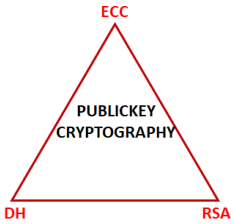
3 MÃ CÔNG KHAI



CẤU TRÚC NHÓM, VÀNH, TRƯỜNG

Nhóm (*group*), vành (*ring*), và trường (*field*) là 3 cấu trúc đại số quan trọng được dùng để thiết lập các hệ mã, đặc biệt các hệ mã công khai phổ biến hiện nay.

Nhóm

Phép toán 2-ngôi

Trên tập hợp G khác rỗng, ta định nghĩa một phép toán 2 ngôi, ký hiệu $\otimes$,

$\otimes$: $G \times G \rightarrow G$,

Sao cho với mọi phân tử a, b, c thuộc G, thỏa 4 tiên đề nhóm sau:

(1) $a \otimes b \in G$.

(2) $a \otimes (b \otimes c) = (a \otimes b) \otimes c$.

(3) $\exists e \in G$: $a \otimes e = e \otimes a = a$. e được gọi là phần tử đơn vị của G.

(4) $\exists a^{-1} \in G$: $a \otimes a^{-1} = e = a^{-1} \otimes a$. a^{-1} được gọi là phần tử nghịc đảo của a.

Tập G với phép toán $\otimes$ tạo thành một cấu trúc đại số có tên là nhóm (*group*), ký hiệu $(G, \otimes)$.

Ví dụ

VD1. Tập $\mathbb{Z}$ với phép cộng thông thường là một nhóm $(\mathbb{Z}, +)$.

Thực vậy, với mọi số nguyên x, y, z, ta có (1) x + y cũng là số nguyên, (2) (x + y) + z = x + (y + z) = x + y + z, (3) 0 là phần tử đơn vị: x + 0 = 0 + x = x, và (3) -x là phần tử nghịch đảo của x.

VD2. Dễ dàng kiểm tra tập các số nguyên chẵn, ký hiệu $2\mathbb{Z}$, với phép cộng thông thường $(2\mathbb{Z}, +)$ thỏa 4 tiên đề nhóm.

VD3. $3\mathbb{Z} = \{2n + 1, n \in \mathbb{Z}\}$, tập các số lẻ, không phải là nhóm với phép cộng thông thường vì cộng của 2 số lẻ là số chẵn không thuộc $3\mathbb{Z}$.

VD4. Tập các số thực $\mathbb{R}$ với phép nhân thông thường không phải nhóm vì số thực 0 không có nghịch đảo. Nhưng $(\mathbb{R}\backslash\{0\}, *)$ là nhóm.

Nếu phép $\otimes$ có tính giao hoán, nghĩa là $a \otimes b = b \otimes a$ với mọi a, b thuộc G, thì nhóm $(G, \otimes)$ được gọi là nhóm giao hoán.

Các nhóm ví dụ trên đều là nhóm giao hoán. Ví dụ sau minh họa một nhóm không giao hoán.

VD5. Tập tất cả các ma-trận cấp n khả nghịch $\mathcal{M}_n$ với các phần tử thực, với phép nhân ma-trận $(\mathcal{M}_n, \cdot)$ là nhóm không giao hoán. Thực vậy, (1) Tích của 2 ma-trận vuông khả nghịc cấp n là ma-trận khả nghịch, (2) (A·B)·C = A·(B·C) đúng với mọi ma-trận vuông cùng cấp, (3) A·I = A = I·A đúng với mọi ma-trận vuông cấp n và I là ma-trận đơn vị cấp n, (4) Do $\mathcal{M}_n$ là tập tất cả các ma-trận cấp n khả nghịch nên A và A^{-1} cũng thuộc về $\mathcal{M}_n$. Mặt khác, A·B thường khác B·A, nên $(\mathcal{M}_n, \cdot)$ là nhóm không giao hoán.

Do tính chất kết hợp của phép ⊗ nên (x ⊗ x) ⊗ x = x ⊗ (x ⊗ x) = x ⊗ x ⊗ x. Trong trưởng hợp tổng quát, n phần tử x kết hợp với nhau, x ⊗ … ⊗ x, ta ký hiệu

$$x \otimes \ldots \otimes x = \begin{cases} e, n = 0 \\ x^n, n > 0 \end{cases}$$

Nhóm con

Cho (G, ⊗) là nhóm, nếu H ⊂ G và (H, ⊗) là nhóm thì H được gọi là nhóm con của G.

Ví dụ $(2\mathbb{Z}, +)$ trong VD2 ở trên là nhóm con của $(\mathbb{Z}, +)$. Hay tập các số hữu tỷ không chứa phần tử 0, $\mathbb{Q}\backslash\{0\}$ với phép nhân thông thường là nhóm con của nhóm $(\mathbb{R}\backslash\{0\}, *)$.

Cho tập S khác rỗng và hữu hạn các phần tử. Đặt G là tập tất cả các phần tử tạo thành bằng cách kết hợp các phần tử trong S theo phép ⊗.

Nếu (G, ⊗) là nhóm, ta gọi (G, ⊗) là nhóm sinh bởi S, ký hiệu (<S>, ⊗) hay đơn giản là <S>.

Nếu S chỉ có 1 phần tử, S = {a}, thì G = <S> = <a> được gọi là nhóm đơn sinh, hay nhóm tuần hoàn (*cyclic group*). Phần tử a gọi là phần tử sinh của G.

VD6. $< 1 > = (\mathbb{Z}, +)$.

VD7. Đặt $\mathbb{Z}_7^+ = \{1, 2, 3, 4, 5, 6\}$ và %: $\mathbb{Z}_7 \times \mathbb{Z}_7 \to \mathbb{Z}_7$ xác định bởi x % y là phần dư của phép chia của tich x*y cho 7. Thì $(\mathbb{Z}_7, \%)$ là nhóm. Thực vậy, (1) $\forall x, y \in \mathbb{Z}_7, x\%y \in \mathbb{Z}_7$, (2) $(x\ \%y)\%\ z \in = x\ \%\ y\ \%\ z$ đúng với mọi $x, y, z \in \mathbb{Z}$, (3) 1 là phần tử đơn vị của $\mathbb{Z}_7$, và (4) $1^{-1} = 1$, $2^{-1} = 4$, $3^{-1} = 5$, $4^{-1} = 2$, $5^{-1} = 3$, $6^{-1} = 6 \in \mathbb{Z}_7$. Từ đây ta viết $\mathbb{Z}_7$ thay cho $\mathbb{Z}_7^+$ nếu không có ghi chú đặc biệt nào khác.

Định lý Lagrange

Nhóm (G, ⊗) là nhóm hữu hạn nếu số phần tử của G hữu hạn, |G| = n < ∞, và n được gọi là bậc (order) của nhóm G, ký hiệu |G|.

Bên cạnh khái niệm bậc của nhóm, các phần tử của nhóm cũng có khái niệm bậc.

Bậc của một phần tử a ∈ (G, ⊗), ký hiệu |a|, là số nguyên dương n nhỏ nhất sao cho a^n = e, với e là phần tử đơn vị của nhóm G.

Từ khái niệm bậc của nhóm và của phần tử thuộc nhóm, ta có kết quả:

Nếu bậc của phần tử a thuộc nhóm G bằng với bậc của nhóm, |a| = |G|, thì <a> = G và a là phần tử sinh của G.

Định lý Lagrange cho biết quan hệ giữa nhóm G và các nhóm con H của G, và được phát biểu như sau:

Nếu H là nhóm con của G thì bậc của H là ước của bậc của G: $|H| \vdots |G|$.

Như vậy, nếu bậc của nhóm G là số nguyên tố thì G chỉ có 2 nhóm con là H = {e} và chính nó, H = G. $\mathbb{Z}_7^+$ trong **VD7** chỉ có 2 nhóm con $\{1\}$ và $\mathbb{Z}_7^+$. Trong nhóm $\mathbb{Z}_7$, phần tử 3 có tính bậc bằng 6, và <3> = $\{3^0 = 1, 3^1 = 3, 3^2 = 2, 3^3 = 6, 3^4 = 4, 3^5 = 5\}$. Một cách tổng quát, ta có

$(\mathbb{Z}_p^+, \%)$ là nhóm đơn sinh nếu p là số nguyên tố.

Vành

Xét tập hợp khác rỗng R với 2 phép toán + và · được định nghĩa trên R có tính chất

(1) + và · có tính đóng.

(2) + và · có tính kết hợp.

(3) Tồn tại phần tử đơn vị cho phép cộng là 0 và cho phép nhân là 1.

(4) Phép + có tính giao hoán.

(5) Tồn tại phần tử nghịch đảo cho phép cộng, ký hiệu là –.

(6) Phép nhân phân phối với phép cộng: a · (b + c) = a·b + a·c, (b + c)·a = b·a + c·a.

R với hai phép + và · định nghĩa như trên gọi là vành (*ring*), ký hiệu (R, +, ·).

VD8. Tập các ma-trận vuông cấp n với phép cộng và nhân ma-trận là một vành.

VD9. Tập số nguyên $\mathbb{Z}$ với các phép cộng và nhân thông thường là vành.

VD10. $\mathbb{Z}_n$ với các phép cộng (modulo n) và phép nhân (modulo n) là vành.

Trường

Vành (F, +, ·) khác rỗng với 2 phép toán + và · định nghĩa như trên, nếu mọi phần tử khác 0 của F khả nghịch, thì F được gọi là trường (*field*).

Trường hữu hạn là trường có số phần tử hữu hạn.

MẬT MÃ KHÓA CÔNG KHAI

Nguyên lý thiết kế

Hàm một chiều có cửa mật

Trong các hệ mã công khai (*public key cryptosystem*) hay bất đối xứng (*asymmetric crypyosystem*), các tiến trình mã hóa và giải mã sử dụng các khóa (*key*) khác nhau. Khóa mã hóa (*enctyption key*),

ký hiệu **e**, có thể được công bố, nên còn gọi là khóa công khai (*public key*), nhưng khóa giải mã (*decryption key*) – **d**, phải giữ bí mật, gọi là khóa cá nhân (*private key*). Các hệ mã công khai thường được xây dựng dựa trên ý tưởng ***hàm* 1-*chiều có cửa mật*** (*one-way function with trapdoor*). Một cách hình thức, được định nghĩa như sau:

Cho các tập hữu hạn khác rỗng S và T. Hàm 1-chiều $f: S \to T$ là hàm khả nghịch có các tính chất:

(1) f dễ thực hiện. Nghĩa là cho $x \in S$, $y = f(x)$ dễ tính theo nghĩa có thể tính được với độ phức tạp tối đa là đa thức.

(2) f^{-1}, hàm ngược của f, khó thực hiện. Nghĩa là cho $y \in$ T, rất khó tìm được $x \in$ S sao cho $f(x) = y$ hay $x = f^{-1}(x)$.

(3) f^{-1} có thể tính được với thời gian tối đa đa thức khi có thêm một số thông tin cửa mật (*trapdoor*).

Một số ví dụ

VD1. f: pq $\to$ n là hàm 1-chiều. Trong đó p và q là hai số nguyên tố lớn và khác nhau, Thực vậy, tích pq có thể dễ thực hiện chỉ bằng phép nhân số nguyên lớn (độ phức tạp đa thức); nhưng tính f^{-1}: n $\to$ (p, q) được chứng minh là bài toán khó, bài toán phân tích ra thừa số (**IFP** – *integer factorization problem*) với độ phức tạp mũ. Nhưng f^{-1} tính p sẽ trở nên dễ nếu ta có *cửa mật* q, khi ấy, chỉ phải thực hiện 1 phép chia số nguyên lớn.

VD2. $f_{e,n}$: $x \to x^e \pmod n$ là hàm 1-chiều, với $n = pq$ là tích 2 số nguyên tố bí mật, lớn và phân biệt, còn e là số nguyên nguyên tố cùng nhau với $\varphi = (p - 1)(q - 1)$. Thực vậy, phép tính $x^e \pmod n$ có độ phức tạp đa thức, nhưng f^{-1}: $x^e \to x$ chỉ tính được khi biết cửa mật p (hay q) mà chính là hàm 1-chiều ở ví dụ 1. Nếu biết cửa mật p, việc tính $\varphi = (p - 1)(q - 1)$ dễ, dễ tìm được d sao cho $de = 1 \pmod \varphi$ bằng thuật toán Euclide. Khi ấy, có thể dễ dàng tính lại được x theo công thức $x = (f^{-1})^d = (x^e)^d = x^{ed}$.

Quy trình thiết kế hệ mã công khai

Nguyên lý cơ bản để thiết kế một hệ mã công khai là định nghĩa hàm-một-chiều-có-cửa-bẫy và công khai các thông tin về hàm một chiều nhưng giữ bí mật thống tin cửa bẫy.

Có thể có nhiều cách xây dựng hàm một chiều, nhưng cách phổ biến và dễ thực hiện là sử dụng một bài toán khó (*hard problem*) đã và còn đang được công nhận trong lãnh vực khoa học máy tính. Trong khoa học máy tính, bài toán khó là bài toán có thuật giải (*algorithm*) nhưng chương trình máy tính (*computer program*) giải nó bằng máy tính là không thể thực hiện trong thời gian chấp nhận được. Các bài toán khó đang được sử dụng phổ biến trong mã hóa-mật mã phải kể đến gồm:

(**IFP**) *Integer Factorization Problem* – bài toán phân tích số nguyên lớn ra thừa số nguyên tố. Bài toán được mô tả là:

IFP. Cho 2 số nguyên lớn và khác nhau. Việc tính n = pq là dễ, nhưng việc tìm lại p hay q từ tich n là rất khó, trừ khi biết q.

(**DLP**) *Discrete Logarithm Problem* – bài toán logarithm rời rạc, gọi tắt là ***log rời rạc***. Bài toán được phát biểu như sau:

DLP. Cho g là phần tử sinh (*generator*) trên trường hữu hạn (*finite field*) F, và *x* là một số nguyên. Việc tính $y = g^x \in F$ là dễ, nhưng để tính *x* khi biết *y*, g là rất khó.

Bên cạnh đó bài toán giải hệ phương trình lấy nghiệm nguyên gần đúng của một hệ phương trình tuyến tính có số phương trình ít hơn số ẩn cũng được xem là khó và có thể được dùng để xây dựng các bài toán khó trên đó. Các bài toán như bài toán tổng con (*subset problem*) hay các bài toán liên quan đến dàn (*lattice*) là những ví dụ cho ứng dụng này. Tài liệu này giới hạn trong các hệ mã phổ biến đang được sử dụng cho chuỗi khối (*blockchain*) dựa trên 2 bài toán **IFP** và **DLP**.

Hệ mã RSA

RSA – Rivest–Shamir–Adleman, viết tắt tên của 3 tác giả phát minh ra hệ mã RSA. RSA được xây dựng chính dựa trên giải thuyết khó giải bài toán phân tích ra thừa số. Ta sẽ phân tích và cài đặt định lý RSA.

Định lý RSA

Định lý (RSA). Cho p và q là 2 số nguyên tố khác nhau. Đặt n = pq, φ = (p – 1)(q – 1), và chọn 2 số e và d thỏa ed % φ = 1, với % là phép chia lấy phần dư. Với mọi $m \in \mathbb{Z}_n = \{0, 1, 2, \dots, n-1\}$, nếu $c = m^e \% n$ thì $m = c^d \% n$.

Lớp tương đương – ≡

Trong định lý này, $\mathbb{Z}_n$ với phép + (module n), định nghĩa bởi $x + y \ (mod\ n) = (x + y) \% n$

và phép * (modulo n), định nghĩa bởi $x * y \ (mod\ n) = xy \% n$, hình thành nên một vành hữu hạn *n* phần tử, vành $(\mathbb{Z}_n, +, *)$.

Theo ký hiệu phép * (mod) này, biểu thức $ed \% \varphi = 1$ được viết lại thành $e * d \ (mod\ \varphi) = 1$, ký hiệu là $e * d \equiv 1 \ (mod\ \varphi), \forall e, d \in \mathbb{Z}$. Một cách tổng quát:

$y \equiv x \ (mod\ z)$ là tập tất cả các số nguyên *y* chia cho *z* dư *x*, và gọi là lớp tương đương của *x*.

Sinh khóa – **key generating**

Chọn *e* và *d* thỏa ràng buộc $e * d \equiv 1 \ (mod\ \varphi)$ có thể thực hiện bằng cách chọn một số $e \in \mathbb{Z}_\varphi$ khả nghịch và *d* chính là e^{-1}: $d \equiv e^{-1} \pmod{\varphi}$. Về mặt cài đặt, có thể sử dụng thuật giải Euclide mở rộng (*extended Euclidian Algorithm*) hay còn gọi là định lý Bezout, người đã mở rộng định lý Euclide tính ước chung lớn nhất của 2 số nguyên dương, để chọn e và d.

Định lý (Bezout). Với mọi số nguyên dương a, b, luôn tồn tại hai số nguyên x, y thuộc ℤ sao cho gcd(a, b) = ax + by, trong đó gcd(a, b) là ước chung lớn nhất của a và b.

Nhu vậy, thuật toán tìm cặp e và d, gọi là thuật toán sinh khóa (**keyGen** – *key generating*) có thể mã giả như sau:

Algorithm keyGen(φ): #φ = (p – 1)(q – 1) với p và q là 2 số nguyên tố phân biệt.

(1) Chọn ngẫu nhiên một số lẻ e $\in \mathbb{Z}_{\varphi}$.

(2) Nếu gcd(e, φ) = ex + φy = 1, thì trả về d = x; nếu không, quay lại (1).

Mã hóa và giải mã – **Encryption and Decryption**

Nếu xem m $\in \mathbb{Z}_n$ là thông điệp (*message*) thì c = m^e (mod n) $\in \mathbb{Z}_n$ là bản mã tương ứng được mã bằng khóa công khai e. Khi ấy, để giải mã, ta thực hiện phép tính c^d (mod n). Theo định lý RSA, ta có $c^d \equiv m$ (mod n). Ta sẽ sử dụng định lý Fermat nhỏ (*little Fermat*) và số dư Trung Hoa (**CRT** – *Chinese Remainder Theorem*) để chứng minh phát biểu này.

Định lý (Fermat nhỏ). Nếu p là số nguyên tố và b là số số nguyên- bất kỳ không phải bội số của p, thì $b^{p-1} \equiv 1$ (mod p).

Ví dụ cho p = 5. Với b = 2, ta có $2^{5-1} \equiv 2^4 \equiv 16 \equiv 1$ (mod 5). Với b = 3, $3^4 \equiv 91 \equiv 1$ (mod 5). Bây giờ, cho p = 11 và chọn b = 2, $2^{10} \equiv 1024 \equiv 1$ (mod 11); với b = 9, $9^{10} \equiv 3486784401 \equiv 1$ (mod 11).

Định lý (CRT). Nếu $m_1,\ldots, m_k$ là k số nguyên đôi một nguyên tố cùng nhau (*co-prime*), gcd(m_i, m_j) = 1 với mọi i, j = 1, 2, …, k và i≠j; và $a_1,\ldots, a_k$ là k số nguyên bất kỳ, thì hệ

$$\begin{cases} x \equiv a_1 \ (mod\ m_1) \\ \quad\quad \ldots \\ x \equiv a_k \ (mod\ m_k) \end{cases},$$

Có duy nhất nghiệm trong $\mathbb{Z}_m$, với m = $m_1 \ldots m_k$.

Chứng minh định lý **RSA – prove RSA**

Ta có,

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

Do ed ≡ 1 (mod φ) nên có một k sao cho ed = kφ + 1. Suy ra

$m^{ed} \equiv m^{k\varphi + 1} \equiv m.m^{k\varphi}$ (mod n) (*)

(*) luôn đúng với m = 0. Trương hợp m khác p và q,

$m^{ed} \equiv m.m^{k(p-1)(q-1)} \equiv m$ (mod p) #theo định lý Fermat nhỏ $m^{(p-1)}$ mod p = 1

tương tự $m^{ed} \equiv m$ (mod q).

Do p và q là 2 số nguyên tố nên nguyên tố cùng nhau, đặt X ≡ m^{ed} (mod n) hệ

$$\begin{cases} X \equiv m \ (mod\ p) \\ X \equiv m \ (mod\ q) \end{cases} (**),$$

có duy nhất nghiệm trong $\mathbb{Z}_{pq} = \mathbb{Z}_n$. Ta thấy, X ≡ m (mod pq) ≡ m (mod n) là nghiệm của hệ (**). Do tính duy nhất nghiệm, ta có $X \equiv m^{ed} \equiv (m^e)^d \equiv c^d \equiv m$ (mod n).

Giao thức RSA

Giao thức mã công khai RSA (*RSA protocol*) là cài đặt trực tiếp của định lý RSA trên. Theo đó, Bob cần gửi thông điệp mật của m cho Alice, Alice và Bob theo các bước của giao thức sau:

Giao thức – **protocol**

Alice	Bob
Bước 1: sinh khóa – key generating . p, q ← **primeGen**(λ) #sinh 2 số nguyên tố lớn λ-bit . n ← pq . φ ← (p – 1)(q – 1) . e, d ← **keyGen**(φ) #Sau đó, Alice công bố (e, n) và giữ bí mật d, p, q, và φ.	(e, n) được cho công khai để ai cũng có thể gửi tin mật cho Alice.
c được chuyển trên mọi kênh, kể cả kênh không bảo mật và ai cũng có thể lấy được c.	**Bước 2**: mã hóa – encryption . $c = m^e \% n$ #Và chuyển c cho Alice trên kênh không bảo mật.
Bước 3: giải mã – decryption . $m \leftarrow c^d \% n$	chỉ Alice có khóa cá nhân d mới giải mã và đọc được c

Bên cạnh các phép tính trên các số nguyên lớn mà đã có nhiều thư viện cung cấp. Giao thức RSA cần cung cấp hàm sinh khóa – keyGen (là hàm ta đã có thuật giải như mã giả ở phần trước), và hàm sinh số nguyên tố lớn – primeGen. Ta có thể sử dụng định lý sau để tạo số nguyên tố lớn.

Định lý (dùng tạo số nguyên tố). Cho p là số nguyên tố lẻ và k, $1 < 2(p+1)$, là số nguyên tự nhiên không phải bội của p và đặt $n = 2kp + 1$. Các phát biểu sau là tương đương:

(i) n là số nguyên tố.

(ii) Tồn tại một số nguyên tự nhiên a, $2 \le a < n$, thỏa đồng thời

(ii.a) $a^{kp} \equiv 1 \pmod n$ và (ii.b) $\gcd(a^k + 1, n) = 1$.

Hàm primeGen sau mã giả sinh số nguyên tố thỏa định lý trên.

Algorithm primGen(d) #tạo số nguyên tố có d ký số

(1) Chọn 1 số nguyên tố nhỏ p_1 có d_1 ký số, và tìm số $k_1 < 2(p_1 + 1)$ sao cho số $p_2 = 2k_1 + 1$ có d_2 ký số, $d_2 = 2d_1 - 1$ mà có số $a_1 < p_2$ thỏa 2 điều kiện $a_1^{k_1 p_1} \equiv 1 \ (mod\ p_2)$ và $\gcd\left(a_1^{k_1} + 1, p_2\right) = 1$. Theo định lý trên, p_2 là số nguyên tố.

(2) gán $p_2 = p_1$ và lặp lại (1) cho đến khi được số nguyên tố có $d_2 = d$ ký số.

Các thuật toán sinh khóa (bao hàm cả sinh số nguyên tố), có thể được cho mã nguồn trên mạng. Tuy nhiên, cần rất thận trọng khi sử dụng các nguồn mở cho hai hàm này. Tốt nhất là nên tự thực hiện để tránh mọi rủi ro khi triển khai cho chuỗi khối.

Bên cạnh các thuật toán sinh khóa, việc áp dụng RSA vào thực tế, nhất là cho chuỗi khối, tận dụng các tri thức về p, q giúp cho việc giải mã nhanh. Thực vậy, giả sử cần tính c^d để giả mã bản mã c. Ta thực hiện theo cách sau:

. Đặt $d_1 = d \% (p - 1)$ và $d_2 = d \% (q - 1)$. Khi đó, tồn tại n_1 và n_2 thỏa $d = d_1 + n_1(p - 1) = d_2 + n_2(q - 1)$. Và

. $c^d \equiv c^{d_1+n_1(p-1)} \equiv c^{d_1}c^{n_1(p-1)} \equiv c^{d_1}(mod\ p)$ (*). Tương tự $c^d \equiv c^{d_2}(mod\ q)$. (**)

. Do p và 1 là 2 số nguyên tố phân biệt nên hệ (*) và (**) có duy nhất nghiệm trong $\mathbb{Z}_{pq} = \mathbb{Z}_n$.

Thuật giải sau giải hệ đồng dư Trung hoa tổng quát trong định lý CRT phần trước.

Algorithm CRT

(1) $m \leftarrow m_1 \cdot \ldots \cdot m_k$

(2) $n_i \leftarrow m/m_i$, $\forall i = 1, 2, \ldots, k$

(3) $N_i \leftarrow n_i^{-1}\ (mod\ m_i), \forall i = 1,2, \ldots, k$

(4) $X \leftarrow \left(\sum_{i=1}^{k} a_i n_i N_i (mod\ m_i)\right)\ (mod\ m)$

(5) return *X*.

Độ an toàn của RSA

Độ an toan của hệ mã RSA dựa trên giả thuyết RSA:

RSA-conjecture. Mọi phương pháp thám mã RSA phải khó như giải bài toán phân tích thừa số.

Tuy nhiên, RSA có thể không an toàn trong những tình huống cụ thể. Chẳng hạn, hai trường hợp sau dễ khiến RSA bị phá vỡ.

Khóa cá nhân nhỏ. Khi khóa cá nhân d nhỏ, chiều dài bit nhỏ hơn ¼ số bit của n. Wiener đã xây dựng thành công thuật toán phục hồi khóa d nhỏ dựa trên liên phân số. Sau này, các tấn công dàn cũng thành công khi d nhỏ. Vì thế, để an toàn, d phải lớn.

Khóa công khai có bậc nhỏ. Khi khóa cá nhân có bậc r nhỏ trong một nhóm $\lambda(n)$. Khi ấy $e^r \equiv 1 \pmod{\lambda(n)}$, vì thế $m^{e^r} \equiv m\ (mod\ n)$. Đây chính là lần lặp thứ r của hàm mã hóa thông điệp m. Vì thế, để an toàn, r phải lớn.

HỆ MÃ DIFFIE-HELLMAN

Hệ mã hỗn hợp

Khác với RSA mà dựa trên bài toán phân tích ra thừa số, Diffie-Hellman, tên của 2 tác giả phát mính ra, được xây dựng dựa trên bài toán log-rời rạc: ***cho g, p, g^x mod p, tìm p***. Nếu như RSA thiết lập hàm một chiều có cửa mật một cách gián tiếp từ bài toán nền, thì Diffie-Hellman thiết bằng cách dùng lập trực tiếp *x* trong g^x là cửa mật. Và thay vì mã hóa thông điệp, Diffie-Hellman thiết lập bí mật chung giữa các đối tác trong giao thức. Sau pha thiết lập khóa, khóa bí mật chung giữa

các đối tác có thể được sử dụng để trao đổi tin mật theo cách mã đối xứng. Giao thức sau minh họa một hệ mã hỗn hợp.

Allice	Bob
. p ← primeGen(λ) #Allice và Bob quy ước dùng chung một trường $\mathbb{Z}_p$ và một phần tử sinh g	
Bước 1: thiết lập khóa chung – **key exchange** . $x \xleftarrow{\$} \mathbb{Z}$ #chọn ngẫu nhiên một số ngẫu nhiên . $k_A \leftarrow g^x \% p$ (giữ bí mật x và chuyển k_A cho Bob) . $k \leftarrow k_B^x \% p$	. $y \xleftarrow{\$} \mathbb{Z}$. $k_B \leftarrow g^y \% p$ (giữ bí mật y và chuyển k_B cho Allice) . $k \leftarrow k_A^y \% p$
Sau bước này, Allice và Bob có chung khóa bí mật $k \equiv g^{xy} \pmod p$	
Bước 2: mã hóa thông điệp m - **encryption** . c ← m*k % p (chuyển c cho Bob)	
	Bước 3: giải mã – **decryption** . $m \leftarrow c*k^{-1} \% p$

Hệ mã trên sử dụng giao thức Diffie-Hellman cho pha thiết lập khóa bí mật chung (key exchange protocol). Giao thức này sử dụng giả thiết Diffie-Hellman, mà thực chất là bài toán khó log-rời rạc (**DLP** – *Discrete Logarithm Problem*) , đảm bảo độ an toàn của hệ mã.

Giả thuyết Diffie-Hellman (DH conjecture). Hầu như không thể tính được g^{xy} chỉ từ g^x, g^y.

Mặc dầu vậy, nếu số nguyên tố p là số Mersenne, p = 2q + 1, với q cũng là số nguyên tố, hệ DH sẽ bị phá vỡ. Thực vậy, sử dụng kết quả của định lý sau:

Định lý. x là phần tử sinh của $\mathbb{Z}_m \Leftrightarrow x^{\frac{\phi(m)}{q}} \not\equiv 1 \ (mod\ m), \forall q | \phi(m), q$ là số nguyên tố. Với ϕ(m) là hàm ϕ-Euler trả về số các số nguyên dương nhỏ hon m và nguyên tố cùng nhau với m.

Nếu

p = 2q + 1, thì theo định nghĩa của hàm ϕ-Euler, ϕ(p) = 2q, và với phần tử sinh $x \in (1, p)$ ta có $x^2 \equiv 1 \pmod p$ và $x^q \equiv 1 \pmod p$.

Vậy $1 = g^{p-1} \equiv (g^q)^2 \pmod p = \beta^2$.

Theo định lý trên, $\beta \neq 1$ và $\beta^2 = 1$, suy ra $\beta = -1$.

Khi đó, $k \equiv g^{xyq} \equiv \beta^{xy} \pmod p \in \{-1, 1\}$.

Vì vậy, tin tặc có thể sử dụng tấn công trung gian (**MiM** attack – *Man in the Middle attack*), thay 2 thông điệp g^x, g^y bằng các thông điệp mới g^{xq}, và g^{yq}, thì khóa bí mật chung chỉ có 2 giá trị là -1 và 1.

HỆ MÃ ĐƯỜNG CONG ELLIPTIC

Mã đường cong elliptic – **ECC** (Elliptic Curve Cryptosystem) là hệ mã khóa công khai hiện đang được sử dụng trong các chuỗi khối phổ biến như Bitcoin, Ethereum. Sẽ có bài riêng về hệ mã này. Ở đây, chúng tôi chỉ trình bài ý tưởng chính của ECC.

Để cài đặt ECC, cần thực hiện trước các phép tính nhúng thông điệp vào đường cong elliptic. Mục đích là để có thể thực hiện mã hóa thông điệp bằng ECC trong một trường F_q. Cụ thể, muốn nhúng thông điệp rõ thành một điểm trên đường cong elliptic xác định trên trường F_q, giả sử $q = p^r$ và p là số nguyên tố. Xem thông điệp m như một số nguyên $0 \le m \le M$, và chọn k là một số nguyên, $lower \le k \le upper$. Với đường cong (E): $y^2 = x^3 + ax + b$ xác định trên F_q. Thông điệp m được nhúng như sau:

(1) Tính $x = mk + j$, $j = 0, 1, 2,\ldots$ cho đến khi tìm được $x^3 + ax + b$ là số chính phương (mod p), trả về điểm $(x, \sqrt{x^3 + ax + b})$ trên (E).

(2) Để chuyển điểm (x, y) trên (E) lại thành thông điệp m, tính $m = \lfloor x/k \rfloor$.

Ta sẽ quay trở lại hệ mã đường cong elliptic trong bài sau.