10 và λ =131

b/. $tính\ 3\ P = P + 2P = (0,5) + (144,18)$

$$\begin{cases} \lambda = \dfrac{m_1}{m_2} = \dfrac{13}{144} \equiv 178\ (mod\ 187) \\ x_3 = 124\ (mod\ 187) \\ y_3 = 176\ (mod\ 187) \end{cases}$$

Vậy 3P =(124, 176) với m_2 = 144 và λ =178

c/. $tính\ 6P = 2(3P) = 3P + 3P = (124,176) + (124,176)$

$$\lambda = \frac{m_1}{m_2} = \frac{46129}{352} \equiv \frac{127}{165} \equiv O_E(mod\ 187)$$

$$với\ m_1 = 127\ và\ m_2 = 165\ đặt\ z = m_2 = 165$$

d/. $d = gcd(n, z) = gdc(187, 165) = 11$. Vì 1 <11<187 nên 11 là một thừa số nguyên tố của 187. Thực vậy, 187 = 11.17.

3.2.3. Phương pháp phân tích tổng quát

Các phương pháp phân tích đặc biệt ở trên không đủ nhanh để phân tích các *modulo* lớn được sử dụng trong các hệ mã RSA. Hiện nay, các phương pháp phân tích tổng quát như sàng toàn phương (Quadratic Sieve – QS) và sàng trường số tổng quát (General Number Field Sieve – GNFS) đã dần thay thế các phương pháp phân tích đặc biệt trên. Đây là các phương pháp phân tích tổng quát do sự hiệu quả của các

phương pháp này chỉ phụ thuộc vào kích thước của số cần phân tích chứ không phụ thuộc các tính chất đặc biệt của nó.

3.2.4. Phương pháp sàng toàn phương – QS (Quadratic Sieve)

Đây là phương pháp nhanh nhất được biết đến để phân tích các số nhỏ hơn 110 chữ số thập phân và được sử dụng rất rộng rãi. Phiên bản nhanh hơn của thuật toán này được gọi là the Multiple Polynomial Quadratic Sieve (MPQS).

- Giải thuật QS

Ví dụ giải thuật QS:

Cho biết n = 1829.

Input: cho biết n là một số nguyên lẻ không chính phương.

Output: giải thuật tìm thừa số f của n thỏa mãn 1<f<n

Begin

[1] $FB = \{-1, p_1, p_2, \dots, p_k \leq B\}$

[2] Tìm $a_1, a_2, \dots, a_k$ gần với $\sqrt{n}$ với $a_i = \lfloor \sqrt{n} \rfloor + i, i = 1,2,..,k$

Với mỗi $Q(a_i) = a_i^2 - n$

[3] Tìm $x^2 \equiv y^2 \ (mod\ n)$ sử dụng đại số tuyến tính để tìm ra tập con U của $Q(a_i) = a_i^2 - n$. Khi đó,$y^2 \equiv \prod a_i^2 - n$. Cho x là một giá trị của a_i, khi đó

$$x^2 \equiv \left(\prod_{i \in U} a_i\right)^2$$
$$\equiv \prod_{i \in U} (a_i^2 - n)$$
$$\equiv \prod_{i \in U} Q(a_i)$$
$$\equiv \left(\prod_{i \in U} p_j^{\alpha_{i,j}}\right)^2$$
$$\equiv y^2 \ (mod\ n)$$

[4] $(f, g) = gcd(x \pm y, n)$

[5] $if\ 1 < f, g < n, print\ (f, g)$ và $goto$ [6]$else\ goto$ [3] $tìm\ giá\ trị\ x$ và $y\ mới.$

[6] $exit$

[7] End

- Ta có: FB = {-1,2,5, 7, 11} – do 3< 11 là một số nguyên tố, nhưng do n không phải là một thặng dư bình phương nên loại 3 ra khỏi FB.
- Chọn $a_i \sim \lfloor\sqrt{1829}\rfloor = 29$. Do đó, $a_i = 27, 28, 29, \ldots$. Thực hiện tính toán $Q(a_i) = a_i^2 - n$, nhận các vecto theo hệ số 2 như dưới đây:

		-1	2	5	7	11
1)	$Q(27) = 27^2 - N = -1100 = -2^2 \cdot 5^2 \cdot 11 \longleftrightarrow$	1,	0,	0,	0,	1)
2)	$Q(38) = 38^2 - N = -385 = -5 \cdot 7 \cdot 11 \longleftrightarrow$	1,	0,	1,	1,	1)
3)	$Q(39) = 39^2 - N = -308 = -2^2 \cdot 7 \cdot 11 \longleftrightarrow$	1,	0,	0,	1,	1)
4)	$Q(43) = 43^2 - N = 20 = 2^2 \cdot 5 \longleftrightarrow$	0,	1,	1,	0,	0)
5)	$Q(45) = 45^2 - N = 196 = 2^2 \cdot 7^2 \longleftrightarrow$	0,	0,	0,	0,	0)
6)	$Q(52) = 52^2 - N = 875 = 5^3 \cdot 7 \longleftrightarrow$	0,	0,	1 ,	1,	0)
7)	$Q(53) = 53^2 - N = 980 = 2^2 \cdot 5 \cdot 7^2 \longleftrightarrow$	0,	0,	1,	0,	0)

- Từ bảng vecto trên ta tính được:
  - Tổng các vecto 1), 2) và 6) (theo hệ số 2) là số 0. Thật vậy:

$$\begin{array}{cll} & (1,0,0,0,1) & \text{1st} \\ & (1,0,1,1,1) & \text{2nd} \\ \oplus & (0,0,1,1,0) & \text{6th} \\ \hline & (0,0,0,0,0,0) & \Longrightarrow \text{Successful} \end{array}$$

Từ đó tìm được cặp giá trị (x,y) thỏa mãn: $(27.38.52)^2 \equiv (2.5^3.7.11)^2$. Ta có:

$$\begin{aligned} x &= 27.38.52 \\ &= 53352 \\ &\equiv 311 \ (mod\ 1829) \\ y &= 2.5^3.7.11 \\ &= 19250 \\ &\equiv 960 \ (mod\ 1829) \end{aligned}$$

  - Tổng các vecto 2), 3), 7) cũng là một số chính phương

$$\begin{array}{cll} & (1,0,1,1,1) & \text{2nd} \\ & (1,0,0,1,1) & \text{3rd} \\ \oplus & (0,0,1,0,0) & \text{7st} \\ \hline & (0,0,0,0,0,0) & \Longrightarrow \text{Successful} \end{array}$$

Từ đó, $(38.39.53)^2 \equiv (2^2.5.7^2.11)^2$, giá trị x, y được tính như sau:

$$x = 38.39.53$$
$$= 78546$$
$$\equiv 1728\ (mod\ 1829)$$
$$y = 2^2.5.7^2.11$$
$$= 10780$$
$$\equiv 1635\ (mod\ 1829)$$

- Tính $(f,g) = gcd(x \pm y, n)$, ta có 2 cặp giá trị (x,y) lần lượt là (311,960) và (1728,1635). Khi đó (f,g) sẽ là cặp thừa số nguyên tố (p,q) của n.
  - Ta có: $(f,g) = gcd(311 \pm 960,1829) = (31,59)$. Vậy, 1829 = 31.59
  - Ta có: $(f,g) = gcd(1728 \pm 1635,1829)$=(59,31). Vậy, 1829=59.31

3.2.5. Phương pháp sành trường số tổng quát – GNFS (General Number Field Sieve)

Đây là thuật toán phân tích thành thừa số nhanh nhất được biết đến để phân tích các số lớn hơn 110 chữ số, chính là những số được dùng phổ biến trong RSA hiện nay. Nó không thực tế khi được đề xuất nhưng đã dần thay đổi qua hàng loạt các cải tiến trong những năm gần dây. Phiên bản đầu tiền được sử dụng để phân tích số Fermat thứ 9 là $2^{512} + 1$.

Vào tháng 3 năm 1991, RSA Data Security, Inc. đã thiết lập cuộc thi phân tích RSA (RSA Factoring Challenge). Cuộc thi bao gồm danh sách các số "khó"[16], mỗi số là tích của hai số nguyên tố có kích thước xấp xỉ nhau. Có 42 số trong cuộc thi, có độ dài từ 100 đến 500 chữ số, số này cách số kia 10 chữ số (có thêm một số 129 chữ số).

Hiện nay, RSA-100, RSA-110, RSA-120, và RSA-129 đã được phân tích và đều bằng QS. RSA-129 được phân tích vào ngày 2/4/1994, là số dài nhất được công bố sử dụng phương pháp QS cho đến khi NFS phân tích thành công RSA-130 vào ngày 10/4/1996. Tất cả các RSA cho đến bây giờ đều được phân tích bởi NFS. Các cải tiến gần đây trong NFS làm cho NFS hiệu quả hơn MPQS trong việc phân tích các số lớn hơn khoảng 115 chữ số, trong khi MPQS tốt hơn cho các số nguyên nhỏ.

Trong khi RSA-129 (129 chữ số thập phân) bị phân tích sử dụng một biến thể của MPQS, còn một biến thể của NFS đã được sử dụng gần đây để phân tích RSA-155.Như vậy, ước tính nếu NFS được sử dụng để phân tích RSA-129, nó sẽ chỉ cần một phần từ thời gian mà MPQS đã mất. Có thể nói, NFS đã vượt qua MPQS như là thuật toán phân tích được sử dụng rộng rãi nhất.

Vào ngày 9/5/2005, RSA-200 (200 chữ số thập phân tương ứng với 663bit) đã bị phân tích bằng phương pháp GNFS do F. Bahr, M. Boehm, J. Franke và T. Kleinjung thực hiện. Theo chuyên gia Arjen Lenstra của tổ chức EPFL Thụy Sĩ (Ecole Polytechnique Fédérale de Lausanne) thì khả năng phá khóa RSA-1024 sẽ chỉ đạt được trong 5 – 10 năm nữa, nhưng đã đến lúc tìm kiếm giải pháp bảo mật mạnh mẽ hơn.

Để tăng độ an toàn của hệ mã trước các phương pháp này, độ dài khóa (*modulo* n) được chọn ngày càng lớn hơn. Hiện nay, độ dài *modulo* n tối thiểu là 1024 bit.

3.3. TẤN CÔNG DỰA TRÊN SỐ MŨ CÔNG KHAI BÉ

Định lý 2 (Coppersmith)

Cho n là một số nguyên và $f \in Z[x]$ là một đa thức mà có độ đo là d. Đặt $x = n^{1/d-e}$ cho $e \geq 0$. Sau đó biết (n,f) Marvin có thể tìm tất cả số nguyên $|x_0| < x$ thỏa mãn $f(x_0) = 0 \bmod n$. Thời gian chạy phụ thuộc vào thời gian chạy thuật toán LLL với trên một lưới có khoảng cách là $O(\omega)$ với $\omega = \min(1/e, \log_2 n)$.

Định lý cung cấp một thuật toán có thể tìm kiếm hiệu quả tất cả gốc f của n ít hơn $x = n^{1/d}$. Với x nhỏ hơn, thời gian chạy thuật toán cũng giảm. Sức mạnh của thuật toán là có thể tìm được gốc của n trong thời gian đa thức. Định lý Coppersmith làm việc rất hiệu quả với một số nguyên tố.

Hastad's Broadcast Attack

Để đơn giản ta coi e_i là thành phần công khai bằng 3. Marvin tìm ra m rất đơn giản nếu $k \geq 3$. Thực vậy, Marvin có được c_1, c_2, c_3, thỏa mãn:

$c_1 = m^3 \bmod n_1$, $c2 = m^3 \bmod n2$, $c3 = m^3 \bmod n3$.

Do đó với e = 3, gửi các thông điệp giống nhau được gửi đến 3 người nhận là không an toàn. Giải pháp chống tấn công này chúng ta gắn các thông điệp trước khi mã hóa với đa thức ?

Định lý Hastad

Cho $n_{1,..},n_k$ là những số nguyên tố và tập $n_{min} = min_i(n_i)$ từng đôi một. Với $g_i \in Z_{ni}[x]$, k là đa thức có giá trị nhỏ nhất là d. Tồn tại $m < n_{min}$ thỏa mãn: $g_i(m) = m \bmod n_i$ với tất cả i = 1,...,k. Giả thiết rằng k > d, có thể tìm m khi cho $(n_i,g_i)^k_{i=1}$.

Định lý chỉ ra rằng một hệ thống đồng biến với các đa thức nguyên tố hỗn hợp có thể giải quyết hiệu quả, giả thiết rằng các hàm được cung cấp đầy đủ. Bằng cách cài đặt $g_i = f_i^{ei} – c_i \bmod n_i$, chúng ta thấy rằng Marvin có thể tìm được m từ bản mã được cho với số thành viên ít nhất là d, khi đó d là giá trị lớn nhất của $e_i deg(f_i)$ với i = 1,…,k.

Chúng ta lưu ý rằng để chống lại tấn công broadcast ở trên chúng ta sử dụng một cặp số ngẫu nhiên thay vì gắn cứng vào một giá trị.

Franklin-Reiter Related Message Attack.

<u>Hệ quả (FR):</u> *Giả sử rằng với e =3 và (n,e) là một khóa công khai của RSA. Cho $m_1 \neq m_2 \in Z^*_N$ thỏa mãn $m_1 = f(m_2) \bmod n$ trong đó $f = ax + b \in Z^*_n$ là đa thức tuyến tính với $b \neq 0$. Khi đó cho trước (n, e, c_1,c_2, f), Marvin có thể tìm được m_1, m_2 với thời gian là đa thức bậc hai logn.*

Để chứng minh hệ quả FR ta tính gcd của hai đa thức. Với e = 3 thì giá trị gcd phải là giá trị tuyến tính. Thật vậy, đa thức $x^3 – c_2$ phân tích thành p và q là phép phân tích tuyến tính và không thể rút gọn về nhân tố bậc hai (ta nhớ rằng gcd(e, Φ (n)) = 1 và vì thế $x^3 – c_2$ chỉ có giá trị gốc nằm trong Zn). Khi đó g_2 không thể chia cho g_1, gcd phải là một hàm tuyến tính. Với e = 3 hàm gcd luôn là tuyến tính. Tuy nhiên, đối với một vài m_1, m_2 và f, gcd có thể không phải là tuyến tính, trong trường hợp này việc tấn công là thất bại.

Thường nó chỉ áp dụng với khi số mũ công khai e được sử dụng với giá trị nhỏ. Với e lớn, công việc tính toán gcd là rất khó. Một câu hỏi thú vị (nếu không nói là

khó) đặt ra là liệu việc tấn công với một số e bất kỳ sẽ như thế nào. Khí đó việc tính toán gcd của g_1 và g_2 theo cách trên có trong thời gian đa thức đối với log e ?

3.4. TẤN CÔNG DỰA TRÊN SỐ MŨ RIÊNG BÉ

Trong thực tế, để giải mã nhanh đòi hỏi số d nhỏ và điều này để lộ lỗ hổng mà kẻ tấn công có thể thực hiện như sau. Trước hết ta nghiên cứu định lý Wiener

Định lý 1 (M. Wiener)

Cho n = pq với q < p< 2q. Giả sử $d < 1/3n^{1/4}$. Cho trước (n,e) với ed = 1 mod Φ (n), Marvin có thể tìm được d hiệu quả. Việc chứng minh định lý trên dựa trên xấp xỉ hóa phân số liên tục như sau:

Khi ed = 1 mod $\Phi(n)$, tồn tại một số k thỏa mãn ed - k$\Phi(n)$ = 1. Vì thế: $\left|\frac{e}{\Phi(n)}-\frac{k}{d}\right|=\frac{1}{d\Phi(n)}$ Do đó, $\frac{k}{d}$ là xấp xỉ của $\frac{e}{\Phi(n)}$. Mặc dù Marvin không biết Φ(n), anh ta có thể sử dụng n để xấp xỉ nó. Hơn nữa, từ $\Phi(n)$ = n- p- q +1 và p + q-1< $3\sqrt{n}$, chúng ta có:

| n - $\Phi(n)$ | < $3\sqrt{n}$.

Sử dụng n thay vào Φ (n), chúng ta có:

$$\left|\frac{e}{n}-\frac{k}{d}\right|=\left|\frac{ed-k\Phi(n)-kn+k\Phi(n)}{nd}\right|=\left|\frac{1-k(n-\Phi(n))}{nd}\right|\le\left|\frac{3k\sqrt{n}}{nd}\right|=\frac{3k}{d\sqrt{n}}$$

Bây giờ, kΦ (n) = ed – 1 < ed. Từ e < Φ (n), chúng ta thấy rằng k < d < $\frac{1}{3}n^{1/4}$.

Vì thế ta có: $\left|\frac{e}{n}-\frac{k}{d}\right|\le\frac{1}{dn^{1/4}}<\frac{1}{2d^2}$ Đây là hệ thức xấp xỉ cổ điển. Phân số $\frac{k}{d}$ với d < n là xấp xỉ của $\frac{e}{n}$ nên bị chặn tại $\log_2 n$.

Trong thực tế, tất cả các nhân tử thu được từ phân tích đều hội tụ tại kết triển khai mở rộng phân số $\frac{e}{n}$. Tất cả kết quả đó đều thu được từ việc tính toán logn hội tụ của việc tính toán phân số $\frac{e}{n}$. Một trong những kết quả đó sẽ là $\frac{k}{d}$.

Khi đó ed - k Φ (n) = 1, chúng ta có gcd(k,d) = 1, và vì thế $\frac{k}{d}$ là rút gọn phân số. Thuật toán tìm khóa riêng d là thuật toán có thời gian tuyến tính.

Độ lớn e

Thay vì rút gọn e trong Φ (n), ta sử dụng (n,e') cho khóa công khai thỏa mãn e' = e + t. Φ (n) trong đó số t lớn. Rõ ràng có thể sử dụng e' thay thế e để mã hóa thông điệp. Tuy nhiên, khi số e có giá trị lớn, theo chứng minh ở trên thì số k không thể nhỏ hơn. Một tính toán đơn giản chỉ ra rằng nếu $e' > n^{1.5}$ thì sẽ không có vấn đề gì xẩy ra mặc dù số d nhỏ và tấn công ở trên không thể thực hiện được. Nhưng điều bất tiện là số e lớn sẽ là tăng thời gian mã hóa.

Sử dụng CRT

Một cách tiếp cận khác là sử dụng định lý đồng dư trung hoa (Chinese Remainder Theorem - CRT). Ta chọn một số d sao cho cả $d_p = d \bmod (p - 1)$ và $d_p = d \bmod (q - 1)$ đều nhỏ 128 bits. Để giải mã nhanh bản c ta có thể tiến hành: Trước hết ta tính $m_p = c^{dp} \bmod p$ và $m_q = c^{dq} \bmod q$. Sau đó sử dụng CRT để tính giá trị $m \in Z_N$ thỏa mãn $m = m_p \bmod p$ và $m = m_q \bmod q$. Kết quả m phải thỏa mãn $m = c^d \bmod n$ là bắt buộc. Mặc dầu d_p và d_q là nhỏ song giá trị d mod Φ (N) có thể lớn, tùy thuộc vào Φ (N). Theo kết quả, sự tấn công của định lý 2 không được áp dụng. Chúng ta lưu ý rằng nếu (n,e) được biết thì kẻ địch có thể tấn công n trong thời gian O(min($\sqrt{dp}$, $\sqrt{dq}$)), vì thể d_p và d_q không thể quá nhỏ.

Mặt khác ta không thể biết được điều gì xẩy ra đối với vấn đề an ninh này. Chúng ta chỉ biết thông qua tấn công hữu hiệu của Wiener. Định lý 1 gần đây đã được cải thiện bởi Boneh và Durfee , họ chỉ ra rằng số với $d < n^{0.292}$, kẻ tấn công có thể tính được d từ (n,e). Kết quả này chỉ ra ranh giới của Wiener là không rõ ràng. Nó có vẻ như là $d< n^{0.5}$, đây là một bài toán mở.

<u>Bài toán mở :</u> *Cho n = pq và $d < n^{0.5}$. Nếu Marvin biết (n,e) với ed = 1 mod* Φ (n) *và e <* Φ (n), *anh ta có thể tìm được d không ?*

3.5. CÀI ĐẶT MỘT SỐ THUẬT TOÁN

3.5.1. Cơ sở toán học.

Bổ đề 1

Giả sử rằng n=p.q với p ≠ q là hai số nguyên tố lẻ. Ngoài ra ta giả thiết rằng p<q. Khi đó:

i/ $p < \sqrt{n} < q$

ii/ Số p gần $\sqrt{n}$ *hơn số q. Tức là giả sử* $\alpha, \beta > 0$ *sao cho:* $p + \alpha = \sqrt{n} = q - \beta$, *khi đó* $\alpha < \beta$

Chứng minh:

i/ Hiển nhiên đúng

ii/ Từ kết quả ở i/ ta suy ra có tồn tại hai số dương α, β sao cho: $p = \sqrt{n} - \alpha$ và $q = \sqrt{n} + \beta$

Từ đó: $n = p.q = (\sqrt{n} - \alpha)(\sqrt{n} + \beta) = n - \alpha\sqrt{n} + \beta\sqrt{n} - \alpha\beta$

Hay: $(\beta - \alpha)\sqrt{n} - \alpha\beta = 0$ (3.1)

$\Rightarrow (\beta - \alpha)\sqrt{n} = \alpha\beta = 0$ hay $\sqrt{n} = \dfrac{\alpha\beta}{\beta - \alpha}$

Do α, β > 0 và $\sqrt{n} > 0$ nên β-α>0 ⇒ β≠α vì nếu β=α thì từ (3.1) ta suy ra αβ=0. Từ đó hoặc α=0 hoặc β=0. Nhưng nếu α=0 thì $p = \sqrt{n}$ vô lý, tương tự nếu β=0 thì $q = \sqrt{n}$ (cũng vô lý). Mệnh đề được chứng minh.

Từ bổ đề 1 ta suy ra rằng giữa hai nhân tử nguyên tố của số n thì nhân tử bé hơn p gần $\sqrt{n}$ hơn.

Bổ đề 2

Cho trước n=p.q với p<q là hai nhân tử nguyên tố.

Ta kí hiệu:

$\phi(n) = (p-1)(q-1)$ *(được gọi là hàm Euler)*

Khí đó:

$\phi(n) \le n - 2\sqrt{n} + 1$ (3.2)

Chứng minh:

Thật vậy, ta có: $\phi(n) = (p-1)(q-1) = n - (pq) + 1$ (3.3)

Do p, q >0 nên ta ứng dụng bất đẳng thức:

$\frac{a+b}{2} \geq \sqrt{ab}; \forall a, b > 0$ và dấu "=" đạt được khi và chỉ khi a=b, ta có:

$\frac{p+q}{2} \geq \sqrt{pq} = \sqrt{n}$

Hay: $p + q \geq 2\sqrt{n}$; từ đây ta có ngay:

$n - (p+q) + 1 \leq n - 2\sqrt{n} + 1$ (điều phải chứng minh)

Bổ đề 3.

Cho trước n=p.q (p<q) đối với mọi số nguyên tố lẻ p, q. Đặt:

$\phi(n) = n - (p+q) + 1$

Khi đó: $\lim_{n \to +\infty} \frac{\phi(n)}{n} = 1$

Để chứng minh bổ đề 3, ta dùng đến bổ đề sau:

Bổ đề 4

Giả sử n không phải là số chính phương (perfect square) và sao cho: $n-1 > \phi(n) > n - n^{2/3}$*. Khi đó n=p.q với p, q là 2 số nguyên tố phân biệt.*

Chứng minh: Thật vậy, n không phải là số nguyên tố vì nếu n là số nguyên tố thì theo định nghĩa hàm ϕ(.) ta có ϕ(n)=n-1. Điều này trái với giả thiết của bổ đề. Bây giờ giả sử n không phải là tích của 2 số nguyên tố phân biệt, thế thì n phải là tích của nhiều hơn 2 (từ 3 trở lên) số nguyên tố (không nhất thiết phải là phân biệt). Gọi p là số nguyên tố bé nhất trong chúng, thế thì $p \leq n^{1/3}$ (vì nếu $p > n^{1/3}$ thì vô lý vì p là số nguyên tố bé nhất trong nhân tử của n) và do đó theo định nghĩa ta có:

$$\phi(n) = n \prod_{p_i / n} (1 - p_i) \leq n\left(1 - \frac{1}{p}\right) \leq n\left(1 - n^{\frac{1}{3}}\right) = n - n^{\frac{2}{3}}$$

Điều này mâu thuẫn với giả thiết của bổ đề. Vậy n=p.q, với p, q là 2 số nguyên tố phân biệt (điều phải chứng minh).

Chú ý: mệnh đề ngược lại cũng đúng, tức nếu n=p.q với p, q là 2 số nguyên tố phân biệt thì:

$$n-1 > \phi(n) > n-n^{2/3}$$

Thật vậy nếu trái lại thì tức hoặc $\phi(n) \geq n-1$ hoặc $\phi(n) \leq n-n^{2/3}$

Nhưng nếu $\phi(n) = n-1$ thì n là số nguyên tố. Điều này mâu thuẫn với giả thiết rằng n là tích của 2 số nguyên tố phân biệt. Còn nếu $\phi(n) \leq n-n^{2/3}$ thì n phải là tích của nhiều hơn 2 số nguyên tố (không nhất thiết phải là tích của các (nhiều hơn 2) số nguyên tố phân biệt).

Vậy từ bổ đề 4, ta suy ra số nguyên dương n là tích của 2 số nguyên tố phân biệt $\Leftrightarrow n-1 > \phi(n) > n-n^{2/3}$.

Bây giờ trên cơ sở các bổ đề 2,4 chúng ta chứng minh bổ đề 3 như sau:

Theo bổ đề 4, ta có:

$$\frac{\phi(n)}{n} > \frac{n-n^{\frac{2}{3}}}{n} \tag{3.4}$$

Mặt khác theo bổ đề 2 ta có:

$$\frac{\phi(n)}{n} \leq \frac{n-2\sqrt{n}+1}{n} \tag{3.5}$$

Từ (3.4) và (3.5) ta có bất đẳng thức kép sau đây:

$$\frac{n-n^{\frac{2}{3}}}{n} < \frac{\phi(n)}{n} \leq \frac{n-2\sqrt{n}+1}{n} \tag{3.6}$$

Bây giờ lấy giới hạn:

$\lim\limits_{n\to+\infty} \frac{n-n^{\frac{2}{3}}}{n}$ và $\lim\limits_{n\to+\infty} \frac{n-2\sqrt{n}+1}{n}$ ta có 1 là giới hạn chung. Vậy áp dụng định lý "bị kẹp" trong giải tích toán học ta suy ra $\lim\limits_{n\to+\infty} \frac{\phi(n)}{n} = 1$ (bổ đề 3 được chứng minh).

Từ bổ đề 3 nếu chuyển qua ngôn ngữ giới hạn là:

$\forall \varepsilon > 0$, $\exists N_0$ sao cho: $\forall n \geq N_0$, ta có:

$$\left| \frac{\phi(n)}{n} - 1 \right| < \varepsilon \tag{3.7}$$

từ (3.7) ta có thể viết:

$$(1-\varepsilon)n < \phi(n) < (1+\varepsilon)n; \forall n \geq N_0 \quad (3.8)$$

Số N_0 phụ thuộc vào ε: $N_0=N_0(\varepsilon)$

Bây giờ ta hãy tìm mối quan hệ của p, q với ϕ(n). Trong đó n=p.q; 2<p<q (là 2 số nguyên tố lẻ).

Theo định nghĩa ta có:

$$\phi(n) = (p-1)(q-1) = n-(p+q)+1 = n-\left(p+\frac{n}{p}\right)+1$$

$$= \frac{np-(p^2+n)+p}{p}$$

Vậy:

$$p\phi(n) = np-p^2-n+p \Leftrightarrow p^2-np+p\phi(n) = p+n = 0$$
$$\Leftrightarrow p^2-p(n+1-\phi(n))+n = 0 \quad (3.9)$$

Do n là một số lẻ nên n+1 là một số chẵn, đồng thời ϕ(n) là một số chia hết cho 4(vì ϕ(n)=(p-1)(q-1) trong đó p, q là lẻ). Vậy phương trình đối với ẩn số p cho ở (3.9) có thể được viết như sau:

$$p^2-2(n'-\phi'(n))p+n = 0 \quad (3.10)$$

Trong đó:

$$n' = \frac{n+1}{2} \;;\; \phi'(n) = \frac{\phi(n)}{2}$$

Như vậy $\phi'(n)$ chia hết cho 2.

Ta giải phương trình (3.10) với ẩn là p

Trước hết ta tính: $\Delta = \left(n'-\phi'(n)\right)^2 - n$ (3.11)

Ta có: $p_1 = \left(n'-\phi'(n)\right)-\sqrt{\Delta}$ và $p_2 = \left(n'-\phi'(n)\right)+\sqrt{\Delta}$ (3.12)

Ta thấy rằng p_1, p_2 là các nghiệm nguyên dương.

Do đó Δ phải là số chính phương.

Từ 4 bổ đề vừa được trình bày cùng các lý luận trên đây em giới thiệu hai thuật toán sau đây:

3.5.2. Xây dựng thuật toán demo

Thuật toán 1: Thuật toán 1 sẽ tiến hành tìm một nhân tử nguyên tố (cụ thể là tìm p) trong khoảng: $\left(2\left[0{,}01\sqrt{n}\right]+1, \sqrt{n}\right)$.

Thuật toán này là sự cải tiến của thuật toán *3.1.4.1/.* được nêu trong chương III). Thuật toán *3.1.4.1/.* tìm nhân tử số nguyên tố lớn nhất của n mà bé hơn hay bằng $\sqrt{n}$. Thuật toán *3.1.4.1/.* tìm số nguyên tố đó trong khoảng $\left(2\left[n^{\frac{1}{3}}\right]+1, \sqrt{n}\right)$, còn thuật toán 1 sẽ tiến hành tìm một nhân tử nguyên tố (cụ thể là tìm p) trong khoảng: $\left(2\left[0{,}01\sqrt{n}\right]+1, \sqrt{n}\right)$.

Nếu số n không lớn lắm thì thuật toán *3.1.4.1/.* và thuật toán 1 không có gì khác mấy về độ phức tạp của thuật toán (tức là thời gian để thuật toán kết thúc) nhưng nếu n lớn, chẳng hạn $n=10^{24}$ thì thuật toán 1 nhanh hơn nhiều. Thật vậy, đối với thuật toán *3.1.4.1/.*, khoảng phải tìm số p là: $(2.10^8+1;\ 10^{12})$ còn với thuật toán 1, khoảng đó là: $(2.10^{10}+1;\ 10^{12})$.

Số n càng lớn thuật toán 1 càng có ý nghĩa hơn so với thuật toán *3.1.4.1/.*. Đó là chưa tính đến việc lấy căn bậc 3 của một số lớn phức tạp hơn nhiều so với căn bậc 2.

Một câu hỏi được đặt ra ở đây là vậy liệu ta chọn cận dưới của khoảng là $\left(2\left[0{,}01\sqrt{n}\right]+1\right)$ có thể vét hết nhân tử nguyên tố $p < \sqrt{n}$ của số n hay không?

Câu trả lời là như sau: Nếu theo qui định của nhà thiết kế RSA thì độ dài p và độ dài q phải bằng nhau và bằng một nửa độ dài của n, hay cùng lắm là xấp xỉ nhau nhất có thể đựợc thì gần như chắc chắn rằng một nhân tử nguyên tố p của n (giả sử p<q) sẽ nằm trong khoảng $\left(2\left[0{,}01\sqrt{n}\right]+1; \left[\sqrt{n}\right]\right)$ ít nhất là với xác suất rất lớn. Thật vậy, từ kết quả của bổ đề 1 vừa được giới thiệu ở trên, một mặt số p chắc chắn bé hơn hay bằng $[\sqrt{n}]$; mặt khác theo giả thiết về tiêu chuẩn RSA thì số q cùng lắm là gấp 100 lần số p (không thể hơn). Trong lúc đó, theo bổ đề 1 số q cách xa $[\sqrt{n}]$ hơn số p. Do đó số p nhỏ hơn $[\sqrt{n}]$ không quá 50 lần, tức là:

$$p \geq \frac{\sqrt{n}}{50} = 2\frac{\sqrt{n}}{100}$$

Do p là số lẻ nên:

$$p \geq 2\left[\frac{\sqrt{n}}{100}\right] + 1 = 2\left[0{,}01\sqrt{n}\right] + 1 \text{ (điều phải chứng minh)}$$

Thuật toán 1 cải tiến đáng kể thời gian tìm kiếm số nguyên tố p so với thuật t

Ví dụ:

Cho p=11, q=997. Ta có:

N=p.q=11.997=10967. Từ đó:

$2\left[0{,}01\sqrt{n}\right] + 1 = 2.1 + 1 = 3$ và [$\sqrt{n}$]=104. Rõ ràng là: 3≤p≤104

Chú ý: trong trường hợp này số q cũng chỉ xấp xỉ 90 lần so với số nguyên tố p.

Thuật toán 2:

+ Cơ sở lý thuyết dẫn đến thuật toán:

Đặt lại bài toán: Cho trước n=p.q; với 2<p<q; sao cho độ dài l(p)=l(q)=l(n)/2

p, q là 2 số nguyên tố.

Cho số mũ công khai b sao cho: (b, ϕ(n))=1. Trong đó: ϕ(n)=(p-1)(q-1). Hãy tìm hoặc là số d: b.d ≡ 1 mod ϕ(n) hoặc là số p (từ đó suy ra số q) hoặc là hàm ϕ(n).

Chúng ta biết rằng để giải được bản mã được mã bởi hệ mật RSA, chúng ta chỉ cần biết một trong ba tham số mà thôi.

Thật vậy, giả sử ta biết được số p và từ đó suy ra số $q = \frac{n}{p}$. Do đó ta tìm được hàm số ϕ(n)=(p-1)(q-1). Trên cơ sở ϕ(n) chúng ta tìm được số mũ bí mật (số mũ giải mã) d nhờ thuật toán ơclit: b.d ≡ 1 mod ϕ(n)

Do b cho trước và (b, ϕ(n))=1 nên số nguyên dương d như vậy luôn luôn tồn tại. Từ đó suy ra rằng để xác định được số mũ bí mật d, chỉ cần biết được hàm Euler ϕ(n) là đủ. Hệ thức giữa số nguyên tố p và ϕ(n) được suy ra nhau từ (3.9) hoặc (3.10).

Chúng ta sẽ xác định ϕ(n) bằng cách xác định ϕ'(n) (vì $\phi'(n) = \frac{\phi(n)}{2}$ từ hệ thức (3.11).

Như vậy vấn đề đặt ra là hãy tìm ϕ'(n) trong khoảng nào đó sao cho:

$\Delta = \left(n^{'} - \phi^{'}(n)\right)^2 - n$ là số chính phương.

Trong đó: $n^{'} = \frac{n+1}{2}$; tức là $\sqrt{\Delta}$ là một số nguyên dương.

Thuật toán từ đó chia làm 2 phần: phần off-line và phần on-line.

Off-line: Xác định khoảng mà ϕ'(n) nằm trong đó với xác suất lớn (n càng lớn xác suất càng lớn). Ta xác định khoảng (ta ký hiệu là (a_n, b_n) như sau:

1) Lập tỷ số $\frac{n}{n-2\left[\sqrt{n}\right]+1} = 1+\delta(n)$ và đặt $n^{'} = \frac{n+1}{2}$

2) Chọn tham số a $\in$ (1,7) (Thường chọn a$\in$(1,1 ; 1,3; 1,4; 1,5)

3) Đặt ε = a.δ

4) từ bổ đề 2 và bổ đề 3 ta có:

$$\left[\frac{1-\varepsilon}{2}n\right] \le \phi^{'}(n) \le \frac{n-2[n]+1}{2} \qquad (3.13)$$

Đặt: $a_n = \left[\frac{1-\varepsilon}{2}n\right]$; $b_n = \frac{n-2\left[\sqrt{n}\right]+1}{2}$, ta có khoảng cần tìm.

Lưu ý: Trong thực hành ta nên chọn a thuộc khoảng [1,2 ;1,5] là đủ. Tuy vậy cũng có trường hợp phải chọn a=6 (rất ít).

+on-line: Giả sử n=p.q là một số nguyên dương (lẻ) cho trước, chọn a$\in$[1,2 ; 1,5] (thường chọn a=1,4).

+ Bước 1: Xác định khoảng (a_n; b_n) bởi công thức cho ở (3.13)

+ Bước 2:

Đặt : $\Delta_i = (n^{'} - \phi_i^{'}(n))^2 - n = \delta_i^2 - n; i = 1,2,....$

với: $\phi_1^{'}(n) = b_n$ (hoặc a_n) nếu bn là số chẵn và $\phi_1^{'}(n) = b_n - 1$ nếu b_n lẻ.

(Vì $\phi_1^{'}(n)$ luôn luôn là số chẵn).

Nếu chọn $\phi_1^{'}(n) = a_n$ thì cần lưu ý:

$$\phi_1^{'}(n) = \begin{cases} a_n & \text{- Nếu } a_n \text{ là số chẵn} \\ a_n + 1 & \text{- Nếu } a_n \text{ là số lẻ} \end{cases}$$

Tổng quát:

$\Delta_{i+1} = (\delta_i + 2)^2 - n$ (nếu chọn $\phi_1'(n) = b_n$ (hoặc b_n-1))

$\Delta_{i+1} = (\delta_i - 2)^2 - n$ (nếu chọn $\phi_1'(n) = a_n$

+ Bước 3:

- Nếu Δ_{i+1} là số chính phương ($\sqrt{\Delta_{i+1}}$ là số nguyên) thì thuật toán dừng và

$\phi'(n) = n' - (\delta_i + 2)$; $\phi'(n) = \phi_{i+1}' = n' - (\delta_i + 2)$

- Ngược lại: nếu $\delta_i = n' - a_n$ (hoặc a_n+1) thì thuật toán dừng và không có số chính phương trong khoảng (a_n, b_n) và do đó không tìm được số ϕ'(n).

hoặc nếu $\delta_i < n' - a_n$ thì chuyển đến bước 2

Các ví dụ:

+ Ví dụ 1: cho n=18721

Ta có: $n' = \frac{n+1}{2} = 9361$

cận trên b_n: $b_n = \frac{n - 2\lfloor\sqrt{n}\rfloor + 1}{2} = \frac{18450}{2} = 9225$

cận dưới:

- Đặt: $\frac{n}{n - 2\lfloor\sqrt{n}\rfloor + 1} = \frac{18721}{18450} \approx 1 + 0{,}01469 = 1 + \delta$

- Chọn a=1,4. Ta có ε=a.δ=1,4.0,01469=0,020566≈0,02.

- Vậy cận dưới $a_n = \left\lfloor \frac{1-\varepsilon}{2} n \right\rfloor = 9173$

Lúc đó ta cần tìm ϕ'(n) trong khoảng (9173, 9225) vì 9173, 9225 đều lẻ nên ta chỉ cần xét trong khoảng (9174, 9224).

+ On-line:

$\Delta_1 = (n' - \phi_1'(n))^2 - n = (9361 - 9224)^2 - 18721 = 137^2 - 18721 = 48$ (không chính phương)

$\Delta_2 = 139^2 - 18721 = 600$ (không chính phương)

$\Delta_3 = 141^2 - 18721 = 1160$ (không chính phương)

$\Delta_4 = 143^2 - 18721 = 1728$ (không chính phương)

$\Delta_5 = 145^2 - 18721 = 2304 = 48^2$ (Do đó Δ_5 là số chính phương).

Vậy: $\phi'(n) = n' - 145 = 9361 - 145 = 9216$

Do đó hàm ϕ-Euler: ϕ(n)=2ϕ'(n)=18432.

Kiểm tra: bằng thuật toán 1 ta tìm được p=97, q=193, tức là n=18721=97.193.

Do vậy ϕ(n)=(p-1)(q-1)=96.192=18432. Vậy kết quả trên là chính xác.

Ví dụ 2: Cho trước n=17018759. Hãy xác định ϕ(n) và từ đó xác định các nhân tử p, q của n.

Giải :

Bước 1:(off-line)

$n' = \frac{n+1}{2} = \frac{17018759 + 1}{2} = 8509380$

$\lfloor \sqrt{n} \rfloor = 4125$

+ Cận trên là: $b_n = \frac{n - 2\lfloor \sqrt{n} \rfloor + 1}{2} = \frac{17010510}{2} = 8505255$

+ Cận dưới: Xác định δ: $\frac{n}{n - 2\lfloor \sqrt{n} \rfloor + 1} = \frac{17018759}{17010510} = 1 + 0{,}000485 = 1 + \delta$

Vậy δ=0,000485.

Chọn a=1,7, ta có ε=a.δ=0,000679≈0,00068

Vậy cận dưới là: $a_n = \left\lceil \frac{1-\varepsilon}{2} n \right\rceil = 8503593$

Khoảng xác định của $\phi'(n)$ là [8509380;8505254]

Bây giờ ta chỉ việc tìm $\phi'(n)$ trong khoảng đã cho

Bước 2:(on-line)

$\Delta_1 = \left(n - \phi'(n)\right)^2 - n = (8509380 - 8503594)^2 - 17018759$

$= 5786^2 - 17018759 = 16459037$

(không chính phương)

$\Delta_2 = 5784^2 - 17018759 = 16435897$ (không chính phương)

$\Delta_3 = 5782^2 - 17018759 = 16412765$ (không chính phương)

….

$\Delta_{157} = 5472^2 - 17018759 = 12924025 = 3595^2$ Vậy Δ_{157} chính phương

Vậy $\phi'(n) = n' - 5472 = 8503908$ và $\phi(n) = 2\phi'(n) = 17007816$

Từ đó: $p = n' - \phi'(n) - 3595 = 1877$

$q = n' - \phi'(n) + 3595 = 9067$

Thử lại 17018759=1877.9067. Vậy thuật toán cho kết quả chính xác.

Trong trường hợp khi thực hiện tính toán trên mạng song song, ta chia bài toán thành từng phần và mỗi máy thực hiện một phần. Chẳng hạn trong bài toán của chúng ta, ta sẽ chia khoảng (a_n, b_n) thành các khoảng con sau đó mỗi máy có nhiệm vụ tìm ϕ'(n) trong khoảng được giao.

Ví dụ 3: Cho trước n=1690861. Hãy xác định ϕ(n) và từ đó xác định các nhân tử p, q của n.

Giải :

Bước 1:(off-line)

$n' = \frac{n+1}{2} = \frac{1690861 + 1}{2} = 845431$

$\lfloor \sqrt{n} \rfloor = 1300$

+ Cận trên là: $b_n = \frac{n - 2\lfloor \sqrt{n} \rfloor + 1}{2} = \frac{1690861 - 2600 + 1}{2} = 844131$

+ Cận dưới: Xác định δ: $\frac{n}{n - 2\lfloor \sqrt{n} \rfloor + 1} = \frac{1690861}{1688262} = 1 + 0{,}00154 = 1 + \delta$

Vậy δ=0,00154.

Chọn a=1,2, ta có ε=a.δ=1,2.0,00154=0,00185

Vậy cận dưới là: $a_n = \left\lfloor \frac{1-\varepsilon}{2} n \right\rfloor = 843866$

Khoảng xác định của $\phi^{'}(n)$ là [843866;844131]

Bây giờ ta chỉ việc tìm $\phi^{'}(n)$ trong khoảng đã cho

Bước 2:(on-line)

$\Delta_1 = \left(n - \phi^{'}(n)\right)^2 - n = (8509380 - 8503594)^2 - 17018759$
$= 5786^2 - 17018759 = 16459037$

(không chính phương)

$\Delta_2 = 5784^2 - 17018759 = 16435897$ (không chính phương)

$\Delta_3 = 5782^2 - 17018759 = 16412765$ (không chính phương)

$\Delta_{157} = 5472^2 - 17018759 = 12924025 = 3595^2$ Vậy Δ_{157} chính phương

Vậy $\phi^{'}(n) = n^{'} - 5472 = 8503908$ và $\phi(n) = 2\phi^{'}(n) = 17007816$

Từ đó: $p = n^{'} - \phi^{'}(n) - 3595 = 1877$

$q = n^{'} - \phi^{'}(n) + 3595 = 9067$

Thử lại 17018759=1877.9067. Vậy thuật toán cho kết quả chính xác.

Trong trường hợp khi thực hiện tính toán trên mạng song song, ta chia khoảng (a_n,b_n) thành 2 khoảng con bởi điểm chia là:

$\frac{a_n + b_n}{2} = c_n$. Ở đây $c_n = \frac{843866 + 844130}{2} = 843998$

Bây giờ khoảng: $(a_n; b_n) = (a_n; c_n) \cup (c_n; b_n)$. Cụ thể là:

$(843866; 843998) \cup (844000; 844130) = (843866; 844130)$

+Máy tính 1 thực hiện tìm $\phi^{'}(n)$ trên khoảng (c_n, b_n)

+Máy tính 2 thực hiện tìm $\phi^{'}(n)$ trên khoảng (a_n, c_n)

Giả sử đối với máy 1(tìm từ dưới lên):

$\Delta_1 = \left(n^{'} - \phi_1^{'}(n)\right)^2 - n = (845431 - 844000)^2 - 1690861 =$
$= 1431^2 - 1690861 = 356900$

(không chính phương)

$\Delta_2 = 1429^2 - 1690861 = 351180$ (không chính phương)

$\Delta_3 = 1427^2 - 1690861 = 345468$ (không chính phương)

$\Delta_4 = 1425^2 - 1690861 = 339764$ (không chính phương)

$\Delta_5 = 1423^2 - 1690861 = 334068$ (không chính phương)

$\Delta_6 = 1421^2 - 1690861 = 328380$ (không chính phương)

$\Delta_7 = 1419^2 - 1690861 = 322700$ (không chính phương)

$\Delta_8 = 1417^2 - 1690861 = 317028$ (không chính phương)

$\Delta_9 = 1415^2 - 1690861 = 311364 = 558^2$ (chính phương)

Vậy Δ_9 là số chính phương. Từ đó

$$\Rightarrow \phi'(n) = n' - 1415 = 845431 - 1415 = 844016$$

Từ đây suy ra:

$$\phi(n) = 2\phi'(n) = 2.844016 = 1688032$$

$\Rightarrow$ p=857 và q=1973

Rõ ràng là trong trường hợp này nếu không chia đôi khoảng thì việc tìm kiếm ϕ'(n) sẽ chậm hơn vì ta phải tìm hoặc từ trên xuống hoặc từ dưới lên, nên để đạt được kết quả, số bước tìm sẽ nhiều hơn 9 bước. Cụ thể là nếu tìm từ trên xuống chúng ta phải thực hiện 57 bước, còn nếu tìm từ dưới lên thì phải mất 75 bước. Nếu chia cho 4 máy thực hiện trên 4 khoảng, chúng ta sẽ cho kết quả nhanh hơn nữa.

Nhận xét, đánh giá:

Theo đánh giá của người dùng và theo lý thuyết thì độ dài tham số modul n càng lớn càng đảm bảo an toàn. Nhưng qua thuật toán thì kết luận đó chưa chắc đã đúng vì n càng lớn thì hàm Euler Ø (n) càng gần với số n và do đó khả năng vét cạn để tìm được Ø (n) là rất khả thi và từ đó xác định được số mũ bí mật. Đó chính là đóng góp của đề tài luận văn.

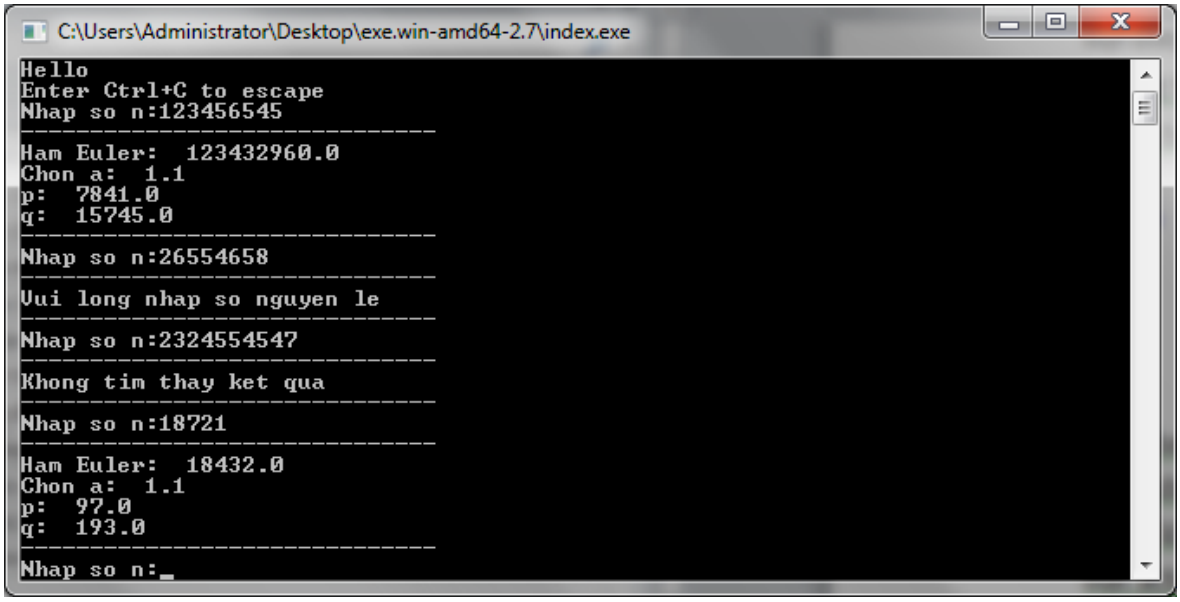
3.5.3. Chương trình thử nghiệm

Môi trường cài đặt

Hệ thống thử nghiệm được xây dựng trên môi trường lập trình Python, sử dụng ngôn ngữ lập trình Python.

Thực hiện trên Windown 7.

Giao diện chương trình


C:\Users\Administrator\Desktop\exe.win-amd64-2.7\index.exe
Hello
Enter Ctrl+C to escape
Nhap so n:123456545
Ham Euler: 123432960.0
Chon a: 1.1
p: 7841.0
q: 15745.0
Nhap so n:26554658
Vui long nhap so nguyen le
Nhap so n:2324554547
Khong tim thay ket qua
Nhap so n:18721
Ham Euler: 18432.0
Chon a: 1.1
p: 97.0
q: 193.0
Nhap so n:

KẾT LUẬN

Hơn hai thập niên nghiên cứu tìm hiểu nghịch đảo của RSA để tìm sự tấn công hiệu quả nhưng không có một tấn công hiệu quả nào được tìm ra. Những sự tấn công được khám phá cho đến nay chủ yếu minh họa các cạm bẫy phải tránh khi cài đặt RSA. Lúc này có vẻ như sự cài đặt đúng cách có thể đảm bảo được an ninh trong thế giới số.

Chúng ta phân loại tấn công trên RSA thành 5 loại:

(1) Tấn công cơ bản khai thác sự sai sót của hệ thống

(2) Tấn công khóa riêng có số mũ thấp không đủ, khóa riêng có số mũ thấp không bao giờ được sử dụng.

(3) Tấn công khóa công khai có số mũ thấp

(4) Tấn công trong cài đặt

(5) Tấn công bằng cách nhân tử hóa.

Theo đánh giá của người dùng và theo lý thuyết thì độ dài tham số modul n càng lớn càng đảm bảo an toàn. Nhưng qua thuật toán thì kết luận đó chưa chắc đã đúng vì n càng lớn thì hàm Euler Ø (n) càng gần với số n và do đó khả năng vét cạn để tìm được Ø (n) là rất khả thi và từ đó xác định được số mũ bí mật. Đó chính là đóng góp của đề tài luận văn. Mặc dù đã có nhiều cố gắng nhằm hoàn thành luận văn có chất lượng nhất có thể được, song do hạn chế về mặt thời gian và trình độ toán học, trong luận văn của em còn một số vấn đề chưa giải quyết được. Đó là so sánh độ phức tạp thuật toán mới được giới thiệu với độ phức tạp của các thuật toán khác. Hy vọng trong thời gian tới, với sự góp ý, giúp đỡ của các thầy cô em sẽ hoàn thiện các vấn đề đặt ra trong phương pháp này.

TÀI LIỆU THAM KHẢO

Tài liệu tiếng việt:

[1] Đặng Văn Cương - Vấn đề an toàn của hệ mật mã khoá công khai - Luận văn thạc sĩ, Khoa công nghệ thông tin - Đại học công nghệ 2003

[2] Nguyễn Thị Miền – Thanh toán từ xa – Luận văn đại học, Khoa công nghệ thông tin - Đại học công nghệ 2008

[3] Nguyễn Minh Hải - Đấu thầu từ xa - Luận văn đại học, Khoa công nghệ thông tin - Đại học công nghệ 2008

[4] Đặn Thị Lan Hương - Vấn đề an toàn thông tin trong thương mại điện tử - Luận văn đại học, Khoa công nghệ thông tin - Đại học công nghệ 2008

[5] Phan Đình Diệu – Lý thuyết mật mã và an toàn thông tin, Đại học quốc gia Hà Nội 2002

[6] Trịnh Nhật Tiến – Giáo trình an toàn dữ liệu – Khoa công nghệ thông tin, Đại học quốc gia Hà Nội

Tài liệu tiếng anh:

[7] D.Bleichenbacher. Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS #1

[8] D.Boneh, R.Demillo, and R.Lipton. On the importance of checking cryptographic protocols for faults.

[9] D.Boneh and G.Durfee. New results on cryptanalysis of low private exponent RSA. Preprint, 1998

[10] Mark Stamp Richard M.Low: “Applied Cryptanalysis”, A John Wiley & Sons INC publication, San Jose state University, San Jose CA 2007

[11] M. Wiener. Cryptanalysis of short RSA secret exponents. IEEE Transactions on Information Theory, 1990

[12] Neal Koblitz: “ A course in Number theory and Cryptography” New York, Berlin Heidelberg, London, Paris, Tokyo, 1987

[13] J. Hastad. Solving simultaneous modular equation of low degree. SIAM J. of Computing, 1988

[14] http://www.RSA.com

[15] http://www.RSAsercurity.com

[16] S. Goldwasser. The search for provably secure cryptosystems. In Cryptology and computational number theory, volume 42 of Proceeding of the 42nd Symposium in Applied Mathematics. American Mathematical Society, 1990