MỤC LỤC

MỞ ĐẦU

Năm 1946 chiếc máy tính đầu tiên được khai sinh tại Hoa kỳ từ đó đã phát triển rất mạnh cho đến nay. Trải qua nhiều thế hệ máy tính đã có đã có những cải tiến vượt bậc, đã và đang thâm nhập sâu rộng vào hầu hết các lĩnh vực. Nhờ sự tiện lợi, tốc độ xử lý cao, khả năng lưu trữ lớn nó đã đang dần thay thế các phương thức lưu trữ, xử lý dữ liễu xưa thủ công. Vào khoảng cuối thập niên 80 đầu thập niên 90. Sự bùng nổ của mạng Internet một lần nữa đưa máy tính bước sang một trang sử mới. Dữ liệu được lưu thông một cách nhang chóng, thuận tiện. Ứng dụng máy mạng máy tính giúp dễ dàng trong việc trao đổi dữ liệu đã ra đời và ngày một phổ biến. Các phương thức thanh toán, trao đổi thông dữ liệu qua mạng thay thế hầu hết những phương thức thủ công.

Trong một môi trường dữ liệu "mở" như vậy, dữ liệu có thể được nhiều người khai thác và xử dụng vào nhiều mục đích khác nhau bên cạnh đó việc lưu trữ và trao đổi thông tin kém an toàn sẽ là một cơ hội cho những kẻ xấu muốn phá hoại thông tin hoặc xử dụng dữ liệu sai mục đích hoành hành. Vì để đảm bảo rằng dữ liệu lưu trữ không bị thay đổi hay truy cập trái phép, tin tứ truyền trên mạng đến đúng đích cần đến mà không bị bên thứ ba can thiệp, việc tạo ra các cơ chế bảo mật, xác thức thông tin là rất cần thiết.

Trong đề tài này em xin chỉ đề cập đến các vấn đề liên quan mã hóa và xác thực khi truyền tin và bỏ qua phần lưu trữ.

Mục tiêu cơ bản của mật mã là cho phép hai người, giả sử A và B liên lạc với nhau qua kênh không an toàn theo cách mà người thứ ba O (được nói đến như người thám mã) khó có thể hiểu đươc hai người đang liên lạc gì với nhau. Kênh này có thể là đường điện thoại, mạng máy tính hay đơn thuần là thư tay. Thông tin mà A gửi cho B được gọi là "bản rõ" (plaintext), có thể là bất kỳ văn bản tài liệu nào. A sẽ mã hóa" bản rõ " với 1 hoặc nhiều khóa bằng một thuật toán mã hóa cho trước. Sau khi mã hóa dữ liệu A gửi cho B "bản mã" thông qua kênh truyền tin công cộng hoặc bí mật. B nhận "bản mã" sẽ dùng nó kết hợp với 1 hoặc nhiều khóa có sãn cùng với thuật toán giải mã đúng. Sẽ cho ra kết quả là gói dữ liệu ban đầu trước khi A mã hóa. Nếu trên đường truyền tin, O đánh cắp bản mã. O không có trong tay "khóa" và thuật toán giải mã. Trong tay O chỉ là 1 gói dữ liệu hỗn độn và không có giá trị.

Có hai loại hệ mật gồm hệ mật mã khóa bí mật và hệ mật mã khóa công khai. Trong hệ mật mã khóa công khai, hai người muốn trao đổi thông tin với nhau phải thỏa thuận với nhau một cách bí mật khóa k. Trong hệ mật mã này có hai hàm lập mã ek và hàm giải mã dk. Nếu tiết lộ khóa k sẽ làm cho hệ thống không an toàn. Trong thực tế, Độ an toàn hệ thống chính là độ an toàn tính toán. Một hệ mật là "an toàn tính toán" nếu phương pháp tốt nhất đã biết để phá nó yêu cầu một số lớn không hợp lý thời gian tính toán, nghĩa là quá trình thực hiện tính toán cực kỳ phức tạp, phức tạp đến mức ta coi "không thể được". Hệ mã khóa công khai đã đáp ứng được yêu cầu đó. Ý tưởng của hệ mã khóa công khai là ở chỗ nó có thể tìm ra một hệ mã khó có thể tính toán xác định dk khi biết ek. quy tắc mã ek có thể công khai. Hàm mã hóa công khai ek phải dễ dàng tính toán nhưng việc giải mã phải khó đối với bất kì người nào ngoài người lập mã. Tính chất dễ tính toán và khó đảo ngược này thường được gọi là tính chất một chiều. Điều này bảo đảm tính bí mật cao.

Như chúng ta đã biết, trong cách thức giao dịch truyền thống, thông báo được truyền đi trong giao dịch thường dưới dạng viết tay hoặc đánh máy kèm theo chữ ký(viết tay) của người gửi ở bên dưới văn bản. Chữ ký đó là bằng chứng xác nhận thông báo đúng là của người ký, tức là chủ thể giao dịch. Chữ ký viết tay có nhiều ưu điểm đó là dễ kiểm thử, không sao chép được chữ ký của một người là giống nhau trên nhiều văn bản…

Ngày nay, cùng với sự phát triển của khoa học và công nghệ thông tin đặc biệt là sự bùng nổ của mạng máy tính thì nhu cầu trao đổi thông tin trên mạng ngày càng phổ biến. Khi chúng ta chuyển sang cách thức truyền tin bằng các phương tiện hiện đại, các thông báo được truyền đi trên các mạng truyền tin số hóa, bản thân các thông báo cũng biểu diễn dưới dạng số hóa tức là dưới dạng bít nhị phân, "chữ ký" nếu có cũng ở dưới dạng các dãy bit, thì các mối quan hệ tự nhiên kể trên không còn giữ được nữa. Chẳng hạn, "chữ ký" của một người gửi trên những văn bản khác nhau phải thể hiện được sự gắn kết trách nhiệm của người gửi đối với từng văn bản đó thì tất yếu phải khác nhau chứ không thể là những đoạn bit giống nhau như các chữ ký giống nhau trên các văn bản thông thường. Chữ ký viết tay có thể được kiểm thử bằng cách so sánh với nguyên mẫu, nhưng "chữ ký" điện tử thì không thể có "nguyên mẫu" để mà so sánh, việc kiểm thử phải được thực hiện bằng những thuật toán đặc biệt. Một vấn đề nữa đó là chữ ký điện tử có thể sao chép tùy ý khó có thể phân biệt được bản sao và bản gốc nên có thể có nguy cơ dùng lại nhiều lần. Vậy làm thế nào để ngăn chặn nguy cơ đó và làm thế nào để có thể ngăn cản được người ký chối bỏ chữ ký của mình hoặc người kiểm tra chối bỏ việc mình đã nhận đọc thông báo.

Trước những yêu cầu đó, để nâng cao tính an toàn của chữ ký điện tử và để nâng cao trách nhiệm của người ký và người kiểm tra, đòi hỏi người ta phải đưa ra một lược đồ chữ ký sử dụng các giao thức để có thể khắc phục được những nhược điểm của chữ ký số.

Đó là lý do em chọn đề tài "Các Chữ ký không chối bỏ được và ứng dụng"làm đề tài nghiên cứu của mình.

Trong đồ án này em đi sâu tìm hiểu về lược đồ chữ chống chối bỏ và ứng dụng.

CHƯƠNG i. CƠ SỞ MẬT MÃ VÀ CHỮ KÝ SỐ

1.1. CƠ SỞ TOÁN HỌC

1.1.1. Phép chia hết

- Định nghĩa: cho a,b ∈ Z a. Ta nói a chia hết cho b nếu ∃ số c sao cho a = b.c ; Ký hiệu: b|a
- Tính chất: a,b,c ∈ Z
• a|a
• a|b , b|c → a|c
• a|b , a|c → a|(x.b+y.c) ∀ x,y ∈ Z
• a|b , b|a → a ≡ ±b

1.1.2. Không chia hết

- ***Định nghĩa:*** Phép chia gọi là không chia hết nếu tồn tại số r (0 < r < b) sao cho:

a = b.q + r

Với: q là phần nguyên

r là phần dư

1.1.3. Ước số

- ***Định nghĩa:*** Ước số của a và b là c nếu c|a và c|b
- ***Ước số chung lớn nhất :*** Là số lớn nhất mà a và b chia hết

Ký hiệu : c = gcd(a,b) ; (great common d4.11sor)

- ***Bội số chung nhỏ nhất :*** d là BSCNN của a và b nếu ∀ c mà a|c , b|c → d|c

Ký hiệu: d = lcm(a,b) ; (least common multiple)

- ***Tính chất:*** lcm(a,b) = a.b/gcd(a,b)

1.1.4. Nguyên tố cùng nhau

- ***Định nghĩa:*** a,b gọi là hai nguyên tố cùng nhau khi gcd(a,b) = 1 đơn giản (a,b) = 1

1.1.5. Số nguyên tố

- ***Định nghĩa:*** Số nguyên tố là số chỉ chia hết cho 1 và chính nó
- ***Tính chất:***

•Giả sử p là số nguyên tố và p|a.b thì p|a hoặc p|b hoặc cả hai đều chia hết cho p.

•Có vô số số nguyên tố.

1.1.6. Định nghĩa hàm phi Euler

- Định nghĩa: Với n≥1 chúng ta gọi φ (n) là tập các số nguyên tố cùng nhau với n nằm trong khoảng [1,n]
- Tính chất :

• Nếu p là số nguyên tố → φ (p) = p-1

1.1.7. Định nghĩa thặng dư bậc 2

- Định nghĩa: Cho a ∈ Z*n gọi a là thặng dư bậc 2 theo modulo n nếu tồn tại x Z*n sao cho x2≡a(modn) và nếu không tồn tại thì gọi a là bất thặng dư bậc 2 theo modulo n. Tập các thặng dư bậc 2 ký hiệu là Qn và các tập bất thặng dư bậc 2 ký hiệu là Qn .

1.1.8. Số Blum

- Định nghĩa: Số Blum là một hợp tử n=p.q nếu p,q là hai số nguyên tố khác nhau và đồng dư với 3mod4.11

i.2. TÌM HIỂU MẬT MÃ

i.2.1. Giới thiệu

Mật mã đã được sử dụng từ rất sớm, khi con người biết trao đổi thông tin cho nhau và trải qua bao nhiêu năm nó đã được phát triển từ những hình thức sơ khai cho đến hiện đại và tinh vi. Mật mã được sử dụng trong rất nhiều lĩnh vực của con người và các quốc gia, đặc biệt trong các lĩnh vực quân sự, chính trị, ngoại giao và thương mại. Mục đích của mật mã là tạo ra khả năng trao đổi thông tin trên một kênh thông tin chung cho những đối tượng cùng tham gia trao đổi thông tin và không muốn một đối tượng thứ ba khác biết được những thông tin mà họ trao đổi.

Khi một đối tượng A muốn gửi một thông điệp cho những người nhận, A sẽ phải mã hóa thông điệp và gửi đi, những người nhận được thông điệp mã hóa muốn biết được nội dung thì phải giải mã thông điệp mã hóa. Các đối tượng trao đổi thông tin cho nhau phải thỏa thuận với nhau về cách thức mã hóa và giải mã, quan trọng hơn là khóa mật mã đã sử dụng trong quá trình mã hóa và giải mã, nó phải tuyệt đối được giữ bí mật. Một đối tượng thứ ba mặc dù có biết được nhưng sẽ không biết được nội dung thông điệp đã mã hóa.

Có hai phương pháp mã hóa dữ liệu là Mã hóa khóa đối xứng và Mã hóa khóa công khai.

i.2.2. Sơ đồ hệ thống mật mã

Là một bộ năm (P, C, K, E, D) trong đó:

+ P là một tập hữu hạn các bản rõ.
+ C là một tập hữu hạn các bản mã.
+ K là một tập hữu hạn các khoá.
+ Với mỗi k ϵ K, có một hàm lập mã e_k ϵ E

$$e_k : P \rightarrow C$$

và một hàm giải mã d_k ϵ D

$$d_k : C \rightarrow P \text{ sao cho } d(e(x)) = x \text{ với mọi } x_k \epsilon P_k$$

i.2.3. Mật mã khóa đối xứng

Phương pháp mã hóa đối xứng (symmetric cryptography) còn được gọi là mã hóa khóa bí mật (secret key cryptography). Với phương pháp này, người gửi và người

nhận sẽ dùng chung một khóa để mã hóa và giải mã thông điệp. Trước khi mã hóathông điệp gửi đi, hai bên gửi và nhận phải có khóa chung và phải thống nhất thuật toán dùng để mã hóa và giải mã. Có nhiều thuật toán ứng dụng cho mã hóa khóa bí mật DES - Data Encrytion Standard, 3DES - triple-strength DES, RC2 - Rons Cipher 2 và RC4, 5.I4.2.. và sơ khai nhất là các hệ mật mã cổ điển.

Nhược điểm chính của phương pháp này là khóa được truyền trên kênh an toàn nên chi phí tốn kém và không kip thời. Ưu điểm là tốc độ mã hóa và giải mã rất nhanh.

Một số hệ mật mã cổ điển

Định nghĩa: Mã dịch chuyển: (P, C, K, E, D)

P = C = K = Z_{20} với k ϵ K, định nghĩa $e_k(x) = (x + k) \bmod 26$ $d_k(y) = (y – k) \bmod 26$ (x, y ϵ Z_{26})

Ví dụ: Dùng khoá k = 9 để mã hoá dòng thư: “toinaydichoi” dòng thư đó tương ứng với

dòng số

t	o	i	n	a	y	d	i	c	h	o	i
19	14	8	12	0	24	3	8	2	7	14	8

qua phép mã hoá e_9 sẽ được:

2	23	17	22	9	7	12	17	11	16	23	17
c	x	r	w	j	h	m	r	l	q	x	r

bản mã sẽ là: “qnwcxrcqdkjh”

Nhận được bản mã đó, dùng d9 để nhận được bản rõ. Cách đây 2000 năm mã dịch chuyển đã được Julius Ceasar sử dụng, với khoá k=3 mã địch chuyển được gọi là mã Ceasar. Tập khoá phụ thuộc vào Zm với m là số khoá có thể. Trong tiếng Anh tập khoá chỉ có 26 khoá có thể, việc thám mã có thể được thực hiện bằng cách duyệt tuần tự 26 khoá đó, vì vậy độ an toàn của mã dịch chuyển rất thấp.

i.**2.3.1 Mã thay thế**

Định nghĩa Mã thay thế: (P, C, K, E, D)

P = C = Z_{26} , K = S (Z) Với mỗi π ϵ K, tức là một hoán vị trên Z_{26} , ta xác định

$e_\pi(x) = \pi(x)$

$d_\pi(y) = \pi^{-1}(y)$

với x, y є Z_{26} , π^{-1} là nghịch đảo của л

Ví dụ: π được cho bởi (ở đây ta viết chữ cái thay cho các con số thuộc Z_{26}):

a	b	c	d	e	f	g	h	i	j	k	l	m	n
x	n	y	a	h	p	o	g	z	q	w	b	t	s

o	p	q	r	s	t	u	v	w	x	y	z
f	l	r	c	v	m	u	e	k	j	d	i

bản rõ: "toinaydichoi"

sẽ được mã hoá thành bản mã (với khoá π): "mfzsxdazygfz"

Dễ xác định được π^{-1} , và do đó từ bản mã ta tìm được bản rõ.

Mã thay thế có tập hợp khoá khá lớn - bằng số các hoán vị trên bảng chữ cái, tức số các hoán vị trên Z26 , hay là 26! > 4.1110 26 . Việc duyệt toàn bộ các hoán vị để thám mã là rất khó, ngay cả đối với máy tính. Tuy nhiên, bằng phương pháp thống kê, ta có thể dễ dàng thám được các bản mã loại này, và do đó mã thay thế cũng không thể được xem là an toàn

i.2.3.2 Mã Anffine:

Định nghĩa Mã Anffine: (P, C, K, E, D)

$$P = C = Z_{26}, K = \{ (a, b) \in Z_{26} \times Z_{26} : (a, 26) = 1 \}$$

với mỗi k = (a, b) є K ta định nghĩa:

$$e_k(x) = ax + b \bmod 26$$

$$d_k(y) = a\ (y - b) \bmod 26$$

Trong đó x,y є $Z_{26.}$

Ví dụ: Lấy k = (5, 6).

Bản rõ:

"toinaydichoi"

	t	o	i	n	a	y	d	i	c	h	o	i
x	19	14	8	13	0	14	3	8	2	7	14	8

y=5x + 6 mod 26

y	23	24	20	19	6	24	21	20	16	15	24	20
	x	y	u	t	g	y	v	u	q	p	y	u

Bản mã: "xyutgyvuqpyu"

Thuật toán giải mã trong trường hợp này có dạng: $d_k(y) = 21(y - 6) \bmod 26$

Với mã Apphin, số các khoá có thể có bằng (số các số ≤ 26 và nguyên tố với 26) × 26, tức là 12 × 26 = 312. Việc thử tất cả các khoá để thám mã trong trường hợp này tuy khá mất thì giờ nếu tính bằng tay, nhưng không khó khăn gì nếu dùng máy tính. Do vậy, mã Apphin cũng không phải là mã an toàn.

i.2.3.2. Mã Vigenère:

Định nghĩa Mã Vigenere: (P, C, K, E, D)

Cho m là số nguyên dương. $P = C = K = Z_{26}^m$

với mỗi khoá $k = (k_1, k_2, \ldots, k_m) \in K$ có:

$$e_k(x_1, x_2, \ldots, x_m) = (x_1 + k_1, x_2 + k_2, \ldots, x_m + k_m)$$

$$d(y_1, y_2, \ldots, y_m) = (y_1 - k_1, y_2 - k_2, \ldots, y_m - k_m)$$

các phép cộng phép trừ đều lấy theo modulo 26

Ví dụ: Giả sử m = 6 và khoá k là từ CIPHER - tức k=(2, 8, 15, 7, 4, 17).

Bản rõ:

"toinaydichoi"

	t	o	i	n	a	y	d	i	c	h	o	i
x	19	14	8	13	0	24	3	8	2	7	14	8
k	2	8	15	7	4	17	2	8	15	7	4	17
y	21	22	23	20	4	15	5	16	17	14	18	25
	v	w	x	u	e	p	f	q	r	o	s	z

Bản mã

"vwxuepfqrosz"

Từ bản mã đó, dùng phép giải mã dk tương ứng, ta lại thu được bản rõ.

Chú ý: Mã Vigenere với m = 1 sẽ trở thành mã Dịch chuyển.

Tập hợp các khoá trong mã Vigenere mới $m \geq 1$ có tất cả là $26m$ khoá có thể có.

Với m = 6, số khoá đó là 308.915.776, duyệt toàn bộ chừng ấy khoá để thám mã bằng tính tay thì khó, nhưng với máy tính thì vẫn là điều dễ dàng.

i.2.3.3. Mã Hill:

Định nghĩa Mã Hill: (P, C, K, E, D)

Cho m là số nguyên dương. $k = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$

P = C = Z_{26m}

K = { k є Z_{26mxm} : (det(k), 26) = 1 }

với mỗi k є K định nghĩa: $k^{-1} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1}$

$e_k (x_1 , x_2 ,\ldots, x_m) = (x_1 , x_2 ,\ldots, x_m).k$

$d_k (y_1 , y_2 ,\ldots, y_m) = (y_1 , y_2 ,\ldots, y_m).k^{-1}$

Ví dụ: Lấy m = 2, và $k = \begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1} \begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} \frac{d}{ad-bc} & \frac{-b}{ad-bc} \\ \frac{-c}{ad-bc} & \frac{a}{ad-bc} \end{pmatrix} \begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$

Với bộ 2 ký tự (x_1, x_2), ta có mã là (y_1, y_2) = (x_1 , x_2).k được tính bởi

$Y_1 = 1i.x_1 + 3.x_2$

$Y_2 = 8.x_1 + 7.x_2$

Giả sử ta có bản rõ: "**tudo**", tách thành từng bộ 2 ký tự, và viết dưới dạng số ta được 19 20 | 03 14 , lập bản mã theo quy tắc trên, ta được bản mã dưới dạng số là: 09 06 | 2318, và dưới dạng chữ là "**fgxs**".

Chú ý: Để đơn giản cho việc tính toán, thông thường chọn ma trận vuông 2×2. Khi đó có thể tính ma trận nghịch đảo theo cách sau :

Giả sử ta có :

Ta có ma trận nghịch đảo

Và được tính như sau :

Một chú ý là để phép chia luôn thực hiện được trên tập Z26 thì nhất thiết định thức của k : det(k) = (ad – bc) phải có phần tử nghịch đảo trên Z26, nghĩa là (ad – bc) phải là một trong các giá trị : 1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23, hoặc 25. Đây cũng là điều kiện để ma trận k tồn tại ma trận nghịch đảo.

Khi đó: k^{-1} .k = I là ma trận đơn vị (đường chéo chính bằng 1)

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1}\begin{pmatrix} a & b \\ c & d \end{pmatrix}=\begin{pmatrix} \frac{d}{ad-bc} & \frac{-b}{ad-bc} \\ \frac{-c}{ad-bc} & \frac{a}{ad-bc} \end{pmatrix}\begin{pmatrix} a & b \\ c & d \end{pmatrix}=\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Định thức của : $\begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix}$

Là 11*7 – 8*3 = 1 ≡ 1 mod 26

Khi đó $\begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix}^{-1}=\begin{pmatrix} 7 & -8 \\ -3 & 11 \end{pmatrix}\equiv\begin{pmatrix} 7 & 18 \\ 23 & 11 \end{pmatrix} mod\ 26$

i.2.3.4. Mã hoán vị:

Định nghĩa Mã hoán vị: (P, C, K, E, D) :Cho m là số nguyên dương.

$P=C=Z_{26}$,$K=S_m$

với mỗi k = π ϵ S_m , ta có

$$\begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix}^{-1}=\begin{pmatrix} 7 & -8 \\ -3 & 11 \end{pmatrix}\equiv\begin{pmatrix} 7 & 18 \\ 23 & 11 \end{pmatrix} mod\ 26$$

trong đó π^{-1} là hoán vị nghịch đảo của π

Ví dụ: Giả sử m = 6, và khoá k được cho bởi phép hoán vị π

1	2	3	4	5	6
3	5	1	6	4	2

Khi đó phép hoán vị nghịch đảo π^{-1} là:

1	2	3	4	5	6
3	6	1	5	2	4

Bản rõ:

"toinaydichoi"

	t	o	i	n	a	y	d	i	c	h	o	i
vt	1	2	3	4	5	6	1	2	3	4	5	6
π	1->3	2->5	3->1	4->6	5->4	6->2	1->3	2->5	3->1	4->6	5->4	6->2
vt	3	5	1	6	4	2	3	5	1	6	4	2
	i	a	t	y	n	o	c	o	d	i	h	i

Bản mã: "iatynocodihi"

Dùng hoán vị nghịch đảo, từ bản mật mã ta lại thu được bản rõ.

Chú ý:

Mã hoán vị là một trường hợp riêng của mã Hill. Thực vậy, cho phép hoán vị π của {1, 2,…, m}, ta có thể xác định ma trận $K_\pi = (k_{ij})$, với

$$k_{ij} = \begin{cases} 1 & \text{nếu } \quad i = \pi(j) \\ 0 & \text{nếu ngược lại} \end{cases}$$

Thì dễ thấy rằng mã Hill với khoá K_π trùng với mã hoán vị với khoá π.

Với m cho trước, số các khoá có thể có của mã hoán vị là m!
Dễ nhận thấy với m = 26 ta có số khóa 26! (mã Thay thế).

i.2.3.5. Mã khóa công khai:

Phương pháp mã hóa khóa công khai (public key cryptography) còn được gọi là mã
hóa bất đối xứng (asymmetric cryptography) đã giải quyết được vấn đề của phương pháp mã hóa khóa bí mật (đối xứng) là sử dụng hai khóa: khóa bí mật (pr4.11Iate key) và (public key). Khóa bí mật được giữ kín, trong khi đó được gửi công khai bởi vì tính chất khó tính được khóa bí mật từ khóa công khai. Khóa công khai và khóa bí mật có vai trò trái ngược nhau, một khóa dùng để mã hóa và khóa kia sẽ dùng để giải mã.

Hiện nay các hệ mật mã khóa công khai đều dựa trên hai bài toán "khó" là bài toán logarith rời rạc trên trường hữu hạn và bài toán tìm ước số nguyên tố.
Phương pháp cho phép trao đổi khóa một cách dễ dàng và tiện lợi. Nhưng tốc độ mã hóa khá chậm hơn rất nhiều so với phương pháp mã hóa khóa đối xứng rất nhiều, Tuy nhiên, hệ mật mã khóa công khai có một ưu điểm nổi bật là cho phép tạo chữ ký điện tử.

Một số hệ mật mã khóa công khai

i.2.3.6. Mã RSA:

Hệ mật này sử dụng tính toán trong Zn, trong đó n là tích của 2 số nguyên tố phân biệt p và q. Ta thấy rằng $\varphi(n) = (p - 1).(q - 1)$.
Định nghĩa
Cho n = p.q trong đó p và q là các số nguyên tố. Đặt P = C = Zn và định nghĩa:
$K = \{(n, p, q, a, b): n = p.q; p, q$ là các số nguyên tố, $a.b \equiv 1 \bmod \varphi(n)\}$
Với K = (n, p, q, a, b) ta xác định: $eK = x^b \bmod n$ và $dK = y^a \bmod n$ $(x, y \in Zn)$ Các giá trị n và b được công khai và các gia trị p, q, a được giữ kínVí dụ:
Chọn p = 2, q = 5. Tính n = p.q = 2*5 = 10
$\varphi(n) = (p - 1).(q - 1) = 1*4 = 4$
Do $UCLN(\varphi(n), b) = 1$ nên chọn b = 3
$a.b \equiv 1 \bmod \varphi(n)$ nên chọn a = 7
Giả sử G muốn gửi bản rõ x = 3 tới N, G phải tính:
$y = eK = x^b \bmod n = 33 \bmod 10 = 7$
Khi N nhận được bản mã y = 1, anh ta sử dụng số mũ a mật để tính:
$x = dK = y^a \bmod n = 77 \bmod 10 = 3$
Đó chính là bản rõ mà G đã mã hoá.
Độ mật của hệ RSA được dựa trên giả thiết là hàm mã $eK = x^b \bmod n$ là hàm mộtchiều. Bởi vậy thám mã sẽ khó có khả năng về mặt tính toán để giải mã một bản mã

Cửa sập cho phép N chính là thông tin về phép phân tích thừa số n (n = p.q). Vì N biết phép phân tích này nên anh ta có thể tính φ(n) = (p – 1).(q – 1) và rồi tính số mũ giải mã a bằng cách sử dụng thuật toán Eculide mở rộng.

i.2.3.7. Mã Elgamal:

Mô tả hệ mã Elgamal

Hệ mật mã ElGamal được T. ElGamal đề xuất năm 1985, dựa vào độ phức tạp của bài toán tính lôgarit rời rạc, và sau đó đã nhanh chóng được sử dụng rộng rãi không những trong vấn đề bảo mật truyền tin mà còn trong các vấn đề xác nhận và chữ ký điện tử.

Bài toán logarithm rời rạc trong Zp là đối tượng trong nhiều công trình nghiên cứu và được xem là bài toán khó nếu p được chọn cẩn thận. Cụ thể là không có một thuật toán thời gian đa thức nào cho bài toán logarithm rời rạc. Để gây khó khăn cho các phương pháp tấn công đã biết, p phải có ít nhất 150 chữ số và (p – 1) phải có ít nhất một thừa số nguyên tố lớn

Hệ mật Elgamal là một hệ mật không tất định vì bản mã phụ thuộc vào cả bản rõ x lẫn giá trị ngẫu nhiên k do G chọn. Bởi vậy sẽ có nhiều bản mã được mã từ cùng một bản rõ.

Bài toán logarithm rời rạc trong Zp:

Đặc trưng của bài toán: I = (p, α, β) trong đó p là số nguyên tố, α ∈ Zp là phần tử nguyên thuỷ (hay phần tử sinh), β ∈ Zp*

Mục tiêu: Hãy tìm một số nguyên duy nhất a, $0 \leq a \leq p - 2$ sao cho:

$\alpha a \equiv \beta \pmod p$

Ta sẽ xác định số nguyên a bằng log α β.

Định nghĩa mã khóa công khai Elgamal trong Zp*:

Cho p là số nguyên tố sao cho bài toán logarithm rời rạc trong Zp là khó giải . Cho α ∈ Zp* là phần tử nguyên thuỷ. Giả sử P = Zp*, C = Zp* x Zp*. Ta định nghĩa K = {(p, α, a, β): $\beta \equiv \alpha a \pmod p$}Các giá trị p, α, β được công khai, còn a giữ kín.

Với K =(p, α, a, β) và một số ngẫu nhiên bí mật k ∈ Zp – 1, ta xác định:

eK(x, k) = (y1, y2).

Trong đó: $y1 = \alpha k \bmod p$

$y2 = x.\ \beta k \bmod p$

với y1, y2 ∈ Zp* ta xác định:

dK(y1, y2) = y2(y1a) – 1 mod p

Ví dụ:

Chọn p = 7, α ∈ Zp* là phần tử nguyên thuỷ nên α = 3

Chọn a sao cho $0 \leq a \leq p - 2$ nên a = 2

Khi đó : $\beta = \alpha a \bmod p = 32 \bmod 7 = 2$

Chọn một số ngẫu nhiên bí mật k ∈ Zp – 1, chọn k =3

Giả sử G muốn gửi thông báo x = 3 cho N, G phải tính:

eK(x, k) = (y1, y2)

trong đó:

y1 = αk mod p = 33 mod 7 = 6

y2 = x. βk mod p = 3*23 mod 7 = 3

Khi N thu được bản mã (y1, y2) = (6, 3), anh ta sẽ tính:

x = dK(y1, y2) = y2(y1a)-1 mod p = 3*(62)-1 mod 7 = 3

Đó chính là bàn rõ mà G đã mã hoá.

i.3. CHỮ KÝ SỐ

i.3.1. Giới thiệu chung về chữ ký số:

Như chúng ta đã biết, chữ ký viết tay "thường lệ" gắn với tài liệu được dùng để chỉ ra người đã ký nó. Chữ ký được sử dụng hàng ngày như viết thư, ký hợp đồng…

Ở đây chúng ta tìm hiểu về chữ ký hoàn toàn khác đó là chữ ký số. Nó là phương pháp ký thông báo được lưu dưới dạng điện tử và thông báo được ký có thể truyền trên mạng máy tính. Chữ ký tay và chữ ký số dù có chung nhiệm vụ là ký nhưng có sự khác biệt cơ bản giữa chúng.

Thứ nhất, về việc ký tài liệu: với chữ ký viết tay thì chũ ký là bộ phận vật lý của tài liệu được ký. Tuy nhiên, chữ ký số không một cách vật lý với thông báo được ký mà được gắn với thông báo theo logic, do đó thuật toán được dùng phải "trói " chữ ký với thông báo theo một cách nào đó.

Thứ hai, về việc kiểm tra: chữ ký tay được kiểm tra bằng cách so sánh nó với những cái khác những chữ ký đã được xác thực. Ví dụ, một người ký một tấm séc mua hàng, người bán hàng phải so sánh chữ ký trên tấm séc với chữ ký nằm sau thẻ tín dụng để kiểm tra. Tuy nhiên, phương pháp này không an toàn lắm vì nó tương đối dễ đánh lừa bởi chữ ký của người khác. Khác với chữ ký tay, chữ ký số có thể được kiểm tra bằng cách dùng thuật toán kiểm tra công khai đã biết. Vì vậy bất kì người nào đều có thể kiểm tra chữ ký số, và việc sử dụng lược đồ ký an toàn sẽ ngăn chặn khả năng đánh lừa.

Điều khác nhau cơ bản giữa chữ ký tay và chữ ký số là "bản sao" thông báo số được ký là đồng nhất với bản gốc. Trong khi đó, bản sao tài liệu giấy đã ký thường là khác với bản gốc. Điều này có nghĩa là phải cẩn thận để ngăn chặn thông một thông báo đã ký số bị sử dụng lại. Ví dụ, nếu A ký thông báo số cho B rút 1000$ từ tài khoản trong ngân hàng của mình, A chỉ muốn B làm điều đó 1 lần. Do đó, thông báo đó phải chứa thông tin để ngăn chặn B làm lại việc đó nhiều lần.

Lược đồ chữ ký gồm hai thành phần: một thuật toán ký và một thuật toán kiểm tra. A có thể ký thông báo x nhờ thuật toán ký(bí mật) Sig. Chữ ký thu được Sig(x) sau đó

có thể được kiểm tra bằng thuật toán kiểm tra công khai Ver. Khi cho cặp(x,y) thuật toán kiểm tra trả lời "đúng" hoặc "sai" phụ thuộc vào việc ký có đích thực không?

i.3.2. Định nghĩa lược đồ chữ ký:

Lược đồ chữ ký là một bộ năm phần tử (P,A,K,S,V) thỏa mãn các điều kiện sau:

i. P _ là một tập hữu hạn các thông báo.

2. A _ tập hữu các chữ ký có thể.

3. K _ tập hữu hạn các khóa, không gian khóa.

4.11 Với mỗi k ∈ K, ∃ sigk ∈ S và verk ∈ V

Mỗi sigk: P→ A, verk: P * A→ {true, false}là những hàm sao cho mỗi bức điện x ∈ Pvà mỗi chữ ký y ∈ A thỏa mãn:

$$\text{Ver(x,y)} = \begin{cases} true, & khi \quad y = sig(x) \\ false, & khi \quad y \neq sig(x) \end{cases}$$

Yêu cầu:

- Với mỗi k ∈ K, các hàm sigk và verk là các hàm thời gian đa thức

-

Verk là hàm công khai, sigk là hàm bí mật tránh trường hợp một người B nào đó có thể giả mạo chữ ký của chủ thể A để ký thông báo. Với mỗi x chỉ duy nhất A tính đ ược

chữ ký y sao cho:

Ver(x,y)= True

Lược đồ chữ ký phải an toàn. Bởi vì người thám mã B có thể kiểm tra tất cả các khả năng của chữ ký y nhờ thuật toán kiểm tra công khai Ver cho tới khi đạt được yêu cầu tức là tìm được chữ ký đúng. Do đó, nếu đủ thời gian cần thiết thì B có thể giả mạo được chữ ký của A. Vì vậy, mục đích của chúng ta là tìm các lược đồ chữ ký sao cho B không đủ thời gian thực tế để thử như thế.

i.3.2.1. Lược đồ chữ ký RSA:

Lược đồ chữ ký RSA được định nghĩa như sau:

·Tạo khóa:

Sơ đồ chữ ký cho bởi bộ năm (P,A,K,S,V)

Cho n=p.q; với mỗi p,q là các số nguyên tố lớn khác nhau φ(n) = (p - 1)(q - 1).

Cho P = A = Zn và định nghĩa:

K là tập các khóa, K=(K',K''); với K'=a; K''=(n,b)

a,b ∈ Zn*, thỏa mãn ab ≡ 1mod φ(n).

Các giá trị n,b là công khai, các giá trị p,q,a là các giá trị bí mật.

·Tạo chữ ký:

Với mỗi K=(n.p,q,a,b) xác định:

SigK’(x)= xa mod n

·Kiểm tra chữ ký:

VerK’’(x,y)= true ⇔x ≡ yb mod n; x, y ∈ Zn.

Giả sử A muốn gửi thông báo x, A sẽ tính chữ ký y bằng cách :

y=sigK’(x)= xa mod n (a là tham số bí mật của A)

A gửi cặp (x,y) cho B. Nhận được thông báo x, chữ ký số y, B bắt đầu tiến hànhkiểm tra đẳng thức

x=ybmod(n)(b là khóa công khai A)

Nếu đúng, B công nhận y là chữ ký trên x của A. Ngược lại, B sẽ coi x không phải của A gửi cho mình (chữ ký không tin cậy).

Người ta có thể giả mạo chữ ký của A như sau: chọn y sau đó tính x= verK’’(y), khi đó y= sigK’(x). Một cách khắc phục khó khăn này là việc yêu cầu x phải có nghĩa. Do đó chữ ký giả mạo thành công với xác suất rất nhỏ. Ta có thể kết hợp chữ ký với mã hóa làm cho độ an toàn tăng thêm.

Giả sử trên mạng truyền tin công cộng, ta có hai hệ mật mã khóa công khai δ1 và hệ xác nhận chữ ký δ2. Giả sử B có bộ khóa mật mã K=(K’,K’’) với K’=(n,e) và K’’=d trong hệ δ1, và A có bộ khóa chữ ký Ks=(Ks’,Ks’’) với Ks’= a và Ks’’=(n,b) trong hệ δ2. A có thể gửi đến B một thông báo vừa bảo mật vừa có chữ ký xác nhận như sau: A tính chữ ký của mình là: y= sigA(x), và sau đó mã hóa cả x và y bằng cách sử dụng mật mã công khai eB của B, khi đó A nhận được z= eB(x,y), bản mã z sẽ được gửi tới B. khi nhận được z việc trước tiên B phải giải mã bằng hàm dB để nhận được (x,y). Sau đó B sử dụng hàm kiểm tra công khai của A để kiểm tra xem verA(x,y)= true? Tức là kiểm tra xem chữ ký đó có đúng là của A?.

Ví dụ:

A dùng lược đồ chữ ký số RSA với n=247,(p=13,q=19);

φ(n) = 12.18 = 216. Khóa công khai của A là b=7.

⇒ a = 7-1mod216 = 3i.

A công khai (n,b) = (247,7)

A ký trên thông báo x=100 với chữ ký:

y = xa modn = 10031 mod247 = 74.11

A gửi cặp (x,y) = (100,74) cho B, B kiểm tra bằng cách sử dụng khóa công khai của A như sau:

x = yb modn = 747 mod247 = 100 = x.

B chấp nhận y=74 là chữ ký tin cậy.

i.3.2.2. Lược đồ chữ ký ElGamal:

Lược đồ chữ ký ElGamal được giới thiệu năm 1985 và được Viện tiêu chuẩn và Công nghệ quốc gia Mỹ sửa đổi thành chuẩn chữ ký số. Lược đồ chữ ký ElGammal không tất định cũng giống như hệ mã hóa ElGamal. Điều này có nghĩa là có nhiều chữ ký hợp lệ cho một thông báo bất kỳ. Thuật toán kiểm tra phải có khả năng khả năng chấp nhận bất kỳ chữ ký hợp lệ nào khi xác minh.

Lược đồ chữ ký ElGamal được định nghĩa như sau:

· Tạo khóa:

Cho p là số nguyên tố sao cho bài toán logarit rời rạc trong Zp là khó và giả sử $\alpha \in Z$ p là phần tử nguyên thủy

Cho P = Z *p , A = Z *p × Zp-1 và định nghĩa

K = {(p, a, α, β): β = αa modp }.

Các giá trị p, α, β là công khai, a là bí mật.

· Tạo chữ ký

Với K = (p, a, α, β) và với số ngẫu nhiên $k \in Z^*p-1$,

định nghĩa sigk(γ, δ), trong đó:

γ = αk modp và δ = (x - aγ) k -1mod(p - 1).

· Kiểm tra chữ ký số

Với x, $\gamma \in$ Z *p và $\delta \in$ Zp-1 , ta định nghĩa :

Ver (x, γ, δ) = True ⇔βγ. γδ ≡ αx modp.

Chứng minh:

Nếu chữ ký được thiết lập đúng thì hàm kiểm tra sẽ thành công vì:

⇒βγ γδ ≡ αa.γ αr.δ modp

≡ αx modp (vì aγ + rδ ≡ x mod(p - 1)).

A tính chữ ký bằng cách dùng cả giá trị bí mật a(là một phần của khóa) lẫn số ngẫu nhiên bí mật k (dùng để ký trên x). Việc kiểm tra có thể thực hiện duy nhất bằng thông tin công khai

Ví dụ: Giả sử p=467, α = 2, a = 127

Khi đó: β = αa modp = 2127mod467 = 132

Giả sử A có thông báo x=100 và A chọn ngẫu nhiên k=213 vì (213,466)=1 và

213-1 mod466 = 431, A ký trên x như sau:

γ = αk modp = 2213mod467 = 29

Và δ = (x - aγ)k-1 mod(p -1) = (100 – 127. 29).431 mod466 = 5i.

Chữ ký của A trên x= 100 là (29,51).

Bất kỳ người nào đó cũng có thể kiểm tra chữ ký bằng cách:

13229 . 2951 ≡ 189 mod 467

2100 ≡ 189 mod 467

Do đó, chữ ký là tin cậy.

i.4.Hàm Hash

i.4.1. Giới thiệu:

Đối với xác thực và chữ ký số ta thấy rằng các thuật toán thường nhận đầu vào là các dòng bit có độ dài rất ngắn (6i.128.160 bit) và có tốc độ thực hiện chậm. Mặt khác, các thông báo ký thường có độ dài khác nhau và trong trường hợp chúng có độ dài lớn cỡ vài Kilôbyte hoặc và Megabyte. Do vậy, muốn ký trên một thông báo dài ta phải cắt thông báo ra nhiều đoạn có độ dài hữu hạn và cố định rồi tiến hành ký độc lập từng đoạn và gửi từng đoạn đó đi, khi đó lại xuất hiện một vấn đề như:- Tốc độ sẽ chậm vì phải ký trên quá nhiều đoạn.

- Dễ xảy ra trường hợp không sắp xếp được thông báo theo đúng trật tự ban đầu.
- Có thể bị mất các đoạn riêng biệt trong quá trình truyền tin.

Để giải quyết vấn đề này ta dùng hàm Hash. Hàm Hash chấp nhận một thông báo c ó

độ dài bất kỳ làm đầu vào, Hàm Hash sẽ biến đổi thông báo này thành một thông b áo rút

gọn, sau đó sẽ sử dụng lược đồ chữ ký để ký trên thông báo rút gọn.

Ta có mô hình chung như sau:

Thông báo	X		độ dài tùy ý
	↓		
Thông báo rút gọn	z = h(x)		160 bit
	↓		
Chữ ký		y = sigK(x)	320 bit

Ta sẽ gửi cặp (x,y) cho người nhận. Nếu cần giữ bí mật x thì ta mã hóa x thành x’ rồi sau đó gửi cặp (x’,y).

i.4.2. Định nghĩa:

Hàm Hash là hàm tính toán có hiệu quả khi ánh xạ các dòng nhị phân có độ dài tùy ý

thành những dòng nhị phân có độ dài cố định nào đó.

-Hàm Hash yếu: hàm Hash gọi là yếu nếu cho một thông báo x thì về mặt tính toán không tìm ra được thông báo x’ khác x sao cho:

$h(x') = h(x)$

-Hàm Hash mạnh: hàm Hash được gọi là mạnh nếu về mặt tính toán không tìm ra được hai thông báo x và x’ sao cho:

$x1 \neq x2$ và $h(x1) = h(x2)$

Nói cách khác, tìm hai văn bản khác nhau có cùng một đại diện là cực kỳ khó

Hàm Hash phải là hàm một phía, nghĩa là cho x tính z = h(x) thì dễ, nhưng ngược lại, biết z tính x là công việc cực khó.

Hàm Hash yếu làm cho chữ ký trở lên tin cậy giống như việc ký trên toàn thông báo.

Hàm Hash mạnh có tác dụng chống lại kẻ giả mạo tạo ra hai bản thông báo có nội dung khác nhau, sau đó thu nhận chữ ký hợp pháp cho một bản thông báo dễ được xác nhận rồi lấy nó giả mạo làm chữ ký củathông báo thứ 2 hay nói cách khác tìm 2 văn bản khác nhau có cùng một đại diện là cực kỳ khó.

i.4.2.1. Một số hàm Hash sử dụng trong chữ ký số:

Các hàm Hash đơn giản:

Tất cả các hàm Hash đều được thực hiện theo quy tắc chung là: Đầu vào được biểu diễn dưới dạng một dãy các khối n bit, các khối n bit này được xử lý theo cùng một kiểu và lặp đi lặp lại để cuối cùng cho đầu ra có số bit cố định.

Hàm Hash đơn giản nhất là thực hiện phép toán XOR từng bit một của mỗi khối.

Nó được biểu diễn như sau: $Ci = b1i \oplus b2i \oplus \ldots \oplus bmi$

Trong đó :

C_i : là bit thứ i của mã Hash, $i = \overline{1, n}$

m : là số các khối đầu vào

b_{ji} : là bit thứ i trong khối thứ j

$\oplus$: là phép cộng modulo 2

Sơ đồ hàm Hash sử dụng phép XOR.

Khối 1:	B11	B12	…	B1n
Khối 2:	B21	B22	…	B2n
…	…	…	…	…
Khối m:	Bm1	Bm2	…	Bmn
Mã Hash:	C1	C2	…	Cmn

Ci là bit kiểm tra tính chẵn lẻ cho vị trí thứ i khi ta chia tệp dữ liệu thành từng khối, mỗi khối con vị trí. Nó có tác dụng như sự kiểm tra tổng thể tính toàn vẹn của dữ liệu.

Khi mã hóa một thông báo dài thì ta sử dụng mode CBC (The Cipher Block Chaining), thực hiện như sau:

Giả sử thông báo X được chia thành các khối 64 bit liên tiếp

$$X = X_1X_2 \dots X_n$$

Khi đó mã Hash C sẽ là:

$C = X_{NH} = X_1 \oplus X_2 \oplus \dots \oplus X_n$

Sau đó mã hóa toàn bộ thông báo nối với mã Hash theo mode CBC sản sinh ra bản mã $.Y_1Y_2 \dots Y_{N+1}$

Kỹ thuật khối xích :

Người ta đầu tiên đề xuất kỹ thuật mật mã xích chuỗi nhưng không có khóa bí mật là Rabin.

Kỹ thuật này được thực hiện như sau :

Chia thông báo M thành các khối có cỡ cố định là M1, M2, …, MN, sử dụng hệ mã thuận tiện như DES để tính mã Hash như sau :H_0 = giá trị ban đầu

$$H_i = E_{Mi}(H_{i-1}),\ i = 1, N$$

$$G = H_N$$

Các hàm Hash mở rộng:

Ở trên, ta đề cập đến hàm Hash có nhiều đầu vào hữu hạn. Tiếp theo ta sẽ đề cập tới loại hàm Hash mạnh với đầu vào vô hạn thu được do mở rộng một hàm Hash mạnh có đầu vào độ dài hữu hạn. Hàm này sẽ cho phép ký các thông báo có độ dài tùy ý.

Giả sử h: $(Z2)^m \rightarrow (Z2)^t$ là một hàm Hash mạnh, trong đó $m \geq t + 1$ ta sẽ xây dựng một hàm Hash mạnh :

$$h^*: X \rightarrow (Z_2)_t, \text{ trong đó } X = \cup(Z_2)_i$$

Xét trường hợp $m \geq t + 2$

Giả sử $x \in X$, vậy thì tồn tại n để $x \in (Z_2)^n$, $n \geq m$.

Ký hiệu : |x| là độ dài của x tính theo bit. Khi đó, |x| = n.

Ký hiệu : x || y là dãy bit thu được do nối x với y.

Giả sử $|x| = n \geq m$. Ta có thể biểu diễn x như sau:

$$x = x_1 \,||\, x_2 \,||\, \dots \,||\, x_k$$

Trong đó $|x_1| = |x_2| = \ldots = |x_{k-1}| = m - t - 1$ và $|x_k| = m - t - 1 - d$,

$0 \le d \le m-t-2$

$\Rightarrow |x_k| \ge 1$ và $m - t - 1 \ge 1$, $k \ge 2$.

Thuật toán xây dựng h thành h* được mô tả như sau :

i. Cho i = 1 tới k-1 gán $y_i = x_i$;

2. $y_k = x_k \| 0^d$ (0^d là dãy có d số 0. Khi đó y_k dài m-t-1)

3. y_{k+1} là biểu diễn nhị phân của d ($|y_{k+1}| = m-t-1$)

4.11 $g_1 = h(0^{t+1} \| y_1)$ (g_1 = t, $0^{t+1} \| y_1$ dài m)

5. Cho i=1 tới k thực hiện

$g_{i+1} = h(g_i \| 1 \| y_{i+1})$

a. $h^*(x) = g_{k+1}$

Ký hiệu $y(x) = y_1 \| y_2 \| \ldots \| y_{k+1}$

Ta thấy rằng $y(x) \neq y(x')$ nếu $x \neq x'$

Xét trường hợp m=t+1

Cũng như trên, ta giả sử |x| = n >m

Ta xác định f như sau:

f(0) = 0;

f(1) = 01;

Thuật toán xây dựng h* khi m=t+1 như sau :

i. Cho $y= y_1,y_2, \ldots, y_k =11 \| f(x_1) \| f(x_2) \ldots f(x_n)$ (x_1 là một bit)

2. $g_1 = h(0^t \| y_1)$ (y_1 = m – t)

3. Cho i=1 tới k -1 thực hiện

$g_{i+1} = h(g_i \| y_{i+1})$ (y_i = m – t - 1)

4.11 $h^*(x) = g_{k^*}$

Ngoài ra còn có một số hàm Hash khác như hàm Hash MD4 và hàm Hash MD5.

Chương2 CHỮ KÝ SỐ CHỐNG CHỐI BỎ

2.1. Giới thiệu:

Chữ ký không chối bỏ được công bố bởi Chaum và Van Antverpen vào năm1989. Nó có một nét riêng mới lạ và thú vị. Quan trọng nhất trong số đó là chữ ký khôngthể kiểm tra khi không có sự cộng tác của người ký, A(giả sử người ký là A).

Sự bảo vệ này của A đề phòng khả năng chữ ký trong tài liệu của anh ta bị sao chép và phân bố bởi thiết bị điện tử mà không có sự đồng ý của anh ta.

Ví dụ: A có một phần mềm và chữ ký kèm theo được tạo ra nhờ thuật toán của chữ ký số thông thường. Như vậy, sẽ không tránh khỏi trường hợp phần mềm đó bị sao chép mà B không biết. Người mua sẽ kiểm tra chữ ký kèm theo nhờ thuật toán kiểm tra công khai Ver và công nhận chữ ký đó là đúng. Vì như chúng ta đã biết bản sao của chữ ký số đồng nhất với bản gốc. Đương nhiên như vậy A sẽ bị mất bản quyền. Để tránh điều bất tiện đó A đã dùng chữ ký không chối bỏ. Sự kiểm tra sẽ thành công khi thực hiện giao thức hỏi - đáp.

Lược đồ chữ ký chống chối bỏ gồm 3 phần: thuật toán ký, giao thức kiểm tra, giao thức chối bỏ.

2.2. Lược đồ chống chối bỏ:

2.2.1. Thuật toán ký:

* Tạo khóa:

Cho p,q là các số nguyên tố lẻ sao cho p=2q+1 và bài toán rời rạc trên Z_p là khó. Lấy

$\alpha \in Z_{p^*}$ là một phần tử bậc q(Nếu α_0 là phần tử nguyên thủy của Z_p thì

$\alpha = \alpha_{0(p-1)/q}$ modp) lấy $1 \le a \le q-1$ và xác định: $\beta = \alpha^a$ modp.

Lấy G là phân nhóm nhân của Z_{*p} bậc q (G bao gồm các thặng dư bậc hai theo modun p).

Lấy P=A=G, xác định:

$K = \{ (p, \alpha, a, \beta): \beta = \alpha^a \text{ modp}\}$

Các giá trị p, α, β là công khai, a là bí mật.

* Tạo chữ ký:

Với K= (p, α, a, β) và x ∈ G, xác định chữ ký y trên thông báo x:

$y = sig_k(x) = x^a$ modp

2.2.2 Giao thức kiểm tra :

Với x, y ∈ G, sự kiểm tra được tiến hành theo giao thức sau :

1. A chọn e_1,e_2 ngẫu nhiên, e_1, $e_2 \in Z_{p^*}$.
2. A tính $c = y^{e_1} \beta^{e_2}$ modp gửi nó cho B.
3. B tính $d = c^{a^{-1} \text{modq}}$ modp và gửi nó cho A.

4. A chấp nhận chữ đúng khi và chỉ khi :

$d \equiv x_e \square_e \text{ mod} p.$ (*)

* Vai trò của p, q trong lược đồ:

Lược đồ nằm trong Zp; tuy nhiên chúng ta cần tính toán trong phân nhóm nhân G của Zp* của bậc nguyên tố lẻ. Đặc biệt, chúng ta cần tính phần tử nghịch đảo theo modun |G|, điều này lý giải tại sao |G| nên là nguyên tố lẻ. Nó thuận tiện lấy p=2q+1 với q là số nguyên tố lẻ. Trong trường hợp này, phân nhóm G tồn tại.

Ví dụ: giả sử ta lấy p = 467, từ 2 là căn nguyên thủy => 22 = 4 là thặng dư bậc hai theo modun 267 và 4 là phần tử sinh của G, lấy α = 4.11 Giả sử a=101, ta có:

$\beta = \alpha^a \text{mod} p = 4^{101} \text{ mod} 467 = 449$

A sẽ ký thông báo x=119 với chữ ký:

$y = x^a \text{ mod} p = 119^{101} \text{ mod} 467 = 129$

Giả sử B muốn kiểm tra chữ ký y, B chọn ngẫu nhiên $e_1 = 38, e_2 = 397$.

Ta có: $c = y^{e_1} \beta^{e_2} \text{ mod} p = 129^{38}\ 449^{397} \text{ mod} 467 = 13$

B gửi c=13 cho A và A tính d theo:

$d = c^{a^{-1} \text{mod} q} \text{ mod} p$.

$\rightarrow d = 13^{101^{-1} \text{mod} 233} \text{mod} 467 (q = (p - 1)/2 = (467 - 1)/2 = 233).$

$\rightarrow d = 9.$

B muốn kiểm tra chữ ký y theo bước 4.11 Có:

$x_e \square_e \text{ mod} p = 119^{38}\ 4^{397} \text{ mod} 467 = 9$

$\rightarrow d \equiv x^{e_1} \alpha^{e2} \text{ mod} p$

=> B chấp nhận chữ ký là đúng

2.2.3. Giao thức chối bỏ

Một vấn đề đặt ra, nếu sự cộng tác của chủ thể ký là cần thiết trong việc kiểm tra chữ ký thì điều gì đã ngăn cản anh ta trong việc từ chối chữ ký do anh ta tạo ra. Tất nhiên, anh ta có thể cho rằng chữ ký đúng đó là giả mạo và từ chối kiểm tra nó hoặc anh ta thực hiện một giao thức mà theo đó chữ ký sẽ không được kiểm tra. Vì vậy, một lược đồ chữ ký chống chối bỏ được kết hợp chặt chẽ với một giao thức chối bỏ và nhờ điều đó chủ thể ký có thể chứng minh được chữ ký đó là giả mạo. (Nếu anh ta từ chối thực hiện 1 phần trong giao thức chối bỏ, điều đó đồng nghĩa với dấu hiệu chứng minh chữ ký đó là của anh ta và anh ta đang cố gắng từ chối chữ ký của mình).

Giao thức chối bỏ gồm hai tiến trình của giao thức kiểm tra và có các bước sau:

1. B chọn e_1, e_2 ngẫu nhiên, e_1, $e_2 \in Z_{q^*}$.
2. B tính $c = y^{e_1} \beta^{e_2}$ modp và gửi nó cho A.
3. A tính $d = c^{a^{-1} \bmod q}$ modp s và gửi nó cho B.
4. B kiểm tra $d \neq x^{e1} \alpha^{e_2}$ modp.
5. B chọn f_1,f_2 ngẫu nhiên, f_1, $f_2 \in Z_{q^*}$.
6. B tính $C = y^{f1} \beta^{f2}$ modp và gửi nó cho A.
7. A tính $d = c^{a^{-1} \bmod q}$ modp và gửi nó cho B.
8. B kiểm tra $d \neq x^{f1} \alpha^{f_2}$ modp.
9. B kết luận rằng y là chữ ký giả mạo khi và chỉ khi

$(d\alpha^{-e_2})^{f1} \equiv (D\alpha^{-f2})^{e}$ modp

Ví dụ: Lấy p=467, α = 4, a = 101, β = 449. Ký trên thông báo x=286 với chữ ký y= 83 (là giả mạo). A muốn thuyết phục B rằng chữ ký đó là không đúng. Vậy phải thực hiện như sau:

Chọn ngẫu nhiên e1 = 45, e2 = 237. B tính c=305 và A trả lời với d= 109.

B tính $286^{45} .4^{237} \bmod 467 = 149$.

Vì 149 ≠ 109 nên ta phải thực hiện giao thức chối bỏ

B chọn tiếp $f_1 = 125$, $f_2 = 9$, ngẫu nhiên, B tính C=270 và A trả lời với D=68.

B tính: $286.4^{9} \bmod 467 = 25$.

Vì 25 ≠ 68 nên B thực hiện tiếp bước cuối cùng của giao thức là thực hiện kiểm tra tính chính xác.

Ta có: $109.4^{-237})^{125} \equiv 188 \bmod 467$

và $(68.4^{-9})^{45} \equiv 188 \bmod 467$;=> $(d\alpha^{-e_2})^{f1} \equiv (D\alpha^{-f2})^{e1}$ modp

Vậy B tin chắc rằng đó là chữ ký không đúng

Bây giờ vấn đề đặt ra là:

- A có thuyết phục được B rằng chữ ký không đúng đó là giả mạo
- A không thể làm cho B bị thuyết phục rằng chữ kí đó đúng là giả mạo ngoại trừ xác suất rất nhỏ.

2.3. Các định lý:

2.3.1. Định lý 1:

Nếu $y \neq x^a$ modp B sẽ chấp nhận y như là một chữ ký đúng của x với xác suất 1/q.

Chứng minh: Trước tiên, ta nhận xét rằng mỗi yêu cầu c sẽ xảy ra tương ứng chính xác với một cặp (e1,e2) bậc q. (Bởi vì y và β đều là phần tử thuộc nhóm nhân G có bậc nguyên tố lẻ q). Khi A nhận yêu cầu c, A không biết B đã dùng cặp (e1,e2) nào để xây dựng c.

Chúng ta cần phải chứng minh rằng, nếu $y \neq x^a \bmod p$ thì các câu trả lời của A $d \in G$ có thể đúng duy nhất một cặp (e1,e2) trong các cặp (e1, e2) bậc q.

Từ phần tử sinh α của nhóm G, chúng ta có thể viết được một số phần tử của G như là một khả năng của α với số mũ xác định duy nhất theo modun của q.

Như vậy, ta có thể viết $c = \alpha^i$, $d = \alpha^j$, $x = \alpha^k$, $y = \alpha^l$ với $i, j, k, l \in Z_p$ và tất cả tính theo modun của p.

Ta xét 2 đồng dư sau:

$$c \equiv y^{e_1}\beta^{e_2} \bmod p \quad (1)$$

$$d \equiv x^{e_1}\alpha^{e_2} \bmod p \quad (2)$$

$$(1) \Leftrightarrow \alpha^i \equiv \alpha^{l.e_1}.\beta^{e_2} \bmod p$$

Với $\beta = \alpha^a \bmod p$

$$\Rightarrow \alpha^i \equiv \alpha^{l.e_1}.\alpha^{a.e_2} \bmod p$$

$$\Leftrightarrow \alpha^i \equiv \alpha^{l.e_1 + a.e_2} \bmod p$$

$$\Rightarrow i \equiv l.e_1 + a.e_2 \bmod q \quad (3)$$

$$(2) \Rightarrow \alpha^j \equiv \alpha^{k.e_1}.\alpha^{e_2} \bmod p$$

$$\Leftrightarrow \alpha^j \equiv \alpha^{k.e_1 + e_2} \bmod p$$

$$\Rightarrow j \equiv k.e_1 + e_2 \bmod q \quad (4)$$

Từ (3) và (4) ta có hệ:

$$i \equiv l.e_1 + a.e_2 \bmod q$$

$$j \equiv k.e_1 + e_2 \bmod q$$

Xét $D = \begin{vmatrix} l & k \\ a & 1 \end{vmatrix} = l - a.k$ (5) mặt khác: $y \neq x^a \bmod p$ (gt)

$$\Leftrightarrow \alpha^l \neq \alpha^{k.a} \bmod p$$

$$\Rightarrow l \neq a.k \bmod q \quad (6)$$

Từ (5) và (6) => $D \neq 0$

Vì hệ số ma trận của hệ đồng dư theo modulo $q \neq 0$ nên hệ có 1 nghiệm duy nhất nghĩa là tìm được duy nhất một cặp (e1, e2) $\forall$ i, j, k, $l \in Z_p$.

Do đó, $\forall d \in G$ là câu trả lời thì tất cả các câu trả lời đó chỉ đúng với 1 cặp (e1, e2) trong các cặp (e1, e2) bậc q.

Vậy xác suất A đưa cho B câu trả lời d mà sẽ được kiểm tra 1/q, đồng nghĩa với việc B chấp nhận y là chữ ký của A với xác suất 1/q.

2.3.2. Định lý 2:

Khi A và B thực hiện giao thức chối bỏ. Nếu $y \neq x^a \text{mod} p$ thì

$$(d\alpha^{-e_2})^{f_1} \equiv (D\alpha^{-f_2})^{e_1} \text{ mod} p.$$

Chứng minh:

Ta có: $d \equiv ca \text{ mod} p.$

Mà $c \equiv y^{e_1}\beta^{e_2} \text{ mod} p.$

-> $d \equiv y^{ei.a-1} . \alpha^{e2.a-1} \text{ mod} p.$

Mặt khác: $\beta \equiv \alpha^a \text{ mod} p_1$

$\Rightarrow d \equiv y^{e_1.a} . \alpha^{e_2.a.a} \text{ mod} p$

Do vậy :

$$(d.a^{-e_2})^{f1} \equiv (y^{e_i.a-i.f_i.} \alpha^{e_2.a-i.a} . \alpha^{-e2})^{f_1} \text{mod} p.$$

$$\equiv y^{e_i.a-i.f1} .\alpha^{e_2.f_1-e_2.f1} \text{ mod} p$$

$$\equiv y^{e_i.a-i.f1} \text{mod} p \ (1)$$

Tương tự như trên ta tính được :

$$(D.\alpha^{-f_2})^{e_1} \equiv y^{e_1.a-i.f_1} \text{ mod} p(2)$$

Với $D \equiv C^{a-1} \text{ mod} p$

$C \equiv y^{f_1}\beta^{f_2} \text{ mod} p$

$\beta \equiv \alpha^a \text{ mod} p$

Từ (1) và (2) $\Rightarrow (d\alpha^{-e_2})^{f_1} \equiv (D\alpha^{-f_2})^{e_1} \text{ mod} p.$

Vì vậy, nếu y là chữ ký giả mạo thì A có thể thuyết phục được B tin chữ ký đó là gi ả

mạo.

2.3.3. Định lý 3:

Giả sử $y \equiv x \text{amod} p$ B thực hiện giao thức chối bỏ.

Nếu $d \neq xe\ 1\ \alpha e\ \ 2 \text{ mod} p$, $D \neq xf\ 1\ \alpha f\ 2 \text{ mod} p$ thì khả năng $(d\alpha\text{-}e\ 2)f\ 1 \neq (D\alpha\text{-}f\ 2)e\ 1 \text{ mod} p$ có xác suất là 1-1/q.

Ở đây ta xét trường hợp A có thể từ chối chữ ký đúng của anh ta. Trong trường hợp này, chúng ta có thể không giả định A làm theo giao thức nghĩa là A không xây dựng d và D như lý thuyết bởi giao thức, chúng ta chỉ giả định A tạo ra 2 giá trị d và D thỏa mãn điều kiện bước 4, 8, 9 của giao thức chối bỏ.

Giả thuyết chúng ta có.

$$y \equiv x^a \bmod p$$

$$d \neq x^{e_1}\alpha^{e_2} \bmod p$$

$$D \neq x^{f_1}\alpha^{f_2} \bmod p$$

$$(d\alpha^{-e_2})^{f_1} \equiv (D\alpha^{-f_2})^{e_1} \bmod p$$

Từ $(d\alpha^{-e_2})^{f_1} \equiv (D\alpha^{-f_2})^{e_1} \bmod p$ có:

2.3.4. Định lý 4:

Giả sử y ≡ xamodp B thực hiện giao thức chối bỏ.

Nếu d ≠ xe 1 αe 2 modp, D ≠ xf 1 αf 2 modp thì khả năng (dα-e 2)f 1 ≠ (Dα-f 2)e 1 modp có xác suất là 1-1/q.

Ở đây ta xét trường hợp A có thể từ chối chữ ký đúng của anh ta. Trong trường hợp này, chúng ta có thể không giả định A làm theo giao thức nghĩa là A không xây dựng d và D như lý thuyết bởi giao thức, chúng ta chỉ giả định A tạo ra 2 giá trị d và D thỏa mãn điều kiện bước 4, 8, 9 của giao thức chối bỏ.

2.3.5. Vấn đề cần giải quyết:

Ba định lý trong phần này đều mới chỉ đề cập tới một khía cạnh là A chấp nhận hay chối bỏ chữ ký của mình chưa nói đến một khía cạnh khác là B có thể chối bỏ việc mình đã đọc thông báo do A gửi. Ta giả định rằng, nếu A gửi cho B một thông báo đòi nợ nhưng B chưa muốn trả hoặc không muốn trả thì anh ta sẽ lờ đi coi như chưa nhận hay chưa đọc thông báo đó. Vậy A có thể làm cách nào để chứng minh B đã mở thông báo?

Để giải quyết vấn đề cả A và B thực hiện theo giao thức sau:

Trước tiên, A và B phải xây dựng khóa K theo lược đồ trao đổi khóa Diffie- Hellman.

Giao thức như sau:

Giả sử p là số nguyên tố, α là căn nguyên thủy của Zp*; α, p là công khai cuộc trao đổi khóa giữa A và B diễn ra như sau:

i. A chọn ngẫu nhiên aA : $0 \leq aA \leq p-2$.

2. A tính $\alpha^{aA} \bmod p$ rồi gửi nó cho B.

3. B chọn ngẫu nhiên aB : $0 \leq aB \leq p-2$.

4.11 B tính α^{aB} và gửi nó cho A.

5. A tính $K = (\alpha^{aB})^{aA} \bmod p$.

6. B tính K = $(\alpha^{a\,B})^{a\,A}$ mod p.

Sau đó, A tiếp tục xây dựng một khóa K1, K1 bí mật. A có thể xây dựng K1 theo hệ mật đối xứng (DES, AES – đó là một hệ khóa. Các khóa lập mã và giải mã là như nhau hay dễ dàng xác định lẫn nhau. Các hệ một khóa cung cấp một cách tuyệt vời cho việc mã hóa các tệp riêng của người dùng). Avà B tiến hành theo các bước sau đây:

i. A dùng K1 để mã hóa thông báo x và chữ ký kèm theo:

y = sigA(x)

i = eK 1(x, y) A gửi i cho B

2. B gửi lại thông báo x1 kèm theo chữ ký y1 = sigB(x1) và mã y1 bằng

K: j=eK(y1) rồi gửi cho A. Trong đó x1 chứa ngày, giờ, lời yêu cầu

và chứa cả i.

3. A tính i1 = eK(K1) và gửi nó cho B.

Khi A và B tiến hành theo giao thức trên, muốn đọc được thông thì B phải gửi lại một thông báo (đã được mã hóa bằng khóa K) tới A, yêu cầu A gửi khóa K1 cho mình, bởi vì K1 chỉ mình A biết. A kiểm tra thông báo của B theo thuật toán kiểm tra công khai Bver để xác định thông báo có đúng là của B gửi hay không? Nếu đúng, anh ta gửi K1 cho B mà K1đã được mã hóa theoK.

A thực hiện theo cách trên sẽ có đủ chứng cứ để chứng minh trước tòa rằng B có mở và đọc thông báo anh ta gửi tới bằng cách đưa ra thông báo có kèm theo chữ ký của B và cả ngày, giờ B đọc thông báo đó.

Chương 3 Ứng dụng mô phỏng

3.1. GIỚI THIÊU SƠ BỘ .NET

3.2. Giới thiệu chung về nền tảng .NET (.NET platform)

Nền .NET là một khái niệm mới trong khoa học máy tính; nó vượt ra ngoài khuụn khổ của một ngụn ngữ lập trỡnh, một bộ thư viện; nú chưa phải là một hệ điều hành, chỳng ta cú thể hiểu đơn giản nú là một nền để từ đú cú thể phỏt triển cỏc ứng dụng cả trờn Windows lẫn trờn Internet thuận tiện hơn. Nền .NET được thiết kế để phục vụ cỏc mục đớch sau:

- Cung cấp một mụi trường lập trỡnh hướng đối tượng tuyệt đối, mó của chương trỡnh được thực thi trờn một mỏy hay cũng cú thể thực thi từ một mỏy từ xa thụng qua Internet.
- Giảm thiểu tối đa xung đột giữa cỏc version của một phần mềm
- Đem lại một mụi trường cho phộp cỏc ngụn ngữ lập trỡnh cú thể giao tiếp với nhau, tớch hợp với nhau

Chỳ ý: chỳng ta cũng cần phải phõn biệt giữa hai thuật ngữ: .NET và nền .NET. .NET bao gồm 3 thành phần cơ bản :

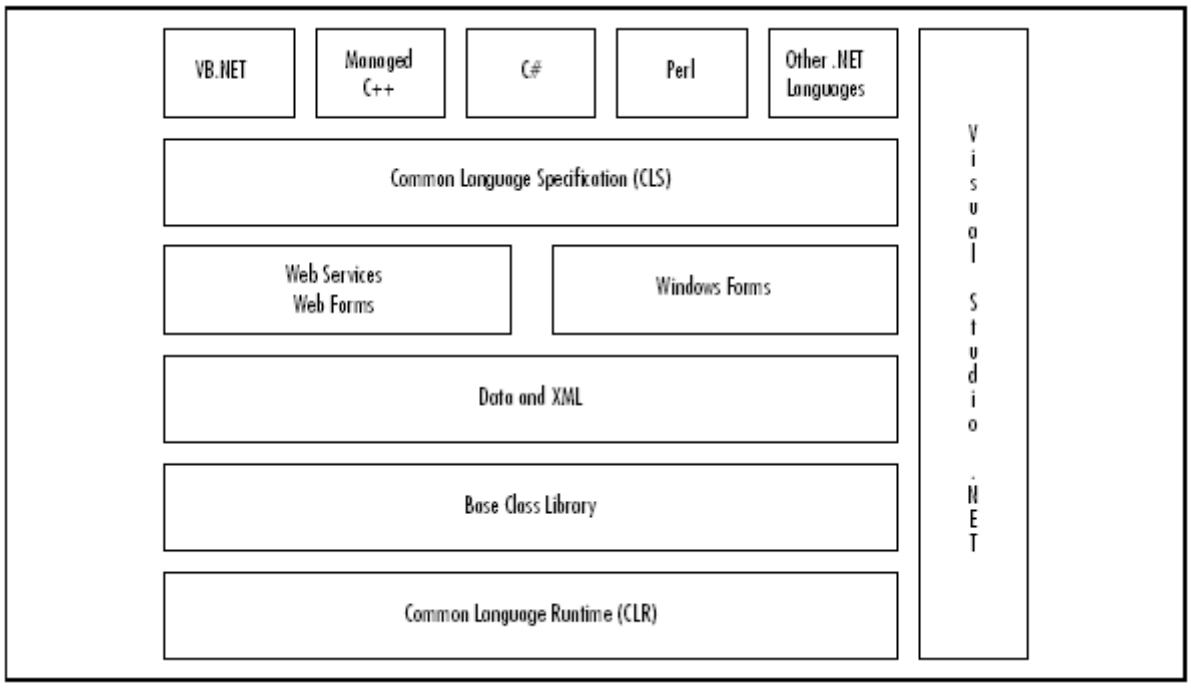
- *Nền .NET: một nền cho phộp phỏt triển cỏc ứng dụng*
- *Cỏc sản phẩm .NET: bao gồm tất cả cỏc sản phẩm của Microsoft dựa trờn nền .NET.*
- *Cỏc dịch vụ .NET: cỏc dịch vụ được cung cấp bởi Microsoft phục vụ cho việc phỏt triển cỏc ứng dụng chạy trờn nền .NET.*

Như vậy nền .NET chỉ là một thành phần của .NET.

Nền .NET gồm hai thành phần chớnh: Common language runtime (CLR) và thư viện lớp nền .NET. Hai thành phần này sẽ được trỡnh bày cụ thể ở những phần sau.

3.3. Kiến trúc phân lớp nền .NET

Hình 1 biểu diễn kiến trúc nền .NET. Mỗi ngôn ngữ thuộc gia đình .NET (phiên bản đầu tiên gồm các ngôn ngữ : VC.NET, VB.NET, C#, sau đó có thêm VJ#)



Hình 1 Kiến trúc nền .NET

đều được dịch sang ngôn ngữ trung gian Microsoft (MSIL hay gọi ngắn là IL) – ngôn ngữ dựa theo tiêu chuẩn của Common Language Specification (CLS). Có 3 loại ứng dụng cơ bản là: các ứng dụng Web, các dịch vụ Web, các ứng dụng Form trên Windows. Những ứng dụng này sử dụng các đối tượng, phương thức từ thư viện lớp cơ sở và chạy trong môi trường CLR.

3.4. Những đặc trưng của nền .NET

Những đặc trưng chủ chốt của nền .NET chủ yếu nssằm trong CLR, thư viện lớp cơ sơ và CLS.emchỉ xin trình bày một số đặc trưngemcho là dễ nhận biết và nắm bắt nhất

3.4.1. Phát triển đa ngôn ngữ

Trước đây, vấn đề sử dụng đa ngôn ngữ (multilanguage) hay giao thoa ngôn ngữ lập trình (cross – language) đã được đề cập nhiều khi phát triển các ứng dụng. Đa ngôn ngữ có thể hiểu là việc sử dụng nhiều ngôn ngữ phát triển một ứng dung, mỗi ngôn ngữ viết lên một phần ứng dụng. Với giải pháp này, người lập trình có thể sử dụng một ngôn ngữ mà mình quen thuộc kết hợp sử dụng lại những đoạn mã được viết trên những ngôn ngữ khác phù hợp với mục đích của một phần chương trình nhất định để xây dựng lên một ứng dụng hoàn chỉnh. Một phương pháp truyền thống để thực hiện giải pháp này là xây dựng nên các thư viện động .dll. Phương pháp này được áp dụng trong VS 6.0. Mỗi ngôn ngữ đều có thể xây dựng nên một thư viện .dll.

Một ngôn ngữ khác sẽ sử dụng file .dll đó như là một phần thư viện của mình. Phương pháp thư hai là sử dụng mô hình đối tượng hướng thành phần – COM (trong đề tài này sẽ không trình bày về COM, ở đâyemchỉ điểm qua). Cả hai phương pháp trên đều sử dụng ngôn ngữ định nghĩa giao diện (IDL). Với nền .NET, chúng ta có thể thực hiện việc phối hợp ngôn ngữ dễ dàng hơn. Nền .NET cho phép ngôn ngữ này có thể tích hợp với ngôn ngữ khác bằng việc sử dụng ngôn ngữ trung gian là MSIL Tất cả các ngôn ngữ khi soạn thảo có thể khác nhau, sau đó được dich bởi một chương trình dịch thích hợp, chúng đều trở thành dạng ngôn ngữ trung gian, khác biệt giữa các ngôn ngữ hoàn toàn bị xoá bỏ. Ngôn ngữ trung gian sẽ được đưa vào CLR để thực thi.

3.4.2. Chương trình ứng dụng độc lập với hệ điều hành và bộ vi xử lí

Ngôn ngữ trung gian IL là ngôn ngữ độc lập với bộ vi xử lí, nó là ngôn ngữ ở cấp cao hơn ngôn ngữ máy. Khi nào cần thực thi, IL sẽ được dịch ra ngôn ngữ máy thích hợp. Bất cứ hệ điều hành nào hỗ trợ nền .NET thì ứng dụng .NET sẽ chạy và không gặp khó khăn gì. Đối với các hệ điều hành thuộc họ Windows từ Win 98 trở nên đều hỗ trợ nền .NET. Tháng 6 – 2001, khi mới cho ra đời .NET, Microsoft đã thông báo rằng họ đã đạt được thoả thuận phát triển .NET trên Unix, tuy nhiên đến nay vẫn chưa có kết quả chính thức. Tháng 10 – 2001, Microsoft cho phép Ximian, người đã phát triển giao diện GNOME thông dụng trên Linux, phát triển một chương trình dịch C# và CLR trên Linux. Phiên bản đầu tiên có tên Mono có thể tìm trên www.go-mono.net. Công việc hiện đang tiến hành ở giai đoạn xây dựng thư viện cơ sở trên Linux.

3.4.3. Quản lí bộ nhớ tự động

Rò rỉ bộ nhớ luôn là vấn đề phức tạp trong lập trình khi ta không quản lý nổi những vùng nhớ đã được cấp phát. Trong Visual Basic, quản lý bộ nhớ được thực hiện bởi kĩ thuật đếm số lần truy cập. Trong C và C++, cách tốt nhất để quản lý bộ nhớ là tự mình trả lại cho hệ điều hành những vùng nhớ không dùng nữa. Trong .NET, có một bộ phận là GC(Garbage Collection) làm nhiệm vụ thu hồi lại vùng nhớ hiệu quả hơn những cách trên.

3.4.4. Hỗ trợ phiên bản

Những lập trình viên đã từng lập trình với thư viện động DLL chắc hẳn đều biết đến thuật ngữ ‘DLL Hell’. DLL Hell có thể miêu tả như sau : bạn đang sử dụng một chương trình ứng dụng với một DLL phiên bản i.0, sau đó bạn cài thêm một ứng dụng khác cũng sử dụng một DLL giống như vậy với phiên bản 1.1. Khi đó ứng dụng cũ lập tức sẽ có vấn đề, có thể không chạy. Khi bạn thay thế DLL đó với DLL phù hợp với ứng dụng cũ thì ứng dụng mới lại không chạy. Trong .NET, các thành phần của đối tượng luôn được phân tách riêng rẽ, một ứng dụng chỉ load những thành phần đã được xây dựng, kiểm tra chạy thử với ứng dụng đó. Sau khi một ứng dụng đã cài đặt

và chạy thử thành công thì nó luôn chạy. .NET thực hiện vấn đề này bằng cách sử dụng thêm thành phần là assemblies. Những thành phần được đóng gói lại trong một assembly. Assembly có chứa thông tin về phiên bản, và CLR trong .NET sẽ sử dụng thông tin này để nạp đúng những thành phần phục vụ cho ứng dụng

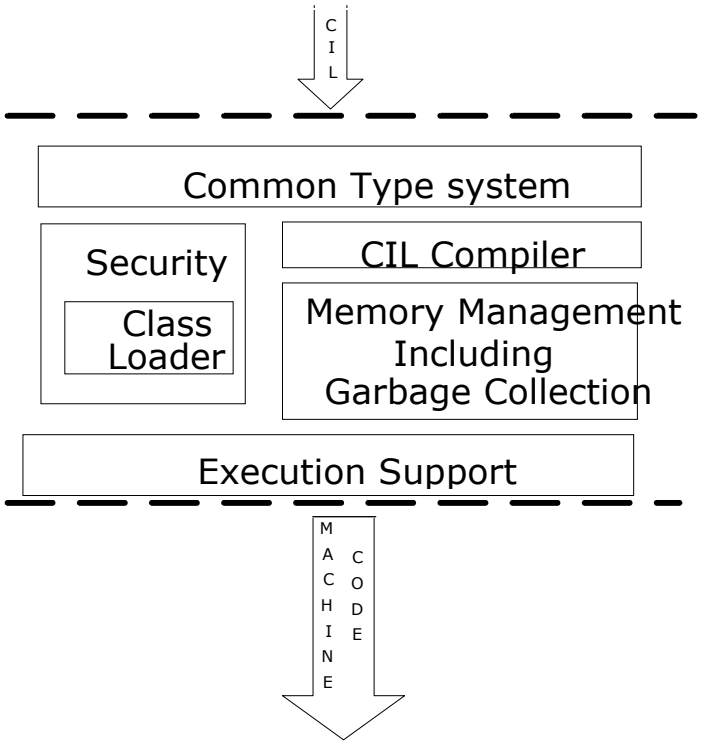
3.4.5. Những thành phần của nền .NET

Như chúng ta đã xem ở phần trước, có nhiều thành phần trong nền .NET. Trong phần này chúng ta sẽ trình bày các thành phần nổi bật về tính chất và vai trò của chúng trong cả hệ thống

.

3.5. CLR

CLR có thể được coi như trái tim của nền .NET. CLR nằm ở cấp cuối cùng trong sơ đồ phân cấp của nền .NET, trực tiếp giao tiếp với hệ điều hành hay các thiết bị phần cứng. Vai trò của nó là nhận mã IL, dịch chuyển sang mã máy thích hợp. Từ IL trở xuống CLR giống nhau cho mọi ngôn ngữ thuộc dòng .NET, điều này giải quyết được vấn đề đa ngôn ngữ trong một ứng dụng



3.5.1. Mã quản lí và mã không quản lí (Managed/Unmanaged Code)

Những mã được soạn thảo, dịch nhằm mục đích được chạy trong môi trường CLR thì được gọi là mã mã quản lí (managed code). Có thể hiểu đơn giản hơn, mã quản lí là loại mã mà chương trình thực thi mã đó được quản lí bởi CLR và nó được thừa hưởng mọi dịch vụ mà CLR có. Thông thường, mã quản lí là những mã được tích hợp sẵn ở trong các thư viện lớp hay những mã được dịch bởi một chương trình dịch tuân theo chuẩn CLS tạo ra ngôn ngữ trung gian. Mã không quản lí (unmanaged code) là những mã không được soạn thảo, dịch trong môi trường .NET và không nhằm mục đích chạy trong CLR tuy nhiên CLR vẫn nạp những mã này vào chạy, nó chỉ không hỗ trợ các dịch vụ cho loại mã này. Điển hình cho loại mã này là các thư viện DLL có từ trước .NET và thư viện Windows APIs, những chương trình .NET sử dụng Windows APIs có nghĩa là nó đã sử dụng mã không quản lý.

3.5.2. Ngôn ngữ trung gian , hệ thống kiểu thông thường và CLS

Ngôn ngữ trung gian MSIL trong .NET, hệ thống kiểu thông thường và CLS là 3 yếu tố gắn liền với nhau tạo nên khả năng phối hợp đa ngôn ngữ và độc lập với môi trường của các ứng dụng .NET.

Hệ thống kiểu thông thường (common type system) bao gồm các kiểu dữ liệu mà các ngôn ngữ .NET có thể sử dụng cũng như qui cách người dùng phải tuân theo để xây dựng nên những kiểu dữ liệu của người dùng. Các kiểu dữ liệu trong hệ thống kiểu thông thường được chia thành 2 loại :

- Loại tham trị: những kiểu tham trị trực tiếp lưu trữ các dữ liệu, được cấp phát ở vùng nhớ stack. Những dữ liệu kiểu này thường là kiểu dữ liệu xây dựng sẵn như Int, long, boolean,.. hay kiểu struct do người dùng định nghĩa.
- Loại tham biến: kiểu tham biến lưa giữ địa chỉ chỉ tới một vùng dữ liệu, chúng được cấp phát ở vùng nhớ Heap. Những dữ liệu kiểu này thường là các biến đối tượng.

CLS (common language specification) là một tập hợp các đặc điểm ngôn ngữ mà tất cả các ngôn ngữ lập trình trên .NET phải tuân theo, nó cũng bao gồm các kiểu dữ liệu và các qui cách trong hệ thống kiểu thông thường. Những người muốn phát triển một ngôn ngữ trên .NET thì cũng phải dựa theo CLS để xây dựng chương trình dịch gọi là chương trình dịch CLS .

Ngôn ngữ trung gian IL được dịch ra từ mã nguồn của một ngôn ngữ lập trình cấp cao bằng một chương trình dịch CLS, ngôn ngữ trung gian IL sau đó được CLR dịch lại một lần nữa ra mã máy để thực thi.

3.5.3. Thư viện lớp cơ sở của .NET

.NET có một thư viện đồ sộ những kiểu dữ liệu có thể sử dụng lại, được tích hợp chặt chẽ với CLR. Thư viện lớp này hoàn toàn hướng đối tượng, cung cấp những kiểu dữ liệu mà chúng ta có thể sử dụng rất nhiều chức năng từ đó. Nhờ sử dụng thư viện lớp cơ sở chúng ta có thể phát triển các kiểu ứng dụng sau:

- Ứng dụng vào ra Console
- Những ứng dụng Windows với giao diện đồ hoạ
- Những ứng dụng ASP.NET
- Dịch vụ Web
- Các thư viện

Khi muốn lập trình trên Windows chúng ta có thể sử dung các lớp Form, Button, CheckBox, Text… để phát triển các giao diện đồ hoạ. Khi muốn phát triển một ứng dụng Web, chúng ta có thể sử dụng các lớp Web Forms. Tất cả các ngôn ngữ của .NET đều sử dụng thư viện này, điều này làm cho việc sử dụng đa ngôn ngữ cũng dễ dàng hơn.

3.5.4. Assembly và metadata

Nếu chúng ta muốn trình bày kĩ về assembly và metadata thì cần phải có một đề tài chuyên về mảng này, trong giới hạn đề tài này,emchỉ xin trình bày mang tính khái niệm về hai vấn đề trên. Assembly có thể hiểu như là một gói cả mã chương trình, các thành phần, các tài nguyên. Một assembly bao gồm thông tin metadata, mã chương trình ở dạng IL, các file tài nguyên ví dụ như các file ảnh, âm nhạc, các thư viện thành phần.

Metadata là tập hợp dữ liệu ở dạng nhị phân diễn tả các thành phần của chương trình. Metadata được lưu trữ ở file có thể thực thi (executable hay .exe , .dll) cùng với mã IL của chương trình. Metadata chứa những loại dữ liệu cụ thể sau:

- Tên assembly
- Số hiệu phiên bản
- Culture : thông tin về loại ngôn ngữ mà assembly hỗ trợ
- Thông tin về strong name
- Danh sách tất cả các file được đóng gói
- Thông tin về tham chiếu kiểu dữ liệu: CLR sử dụng thông tin này để tìm ra những file định nghĩ kiểu dữ liệu đó.
- Thông tin phục vụ cho tham chiếu đến các assembly khác

CLR hoàn toàn dựa những thông tin này để điều khiển ứng dụng. Assembly và metadata được tạo ra ngay khi ta biên dịch mã nguồn

3.5.5. Chương trình dịch Just in time

Chương trình dịch Just In Time là nằm trong CLR, có nhiệm vụ chuyển mã IL sang mã máy thích hợp. Trong .NET có 3 loại chương trình dịch JIT:

- Pre-JIT: loại JIT này dịch ngay toàn bộ mã IL sang mã máy khi nó được gọi tới.
- Econo-JIT: loại này sử dụng cho các hệ thống hạn chế bộ nhớ, nó dịch mã IL sang mã máy từng bit một, những mã máy sau khi được dịch và

đưa vào thực thi nó còn được để ở vùng nhớ đệm, nếu hết vùng nhớ đệm JIT sẽ xoá các mã máy này.

- Normal JIT: đây là loại ngầm định, dịch mã IL chỉ khi nó được gọi tới, mã máy sau khi dịch sẽ được đưa vào thực thi đòng thời được đặt vào trong bộ nhớ đệm.

3.5.6. Quản lí bộ nhớ (Garbage Collection)

Những người lập trình thường gặp nhiều khó khăn khi giải quyết vấn đề cấp phát bộ nhớ, rò rỉ bộ nhớ, công việc này làm giảm năng suất lập trình. Để giải quyết vấn đề này, .NET đưa ra hệ thống thu gom bộ nhớ GC. Khi chương trình đòi cấp phát thêm bộ nhớ, bộ phận cấp phát bộ nhớ trong phần quản lí bộ nhớ trong CLR sẽ thực hiện, nếu không còn đủ bộ nhớ nó sẽ thông báo là không còn bộ nhớ để cấp phát. GC bắt chạy, nó giả định rằng tất cả mọi thứ trong bộ nhớ đều có thể thu hồi. Sau đó, nó xem toàn bộ bộ nhớ dành cho chương trình ứng dụng, xây dựng nên một đồ thị diễn tả tất cả các vùng bộ nhớ được tham chiếu bởi chương trình và tham chiếu lẫn nhau. Sau xây dựng xong đồ thị, GC tiến hành thu gọn bộ nhớ Heap bằng cách di chuyển tất cả các vùng nhớ thật sự dùng về vị trí mới bắt đầu tại một vùng nhớ Heap còn trống. Cuối cùng nó cập nhật lại các con trỏ trỏ đến các vùng bộ nhớ vừa được di chuyển. Chúng ta có thể thấy dường như GC thực hiện rất nhiều việc, tuy nhiên nó được thực thực hiện tự động bằng CLR, giảm nhẹ đi rất nhiều công việc của người lập trình.

3.5.7. Vòng đời của mã

Trong phần này, emsẽ giới thiệu về trình làm việc của một ứng dụng .NET từ khi soạn thảo mã nguồn đến khi chạy chương trình :

- Bắt đầu từ việc soạn thảo mã nguồn trên một ngôn ngữ .NET quen thuộc trên một hệ soạn thảo văn bản thông thường.
- Dùng một chương trình dịch .NET dịch mã nguồn ra mã IL, đồng thời xây dựng assembly cho ứng dụng.
- Khi chương trình ứng dụng thực thi, hệ điều hành sẽ đọc header của chương trình và đưa CLR vào quản lí chương trình, CLR đọc các thông tin metadata, điều khiển Loader nạp các thư viện cần thiết vào bộ nhớ.
- Hàm *_CorExeMain* được chèn vào điểm nhập của chương trình.
- Bộ phận Loader nhảy vào điểm nhập chương trình và gọi hàm *_CorExeMain* thực thi.
- Khi *_CorExeMain* thực thi, nó gọi chương trình dịch JIT ra thực thi.
- JIT dịch mã IL sang mã máy và đưa vào thực thi đồng thời được dự trữ ở bộ nhớ đệm để khi cần không phải dịch lại.

3.5. MÔ PHỎNG

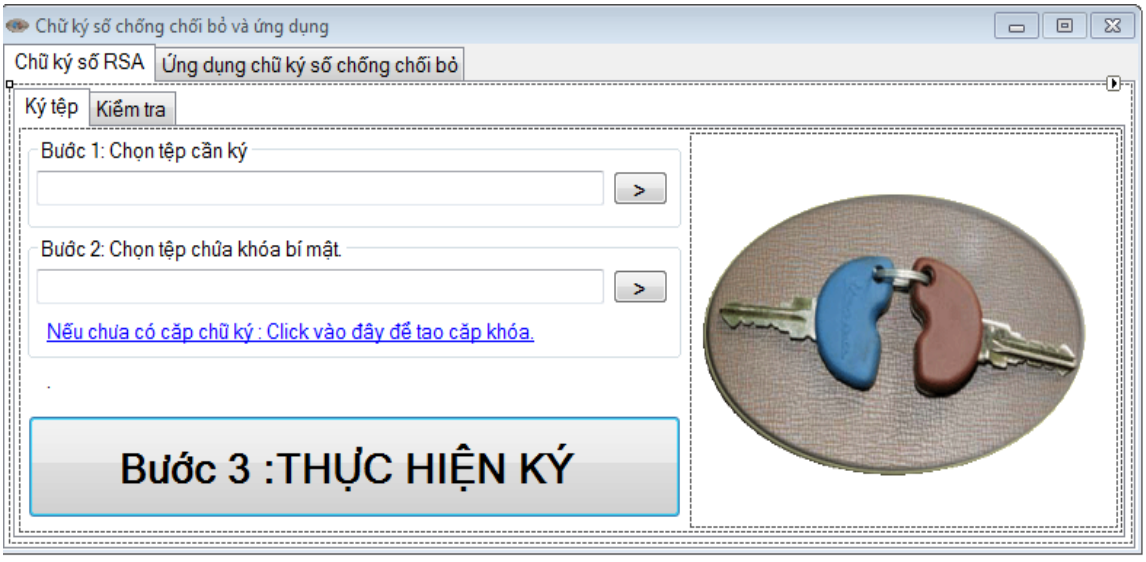
3.5.1. Yêu cầu hệ thống

Hệ điều hành	: Tất cả hệ điều hành Windows hỗ trợ .NET Framework 2.0 trở lên
CPU	: Pentium 233-megahertz (MHz) trở lên.
RAM	: 256MB trở lên.
Card màn hình	: Không yêu cầu.
Card âm thanh	: Không yêu cầu.
.NET Framework	: Phiển bản 2.0 trở lên.
Internet	: Khuyến cáo nên dùng để trao đổi khóa.

3.5.2. Giao diện chương trình

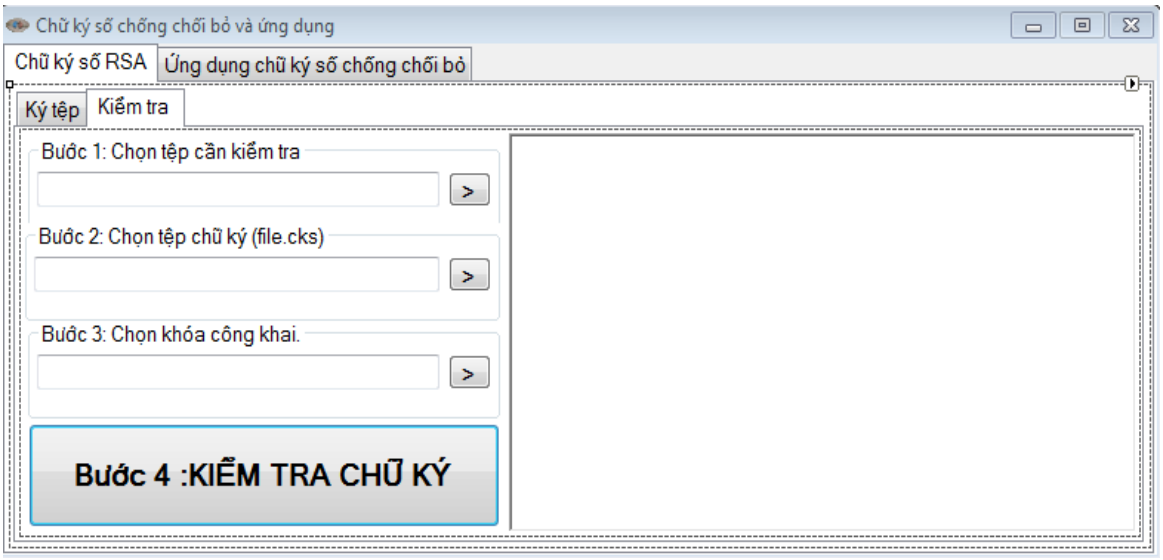
3.5.2.1 Chữ ký số RSA

i. Giao thức ký.



Hình 1: Giao diện ký tệp (Chữ ký số RSA)

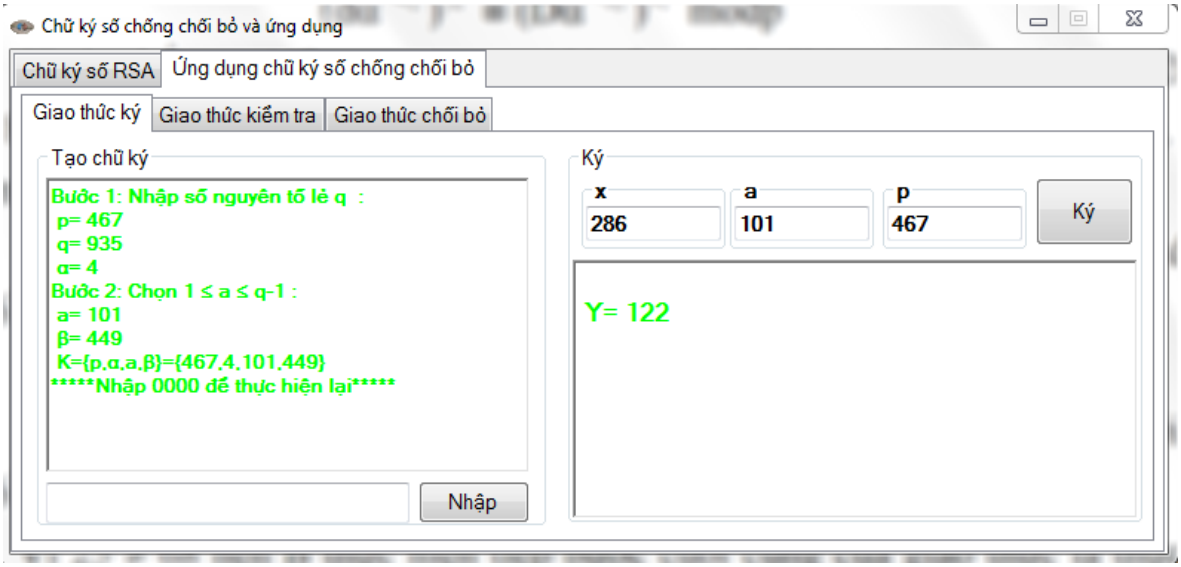
2. Giao thức kiểm tra.



Hình 2: Giao diện kiểm tra chữ ký (Chữ ký số RSA)

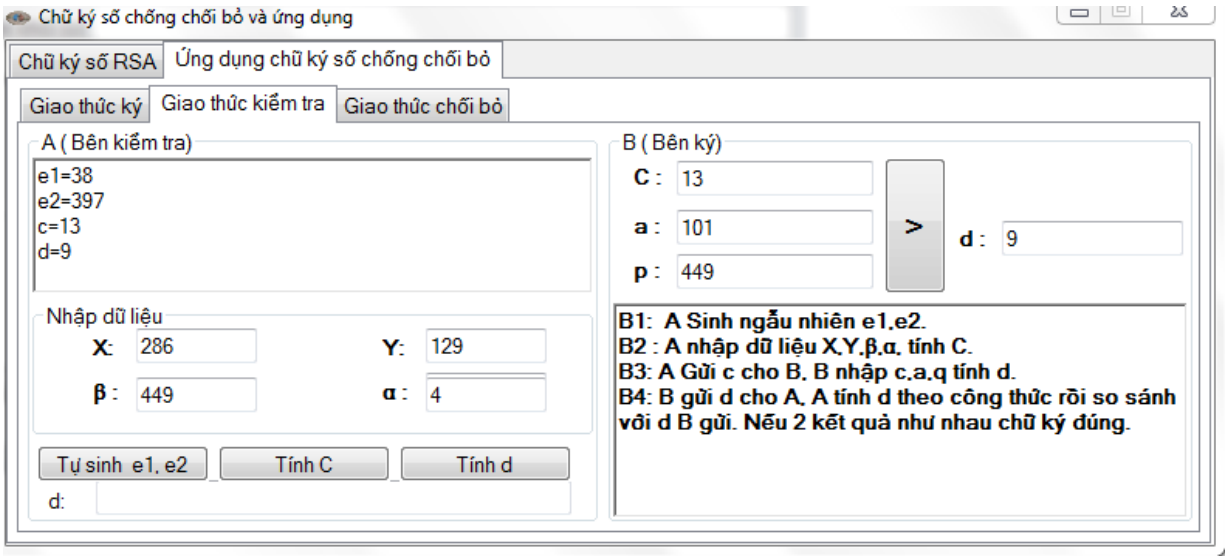
3.5.2.2 Chữ ký số chống chối bỏ

1. **Giao thức ký.**



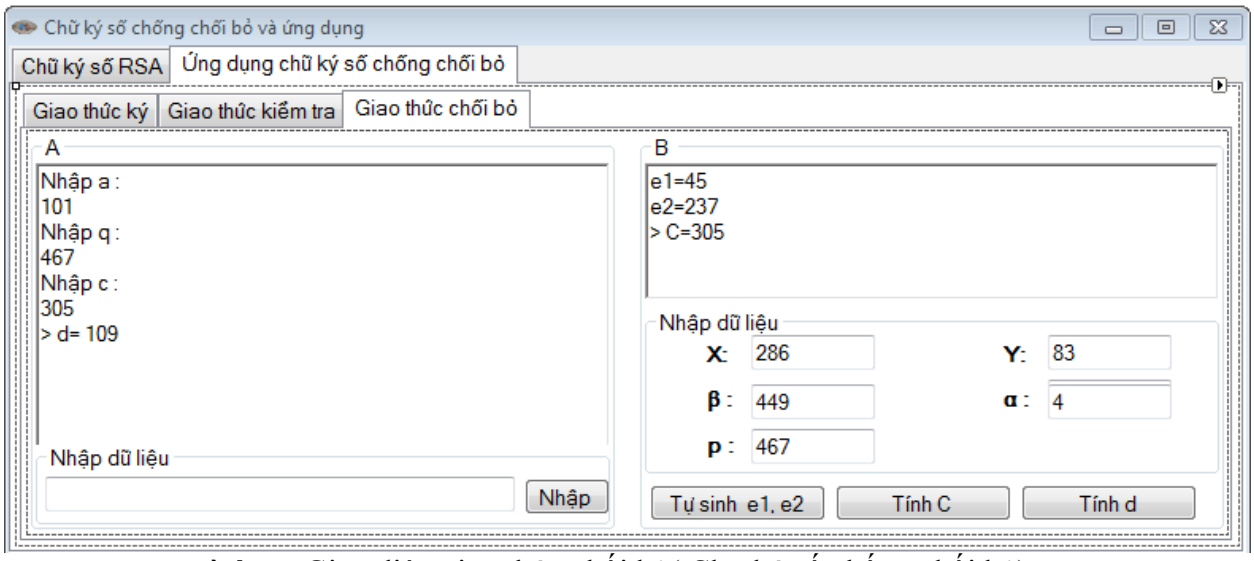
Hình 3: Giao diện tạo khóa và ký (Chữ ký số chống chối bỏ)

2. **Giao thức kiểm tra**



Hình 4: Giao diện kiểm tra(Chữ ký số chống chối bỏ)

3. Giao thức chối bỏ



Hình 5: Giao diện giao thức chối bỏ(Chữ ký số chống chối bỏ)

KẾT LUẬN

Ngày nay, cùng với sự phát triển của khoa học công nghệ hiện đại và Công nghệ thông tin, ngành mật mã đã có những bước phát triển mạnh mẽ, đạt được nhiều kết quả lý thuyết sâu sắc và tạo cơ sở cho việc phát triển các giải pháp bảo mật, an toàn thông tin trong mọi lĩnh vực hoạt động của con người. Đặc biệt là những ưu điểm của chữ ký số.

Chữ ký số được biết đến khi sự trao đổi thông tin ngày càng phổ biến trên các mạng truyền thông ở nơi mà chữ ký tay không thể phát huy tác dụng. Nhưng bên cạnh những ưu điểm của chữ ký số mang lại nó còn bộc lộ những hạn chế nhất là đối với các chữ ký tự xác thực (RSA, Elgamal…), đó là khả năng bảo vệ chữ ký, độ an toàn và xác thực chữ ký…

Trong đồ án này, em đã đi sâu tìm hiểu về lược đồ chữ ký số chống chối bỏ và ứng dụng.

Với lược đồ chữ ký chống chối bỏ nó đã giải quyết được yêu cầu của chữ ký số đó là khả năng bảo vệ chữ ký chống sự sao chép không hợp pháp. Vì chữ ký chống chối bỏ chỉ có thể được kiểm tra khi có sự cộng tác của người ký thông qua giao thức hỏi – đáp.

Tuy nhiên, với lược đồ này lại có một vấn đề nữa là nếu người ký không cộng tác trong việc xác thực chữ ký thì chữ ký sẽ không được kiểm tra hoặc người ký không thực hiện đúng giao thức khi họ muốn chối bỏ chữ ký của mình.

Luận văn tập chung vào nghiên cứu cơ sở lý thuyết và xây dựng chương trình về chữ ký số.Tuy còn nhiều điểm cần phải nghiên cứu và hoàn thiện nhưng do thời gian và trình độ còn hạn chế nên không thể tránh khỏi những nhược điểm, rất mong được sự góp ý của các Thầy, Cô và các bạn.

Cuối cùng em xin cảm ơn nhà trường và các thầy cô trong khoa CNTT trường ĐH Dân Lập Hải Phòng, đặc biệt là thầy giáo T.S Trần Ngọc Thái đã tạo điều kiện và tận tình giúp đỡ em hoàn thành đồ án này.

TÀI LIỆU THAM KHẢO

i. Lý thuyết mật mã và an toàn thông tin – Phan Đình Diệu(NXB ĐHQGHN).

2. TS. Nguyễn Ngọc Cương – "Bài giảng An toàn thông tin".

3. Nguyễn thị Mười Phượng – "Luận văn thạc sĩ".

4.11 D.R Stinson – "Cryptography Theory and Practice", CRC press – 1995.

5. http://google.com , http://vi.wikipedia.org/

Nguồn internet :

(*1) : http://www.vatgia.com/hoidap/4115/77237/lich-su-phat-trien-may-tinh.html