Mật Mã Hóa Khóa Công Khai

Sưu tầm và tổng hợp: Nguyễn Thành Nam, NCS, ĐH Purdue, Indiana, USA

Cao Minh Quang, THPT chuyên Nguyễn Bỉnh Khiêm, Vĩnh Long, VN

Mã Caesar

Phép đồng dư có nhiều ứng dụng trong toán học rời rạc cũng như trong tin học. Một trong những ứng dụng của phép đồng dư là để tạo ra các thư tín bí mật, một lãnh vực của ngành mật mã học. Từ xa xưa, Julius Caesar đã biết ứng dụng phép đồng dư để mã hóa thư tín.

Phương pháp mã hóa của Caesar

Quá trình mã hóa của Caesar được thực hiện theo các bước sau:

Bước 1. Cho tương ứng mỗi chữ cái trong bảng mẫu tự tiếng Anh (gồm 26 chữ cái) với một số nguyên từ 0 đến 25, thứ tự của chữ cái trong mẫu tự sẽ là số tự nhiên tương ứng.

Bước 2. Tiếp theo, sử dụng hàm f để biểu diễn mỗi số nguyên x thuộc tập hợp $\{0;1;2;...;25\}$ tương ứng với giá trị $f(x)$ cũng thuộc tập này, thỏa mãn $f(x) \equiv (x+3) (\mathrm{mod}\, 26)$.

Bước 3. Chuyển tương ứng số $f(x)$ thành kí tự giống như bước 1.

Với cách làm này, Caesar hoàn toàn có thể mã hóa tất cả các bức thư của mình, đồng thời cũng giải mã được các bức thư.

Ta hãy cùng xem xét một số ví dụ sau.

Ví dụ 1. Dùng mật mã Caesar, chuyển bức thư " MEET YOU IN THE SCHOOL" thành bức thư bí mật.

Lời giải. Xét hai tập hợp $X_1 = \{A;B;C;D;...,X;Y;Z\}$ và $X_2 = \{0;1;2;...;24;25\}$. Theo cách mã hóa Caesar, sẽ có tương ứng 1 – 1 mỗi chữ cái của tập hợp X_1 với một số thuộc tập hợp X_2. Thực hiện theo bước 1, ta chuyển bức thư gốc thành dãy số sau

12 4 4 19 – 24 14 20 – 8 13 – 19 7 4 – 18 2 7 14 14 11

Bây giờ, ta thay dãy số trên bằng dãy số tương ứng theo bước 2

15 7 7 22 – 1 17 23 – 11 16 – 22 10 7 – 21 5 10 17 17 14

Tiếp tục thực hiện theo bước 3, ta được bức thư đã được mã hóa là "…"

Để phục hồi bức thư gốc đã được mã hóa theo mật mã của Caesar, ta cần dùng hàm ngược f^{-1} của f. Nói cách khác, để tìm lại các bức thư gốc từ bức thư được mã hóa, mỗi chữ cái được tương ứng với chữ cái cách nó 2 vị trí về phía đầu bảng mẫu tự, riêng ba chữ cái đầu bảng A, B, C thì tương ứng thứ tự vơi X, Y, Z .

Ví dụ 2. Dùng mật mã của Caesar, hãy giải mã các bức thư sau …

Sau đây ta sẽ cùng tìm hiểu mã Caesar áp dụng trên ngôn ngữ tiếng Việt. Về nguyên lý thực hiện cũng tương tự như áp dụng cho tiếng Anh. Trước hết ta lập bảng tương ứng các chữ cái với các số như sau:

a	ă	â	b	c	d	đ	e	ê	g	h	i	k	l	m
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
n	o	ô	ư	p	q	r	s	t	u	ư	v	x	y	
16	17	18	19	20	21	22	23	24	25	26	27	28	29	

Dĩ nhiên ta có thể thêm các số để chỉ các dấu, nhưng để đơn giản ta tạm thời viết các văn bản không dấu.

Sau đây ta xét ví dụ đơn giản sau.

Ví dụ 3. Dùng mật mã Caesar, chuyển bức thư " LY THUYÊT MÂT MA KHÔNG CO GI KHO" thành bức thư bí mật.

Lời giải. Trước hết, để nâng cao tính bảo mật, ta tách bức thư thành từng nhóm 5 chữ cái, nhằm tránh việc một số từ của bức thư dễ bị phát hiện căn cứ vào số chữ cái. Như vậy, bức thư cần mã hóa là

LYTHU YÊTMÂ TMAKH ÔNGCO GIKHO

Nhờ bảng tương ứng giữa chữ và số, ta chuyển bức mã thành dạng chữ số

14 29 24 11 25 29 8 24 15 1 24 15 1 13 11 18 16 10 5 18 10 12 13 11 18

Bức thư bí mật là "…"

Trong hệ mã Caeser, số 3 là khóa của mã, vì nó được dùng để mã hóa và giải mã.

Hơn nữa, ta cũng có thể lập một hệ mật mã mới bằng khóa khác số 3 bằng một số tùy ý từ 0 đến 25 (với ngôn ngữ tiếng Anh) hoặc 1 đến 29 (với ngôn ngữ tiếng Việt).

Qua các ví dụ trên, ta nhận thấy rằng lý thuyết đồng dư trong số học được ứng dụng từ rất lâu và cũng rất thực tế. Hơn nữa, ta có thấy được, nếu thay giá trị của hàm số $f(x)$ ở trên bởi các hàm số khác, thì với mỗi cách chọn hàm $f(x)$ thích hợp, ta sẽ có một cách mã hóa các bức thư. Ta có thể chọn hàm số $f(x)$ như sau:

$$f(x)\equiv(x+2)(\mathrm{mod}\,26); f(x)\equiv(x-5)(\mathrm{mod}\,26); f(x)\equiv(5x+1)(\mathrm{mod}\,26); f(x)\equiv(3x-10)(\mathrm{mod}\,26), ..$$

Một cách tổng quát, ta có định lý sau:

"Với mọi cặp số nguyên $(a;b)$ thỏa điều kiện $(a,26)=1$ thì hàm số đồng dư $f(x)\equiv(ax+b)(\mathrm{mod}\,26)$ cho ta một cách mã hóa và giải mã được tất cả các bức thư theo cách tương tự như Caesar đã làm".

Hệ mã theo công thức $f(x)\equiv(ax+b)(\mathrm{mod}\,26)$ ($f(x)\equiv(ax+b)(\mathrm{mod}\,29)$) được gọi là mã **biến đổi afin.**

Một điều dễ nhận thấy là, khi bắt được một văn bản mật, người ta có thể căn cứ vào tần suất xuất hiện của các chữ cái để đoán ra khóa của mã. Chẳng hạn, nếu chữ *a* xuất hiện nhiều nhất trong các văn bản thì chữ cái nào nào có mặt nhiều nhất trong văn bản mật có thể là chữ *a*, từ đó đoán ra khóa mã. Hơn nữa, chỉ có 26 (hoặc 29) cách khác nhau để chọn khóa cho loại mã Caesar nên người ta cũng dễ dàng tìm ra khóa của mã, đặc biệt là khi áp dụng máy tính. Đối với mã biến đổi afin, chỉ cần dựa vào tần suất xuất hiện để tìm ra hai chữ cái tương ứng với 2 chữ nào đó trong văn bản mật, từ đó có thể xác định *a, b* bằng cách giải hệ hai phương trình đồng dư. Ngoài ra, việc giải những hệ mã biến đổi afin cũng quá dễ dàng đối với máy tính.

Như vây, với những yêu cầu về bảo mật cao hơn, ta phải dùng những hệ mã phức tạp hơn. Một trong những hệ mã có thể thỏa yêu cầu đó là **mã RSA**.

Mã RSA

Trong mật mã học, **RSA** là một thuật toán mật mã hóa khóa công khai. Đây là thuật toán đầu tiên phù hợp với việc tạo ra chữ ký điện tử đồng thời với việc mã hóa. Nó đánh dấu một sự tiến bộ vượt bậc của lĩnh vực mật mã học trong việc sử dụng khóa công cộng. RSA đang được sử dụng phổ biến trong thương mại điện tử và được cho là đảm bảo an toàn với điều kiện độ dài khóa đủ lớn.

Lịch sử

Thuật toán được Ron Rivest, Adi Shamir và Len Adleman mô tả lần đầu tiên vào năm 1977 tại Học viện Công nghệ Massachusetts (MIT). Tên của thuật toán lấy từ 3 chữ cái đầu của tên 3 tác giả.

Trước đó, vào năm 1973, Clifford Cocks, một nhà toán học người Anh làm việc tại GCHQ, đã mô tả một thuật toán tương tự. Với khả năng tính toán tại thời điểm đó thì thuật toán này không khả thi và chưa bao giờ được thực nghiệm. Tuy nhiên, phát minh này chỉ được công bố vào năm 1997 vì được xếp vào loại tuyệt mật.

Thuật toán RSA được MIT đăng ký bằng sáng chế tại Hoa Kỳ vào năm 1983 (Số đăng ký 4.405.829). Bằng sáng chế này hết hạn vào ngày 21 tháng 9 năm 2000. Tuy nhiên, do thuật toán đã được công bố trước khi có đăng ký bảo hộ nên sự bảo hộ hầu như không có giá trị bên ngoài Hoa Kỳ. Ngoài ra, nếu như công trình của Clifford Cocks đã được công bố trước đó thì bằng sáng chế RSA đã không thể được đăng ký.

Hoạt động

Mô tả sơ lược

Thuật toán RSA có hai khóa: khóa công khai (hay khóa công cộng) và khóa bí mật (hay khóa cá nhân). Mỗi khóa là những số cố định sử dụng trong quá trình mã hóa và giải mã. Khóa công khai được công bố rộng rãi cho mọi người và được dùng để mã hóa. Những thông tin được mã hóa bằng khóa công khai chỉ có thể được giải mã bằng khóa bí mật tương ứng. Nói cách khác, mọi người đều có thể mã hóa nhưng chỉ có người biết khóa cá nhân (bí mật) mới có thể giải mã được.

Ta có thể mô phỏng trực quan một hệ mật mã khoá công khai như sau : Bob muốn gửi cho Alice một thông tin mật mà Bob muốn duy nhất Alice có thể đọc được. Để làm được điều này, Alice gửi cho Bob một chiếc hộp có khóa đã mở sẵn và giữ lại chìa khóa. Bob nhận chiếc hộp, cho vào đó một tờ giấy viết thư bình thường và khóa lại (như loại khoá thông thường chỉ cần sập chốt lại, sau khi sập chốt khóa ngay cả Bob cũng không thể mở lại được-không đọc lại hay sửa thông tin trong thư được nữa). Sau đó Bob gửi chiếc hộp lại

cho Alice. Alice mở hộp với chìa khóa của mình và đọc thông tin trong thư. Trong ví dụ này, chiếc hộp với khóa mở đóng vai trò khóa công khai, chiếc chìa khóa chính là khóa bí mật.

Tạo khóa

Giả sử Alice và Bob cần trao đổi thông tin bí mật thông qua một kênh không an toàn (ví dụ như Internet). Với thuật toán RSA, Alice đầu tiên cần tạo ra cho mình cặp khóa gồm khóa công khai và khóa bí mật theo các bước sau:

Bước 1. Chọn 2 số nguyên tố lớn p và q với $p \neq q$, lựa chọn ngẫu nhiên và độc lập.

Bước 2. Tính: $n = pq$.

Bước 3. Tính: giá trị hàm số Euler $\phi(n) = (p-1)(q-1)$.

Bước 4. Chọn một số tự nhiên e sao cho $1 < e < \phi(n)$ và là số nguyên tố cùng nhau với $\phi(n)$.

Bước 5. Tính d sao cho $de \equiv 1 \pmod{\phi(n)}$.

Một số lưu ý.

- Các số nguyên tố thường được chọn bằng phương pháp thử xác suất.
- Các bước 4 và 5 có thể được thực hiện bằng giải thuật Euclid mở rộng (xem thêm: số học modul).
- Bước 5 có thể viết cách khác: Tìm số tự nhiên x sao cho $d = \frac{x(p-1)(q-1)+1}{e}$ cũng là số tự nhiên. Khi đó sử dụng giá trị $d \bmod (p-1)(q-1)$.
- Từ bước 3, sử dụng $\lambda = LCM(p-1, q-1)$ thay cho $\phi = (p-1)(q-1)$.

Khóa công khai bao gồm:

- n, modul, và
- e, số mũ công khai (cũng gọi là *số mũ mã hóa*).

Khóa bí mật bao gồm:

- n, *modul*, xuất hiện cả trong khóa công khai và khóa bí mật, và
- d, số mũ bí mật (cũng gọi là *số mũ giải mã*).

Một dạng khác của khóa bí mật bao gồm:

- p và q, hai số nguyên tố chọn ban đầu,
- $d \bmod (p-1)$ và $d \bmod (q-1)$ (thường được gọi là $dmp1$ và $dmq1$),
- $(1/q) \bmod p$ (thường được gọi là $iqmp$)

Dạng này cho phép thực hiện giải mã và ký nhanh hơn với việc sử dụng định lý số dư Trung Quốc (tiếng Anh: *Chinese Remainder Theorem* - CRT). Ở dạng này, tất cả thành phần của khóa bí mật phải được giữ bí mật.

Alice gửi khóa công khai cho Bob, và giữ bí mật khóa cá nhân của mình. Ở đây, p và q giữ vai trò rất quan trọng. Chúng là các phân tố của n và cho phép tính d khi biết e. Nếu không sử dụng dạng sau của khóa bí mật (dạng CRT) thì p và q sẽ được xóa ngay sau khi thực hiện xong quá trình tạo khóa.

Mã hóa

Giả sử Bob muốn gửi đoạn thông tin M cho Alice. Đầu tiên Bob chuyển M thành một số $m < n$ theo một hàm có thể đảo ngược (từ m có thể xác định lại M) được thỏa thuận trước. Quá trình này được mô tả ở phần Chuyển đổi văn bản rõ.

Lúc này Bob có m và biết n cũng như e do Alice gửi. Bob sẽ tính c là bản mã hóa của m theo công thức:

$$c = m^e \pmod n.$$

Hàm trên có thể tính dễ dàng sử dụng phương pháp tính hàm mũ (theo mdul) bằng (thuật toán bình phương và nhân). Cuối cùng Bob gửi c cho Alice.

Giải mã

Alice nhận c từ Bob và biết khóa bí mật d. Alice có thể tìm được m từ c theo công thức sau:

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

Biết m, Alice tìm lại M theo phương pháp đã thỏa thuận trước. Quá trình giải mã hoạt động vì ta có

$$c^d \equiv \left(m^e\right)^d \equiv m^{cd} \pmod n.$$

Do $ed \equiv 1 \pmod{(p-1)}$ và $ed \equiv 1 \pmod{(q-1)}$, (theo Định lý Fermat nhỏ) nên:

$$m^{ed} \equiv m \pmod p \text{ và } m^{ed} \equiv m \pmod q.$$

Do p và q là hai số nguyên tố cùng nhau, áp dụng định lý số dư Trung Quốc, ta có

$$m^{ed} \equiv m \pmod{pq} \text{ hay } c^d \equiv m \pmod n$$

Ví dụ. Sau đây là một ví dụ với những số cụ thể. Ở đây chúng ta sử dụng những số nhỏ để tiện tính toán còn trong thực tế phải dùng các số có giá trị đủ lớn.

Lấy

$p = 61$	— số nguyên tố thứ nhất (giữ bí mật hoặc hủy sau khi tạo khóa)
$q = 53$	— số nguyên tố thứ hai (giữ bí mật hoặc hủy sau khi tạo khóa)
$n = pq = 3233$	— modul (công bố công khai)
$e = 17$	— số mũ công khai
$d = 2753$	— số mũ bí mật

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

Giả sử ta cần mã hóa thông báo: "ĐA GƯI TIÊN". Trước tiên ta chuyển các chữ cái trong văn bản thành các số tương ứng và nhóm chúng thành từng khối 4 chữ số, ta có:

$$0701 \quad 1026 \quad 1224 \quad 1209 \quad 1628$$

Chú ý rằng, ở mỗi chữ tương ứng với một số có 1 chữ số, ta thêm vào số 0 ở trước mỗi số, chẳng hạn chữ A tương ứng với số 01. Hơn nữa, để khối cuối cùng đủ 4 chữ số, ta thêm chữ X trong văn bản, điều này không gây nhầm lẫn khi đọc thông báo (dĩ nhiên có thể thay chữ X bằng bất cứ chữ cái nào không gây hiểu nhầm)

Ta mã hóa các khối theo công thức $c \equiv m^e \pmod{3233}$. Ta dùng phương pháp bình phương liên tiếp để thức hiện, chẳng hạn với khối đầu tiên, ta có $(0701)^{17} \equiv 140 \pmod{3233}$.

Mã hóa toàn bộ văn bản, ta được văn bản mật là

$$140 \quad 721 \quad 1814 \quad 1819 \quad 361$$

Khi nhận được văn bản mật này, để giải mã, Alice phải tìm một nghịch đảo d của e modulo $\phi(3233)$. Ta có $\phi(53.61) = (53-1).(61-1) = 52.60 = 3120$. Dùng thuật toán Euclid mở rộng, ta tính được $d = 2753$. Như vậy, để giải mã, ta dùng công thức $m \equiv c^d \pmod{3233}$, với $0 \le m \le 3233$. Có thể thử lại điều này:

$$c^d \equiv \left(m^e\right)^d \equiv m^{ed} \equiv m \pmod{n}.$$

Chuyển đổi văn bản rõ

Trước khi thực hiện mã hóa, ta phải thực hiện việc chuyển đổi văn bản rõ (chuyển đổi từ *M* sang *m*) sao cho không có giá trị nào của M tạo ra văn bản mã không an toàn. Nếu không có quá trình này, RSA sẽ gặp phải một số vấn đề sau:

- Nếu $m = 0$ hoặc $m = 1$ sẽ tạo ra các bản mã có giá trị là 0 và 1 tương ứng
- Khi mã hóa với số mũ nhỏ (chẳng hạn $e = 3$) và m cũng có giá trị nhỏ, giá trị m^e cũng nhận giá trị nhỏ (so với n). Như vậy phép modul không có tác dụng và có thể dễ dàng tìm được m bằng cách khai căn bậc e của c (bỏ qua modul).
- RSA là phương pháp mã hóa xác định (không có thành phần ngẫu nhiên) nên kẻ tấn công có thể thực hiện tấn công lựa chọn bản rõ bằng cách tạo ra một bảng tra giữa bản rõ và bản mã. Khi gặp một bản mã, kẻ tấn công sử dụng bảng tra để tìm ra bản rõ tương ứng.

Trên thực tế, ta thường gặp 2 vấn đề đầu khi gửi các bản tin ASCII ngắn với m là nhóm vài ký tự ASCII. Một đoạn tin chỉ có 1 ký tự NUL sẽ được gán giá trị $m = 0$ và cho ra bản mã là 0 bất kể giá trị của e và N. Tương tự, một ký tự ASCII khác, SOH, có giá trị 1 sẽ luôn cho ra bản mã là 1. Với các hệ thống dùng giá trị e nhỏ thì tất cả ký tự ASCII đều cho kết quả mã hóa không an toàn vì giá trị lớn nhất của m chỉ là 255 và 255^3 nhỏ hơn giá trị n chấp nhận được. Những bản mã này sẽ dễ dàng bị phá mã.

Để tránh gặp phải những vấn đề trên, RSA trên thực tế thường bao gồm một hình thức chuyển đổi ngẫu nhiên hóa m trước khi mã hóa. Quá trình chuyển đổi này phải đảm bảo rằng m không rơi vào các giá trị không an toàn. Sau khi chuyển đổi, mỗi bản rõ khi mã hóa sẽ cho ra một trong số khả năng trong tập hợp bản mã. Điều này làm giảm tính khả thi của phương pháp tấn công lựa chọn bản rõ (một bản rõ sẽ có thể tương ứng với nhiều bản mã tuỳ thuộc vào cách chuyển đổi).

Một số tiêu chuẩn, chẳng hạn như PKCS, đã được thiết kế để chuyển đổi bản rõ trước khi mã hóa bằng RSA. Các phương pháp chuyển đổi này bổ sung thêm bít vào M. Các phương pháp chuyển đổi cần được thiết kế cẩn thận để tránh những dạng tấn công phức tạp tận dụng khả năng biết trước được cấu trúc của bản rõ.

Phiên bản ban đầu của PKCS dùng một phương pháp đặc ứng (ad-hoc) mà về sau được biết là không an toàn trước tấn công lựa chọn bản rõ thích ứng (adaptive chosen ciphertext attack). Các phương pháp chuyển đổi hiện đại sử dụng các kỹ thuật như chuyển đổi mã hóa bất đối xứng tối ưu (Optimal Asymmetric Encryption Padding - OAEP) để chống lại tấn công dạng này. Tiêu chuẩn PKCS còn được bổ sung các tính năng khác để đảm bảo an toàn cho chữ ký RSA (Probabilistic Signature Scheme for RSA - RSA-PSS).

Tạo chữ ký số cho văn bản

Thuật toán RSA còn được dùng để tạo chữ ký số cho văn bản. Giả sử Alice muốn gửi cho Bob một văn bản có chữ ký của mình. Để làm việc này, Alice tạo ra một giá trị băm (hash value) của văn bản cần ký và tính giá trị mũ d mod n của nó (giống như khi Alice thực hiện giải mã). Giá trị cuối cùng chính là chữ ký điện tử của văn bản đang xét. Khi Bob nhận được văn bản cùng với chữ ký điện tử, anh ta tính giá trị mũ e mod n của chữ ký đồng thời với việc tính giá trị băm của văn bản. Nếu 2 giá trị này như nhau thì Bob biết rằng người tạo ra chữ ký biết khóa bí mật của Alice và văn bản đã không bị thay đổi sau khi ký.

Cần chú ý rằng các phương pháp chuyển đổi bản rõ (như RSA-PSS) giữ vai trò quan trọng đối với quá trình mã hóa cũng như chữ ký điện tử và không được dùng khóa chung cho đồng thời cho cả hai mục đích trên.

An ninh

Độ an toàn của hệ thống RSA dựa trên 2 vấn đề của toán học: bài toán phân tích ra thừa số nguyên tố các số nguyên lớn và bài toán RSA. Nếu 2 bài toán trên là khó (không tìm được thuật toán hiệu quả để giải chúng) thì không thể thực hiện được việc phá mã toàn bộ đối với RSA. Phá mã một phần phải được ngăn chặn bằng các phương pháp chuyển đổi bản rõ an toàn.

Bài toán RSA là bài toán tính căn bậc e môđun n (với n là hợp số): tìm số m sao cho $m^e=c$ mod n, trong đó (e, n) chính là khóa công khai và c là bản mã. Hiện nay phương pháp triển vọng nhất giải bài toán này là phân tích n ra thừa số nguyên tố. Khi thực hiện được điều này, kẻ tấn công sẽ tìm ra số mũ bí mật d từ khóa công khai và có thể giải mã theo đúng quy trình của thuật toán. Nếu kẻ tấn công tìm được 2 số nguyên tố p và q sao cho: $n = pq$ thì có thể dễ dàng tìm được giá trị $(p-1)(q-1)$ và qua đó xác định d từ e. Chưa có một phương pháp nào được tìm ra trên máy tính để giải bài toán này trong thời gian đa thức (*polynomial-time*). Tuy nhiên người ta cũng chưa chứng minh được điều ngược lại (sự không tồn tại của thuật toán). Xem thêm phân tích ra thừa số nguyên tố về vấn đề này.

Tại thời điểm năm 2005, số lớn nhất có thể được phân tích ra thừa số nguyên tố có độ dài 663 bít với phương pháp phân tán trong khi khóa của RSA có độ dài từ 1024 tới 2048 bít. Một số chuyên gia cho rằng khóa 1024 bít có thể sớm bị phá vỡ (cũng có nhiều người phản đối việc này). Với khóa 4096 bít thì hầu như không có khả năng bị phá vỡ trong tương lai gần. Do đó, người ta thường cho rằng RSA đảm bảo an toàn với điều kiện n được chọn đủ lớn. Nếu n có độ dài 256 bít hoặc ngắn hơn, nó có thể bị phân tích trong vài giờ với máy tính cá nhân dùng các phần mềm có sẵn. Nếu n có độ dài 512 bít, nó có thể bị phân tích bởi vài trăm máy tính tại thời điểm năm 1999. Một thiết bị lý thuyết có tên là TWIRL do Shamir và Tromer mô tả năm 2003 đã đặt ra câu hỏi về độ an toàn của khóa 1024 bít. Vì vậy hiện nay người ta khuyến cáo sử dụng khóa có độ dài tối thiểu 2048 bít.

Năm 1993, Peter Shor công bố thuật toán Shor chỉ ra rằng: máy tính lượng tử (trên lý thuyết) có thể giải bài toán phân tích ra thừa số trong thời gian đa thức. Tuy nhiên, máy tính lượng tử vẫn chưa thể phát triển được tới mức độ này trong nhiều năm nữa.
Bài toán phân tích RSA

Các vấn đề đặt ra trong thực tế

Quá trình tạo khóa

Việc tìm ra 2 số nguyên tố đủ lớn p và q thường được thực hiện bằng cách thử xác suất các số ngẫu nhiên có độ lớn phù hợp (dùng phép kiểm tra nguyên tố cho phép loại bỏ hầu hết các hợp số).

p và q còn cần được chọn không quá gần nhau để phòng trường hợp phân tích n bằng phương pháp phân tích Fermat. Ngoài ra, nếu $p-1$ hoặc $q-1$ có thừa số nguyên tố nhỏ thì n cũng có thể dễ dàng bị phân tích và vì thế p và q cũng cần được thử để tránh khả năng này.

Bên cạnh đó, cần tránh sử dụng các phương pháp tìm số ngẫu nhiên mà kẻ tấn công có thể lợi dụng để biết thêm thông tin về việc lựa chọn (cần dùng các bộ tạo số ngẫu nhiên tốt). Yêu cầu ở đây là các số được lựa chọn cần đồng thời ngẫu nhiên và không dự đoán được. Đây là các yêu cầu khác nhau: một số có thể được lựa chọn ngẫu nhiên (không có kiểu mẫu trong kết quả) nhưng nếu có thể dự đoán được dù chỉ một phần thì an ninh của thuật toán cũng không được đảm bảo. Một ví dụ là bảng các số ngẫu nhiên do tập đoàn Rand xuất bản vào những năm 1950 có thể rất thực sự ngẫu nhiên nhưng kẻ tấn công cũng có bảng này. Nếu kẻ tấn công đoán được một nửa chữ số của p hay q thì chúng có thể dễ dàng tìm ra nửa còn lại (theo nghiên cứu của Donald Coppersmith vào năm 1997)

Một điểm nữa cần nhấn mạnh là khóa bí mật d phải đủ lớn. Năm 1990, Wiener chỉ ra rằng nếu giá trị của p nằm trong khoảng q và $2q$ (khá phổ biến) và $d < n^{1/4}/3$ thì có thể tìm ra được d từ n và e.

Mặc dù e đã từng có giá trị là 3 nhưng hiện nay các số mũ nhỏ không còn được sử dụng do có thể tạo nên những lỗ hổng (đã đề cập ở phần chuyển đổi văn bản rõ). Giá trị thường dùng hiện nay là 65537 vì được xem là đủ lớn và cũng không quá lớn ảnh hưởng tới việc thực hiện hàm mũ.

Tốc độ

RSA có tốc độ thực hiện chậm hơn đáng kể so với DES và các thuật toán mã hóa đối xứng khác. Trên thực tế, Bob sử dụng một thuật toán mã hóa đối xứng nào đó để mã hóa văn bản cần gửi và chỉ sử dụng RSA để mã hóa khóa để giải mã (thông thường khóa ngắn hơn nhiều so với văn bản).

Phương thức này cũng tạo ra những vấn đề an ninh mới. Một ví dụ là cần phải tạo ra khóa đối xứng thật sự ngẫu nhiên. Nếu không, kẻ tấn công (thường ký hiệu là Eve) sẽ bỏ qua RSA và tập trung vào việc đoán khóa đối xứng.

Phân phối khóa

Cũng giống như các thuật toán mã hóa khác, cách thức phân phối khóa công khai là một trong những yếu tố quyết định đối với độ an toàn của RSA. Quá trình phân phối khóa cần chống lại được tấn công đứng giữa (*man-in-the-middle attack*). Giả sử Eve có thể gửi cho Bob một khóa bất kỳ và khiến Bob tin rằng đó là khóa (công khai) của Alice. Đồng thời Eve có khả năng đọc được thông tin trao đổi giữa Bob và Alice. Khi đó, Eve sẽ gửi cho Bob khóa công khai của chính mình (mà Bob nghĩ rằng đó là khóa của Alice). Sau đó, Eve đọc tất cả văn bản mã hóa do Bob gửi, giải mã với khóa bí mật của mình, giữ 1 bản copy đồng thời mã hóa bằng khóa công khai của Alice và gửi cho Alice. Về nguyên tắc, cả Bob và Alice đều không phát hiện ra sự can thiệp của người thứ ba. Các phương pháp chống lại dạng tấn công này thường dựa trên các chứng thực khóa công khai (digital certificate) hoặc các thành phần của hạ tầng khóa công khai (public key infrastructure - PKI).

Tấn công dựa trên thời gian

Vào năm 1995, Paul Kocher mô tả một dạng tấn công mới lên RSA: nếu kẻ tấn công nắm đủ thông tin về phần cứng thực hiện mã hóa và xác định được thời gian giải mã đối với một số bản mã lựa chọn thì có thể nhanh chóng tìm ra khóa *d*. Dạng tấn công này có thể áp dụng đối với hệ thống chữ ký điện tử sử dụng RSA. Năm 2003, Dan Boneh và David Brumley chứng minh một dạng tấn công thực tế hơn: phân tích thừa số RSA dùng mạng máy tính (Máy chủ web dùng SSL). Tấn công đã khai thác thông tin rò rỉ của việc tối ưu hóa định lý số dư Trung quốc mà nhiều ứng dụng đã thực hiện.

Để chống lại tấn công dựa trên thời gian là đảm bảo quá trình giải mã luôn diễn ra trong thời gian không đổi bất kể văn bản mã. Tuy nhiên, cách này có thể làm giảm hiệu suất tính toán. Thay vào đó, hầu hết các ứng dụng RSA sử dụng một kỹ thuật gọi là che mắt. Kỹ thuật này dựa trên tính nhân của RSA: thay vì tính $c^d \bmod n$, Alice đầu tiên chọn một số ngẫu nhiên r và tính $(r^e c)^d \bmod n$. Kết quả của phép tính này là $rm \bmod n$ và tác động của r sẽ được loại bỏ bằng cách nhân kết quả với nghịch đảo của r. Đối với mỗi văn bản mã, người ta chọn một giá trị của r. Vì vậy, thời gian giải mã sẽ không còn phụ thuộc vào giá trị của văn bản mã.

Tấn công lựa chọn thích nghi bản mã

Năm 1981, Daniel Bleichenbacher mô tả dạng tấn công lựa chọn thích nghi bản mã (adaptive chosen ciphertext attack) đầu tiên có thể thực hiện trên thực tế đối với một văn bản mã hóa bằng RSA. Văn bản này được mã hóa dựa trên tiêu chuẩn PKCS #1 v1, một tiêu chuẩn chuyển đổi bản rõ có khả năng kiểm tra tính hợp lệ của văn bản sau khi giải mã. Do những khiếm khuyết của PKCS #1, Bleichenbacher có thể thực hiện một tấn công lên bản RSA dùng cho giao thức SSL (tìm được khóa phiên). Do phát hiện này, các mô hình chuyển đổi an toàn hơn như chuyển đổi mã hóa bất đối xứng tối ưu (Optimal Asymmetric Encryption Padding) được khuyến cáo sử dụng. Đồng thời phòng nghiên cứu của RSA cũng đưa ra phiên bản mới của

PKCS #1 có khả năng chống lại dạng tấn công nói trên.

……

Tài liệu tham khảo

[1] http://vi.wikipedia.org/wiki/Mật_mã_hóa_khóa_công_khai

[2] Hà Huy Khoái, Phạm Hữu Điển, "Số Học Thuật Toán – Cơ sở lý thuyết & Tính toán thực hành", NXB ĐHQG Hà Nội, 2003.

[3] Nguyễn Vũ Thông, "Toán Mật Mã", Tạp chí Toán Tuổi Thơ số 59, 1/2008.