LỜI CẢM ƠN

Trước tiên em xin được bày tỏ lòng biết ơn chân thành tới thầy giáo, PGS. TS Trịnh Nhật Tiến, Khoa Công nghệ thông tin trường Đại học Công nghệ - Đại học Quốc gia Hà Nội đã tận tình chỉ bảo, hướng dẫn em trong suốt thời gian thực hiện luận văn tốt nghiệp.

Em cũng xin chân thành cảm ơn các thầy giáo, cô giáo Khoa Công nghệ thông tin trường Đại học dân lập Hải Phòng đã dạy và truyền đạt những kiến thức cần thiết và bổ ích trong suốt thời gian em học tập tại trường.

Cuối cùng em xin chân thành cảm ơn gia đình và tất cả bạn bè đã đóng góp ý kiến và hỗ trợ em trong quá trình thực hiện luận văn này.

Hải Phòng, tháng 6 năm 2009

Hoàng Thị Thu Trang

MỤC LỤC

VẤN ĐỀ AN TOÀN BẢO MẬT THÔNG TIN

Ngày nay internet cùng với các dịch vụ phong phú của nó có khả năng cung cấp cho con người các phương tiện hết sức thuận tiện để chao đổi, tổ chức, tìm kiếm và cung cấp thông tin. Tuy nhiên, cũng như trong các phương thức truyền thống, việc trao đổi, cung cấp thông tin điện tử trong nhiều lĩnh vực đòi hỏi tính bí mật, tính toàn vẹn, tính xác thực cũng như trách nhiện về các thông tin được trao đổi. Bên cạnh đó, tốc độ xử lý của máy tính ngày càng được nâng cao, do đó cùng với sự trợ giúp của các máy tính tốc độ cao, khả năng tấn công các hệ thống thông tin có độ bảo mật kém rất dễ xảy ra. Chính vì vậy người ta không ngừng nghiên cứu các vấn đề bảo mật và an toàn thông tin để bảo đảm cho các hệ thống thông tin hoạt động an toàn. Cho đến ngày nay với sự phát triển của công nghệ mã hóa phi đối xứng, người ta đã nghiên cứu và đưa ra nhiều kỹ thuật, nhiều mô hình cho phép chúng ta áp dụng xây dựng các ứng dụng đòi hỏi tính an toàn thông tin cao.

Trong văn bản pháp luật của Quốc hội mới ban hành đã công nhận luật giao dịch điện tử - Ngày 29/11/2005. Quốc hội đã thông qua luật giao dịch điện tử 51/2005/QH11. Phạm vi điều chỉnh chủ yếu là giao dịch điện tử trong hoạt động của các cơ quan nhà nước, trong lĩnh vực dân sự, kinh doanh, thương mại... Luật công nhận và bảo vệ hợp đồng điện tử. Trong giao kết và thực hiện giao dịch điện tử, thông báo dưới dạng thông điệp "số" có giá trị pháp lý như thông báo truyền thống.

Việc đòi hỏi an toàn trong giao dịch cũng như trao đổi thông điệp được đặt lên hàng đầu vì vậy việc xác thực thông điệp là một vấn đề rất quan trọng trong giao dịch hiện nay, đặc biệt là trong giao dịch trực tuyến. Khi nhận được một thông điệp như thư, hợp đồng, đề nghị,…vấn đề đặt ra là làm sao để xác định được đúng đối tác giao dịch. Vì vậy đồ án này nghiên cứu một số phương pháp xác thực thông điệp.

Chương 1. MỘT SỐ KHÁI NIỆM CƠ BẢN

1.1. CÁC KHÁI NIỆM TRONG TOÁN HỌC

1.1.1. Khái niệm trong số học

1.1.1.1. Khái niệm số nguyên tố

1/. Khái niệm

Số nguyên tố là số tự nhiên lớn hơn 1 và chỉ có hai ước là 1 và chính nó.

2/. Ví dụ:

Các số 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37 là số nguyên tố.

Số 2 là số nguyên tố ***chẵn*** duy nhất.

Số nguyên tố có vai trò và ý nghĩa to lớn trong số học và lý thuyết mật mã. Bài toán kiểm tra tính nguyên tố của một số nguyên dương n và phân tích một số n ra thừa số nguyên tố là các bài toán rất được quan tâm.

Ví dụ: ***10 số nguyên tố lớn đã được tìm thấy*** [33]

rank	Prime	Digits	Who	when	reference
1	$2^{32582657} - 1$	9808358	G9	2006	Mersenne 44??
2	$2^{30402457} - 1$	9152052	G9	2005	Mersenne 43??
3	$2^{25964951} - 1$	7816230	G8	2005	Mersenne 42??
4	$2^{24036583} - 1$	7235733	G7	2004	Mersenne 41??
5	$2^{20996011} - 1$	6320430	G6	2003	Mersenne 40??
6	$2^{13466917} - 1$	4053946	G5	2001	Mersenne 39??
7	$19249 \cdot 2^{13018586} + 1$	3918900	SB10	2007	
8	$27653 \cdot 2^{9167433} + 1$	2759677	SB8	2005	
9	$28433 \cdot 2^{7830457} + 1$	2357207	SB7	2004	
10	$33661 \cdot 2^{7031232} + 1$	2116617	SB11	2007	

1.1.1.2. Ước số và bội số.

1/. Khái niệm

Cho hai số nguyên a và b, b ≠ 0. Nếu có một số nguyên q sao cho a = b*q, thì ta nói rằng a ***chia hết*** cho b, kí hiệu **b\a.** Ta nói b là ***ước*** của a, và a là ***bội*** của b.

2/. Ví dụ:

Cho a = 6, b = 2, ta có 6 = 2*3, ký hiệu 2\6. Ở đây 2 là ước của 6 và 6 là bội của 2.

Cho các số nguyên a, b ≠ 0, tồn tại cặp số nguyên (q, r) (0 ≤ r < /b/) duy nhất sao cho **a = b*q + r**. Khi đó q gọi là ***thương nguyên***, r gọi là số ***dư*** của phép chia a cho b. Nếu r = 0 thì ta có phép chia hết.

Ví dụ:

Cho a = 13, b = 5, ta có 12 = 5*2 + 3. Ở đây thương là q = 2, số dư là r = 3.

1.1.1.3. Ước số và bội số chung

1/. Khái niệm

Số nguyên **d** được gọi là ***ước chung*** của các sô nguyên $a_1, a_2, ..., a_n$, nếu nó là ***ước*** của tất cả các số đó.

Số nguyên m được gọi là ***bội chung*** của các số nguyên $a_1, a_2, ..., a_n$, nếu nó là ***bội*** của tát cả các số đó.

Một ước chung d > 0 của các số nguyên $a_1, a_2, ..., a_n$, trong đó mọi ước chung của $a_1, a_2, ..., a_n$ đều là ước của d, thì d được gọi là ***ước chung lớn nhất*** (UCLN) của $a_1, a_2, ..., a_n$. Ký hiệu d = gcd ($a_1, a_2, ..., a_n$) hay d = **UCLN**($a_1, a_2, ..., a_n$).

Một bội chung **m** > 0 của các số nguyên $a_1, a_2, ..., a_n$, trong đó mọi bội chung của $a_1, a_2, ..., a_n$ đều là bội của **m**, thì **m** được gọi là ***bội chung nhỏ nhất*** (BCNN) của $a_1, a_2, ..., a_n$. Ký hiệu **m = lcm**($a_1, a_2, ..., a_n$) hay **m = BCNN**($a_1, a_2, ..., a_n$).

2/. Ví dụ:

Cho a = 12, b = 15, **gcd**(12, 15) = 3, **lcm**(12, 15) = 60.

1.1.1.4. Số nguyên tố cùng nhau.

1/. Khái niệm

Nếu **gcd**($a_1, a_2, ..., a_n$) = 1, thì các số $a_1, a_2, ..., a_n$ được gọi là ***nguyên tố cùng nhau.***

2/. Ví dụ :

Hai số 8 và 13 là ***nguyên tố cùng nhau***, vì **gcd**(8, 13) = 1.

1.1.1.5. Đồng dư

1/. Khái niệm

Cho hai số nguyên a, b, m (m > 0). Ta nói rằng a và b ***"đồng dư"*** với nhau theo **modulo m**, nếu chia a và b cho **m**, ta nhận được cùng một số sư.

Ký hiệu: **a ≡ b** (mod m).

2/. Ví dụ:

17 ≡ 5 (mod 3) vì chia 17 và 5 cho 3, được cùng số dư là 2.

1.1.2. Khái niệm trong đại số

1.1.2.1. Nhóm

1/. Khái niệm

Nhóm là một bội (G, *), trong đó G ≠ ∅, * là ***phép toán hai ngôi*** trên G thỏa mãn ba tính chất sau:

+ Phép toán có tính kết hợp: (x*y)*z = x*(y*z) với mọi x, y, z ∈ G.

+ Có phần tử ***trung lập*** e ∈ G: x*e = e*x = x với mọi x ∈ G.

+ Với mọi x ∈ G, có phần tử nghịch đảo x' ∈ G: x*x' = x'*x = e.

Cấp của nhóm G được hiểu là số phần tử của nhóm, ký hiệu là |G|.

Cấp của nhóm có thể là ∞ nếu G có vô hạn phần tử.

Nhóm Abel là nhóm (G, *), trong đó phép toán hai ngôi * có tính giao hoán.

Tính chất: Nếu a*b = a*c, thì b = c.

Nếu a*c = b*c, thì a = b.

2/. Ví dụ :

Tập hợp các số nguyên Z cùng với phép cộng (+) thông thường là nhóm giao hoán, có phần tử đơn vị là số 0. Gọi là ***nhóm cộng*** các số nguyên.

Tập Q^* các số hữu tỷ khác 0 (hay tập R^* các số thực khác 0), cùng với phép nhân (*) thông thường là nhóm giao hoán. Gọi là ***nhóm nhân*** các số hữu tỷ (số thực) khác 0.

Tập các vectơ trong không gian với phép toán cộng vectơ là nhóm giao hoán.

*1.1.2.2. Nhóm con của nhóm (G, *)*

1/. Khái niệm

Nhóm con của G là tập $S \subset G$, $S \neq \phi$, và thỏa mãn các tính chất sau:

+ Phần tử trung lập e của G nằm trong S.

+ S khép kín đối với phép tính (*) trong G, tức là $x*y \in S$ với mọi $x, y \in S$.

+ S khép kín đối với phép lấy nghịch đảo trong G, tức $x^{-1} \in S$ với mọi $x \in S$.

1.1.2.3. Nhóm Cyclic

1/. Khái niệm

Nhóm (**G**, *) được gọi là ***Nhóm Cyclic*** nếu nó được sinh ra bởi một trong các phần tử của nó.

Tức là có phần tử $\mathbf{g} \in \mathbf{G}$ mà với mỗi $\mathbf{a} \in \mathbf{G}$, đều tồn tại $\mathbf{n} \in \mathbf{N}$ để g^n **=g*g*...*g = a.**

(Chú ý **g*g*...*g** là **g*g** với **n** lần).

Nói cách khác: **G** được gọi là Nhóm Cyclic nếu tồn tại $\mathbf{g} \in \mathbf{G}$ sao cho mọi phần tử trong **G** đều là một ***lũy thừa nguyên*** nào đó của g.

2/. Ví dụ :

Nhóm ($\mathbf{Z}^+$, +) gồm các số nguyên dương là Cyclic với phần tử sinh **g** = 1.

1.1.2.4. Tập thặng dư thu gọn theo modulo

1/. Khái niệm

Kí hiệu $\mathbf{Z}_n$ = {0, 1, 2, ..., n-1} là tập các số nguyên không âm < n.

$\mathbf{Z}_n$ và phép cộng (+) lập thành ***nhóm Cyclic*** có phần tử sinh là **1**, pt trung lập **e** = 0.

($\mathbf{Z}_n$, +) gọi là nhóm cộng, đó là nhóm hữu hạn có cấp n.

Kí hiệu $\mathbf{Z}_n^*$ = {**x** ∈ $\mathbf{Z}_n$, **x** là nguyên tố cùng nhau với **n**}. Tức là **x** phải ≠ 0.

$\mathbf{Z}_n^*$ được gọi là ***Tập thặng dư thu gọn theo mod n***, có số phần tử là **ϕ(n)**.

$\mathbf{Z}_n^*$ với phép nhân mod n lập thành một nhóm (nhóm nhân), pt trung lạp e = 1.

Tổng quát ($\mathbf{Z}_n^*$, phép nhân mod n) không phải là nhóm Cyclic.

Nhóm nhân $\mathbf{Z}_n^*$ là Cyclic chỉ khi **n** có dạng: 2, 4, p^k hay $2p^k$ với p là nguyên tố lẻ.

2/. Ví dụ :

Cho n = 21, Z_n^* = {1, 2, 4, 5, 8, 10, 11, 13, 16, 17, 19, 20}.

1.1.2.5. Phần tử nghịch đảo đối với phép nhân

1/. Khái niệm

Cho **a** ∈ $\mathbf{Z}_n$, nếu tồn tại **b** ∈ $\mathbf{Z}_n$ sao cho **a b** ≡ 1 (mod n), ta nói **b** là ***phần tử nghịch đảo*** của **a** trong $\mathbf{Z}_n$ và ký hiệu $\mathbf{a}^{-1}$.

Một phần tử có phần tử nghịch đảo, gọi là khả nghịch.

2/. Ví dụ:

Tìm phần tử nghịch đảo của 3 trong $\mathbf{Z}_7$

Tức là phải giải phương trình 3 **x** ≡ 1 (mod **7**), x sẽ là phần tử nghịch đảo của 3.

I	g_i	u_i	v_i	y
1	7	1	0	
1	3	0	1	2
2	1	1	-2	3
3	0			

Vì **t** = v_2 = -2 < 0 do đó **x** = $\mathbf{a}^{-1}$:= **t** + **n** = -2 + 7 = 5.

Vậy 5 là phần tử nghịch đảo của 3 trong $\mathbf{Z}_7$.

1.1.3. Khái niệm Độ phức tạp của thuật toán

1.1.3.1. Bài toán

Bài toán được diễn đạt bằng hai phần:

Input: Các dữ liệu vào của bài toán.

Ouput: Các dữ liệu ra của bài toán (kết quả).

Không mất tính chất tổng quát, giả thiết các dữ liệu trong bài toán đều là số nguyên.

1.1.3.2. Thuật toán

"Thuật toán" được hiểu đơn giản là cách thức để giải một bài toán. Cũng có thể được hiểu bằng hai quan niệm: Trực giác hay Hình thức như sau:

1). Quan niệm trực giác về "Thuật toán".

Một cách trực giác, Thuật toán được hiểu là một dãy hữu hạn các qui tắc (chỉ thị, mệnh lệnh) mô tả một quá trình tính toán, để từ dữ liệu đã cho (Input) ta nhận được kết quả (Output) của bài toán.

2). Quan niệm toán học về "Thuật toán".

Một cách hình thức, người ta quan niệm thuật toán là một máy Turing.

Thuật toán được chia thành hai loại: Đơn định và không đơn định.

Thuật toán đơn định (Deterministic):

Là thuật toán mà kết quả của mọi phép toán đều được xác định duy nhất.

Thuật toán không đơn định (NoDeterministic):

Là thuật toán có ít nhất một phép toàn mà kết quả của nó là không duy nhất.

1.1.3.3. Hai mô hình tính toán

Hai quan niệm về thuật toán ứng với hai mô hình tính toán.

Ứng với hai mô hình tính toán có hai cách biểu diễn thuật toán.

1). Mô hình ứng dụng: Thuật toán được biểu diễn bằng ngôn ngữ tựa Algol.

+ Đơn vị nhớ: Một ô nhớ chứa trọn vẹn dữ liệu.

+ Đơn vị thời gian: Thời gian để thực hiện một phép tính cơ bản trong số học hay logic như cộng, trừ, nhân, chia,…

2). Mô hình lý thuyết: Thuật toán được biểu diễn bằng ngôn ngữ máy Turing.

+ Đơn vị nhớ: Một ô nhớ chứa một tín hiệu. Với mã nhị phân thì đơn vị nhớ là 1 bít.

+ Đơn vị thời gian: Thời gian để thực hiện một bước chuyển tình trạng.

1.1.3.4. Độ phức tạp của thuật toán

1). Chi phí của thuật toán (Tính theo một bộ dữ liệu vào):

Chi phí phải trả cho một quá trình tính toán gồm chi phí về thời gian và bộ nhớ.

Chi phí thời gian của một quá trình tính toán là thời gian cần thiết để thực hiện một quá trình tính toán. Với thuật toán tựa Algol: Chi phí thời gian là số các phép tính cơ bản thực hiện trong quá trình tính toán.

Chi phí bộ nhớ của một quá trình tính toán là số ô nhớ cần thiết để thực hiện một quá trình tính toán.

Gọi A là một thuật toán, e là dữ liệu vào của bài toán đã được mã hóa bằng cách nào đó. Thuật toán A tính trên dữ liệu vào e phải trả một giá nhất định. Ta ký hiệu:

$\mathbf{t}_A(e)$ là giá thời gian và $\mathbf{I}_A\mathbf{(e)}$ là giá bộ nhớ.

2). Độ phức tạp về bộ nhớ (Trong trường hợp xấu nhất):

$\mathbf{L}_A\mathbf{(n) = max\{ I}_A\mathbf{(e), với |e| \le n\}}$, **n** là "kích thước" đầu vào của thuật toán.

3). Độ phức tạp thời gian (Trong trường hợp xấu nhất):

$\mathbf{T}_A\mathbf{(n) = max\{ t}_A\mathbf{(e), với |e| \le n\}.}$

4). Độ phức tạp tiệm cận: Độ phức tạp PT(n) được gọi là tiệm cận tới hàm f(n) ký hiệu O(f(n)) nếu $\exists$ các số n_0, c mà $PT(n) \le c.f(n)$, $\forall n \ge n_0$.

5). Độ phức tạp đa thức:

Độ phức tạp PT(n) được gọi ***đa thức***, nếu nó ***tiệm cận tới đa thức p(n).***

6). Thuật toán đa thức: Thuật toán được gọi là ***đa thức***, nếu độ phức tạp về thời gian (trong trường hợp xấu nhất) của nó là ***đa thức***.

Nói cách khác:

+ Thuật toán ***thời gian đa thức*** là thuật toán có độ phức tạp thời gian $\mathbf{O(n^t)}$, trong đó t là hằng số.

+ Thuật toán ***thời gian hàm mũ*** là thuật toán có độ phức tạp thời gian $\mathbf{O(t^{f(n)})}$, trong đó t là hằng số và f(n) là đa thức của n.

Thời gian chạy của các lớp thuật toán khác nhau:

Độ phức tạp	Số phép tính ($n = 10^6$)	Thời gian (10^6 ptính/s)
O(1)	1	1 mocro giây
O(n)	10^6	1 giây
$O(n^2)$	10^{12}	11,6 ngày
$O(n^3)$	10^{18}	32 000 năm
$O(2^n)$	10^{301030}	10^{301006} tuổi của vũ trụ

Chú ý

Có người cho rằng ngày nay máy tính với tốc độ rất lớn, không cần quan tâm nhiều tới thuật toán nhanh, chúng tôi xin dẫn một ví dụ đã được kiểm chứng.

- Bài toán xử lý n đối tượng, có ba thuật toán với 3 mức phức tpạ khác nhau sẽ chịu 3 hậu quả như sau: ***Sau 1 giờ:***

Thuật toán A có độ phức tạp O(n) : 3,6 triệu đối tượng.

Thuật toán B có độ phức tạp O(n log n) : 0,2 triệu đối tượng.

Thuật toán C có độ phức tạp $O(2^n)$: 21 đối tượng.

1.1.3.5. Hàm một phía và hàm cửa sập một phía

1). Hàm f(x) được gọi là ***hàm một phía*** nếu tính ***"xuôi"*** y = f(x) thì ***"dễ"***, nhưng tính ***"ngược"*** $x = f^{-1}(y)$ lại rất ***"khó".***

Ví dụ:

Hàm $f(x) = g^x \pmod p$, với p là số nguyên tố lớn, (g là phần tử nguyên thủy mod p) là hàm một phía.

2). Hàm f(x) được gọi là ***hàm cửa sập một phía*** nếu tính y = f(x) thì ***"dễ"***,

tính $x = f^{-1}(y)$ lại rất ***"khó"*** . Tuy nhiên có cửa sổ sập **z** để tính $x = f^{-1}(y)$ là ***"dễ".***

Ví dụ:

Hàm $f(x) = x^a \pmod n$ (với n là tích của hai số nguyên tố lớn n = p*q) là hàm một phía. Nếu chỉ biết a và n thì tính $x = f^{-1}(y)$ rất ***"khó"*** , nhưng nếu biết ***cửa sập*** p và q,, thì tính được $f^{-1}(y)$ là khá ***"dễ".***

1.2. VẤN ĐỀ MÃ HÓA

1.2.1. Giới thiệu về mã hóa

Mật mã được sử dụng để bảo vệ tính bí mật của thông tin khi thông tin được truyền trên các kênh thông tin công cộng như các kênh bưu chính điện thoại, mạng internet v.v… Giả sử một người gửi A muốn gửi đến người nhận B một văn bản (chẳng hạn một bức thư) p, để bảo mật A lập cho p một bản mật mã c, và thay cho việc gửi p, A gửi cho B bản mật mã c, B nhận được c và "giải mã" c để lại được văn bản p như A định gửi. Để A biến p thành c và B biến ngược lại c thành p, A và B phải thỏa thuận trước với nhau các thuật toán lập mã và giải mã, và đặc biệt một khóa mật mã chung K để thực hiện các thuật toán đó.

Người ngoài, không biết các thông tin đó (đặc biệt không biết khóa K), cho dù có lấy trộm được c trên cũng khó tìm được văn bản p mà hai người A và B muốn gửi cho nhau.

1.2.1.1. Khái niệm mật mã

"Mật mã" có lẽ là kỹ thuật được dùng lâu đời nhất trong việc bảo đảm ***"An toàn thông tin".*** Trước đây *"mật mã"* chỉ được dùng trong ngành an ninh quốc phòng, ngày nay việc đảm bảo ***"An toàn thông tin"*** là nhu cầu của mọi ngành, mọi người (do các thông tin chủ yếu được truyền trên mạng công khai), vì vậy kỹ thuật *"mật mã"* là công khai cho mọi người dùng. Điều bí mật nằm ở ***"khóa"*** mật mã.

Hiện nay có nhiều kỹ thuật mật mã khác nhau, mỗi kỹ thuật có ưu, nhược điểm riêng. Tùy theo yêu cầu của môi trường ứng dụng mà ta dùng kỹ thuật này hay kỹ thuật khác. Có những môi trường cần phải an toàn tuyệt đối, bất kể thời gian và chi phí. Có những môi trường lại cần giải pháp dung hòa giữa bảo mật và chi phí thực hiện.

Mật mã cổ điển chủ yếu dùng để "che giấu " dữ liệu. Với mật mã hiện đại, ngoài khả năng "che giấu" dữ liệu, còn dùng để thực hiện: Ký số (ký điện tử), tạo đại diện thông điệp, giao thức bảo toàn dữ liệu, giao thức xác thực thực thể, giao thức xác thực tài liệu, giao thức chứng minh "không tiết lộ thông tin", giao thức thỏa thuận, giao thức phân phối khóa, chống chối cãi trong giao dịch điện tử, chia sẻ bí mật,…

Theo nghĩa hẹp, "mật mã" chủ yếu dùng để bảo mật dữ liệu, quan niệm: Mật mã học là khoa học nghiên cứu mật mã(Tạo mã và phân tích mã)

Phân tích mã là kỹ thuật , nghệ thuật phân tích mật mã, kiểm tra tính bảo mật của nó hoặc phá vỡ sự bí mật của nó. Phân tích mã còn gọi là thám mã.

Theo nghĩa rộng, "mật mã" là một trong những công cụ hiệu quả bảo đảm An toàn thông tin nói chung: bảo mật, bảo toàn, xác thực, chống chối cãi,…

1.2.1.2.Khái niệm mã hóa (Encryption)

1/. *Mã hóa:* là quá trình chuyển thông tin có thể đọc được (gọi là ***bản rõ***) thành thông tin **"khó"** thể đọc được theo cách thông thường (gọi là ***bản mã***).

Đó là một trong những kỹ thuật để bảo mật thông tin.

2/. *Giải mã*: là quá trình chuyển thông tin ngược lại từ ***bản mã*** thành ***bản rõ***.

3/. *Thuật toán mã hóa* hay ***giải mã*** là thủ tục để thực hiện mã hóa hay giải mã.

4/. *Khóa mã hóa* là một giá trị làm cho thuật toán mã hóa thực hiện theo cách riêng biệt và sinh ra bản rõ riêng. Thông thường khóa càng lớn thì bản mã càng an toàn. Phạm vi các giá trị có thể có của khóa được gọi là ***Không gian khóa***.

5/. *Hệ mã hóa* là tập các thuật toán, các khóa nhằm che giấu thông tin, cũng như làm rõ nó.

1.2.1.3. Khái niệm hệ mật mã

Một sơ đồ hệ thống mật mã là bộ năm

S = (P, C, K, E, D) thỏa mãn các điều kiện:

P: là một tập hữu hạn các ký tự bản rõ.

C: là một tập hữu hạn các ký tự bản mã.

K: là một tập hữu hạn các khóa.

E: là một ánh xạ từ KxP vào C, được gọi là phép lập mật mã.

D: là một ánh xạ từ KxC vào P, được gọi là phép giải mã.

Với $k \in K$ ta định nghĩa $e_k \in E$, e_k: $P \rightarrow C$; $d_k \in D$, d_k: $C \rightarrow P$; e_k, d_k được gọi là hàm lập mã và hàm giải mã tương ứng với khóa mật mã k. Các hàm đó phải thỏa mãn hệ thức: $d_k(e_k(x)) = x$ với $\forall x \in P$.

1.2.1.4. Những tính năng của hệ mã hóa

Cung cấp một mức cao về tính bảo mật, toàn vẹn, chống chối bỏ và xác thực.

+ Tính bảo mật: Bảo đảm bí mật cho các thông báo và dữ liệu bằng việc che giấu thông tin nhờ các kỹ thuật mã hóa.

+ Tính toàn vẹn: Bảo đảm với các bên rằng bản tin không bị thay đổi trên đường truyền tin.

+ Chống chối bỏ: Có thể xác nhận rằng tài liệu đã đến từ ai đó, ngay cả khi họ cố gắng từ chối nó.

+ Tính xác thực: Cung cấp hai dịch vụ:

Nhận dạng nguồn gốc của một thông báo, đảm bảo rằng nó là đúng sự thực.

Kiểm tra định danh của người đang đăng nhập hệ thống, tiếp tục kiểm tra đặc điểm của họ trong trường hợp ai đó cố gắng kết nối và giả danh là người sử dụng hợp pháp.

1.2.2. Các phương pháp mã hóa

Hiện nay có 2 loại mã hóa chính: mã hóa khóa đối xứng và mã hóa khóa công khai. **Hệ mã hóa khóa đối xứng** có khóa lập mã và khóa giải mã "giống nhau", theo nghĩa biết được khóa này thì **"dễ"** tính được khóa kia. Vì vậy phải giữ bí mật cả 2 khóa. **Hệ mã hóa khóa công khai** thì có khóa lập mã khác khóa giải mã **(ke ≠ kd)**, biết được khóa nay cũng **"khó"** tính được khóa kia. Vì vậy chỉ cần bí mật khóa giải mã, còn công khai khóa lập mã.

1.2.2.1. Hệ mã hóa khóa đối xứng

1/. Khái niệm

Hệ mã hóa khóa đối xứng là hệ mã hóa mà biết được khóa lập mã thì có thể "dễ" tính được khóa giải mã và ngược lại. Đặc biệt một số hệ mã hóa có khóa lập mã và khóa giải mã trùng nhau (ke = kd), như hệ mã hóa "dịch chuyển" hay DES. Hệ mã hóa khóa đối xứng còn gọi là ***Hệ mã hóa khóa bí mật***, hay ***khóa riêng***, vì phải giữ bí mật cả 2 khóa. Trước khi dùng hệ mã hóa khóa đối xứng, người gửi và người nhận phải thỏa thuận thuật toán mã hóa và ***khóa chung*** (lập mã hay giải mã), khóa phải được bí mật.

Độ an toàn của Hệ mã hóa loại này ***phụ thuộc vào khóa***, nếu để lộ ra khóa này nghĩa là bất kỳ người nào cũng có thể mã hóa và giải mã thông báo trong hệ thống mã hóa.

Sự mã hóa và giải mã của hệ thống mã hóa khóa đối xứng biểu thị bởi:

E_k: P → C và D_k: C → P

2/. Ví dụ:

+ *Hệ mã hóa cổ điển* là Mã hóa khóa đối xứng: dễ hiểu, dễ thực thi, nhưng có độ an toàn không cao. Vì giới hạn tính toán chỉ trông phạm vi bảng chữ cái, sử dụng trong bản tin cần mã, ví dụ Z_{26} nếu dùng các chữ cái tiếng anh. Với hệ mã hóa cổ điển, nếu biết khóa lập mã hay thuật toán lập mã, có thể "dễ" xác định được bản rõ, vì "dễ" tìm được khóa giải mã.

+ *Hệ mã hóa* DES (1973) là Mã hóa khóa đối xứng ***hiện đại***, có độ an toàn cao.

3/. Đặc điểm.

Ưu điểm:

Hệ mã hóa khóa đối xứng mã hóa và giải mã nhanh hơn Hệ mã hóa khóa công khai.

Hạn chế:

(i). Mã hóa khóa đối xứng chưa thật an toàn với lý do sau:

Người mã hóa và người giải mã có "chung" một khóa. Khóa phải được giữ bí mật tuyệt đối, vì biết khóa này "dễ" xác định được khóa kia và ngược lại.

(ii). Vấn đề thỏa thuận khóa và quản lý khóa chung là khó khăn và phức tạp. Người gủi và người nhận phải luôn thống nhất với nhau về khóa. Việc thay đổi khóa là rất khó và dễ bị lộ. Khóa chung phải được gửi cho nhau trên kênh an toàn.

Mặt khác khi hai người (lập mã, giải mã) cùng biết "chung" một bí mật, thì càng khó giữ được bí mật!

4/. Nơi sử dụng hệ mã hóa khóa đối xứng.

Hệ mã hóa khóa đối xứng thường được sử dụng trong môi trường mà khóa chung có thể dễ dàng trao chuyển bí mật, chẳng hạn trong cùng một mạng nội bộ. Hệ mã hóa khóa đối xứng thường dùng để mã hóa những bản tin lớn, vì tốc độ mã hóa và giải mã nhanh hơn hệ mã hóa công khai.

1.2.2.2. Hệ mã hóa khóa phi đối xứng (hệ mã hóa khóa công khai)

1/. Khái niệm

Hệ mã hóa khóa phi đối xứng là Hệ mã hóa có khóa lập mã và khóa giải mã khác nhau (ke ≠ kd), biết được khóa này cũng **"khó"** tính được khóa kia.

Hệ mã hóa này còn được gọi là **Hệ mã hóa khóa công khai** vì:

+ ***Khóa lập mã*** cho ***công khai***, gọi là ***khóa công khai (Public key)***.

+ ***Khóa giải mã*** giữ bí mật, còn gọi là ***khóa riêng (Private key)*** hay ***khóa bí mật.***

Một người bất kỳ có thể dùng khóa công khai để mã hóa bản tin, nhưng chỉ người nào có đúng khóa giải mã thì mới có khả năng đọc được bản rõ.

Hệ mã hóa khóa công khai hay ***Hệ mã hóa phi đối xứng*** do Diffie và Hellman phát minh vào những năm 1970.

2/. Ví dụ

Hệ mã hóa RSA, hệ mã hóa ELGAMAL,….

3/. Đặc điểm.

Ưu điểm:

(i). Thuật toán được viết một lần, công khai cho nhiều lần dùng, cho nhiều người dùng, họ chỉ cần giữ bí mật cho khóa riêng của mình.

(ii). Khi biết các tham số ban đầu của hệ mã hóa, việc tính ra cặp khóa công khai và bí mật phải là "dễ" , tức là trong thời gian đa thức.

Người gửi có bản rõ P và khóa công khai, thì "dễ" tạo ra bản mã C.

Người nhận có bản mã C và khóa bí mật, thì "dễ" giải được thành bản rõ P.

(iii). Người mã hóa dùng khóa công khai, người giải mã giữ khóa bí mật. Khả năng lộ khóa bí mật khó hơn vì chỉ có một người giữ gìn.

Nếu thám mã biết khóa công khai, cố gắng tìm khóa bí mật, thì chúng phải đương đầu với bài toán "khó".

(iv). Nếu thám mã biết khóa công khai và bản mã C, thì việc tìm ra bản rõ P cũng là bài toán "khó", số phép thử là vô cùng lớn, không khả thi.

Nhược điểm:

Hệ mã hóa khóa công khai: mã hóa và giải mã ***chậm hơn*** hệ mã hóa khóa đối xứng.

4/. Nơi sử dụng hệ mã hóa khóa công khai.

Hệ mã hóa khóa công khai thường được sử dụng chủ yếu trên các mạng công khai như Internet, khi mà việc trao đổi chuyển khóa bí mật tương đối khó khăn.

Đặc trưng nổi bật của hệ mã hóa công khai là khóa công khai (public key) và bản mã (ciphertext) đều có thể gửi đi trên một kênh truyền tin ***không an toàn.***

Có biết cả khóa công khai và bản mã, thám mã cũng không dễ khám phá được bản rõ.

Nhưng vì có tốc độ mã hóa và giải mã ***chậm,*** nên hệ mã hóa khóa công khai chỉ dùng để mã hóa những bản tin ngắn, ví dụ như mã hóa khóa bí mật gửi đi.

Hệ mã hóa khóa công khai thường được sử dụng cho cặp người dùng thỏa thuận khóa bí mật của hệ mã hóa khóa riêng.

1.3. VẤN ĐỀ CHỮ KÝ SỐ

1.3.1. Khái niệm "chữ ký số"

1.3.1.1. Giới thiệu "chữ ký số"

Để chứng thực nguồn gốc hay hiệu lực của một tài liệu (ví dụ: đơn xin học, giấy báo nhập học, ...), lâu nay người ta dùng chữ ký ***"tay"***, ghi vào phía dưới của mỗi tài liệu. Như vậy người ký phải ***trực tiếp "ký tay"*** vào tài liệu.

Ngày nay các tài liệu được số hóa, người ta cũng có nhu cầu chứng thực nguồn gốc hay hiệu lực của các tài liệu này. Rõ ràng không thể "ký tay" vào tài liệu, vì chúng không được in ấn trên giấy. Tài liệu "số" (hay tài liệu "điện tử") là một xâu các bít (0 hay 1), xâu bít có thể rất dài (nếu in trên giấy có thể hàng nghìn trang). "Chữ ký" để chứng thực một xâu bít tài liệu cũng không thể là một xâu bít nhỏ đặt phía dưới xâu bít tài liệu. Một "chữ ký" như vậy chắc chắn sẽ bị kẻ gian sao chép để đặt dưới một tài liệu khác bất hợp pháp.

Những năm 80 của thế kỷ 20, các nhà khoa học đã phát minh ra ***"chữ ký số"*** để chứng thực một ***"tài liệu số"***. Đó chính là ***"bản mã"*** của xâu bít tài liệu.

Người ta tạo ra ***"chữ ký số"*** (chữ ký điện tử) trên ***"tài liệu số"*** giống như tạo ra ***"bản mã"*** của tài liệu với "khóa lập mã".

"Chữ ký số" không được sử dụng nhằm bảo mật thông tin mà nhằm bảo vệ thông tin không bị người khác cố tình thay đổi để tạo ra thông tin sai lệch. Nói cách khác, ***"chữ ký số"*** giúp xác định được người đã tạo ra hay chịu trách nhiệm đối với một thông điệp.

Như vậy ***"ký số"*** trên ***"tài liệu số"*** là ***"ký"*** trên từng bít tài liệu. Kẻ gian khó thể giả mạo "chữ ký số" nếu nó không biết "khóa lập mã".

Để kiểm tra một ***"chữ ký số"*** thuộc về một ***"tài liệu số"***, người ta giải mã ***"chữ ký số"*** bằng "khóa giải mã", và so sánh với tài liệu gốc.

Ngoài ý nghĩa để chứng thực nguồn gốc hay hiệu lực của các tài liệu số hóa. Mặt mạnh của ***"chữ ký số"*** hơn ***"chữ ký tay"*** là ở chỗ người ta có thể ***"ký"*** vào tài liệu từ rất xa trên mạng công khai. Hơn thế nữa, có thể ***"ký"*** bằng các thiết bị cầm tay (VD điện thoại di động) tại khắp mọi nơi (Ubikytous) và di động (Mobile), miễn là kết nối được vào mạng. Đỡ tốn bao thời gian, sức lực, chi phí, ...

1.3.1.2. Sơ đồ chữ ký số

Sơ đồ chữ ký là bộ năm **(P, A, K, S, V**), trong đó:

P là tập hữu hạn các văn bản có thể.

A là tập hữu hạn các chữ ký có thể.

K là tập hữu hạn các khóa có thể.

S là tập các thuật toán ký.

V là tập các thuật toán kiểm thử.

Với mỗi khóa $k \in \mathbf{K}$, có thuật toán ký $\mathbf{Sig}_k \in \mathbf{S}$, $\mathbf{Sig}_k : \mathbf{P} \rightarrow \mathbf{A}$,

có thuật toán kiểm tra chữ ký $\mathbf{Ver}_k \in \mathbf{V}$, $\mathbf{Ver}_k : \mathbf{P \times A} \rightarrow$ {đúng, sai},

thỏa mãn điều kiện sau với mọi $\mathbf{x} \in \mathbf{P}$, $\mathbf{y} \in \mathbf{A}$:

$$\mathbf{Ver}_k(\mathbf{x}, \mathbf{y}) = \begin{cases} \text{Đúng, nếu } \mathbf{y} = \mathbf{Sig}_k(\mathbf{x}) \\ \text{Sai, nếu } \mathbf{y} \neq \mathbf{Sig}_k(\mathbf{x}) \end{cases}$$

Chú ý

Thường dùng hệ mã hóa khóa công khai để lập ***"Sơ đồ chữ ký số"***. Ở đây, khóa bí mật **a** dùng làm khóa ***"ký"***, khóa công khai **b** dùng làm khóa kiểm tra ***"chữ ký".*** (Ngược lại với mã hóa, dùng khóa công khai **b** lập mã, khóa bí mật **a** giải mã.)

Điều này là hoàn toàn tự nhiên, ***"ký"*** cần giữ bí mật nên phải dùng khóa bí mật **a** để ***"ký"***. Còn ***"chữ ký"*** là công khai cho mọi người biết, nên họ dùng khóa công khai **b** để kiểm tra.

1.3.2. Phân loại "Chữ ký số"

1.3.2.1. Phân loại chữ ký theo đặc trưng kiểm tra chữ ký

1). Chữ ký khôi phục thông điệp:

Là loại chữ ký, trong đó người gửi chỉ cần gửi ***"chữ ký"*** , người nhận có thể khôi phục lại được thông điệp, đã được ***"ký"*** bởi ***"chữ ký"*** này.

2). Chữ ký đi kèm thông điệp:

Là loại chữ ký, trong đó người gửi chỉ cần gửi ***"chữ ký"***, phải gửi kèm cả thông điệp đã được ***"ký"*** bởi ***"chữ ký"*** này. Ngược lại, sẽ không có được thông điệp gốc.

Ví dụ:Chữ ký Elgamal là chữ ký đi kèm thông điệp, sẽ trình bày trong mục sau.

1.3.2.2. Phân loại chữ ký theo mức an toàn

1). Chữ ký "không thể phủ nhận":

Nhằm tránh việc nhân bản chữ ký để sử dụng nhiều lần, tốt nhất là người gửi tham gia trực tiếp vào việc kiểm thử chữ ký. Điều đó được thực hiện bằng một giao thức kiểm thử, dưới dạng một giao thức mời hỏi và trả lời.

Ví dụ: Chữ ký không phủ định (Chaum- van Antverpen), trình bày trong mục sau.

2). Chữ ký "một lần":

Để bảo đảm an toàn, "Khóa ký" chỉ dùng 1 lần (one - time) trên 1 tài liệu.

Ví dụ: Chữ ký một lần Lamport. Chữ ký Fail - Stop (Van Heyst & Pedersen).

1.3.2.3. Phân loại chữ ký theo ứng dụng đặc trưng

Chữ ký "mù" (Blind Signature).

Chữ ký "nhóm" (Group Signature).

Chữ ký "bội" (Multy Signature).

Chữ ký "mù nhóm" (Blind Group Signature).

Chữ ký "mù bội" (Blind Multy Signature).

1.4. KHÁI NIỆM HÀM BĂM

1.4.1. Vấn đề "Đại diện tài liệu" và "Hàm băm"

1.4.1.1. Một số vấn đề với "chữ ký số"

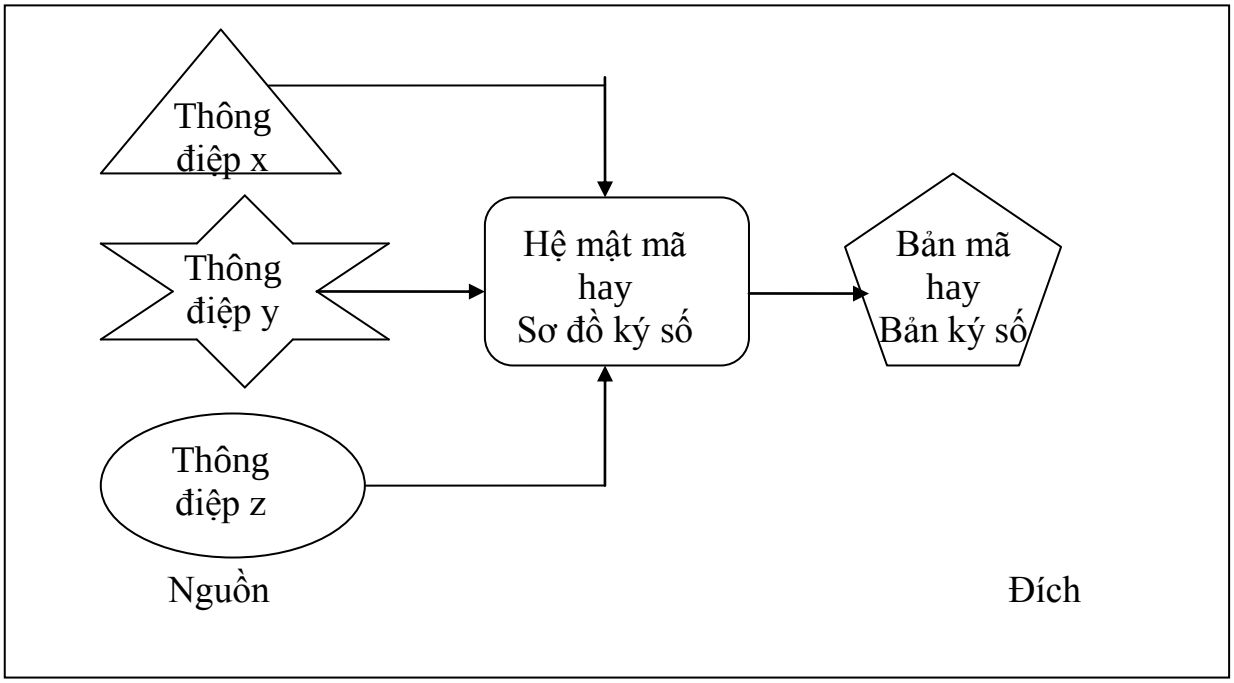
Vấn đề 1:

"Ký số" thực hiện trên từng bít tài liệu, nên độ dài của "***chữ ký số "*** ít nhất cũng bằng độ dài của tài liệu. Một số chữ ký trên bản tin có kích thước gấp đôi bản tin gốc. Ví dụ khi dùng sơ đồ chữ ký DSS để ký vào bản tin có kích thước 160 bit, thì chữ ký số này sẽ có kích thước 320 bit.

Trong khi đó trên thực tế, ta cần phải ký vào các bản tin có kích thước rất lớn, ví dụ vài chục MegaByte (tương ứng hàng nghìn trang tin trên giấy). Như vậy phải tốn nhiều bộ nhớ để lưu trữ "chữ ký", mặt khác tốn nhiều thời gian để truyền "chữ ký" trên mạng.

Vấn đề 2: Với một số sơ đồ chữ ký "an toàn", thì tốc độ ký lại chậm vì chúng dùng nhiều phép tính số học phức tạp như số mũ modulo.

Vấn đề 3: Thực thế có thể xảy ra trường hợp: Với nhiều bản tin đầu vào khác nhau, sử dụng hệ mã hóa hay sơ đồ ký số giống nhau (có thể khác nhau), nhưng lại cho ra bản mã hay chữ ký giống nhau (đó là ánh xạ nhiều – một), như hình dưới. Điều này sẽ dẫn đến phức tạp cho việc xác thực thông tin.



1.4.1.2. Giải quyết vấn đề

Cách 1:

Một cách đơn giản để giải quyết các vấn đề trên với thông điệp có kích thước lớn là ***"chặt"*** bản tin thành nhiều đoạn nhỏ (VD 160 bit), sau đó ký lên các đoạn đó độc lập nhau. Nhưng biện pháp này gặp các vấn đề trên.

Hơn thế nữa còn gặp vấn đề nghiêm trọng hơn. Đó là kết quả sau khi ký, nội dung của thông điệp có thể bị xáo trộn với nhau, hoặc một số đoạn trong chúng có thể bị mất mát. Ta cần phải bảo vệ tính toàn vẹn của bản tin gốc.

Cách 2:

Thay vì phải ký trên tài liệu dài, người ta thường dùng ***"hàm băm"*** để tạo ***"đại diện"*** cho tài liệu, sau đó mới "Ký số" lên ***"đại diện"*** này.

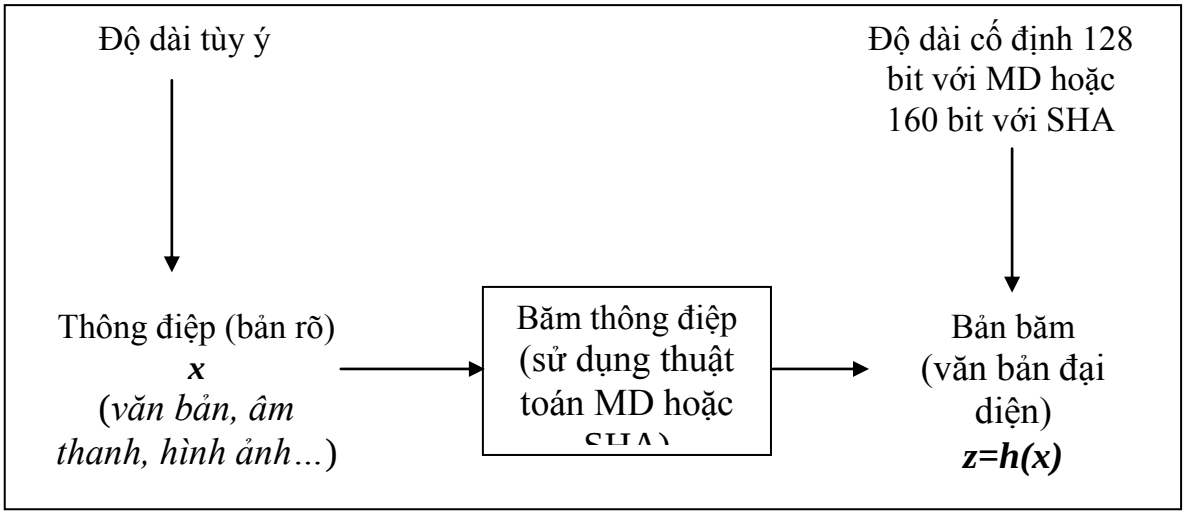
Các tài liệu (bản tin) có thể dưới dạng văn bản , hình ảnh, âm thanh,…và kích thước của chúng tùy ý (vài KB đến vài chục MB), qua các thuật toán băm: MD4, MD5, SHA, các đại diện tương ứng của chúng có kích thước ***cố định***, ví dụ 128 bit với dòng MD, 160 bit với dòng SHA.

"Đại diện" của tài liệu chính là giá trị của ***"hàm băm"*** trên tài liệu, nó còn được gọi là "tóm lược" hay "bản thu gọn" của tài liệu.

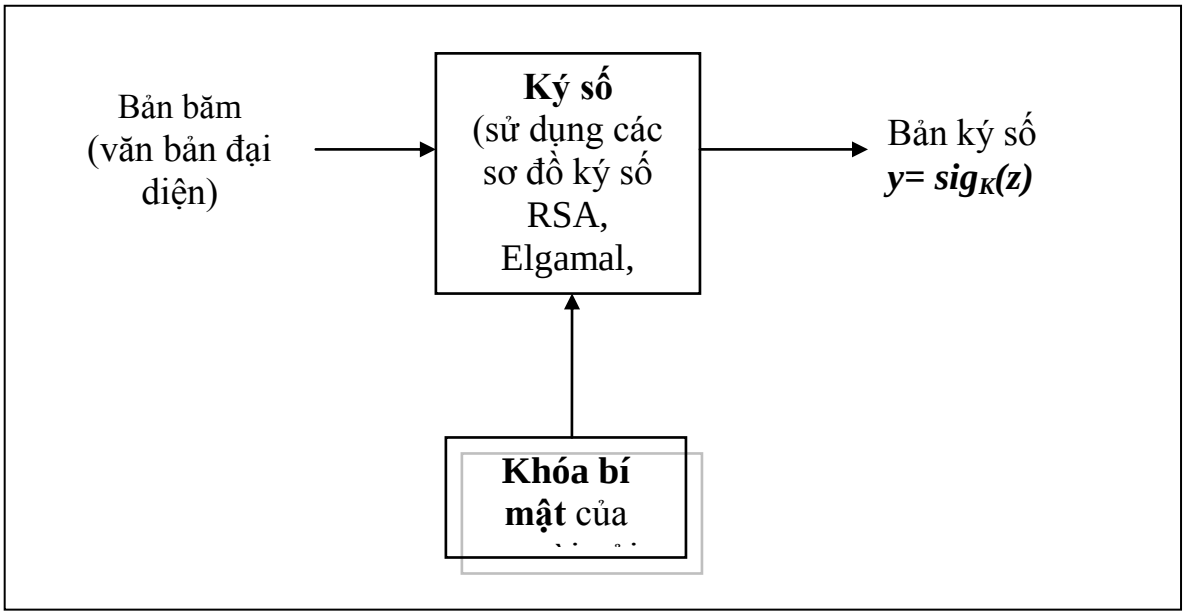
Với mỗi tài liệu (đầu vào), qua ***"hàm băm"*** chỉ có thể tính ra được một ***"đại diện"*** – giá trị băm tương ứng – duy nhất. ***"Đại diện"*** cảu tìa lliệu được xem là ***"đặc thù"*** của tài liệu (thông điệp), giống như dấu vân tay cảu mỗi người.

Trên thực tế, hai tài liệu khác nhau có hai ***"đại diện"*** khác nhau. Như vậy khi đã có ***"đại diện"*** duy nhất cho một tài liệu, thì việc "ký" vào tài liệu, được thay bằng "ký" vào ***"đại diện"*** của nó là hoàn toàn hợp lý. Đó là chưa kể việc tiết kiệm bao nhiêu thời gian cho việc "ký số" , bộ lưu giữ "chữ ký", thời gian truyền "chữ ký" trên mạng,…

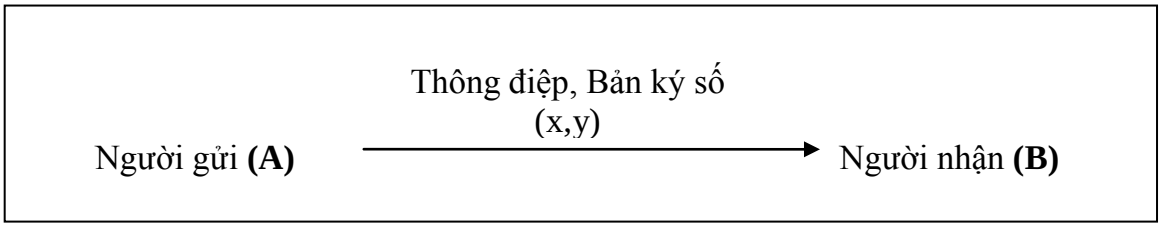
Cơ chế gửi tài liệu cùng "chữ ký" trên nó sử dụng hàm băm được mô tả theo các hình sau.



Băm thông điệp



Ký trên bản băm thông điệp



Truyền thông điệp và chữ ký

1.4.2. Tổng quan về Hàm băm

1.4.2.1. Đặt vấn đề

Trên thực tế, các thông điệp sử dụng chữ ký điện tử có độ dài bất kỳ, thậm chí lên đến vài Megabyte. Trong khi đó, thuật toán ký điện tử lại áp dụng trên các thông điệp có độ dài cố định và thường tương đối ngắn, chẳng hạn như phương pháp DSS sử dụng chữ ký 320 bit trên thông điệp 160 bit.

Để giải quyết vấn đề này, chúng ta có thể chia nhỏ thông điệp cần ký thành các đoạn nhỏ có độ dài thích hợp và ký trên từng mảnh thông điệp này. Tuy nhiên, giải pháp này lại có nhiều khuyết điểm và không thích hợp áp dụng trong thực tế:

+ Nếu văn bản cần được ký quá dài thì số lượng chữ ký được tạo ra sẽ rất nhiều và kết quả nhận được là một thông điệp có kích thước rất lớn. Chẳng hạn như khi sử dụng phương pháp DSS thì thông điệp sau khi được ký sẽ có độ dài gấp đôi văn bản nguyên thủy ban đầu.

+ Hầu hết các phương pháp chữ ký điện tử có độ an toàn cao đều đòi hỏi chi phí tính toán cao và do đó, tốc độ xử lý rất chậm. Việc áp dụng thuật toán tạo chữ ký điện tử nhiều lần trên một văn bản sẽ thực hiện rất chậm.

+ Từng đoạn văn bản sau khi được ký có thể dễ dàng bị thay đổi thứ tự hay bỏ bớt đi mà không làm mất đi tính hợp lệ của văn bản. Việc chia nhỏ văn bản sẽ không thể bảo đảm được tính toàn vẹn của thông tin ban đầu cần được ký.

1.4.2.2. Hàm băm

1). Khái niệm Hàm băm

Hàm băm là thuật toán không dùng khóa để ***mã hóa*** (ở đây dùng thuật ngữ "băm" thay cho " mã hóa "), nó có nhiệm vụ " lọc " (băm) tài liệu (bản tin) và cho kết quả là một giá trị "băm" có kích thước cố định, còn gọi là ***"đại diện tài liệu"*** hay "đại diện bản tin", "đại diện thông điệp".

Hàm băm là ***hàm một chiều***, theo nghĩa giá trị của hàm băm là ***duy nhất,*** và từ giá trị băm này, **"khó thể" suy ngược** lại được nội dung hay độ dài ban đầu của tài liệu gốc.

2). Đặc tính của Hàm băm

Hàm băm **h** là hàm một chiều (One – way Hash) với các đặc tính sau:

1/. Với tài liệu đầu vào (bản tin gốc) **x**, chỉ thu được giá trị băm duy nhất **z=h(x).**

2/. Nếu dữ liệu trong bản tin **x** bị thay đổi hay bị xóa để thành bản tin **x'**, thì giá trị băm **h(x') ≠ h(x).**

Cho dù chỉ là một sự thay đổi nhỏ, ví dụ chỉ thay đổi 1 bit dữ liệu của bản tin gốc **x**, thì giá trị băm **h(x)** của nó cũng vẫn thay đổi. Điều này có nghĩa là: hai thông điệp khác nhau, thì giá trị băm của chúng cũng khác nhau.

3/. Nội dung của bản tin gốc "khó" thể suy ra từ giá trị băm của nó. Nghĩa là: với thông điệp **x** thì "dễ" tính được **z=h(x)**, nhưng lại "khó" tính ngược lại được **x** nếu chỉ biết giá trị băm **h(x)** (Kể cả khi biết hàm băm **h**).

3). Ứng dụng của Hàm băm

1/. Với bản tin dài **x**, thì chữ ký trên **x** cũng sẽ dài, như vậy tốn thời gian " ký ", tốn bộ nhớ để lưu giữ "chữ ký", tốn nhiều thời gian để truyền "chữ ký" trên mạng.

Người ta dùng hàm băm **h** để tạo đại diện bản tin **z=h(x)**, nó có độ dài ngắn (ví dụ 128 bit). Sau đó ký trên **z**, như vậy chữ ký trên **z** sẽ nhỏ hơn rất nhiều so với chữ ký trên bản tin gốc **x.**

2/. Hàm băm dùng để xác định tính toàn vẹn dữ liệu.

3/. Hàm băm dùng để bảo mật một số dữ liệu đặc biệt, ví dụ bảo vệ mật khẩu, bảo vệ khóa mật mã,…

1.4.2.3. Cấu trúc của hàm băm

Hầu hết các hàm băm mật mã đều có cấu trúc giải thuật như sau:

Cho trước một thông điệp M có độ dài bất kỳ. Tùy theo thuật toán được sử dụng, chúng ta có thể cần bổ sung một số bit vào thông điệp này để nhận được thông điệp có độ dài là bội số của một hằng số cho trước. Chia nhỏ thông điệp thành từng khối có kích thước bằng nhau: M_1, M_2, …M_s

Gọi H là trạng thái có kích thước n bit, f là "hàm nén" thực hiện thao tác trộn khối dữ liệu với trạng thái hiện hành

- Khởi gán H_0 bằng một vector khởi tạo nào đó.
- $H_i = f(H_{i-1}, M_i)$ i=1,2,3,…,s

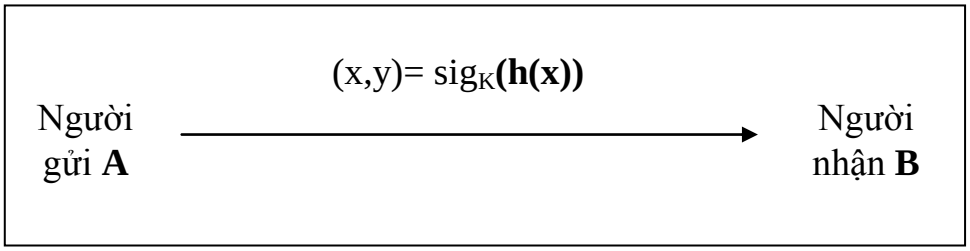
H_s chính là thông điệp rút gọn của thông điệp M ban đầu.

1.4.2.4. Các tính chất của Hàm băm

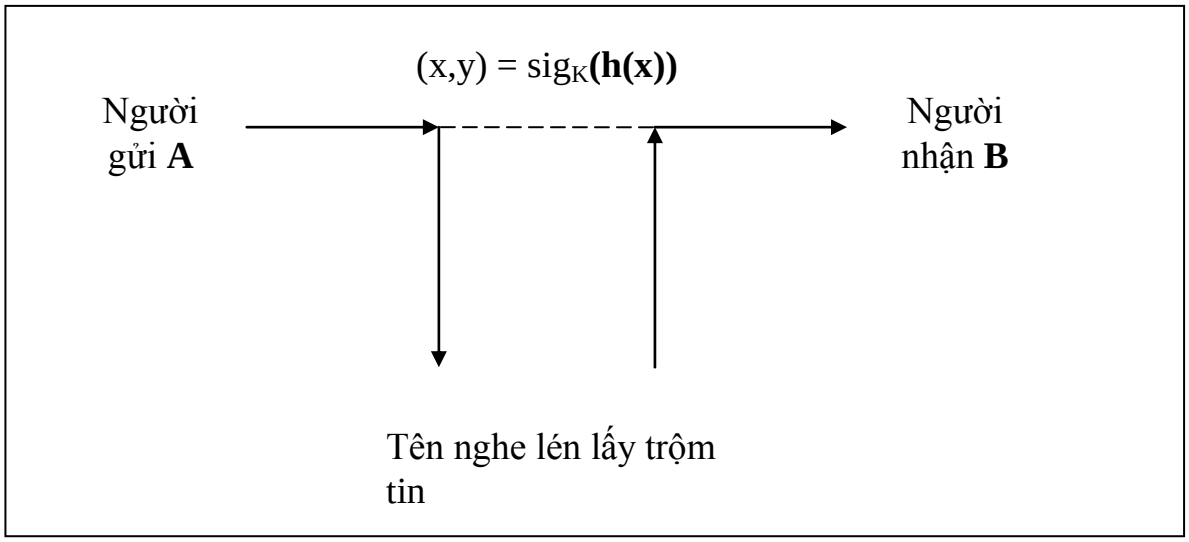
Tính chất 1: Hàm băm **h** là ***không va chạm yếu.***

Ví dụ: Xét kiểu tấn công như sau: *Kiểu tấn công theo tính chất 1.*

*Hình a: Cách đi đúng của thông tin : thông tin được truyền đúng từ A đến B.



*Hình b: Thông tin bị lấy trộm và bị thay đổi trên đường truyền:



**** Kiểu tấn công theo tính chất 1:***

+ Người A gửi cho người B bản tin **(x, y)** với **y**= sig$_K$ (**h(x)**). B không nhận được **(x, y)** vì: trên đường truyền, tin bị lấy trộm. Tên trộm, bằng cách nào đó tìm được một bản tin **x'≠ x** nhưng lại có **h(x') = h(x).** Hắn thay thế **x** bằng **x'**, và chuyển tiếp **(x', y)** cho B.

+ Người B nhận được **(x', y)** và vẫn xác thực được thông tin đúng đắn. Do đó, để tránh kiểu tấn công như trên, hàm **h** phải thỏa mãn tính chất: **không va chạm yếu.**

**** Khái niệm:*** Hàm băm ***không va chạm yếu.***

Hàm băm **h** được gọi là không va chạm yếu, nếu cho trước bức điện **x**, "khó" thể tính toán để tìm ra bức điện **x'≠ x** mà **h(x') = h(x).**

Tính chất 2: Hàm băm ***h*** là ***không va chạm mạnh.***

Ví dụ: Xét kiểu tấn công như sau: *Kiểu tấn công theo tính chất 2.*

+ Đầu tiên, tên giả mạo tìm được hai thông điệp khác nhau **x'** và **x (x' ≠ x)** mà có **h(x) = h(x').** (Coi bức thông điệp **x** là hợp lệ, **x'** là giả mạo).

+ Tiếp theo, hắn thuyết phục ông A ký vào bản tóm lược **h(x)** để nhận được **y**.

Khi đó **(x', y)** là bức điện giả mạo nhưng hợp lệ vì **h(x) = h(x')**.

Để tránh tấn công kiểu này, hàm **h** phải thỏa mãn tính chất: **không va chạm mạnh.**

**** Khái niệm:*** Hàm băm ***không va chạm mạnh.***

Hàm băm b được gọi là **không va chạm mạnh** nếu "khó" thể tính toán để tìm ra hai bức thông điệp khác nhau **x'** và **x (x' ≠ x)** mà có **h(x') = h(x)**.

Tính chất 3: Hàm băm ***h*** là ***hàm một chiều.***

Ví dụ: Xét kiểu tấn công như sau: *Kiểu tấn công theo tính chất 3.*

+ Người A gửi cho B thông tin **(x, z, y)** với **z = h(x), y = $sig_K(z)$**

+ Giả sử tên giả mạo tìm được bản tin **x'**, được tính ngược từ bản tóm lược **z = h(x).**

+ Tên trộm thay thế bản tin **x** hợp lệ bằng bản tin x' giả mạo nhưng lại có **z = h(x').**

Hắn ta ký số trên bản tóm lược **z** của **x'** bằng đúng chữ ký hợp lệ. Nếu làm như vậy thì **(x', z, y)** là bức điện giả mạo nhưng hợp lệ.

Để tránh được kiểu tấn công này, hàm băm **h** cần thỏa mãn ***tính chất một chiều***.

**** Khái niệm:*** Hàm băm ***một chiều.***

Hàm băm **h** được gọi là **hàm một chiều** nếu khi cho trước một bản tóm lược thông báo **z** thì "khó thể" tính toán để tìm ra thông điệp ban đầu **x** sao cho **h(x) = z**.

1.4.2.5. Tính an toàn của hàm băm đối với hiện tượng đụng độ

Hàm băm được xem là an toàn đối với hiện tượng đụng độ khi rất khó tìm được hai thông điệp có cùng giá trị băm.

Nhận xét: Trong một tập hợp mà các phần tử mang một trong N giá trị cho trước với xác suất bằng nhau, chúng ta cần khoảng N phép thử ngẫu nhiên để tìm ra một cặp phần tử có cùng giá trị.

Như vậy, phương pháp hàm băm được xem là an toàn đối với hiện tượng đụng độ nếu chưa có phương pháp tấn công nào có thể tìm ra cặp thông điệp có cùng giá trị hàm băm với số lượng tính toán ít hơn đáng kể so với ngưỡng $2^n/2$, (n là kích thước (tính bằng bit) của giá trị băm.)

Phương pháp tấn công dựa vào đụng độ:

+ Tìm ra 2 thông điệp có nội dung khác nhau, nhưng cùng giá trị băm.

+ Ký trên một thông điệp, sau đó, người ký sẽ không thừa nhận đây là chữ ký của mình mà nói rằng mình đã ký trên một thông điệp khác.

+ Như vậy, cần phải chọn 2 thông điệp "đụng độ" với nhau trước khi ký.

1.4.3. Các loại Hàm băm.

Hàm băm mật mã là hàm toán học chuyển đổi một thông điệp có độ dài bất kỳ thành một dãy bit có độ dài cố định (tùy thuộc vào thuật toán băm). Dãy bit này được gọi là thông điệp rút gọn (message digest) hay giá trị băm (hash value), đại diện cho thông điệp ban đầu.

Dễ dàng nhận thấy rằng hàm băm h không phải là một song ánh. Do đó, với thông điệp x bất kỳ, tồn tại thông điệp $x' \neq x$ sao cho $h(x)= h(x')$. Lúc này, ta nói rằng "có sự đụng độ xảy ra".

Hàm băm h được gọi là an toàn (hay "ít bị đụng độ"), khi không thể xác định được (bằng cách tính toán) cặp thông điệp x và x' thỏa mãn $x \neq x'$ và $h(x) = h(x')$. Trên thực tế, các thuật toán băm là hàm một chiều, do đó, rất khó để xây dựng lại thông điệp ban đầu từ thông điệp rút gọn.

Hàm băm giúp xác định tính toàn vẹn dữ liệu của thông tin: mọi thay đổi, dù là rất nhỏ, trên thông điệp cho trước, ví dụ như đổi giá trị 1 bit, đều làm thay đổi thông điệp rút gọn tương ứng. Tính chất này hữu ích trong việc phát sinh, kiểm tra chữ ký điện tử, các đoạn mã chứng nhận thông điệp, phát sinh số ngẫu nhiên, tạo ra khóa cho quá trình mã hóa…

Hàm băm là nền tảng cho nhiều ứng dụng mã hóa. Có nhiều thuật toán để thực hiện hàm băm, trong số đó, hàm băm SHA-1 và MD5 được sử dụng khá phổ biến từ thập niên 1990 đến nay.

1/. Hàm băm MD4 (Message Digest 4) và MD5 (Message Digest 5)

Hàm băm MD4 được Giáo sư Ron Rivest đề nghị vào năm 1990. Vào năm 1992, phiên bản cải tiến MD5 của thuật toán này ra đời. Thông điệp rút gọn có độ dài 128 bit.

Năm 1995, Hans Dobbertin đã chỉ ra sự đụng độ ngay chính trong bản thân hàm nén của giải thuật (mặc dù chưa thật sự phá vỡ được giải thuật). Năm 2004, nhóm tác giả Xiaoyun Wang, Dengguo Feng, Xuejia Lai và Hongbo Yu đã công bố kết quả về việc phá vỡ thuật toán MD4 và MD5 bằng phương pháp tấn công đụng độ[2] [49].

2/. Hàm băm SHS (Secure Hash Standard)

Hàm băm SHS do NIST và NSA xây dựng được công bố trên Federal Register vào ngày 31/ 1/ 1992, và sau đó chính thức trở thành phương pháp chuẩn từ ngày 13/ 5/ 1993. Thông điệp rút gọn có độ dài 160 bit.

Ngày 26/08/2002, Viện Tiêu chuẩn và Công nghệ quốc gia của Hoa Kỳ (National Institute of Standard and Technology - NIST) đã đề xuất hệ thống chuẩn hàm băm an toàn (Secure Hash Standard) gồm 4 thuật toán hàm băm SHA-1, SHA-256, HA-384, SHA-512. Đến 25/03/2004, NIST đã chấp nhận thêm thuật toán hàm băm SHA-224 vào hệ thống chuẩn hàm băm. Các thuật toán hàm băm do NIST đề xuất được đặc tả trong tài liệu FIPS180-2 [24].

Chương 2. TỔNG QUAN VỀ XÁC THỰC ĐIỆN TỬ

2.1. VẤN ĐỀ XÁC THỰC ĐIỆN TỬ

2.1.1. Khái niệm xác thực

2.1.1.1. Xác thực theo nghĩa thông thường

Xác thực là một chứng thực một cái gì đó (hoặc một người nào đó) đáng tin cậy, có nghĩa là, những lời khai báo do người đó đưa ra hoặc về vật đó là sự thật.

Xác thực một đối tượng còn có nghĩa là công nhận nguồn gốc (provenance) của đối tượng, trong khi, xác thực một người thường bao gồm việc thẩm tra nhận dạng họ. Việc xác thực thường phụ thuộc vào một hoặc nhiều nhân tố xác thực (authentication factors) để minh chứng cụ thể.

2.1.1.2. Xác thực điện tử

Xác thực trong an ninh máy tính là một quy trình nhằm cố gắng xác minh nhận dạng số (digital identity) của phần truyền gửi thông tin (sender) trong giao thông liên lạc chẳng hạn như một yêu cầu đăng nhập. Phần gửi cần phải xác thực có thể là một người dùng một máy tính, bản thân một máy tính hoặc một chương trình máy tính (computer program).

Ngược lại sự tin cậy mù quáng (blind credential) hoàn toàn không thiết lập sự đòi hỏi nhận dạng, song chỉ thiết lập quyền hoặc địa vị hẹp hòi của người dùng hoặc của chương trình ứng dụng mà thôi.

Trong một mạng lưới tín nhiệm, việc "xác thực" là một cách để đảm bảo rằng người dùng chính là người mà họ nói họ là, và người dùng hiện đang thi hành những chức năng trong một hệ thống, trên thực tế, chính là người đã được ủy quyền để làm những việc đó.

2.1.2. Phân loại xác thực điện tử

2.1.2.1. Xác thực dữ liệu

1). Xác thực thông điệp (Message Authentication)

2). Xác thực giao dịch (Transaction Authentication)

3). Xác thực khóa (Key Authentication)

4). Xác thực nguồn gốc dữ liệu (Source của Data)

5). Xác thực bảo đảm toàn vẹn dữ liệu (Data Integrity)

2.1.2.2. Xác thực thực thể

1). Xác thực dựa vào thực thể: Biết cái gì (Something Known)

2). Xác thực dựa vào thực thể: Sở hữu cái gì (Something Possessed)

3). Xác thực dựa vào thực thể: Thừa hưởng cái gì (Something Inherent)

2.2. XÁC THỰC DỮ LIỆU

2.2.1. Xác thực thông điệp

1). Khái niệm

Xác thực thông điệp hay ***Xác thực tính nguyên bản*** của dữ liệu (Data Origin Authentication) là một kiểu ***xác thực đảm bảo một thực thể*** được chứng thực là **nguồn gốc thực sự** tạo ra dữ liệu này ở một thời điểm nào đó.

Xác thực thông điệp bao hàm cả **tính toàn vẹn dữ liệu**, nhưng ***không đảm bảo*** tính duy nhất và sự phù hợp về thời gian của nó.

2.2.2. Xác thực giao dịch

1). Khái niệm

Xác thực giao dịch là ***Xác thực thông điệp*** cộng thêm việc **đảm bảo tính duy nhất** (Uniqueness) và sự phù hợp về **thời gian** (Timeliness) của nó.

Xác thực giao dịch liên quan đến việc sử dụng các tham số thời gian (**TVB-**Time Variant Parameters).

Transaction Authentication = Message Authentication + TVB

Xác thực giao dịch "mạnh hơn" Xác thực thông điệp.

2) Ví dụ

Một thông điệp gửi đi có thể đã bị chặn và phát lại (tương tự như việc đổi tiền bằng một bản sao của Séc). Để ngăn chặn tình huống này, người gửi và người nhận có thể gắn vào thông điệp ***nhãn thời gian*** hoặc ***số thông điệp.***

Số thông điệp là một con số được gắn vào thông điệp. Nó có thể chỉ dùng một lần duy nhất, giá trị không lặp lại, hoặc dùng dưới dạng dãy số tuần tự (Sequence Numbers).

Thám mã không có cách nào để biết được các bit của số này nằm ở vị trí nào trong thông điệp, hoặc không thể biết cách thay đổi các bit để tạo ra dạng mã hóa của số tiếp sau, hoặc không thể biết cách thay đổi các bit này mà không làm gián đoạn việc giải mã phần còn lại của thông báo.

Số thông báo này khó thể bị thay thế, thay đổi hoặc giả mạo. Người nhận phải duy trì việc đếm các số thông báo đã nhận được. Nếu hai người sử dụng một tập các số thì người nhận có thể biết được có thông báo nào trước thông báo hiện thời đã bị mất hoặc bị chậm trễ, vì số được mã hóa của thông báo hiện thời phải lớn hơn số được mã hóa của thông báo trước.

Nếu người gửi có nhiều thông báo thì có thể số thông báo sẽ quá dài. Vì thế, người ta thường đặt lại bộ đếm số thông báo trước khi nó đạt tới giá trị lớn nào đó. Lúc này tất cả bên thu phải được thông báo rằng, số thông báo được gửi tiếp theo sẽ được đặt lại về một số nhỏ (chẳng hạn là 0).

Nhãn thời gian (TimeStamp) là dấu hiệu về thời gian và ngày tháng lấy từ đồng hồ hệ thống hoặc đồng hồ địa phương. Bên gửi: gửi dữ liệu gắn TimeStamp đi. Bên nhận: nhận được dữ liệu, tiến hành lấy TimeStamp tại thời điểm hiện thời, trừ đi TimeStamp nhận được. Dữ liệu nhận được sẽ được chấp nhận nếu:

Độ lệch giữa 2 TimeStamp nằm trong khoảng chấp nhận được.

Không có thông báo nào có cùng TimeStamp được nhận trước đó từ cùng một người gửi. Điều này được thực hiện bằng cách bên nhận lưu giữ danh sách các TimeStamp từ người gửi để kiểm tra hoặc ghi lại TimeStamp gần nhất và chỉ chấp nhận TimeStamp có giá trị lớn hơn.

Như vậy, bên nhận phải đồng bộ và bảo mật về thời gian rất chặt chẽ với bên gửi, ngoài ra phải lưu giữ các TimeStamp.

2.2.3. Xác thực khóa

+ Xác thực không tường minh khóa (Implicit Key Authentication):

Một bên được đảm bảo rằng chỉ có bên thứ hai (và có thể có thêm các bên tin cậy-Trusted Parties) là ***có thể*** truy cập được khóa mật.

+ Khẳng định (Xác nhận) khóa (Key Confirmation):

Một bên được đảm bảo rằng bên thứ hai ***chắc chắn*** đã sở hữu khóa mật.

+ Xác thực tường minh khóa (Explicit Key Authentication)

Bao gồm cả 2 yếu tố trên, nó chứng tỏ được định danh của bên có khóa đã cho.

Chú ý:

Xác thực khóa tập trung vào định danh bên thứ hai có thể truy cập khóa hơn là giá trị của khóa. Khẳng định khóa lại tập trung vào giá trị của khóa.

Ta gọi ngắn gọn Explicit Key Authentication là Key Authentication.

Chú ý:

Xác thực dữ liệu đã bao gồm tính toán vẹn dữ liệu. Ngược lại thì không.

+ Đảm bảo xác thực nguồn gốc dữ liệu➔ phải đảm bảo tính toàn vẹn dữ liệu.

+ Đảm bảo tính toàn vẹn dữ liệu //➔ đảm bảo xác thực nguồn gốc dữ liệu

2.2.4. Xác thực nguồn gốc dữ liệu

Công cụ: Dùng chữ ký số, hàm băm, thủy vân ký.

2.2.5. Xác thực bảo đảm toàn vẹn dữ liệu

Công cụ: Dùng chữ ký số, hàm băm, thủy vân ký, mã xác thực.

2.3. XÁC THỰC THỰC THỂ

Xác thực thực thể (hay Định danh thực thể) là xác thực định danh của một đối tượng tham gia giao thức truyền tin.

Thực thể hay đối tượng có thể là người dùng, thiết bị đầu cuối,…

Tức là: Một thực thể được xác thực bằng định danh của nó đối với thực thể thứ hai trong một giao thức, và bên thứ hai đã thực sự tham gia vào giao thức.

2.3.1. Xác thực dựa vào thực thể: Biết cái gì (Something Known)

Chẳng hạn, mật khẩu, mật khẩu ngữ (pass phrase) hoặc số định danh cá nhân (personal identification number - PIN)

Định danh cá nhân (PIN- Personal Identifier Number) thường gắn với **Something Possessed** để tăng tính bảo mật.

Chú ý:

"Biết cái gì" được dùng trong ***Giao thức định danh,*** đó là ***cơ chế hỏi – đáp*** (Challenge-Response):

Một thực thể (Claimant) chứng tỏ định danh của nó đối với thực thể khác (Verifier) bằng các biểu lộ hiểu biết về một thông tin mật liên quan nào đó cho Verifier, mà không bộc lộ bí mật của nó cho Verifier trong suốt giao thức.

Cơ chế đó gọi là ***"Chứng minh không tiết lộ thông tin".***

Trong cơ chế hỏi – đáp thường dùng một người được uỷ quyền có tín nhiệm TA (Trusted Autherity) để tạo các tham số chung, các thuật toán ký, kiểm tra chữ ký và các chuỗi định danh, dấu xác nhận cho các bên tham gia.

2.3.1.1 .Xác thực dựa trên User name và Password

Sự kết hợp của tên người dùng và mật khẩu là cách xác thực cơ bản nhất. Với kiểu xác thực này, chứng từ ủy nhiệm người dùng được đối chiếu với chứng từ được lưu trữ trên cơ sở dữ liệu hệ thống, nếu trùng khớp tên người dùng và mật khẩu, thì người dùng được xác thực và nếu không người dùng bị cấm truy cập. Phương thức này không an toàn lắm vì chứng từ xác nhận người dùng được gửi đi xác thực trong tình trạng "plain text", tức là không được mã hóa và có thể bị tóm trên đường truyền.

2.3.1.2. Giao thức Chứng thực bắt tay thách thức - Challenge Handshake Authentication Protocol (CHAP)

Giao thức Chứng thực "Bắt tay Thách thức" cũng là mô hình xác thực dựa trên tên người dùng/ mật khẩu. Khi người dùng cố gắng đăng nhập, server đảm nhiệm vai trò xác thực sẽ gửi một thông điệp thử thách (challenge message) trở lại máy tính người dùng. Lúc này máy tính người dùng sẽ phản hồi lại tên người dùng và mật khẩu được mã hóa. Server xác thực sẽ so sánh phiên bản xác thực người dùng được lưu giữ với phiên bản mã hóa vừa nhận. Nếu trùng khớp, ngươi dùng sẽ được xác thực. Bản thân mật khẩu không bao giờ được gửi qua mạng. Phương thức CHAP thường được sử dụng khi người dùng đăng nhập vào các "remote servers" của công ty chẳng hạn như RAS server. Dữ liệu chứa mật khẩu được mã hóa gọi là mật khẩu băm (hash password).

2.3.2. Xác thực dựa vào thực thể: Sở hữu cái gì (Something Possessed)

Ví dụ như sở hữu **khóa bí mật để ký điện tử**

Ví dụ như sở hữu thẻ từ (Magnetic-striped Card), thẻ tín dụng (Credit Card), thẻ thông minh (Smart Card), chứng minh thư (ID card), chứng chỉ an ninh (security token), chứng chỉ phần mềm (software token) hoặc điện thoại di động (cell phone)

2.3.2.1. Phương pháp xác thực Kerberos (Kerberos authentication)

Là phương pháp dùng một Server trung tâm để kiểm tra việc xác thực người dùng và cấp phát thẻ thông hành (service tickets) để người dùng có thể truy cập vào tài nguyên. Kerberos là một phương thức rất an toàn trong xác thực bởi vì dùng cấp độ mã hóa rất mạnh. Kerberos cũng dựa trên độ chính xác của thời gian xác thực giữa Server và máy khách, do đó cần đảm bảo có một "time server" hoặc "authenticating servers" được đồng bộ thời gian từ các "Internet time server". Kerberos là nền tảng xác thực chính của nhiều OS như Unix, Windows.

2.3.2.2. Phương pháp Tokens

Là phương tiện vật lý như các thẻ thông minh (smart cards) hoặc thẻ đeo của nhân viên (ID badges) chứa thông tin xác thực. Tokens có thể lưu trữ số nhận dạng cá nhân - personal identification numbers (PINs), thông tin về người dùng, hoặc mật khẩu.

Các thông tin trên token chỉ có thể được đọc và xử lý bởi các thiết bị đặc dụng, ví dụ như thẻ smart card được đọc bởi đầu đọc smart card gắn trên máy tính, sau đó thông tin này được gửi đến "authenticating server". Tokens chứa chuỗi text hoặc giá trị số duy nhất thông thương mỗi giá trị này chỉ sử dụng một lần.

Ví dụ về Smart cards:

Smart cards là ví dụ điển hình về xác thực tokens (token - based authentication). Một smart card là một thẻ nhựa có gắn một chip máy tính lưu trữ các loại thông tin điện tử khác nhau. Nội dung thông tin của card được đọc với một thiết bị đặc biệt.

2.3.3. Xác thực dựa vào thực thể: Thừa hưởng cái gì (Something Inherent)

Chẳng hạn, vết lăn tay hoặc mẫu hình võng mạc mắt, chuỗi DNA (có đủ loại định nghĩa về cái nào là cái thích hợp và đầy đủ), mẫu hình về giọng nói (cũng có vài định nghĩa ở đây), sự xác minh chữ ký, tín hiệu sinh điện đặc hữu do cơ thể sống tạo sinh (unique bio-electric signals), hoặc những biệt danh sinh trắc (biometric identifier).

2.3.3.1. *Phương pháp Biometrics (phương pháp nhận dạng sinh trắc học)*

Mô hình xác thực dựa trên đặc điểm sinh học của từng cá nhân:

+ Quét dấu vân tay (fingerprint scanner)

+ Quét võng mạc mắt (retinal scanner)

+ Nhận dạng giọng nói (voice-recognition)

+ Nhận dạng khuôn mặt (facerecognition)

Vì nhận dạng sinh trắc học hiện rất tốn kém chi phí khi triển khai nên chưa được sử dụng rộng rãi như các phương thức xác thực khác.

Trong lịch sử, vết lăn tay là một phương pháp xác minh đáng tin nhất, song trong những vụ kiện tòa án (court cases) gần đây ở Mỹ và ở nhiều nơi khác, người ta đã có nhiều nghi ngờ về tính đáng tin cậy của dấu lăn tay. Những phương pháp sinh trắc khác được coi là khả quan hơn (quét võng mạng mắt và quét vết lăn tay là vài ví dụ), song có những bằng chứng chỉ ra rằng những phương pháp này, trên thực tế, dễ bị giả mạo.

Chương 3. PHƯƠNG PHÁP XÁC THỰC THÔNG ĐIỆP

3.1. XÁC THỰC THÔNG ĐIỆP BẰNG CHỮ KÝ SỐ

3.1.1. Ý tưởng chính của phương pháp xác thực bằng chữ ký số

1/. An gửi cho Thu cặp tin (X, Y), trong đó X là bản tin, Y là chữ ký số của bản tin X. Tức là $Y = Sig_k(X)$, Sig_k là thuật toán ký với khóa k.

2/. Khi nhận được (X, Y), Thu tiến hành kiểm tra chữ ký bằng thuật toán Ver (X,Y). Nếu $Ver_k(X, Y)$ = đúng thì Thu chắc chắn rằng X được bảo toàn.

Có hai khả năng:

+ An sử dụng chữ ký khôi phục được thông điệp gốc (chữ ký RSA)

+ An sử dụng chữ ký không khôi phục được thông điệp gốc (chữ ký ELGAMAL, chữ ký DSS).

Ta lấy chữ ký RSA và chữ ký ELGAMAL làm ví dụ cho hai khả năng trên.

3.1.2. Phương pháp chữ ký điện tử RSA

3.1.2.1. Sơ đồ chữ ký

1/. Sơ đồ (đề xuất năm 1978)

**** Tạo cặp khóa (bí mật, công khai)* (a, b):**

Chọn bí mật số nguyên tố lớn p, q, tính n = p * q, công khai n, đặt P = C = Z_n

Tính bí mật $\phi(n) = (p-1).(q-1)$. Chọn khóa công khai $b < \phi(n)$, nguyên tố với $\phi(n)$.

Khóa bí mật a là phần tử nghịch đảo của b theo mod $\phi(n)$: $a*b \equiv 1 \pmod{\phi(n)}$.

Tập khóa (bí mật, công khai) $K = \{(a, b)/\ a,b \in Z_n, a*b \equiv 1(\text{mod } \phi(n))\}$.

**** Ký số:*** Chữ ký trên $x \in P$ là $y = Sig_k(x) = x^a \pmod n$, $y \in A$. (R1)

**** Kiểm tra chữ ký:*** $Ver_k(x, y) =$ đúng $\Leftrightarrow x \equiv y^b \pmod n$. (R2)

2/. Chú ý:

- Việc ký chẳng qua là mã hóa, việc kiểm thử lại chính là việc giải mã:
- Việc "ký số" vào x tương ứng với việc "mã hóa" tài liệu x.
- Kiểm thử chữ ký chính là việc giải mã "chữ ký", để kiểm tra xem tài liệu đã giải mã có đúng là tài liệu trước khi ký không. Thuật toán và khóa kiểm thử "chữ ký" là công khai, ai cũng có thể kiểm thử chữ ký được.

3.1.2.2. Ví dụ

1/. An muốn gửi cho Thu bản rõ x = 2. An tiến hành ký trên x = 2 như sau:

* Tạo cặp khóa (bí mật, công khai) (a, b):

Chọn bí mật số nguyên tố p = 3, q = 5, tính n = p*q = 3*5 = 15, công khai n.

Đặt P = C = Z_n. Tính bí mật ϕ(n) = (p-1).(q-1) = 2*4 = 8.

Chọn khóa công khai b = 3 < ϕ(n), nguyên tố với ϕ(n) = 8.

Khóa bí mật a = 3, là phần tử nghịch đảo của b theo mod ϕ(n): a*b ≡ 1 (mod ϕ(n)).

* Ký số: Chữ ký trên x = 2 ∈ P là

y = $Sig_k(x) = x^a \pmod n = 2^3 \pmod{15} = 8$, y ∈ A.

An sẽ gửi cho Thu cả bản rõ x = 2 và chữ ký y = 8

2/. Thu sau khi nhận được bản rõ x = 2 và chữ ký y = 8 sẽ thực hiện kiểm tra chữ ký

* Kiểm tra chữ ký: $Ver_k(x, y)$ = đúng ⇔ $x \equiv y^b \pmod n$

⇔ $2 \equiv 8^b \pmod{15}$.

Nếu kết quả $Ver_k(x, y)$ đúng bằng 2 như bản rõ thì Thu có thể xác định chữ ký là đúng của bản rõ An gửi.

3.1.3. Phương pháp chữ ký điện tử ElGamal

Chữ ký điện tử ElGamal được giới thiệu vào năm 1985. Sau đó, Viện Tiêu chuẩn và Công nghệ Quốc gia Hoa Kỳ (NIST) đã sửa đổi bổ sung phương pháp này thành chuẩn chữ ký điện tử (Digital Signature Standard– DSS).

3.1.3.1. Bài toán logarit rời rạc

Bài toán logarit rời rạc: Cho số nguyên tố p, gọi $\alpha \in Z_p$ là phần tử sinh (generator) và $\beta \in Z_p^*$. Cần xác định số nguyên dương $a \in Z_{p-1}$ sao cho

$$\alpha^a \equiv \beta \pmod{p}$$

Khi đó, a được ký hiệu là $\log_\alpha \beta$

Trên thực tế, bài toán logarit rời rạc thuộc nhóm bài toán *NP*, nói cách khác, chưa có thuật toán thời gian đa thức nào giải quyết được vấn đề này. Với p có tối thiểu 150 chữ số và $p - 1$ có thừa số nguyên tố đủ lớn, phép toán lũy thừa modulo p có thể xem như là hàm 1 chiều hay việc giải bài toán logarit rời rạc trên Z_p xem như không thể thực hiện được.

3.1.3.2. Sơ đồ chữ ký

1/. Sơ đồ (Elgamal đề xuất năm 1985)

**** Tạo cặp khóa (bí mật, công khai)* (a, h):**

Chọn số nguyên tố p sao cho bài toán logrit rời rạc trong Z_p là "khó" giải.

Chọn phần tử nguyên thủy $g \in Z_p^*$. Đặt $P = Z_p^*$, $A = Z_p^* \times Z_{p-1}$.

Chọn khóa bí mật là $a \in Z_p^*$. Tính khóa công khai $h \equiv g^a \bmod p$.

Định nghĩa tập khóa: $K = \{(p, g, a, h): h \equiv g^a \bmod p\}$.

Các giá trị p, g, h được công khai, phải giữ bí mật a.

**** Ký số:*** Dùng 2 khóa ký: khóa a và khóa ngẫu nhiên bí mật $r \in Z_{p-1}^*$.

(Vì $r \in Z_{p-1}^*$, nên nguyên tố cùng p -1, do đó tồn tại $r^{-1} \bmod (p-1)$).

Chữ ký trên $x \in P$ là $y = Sig_k(x,r) = (\gamma, \delta)$, $y \in A$ (E1)

Trong đó $\gamma \in Z_p^*$, $\delta \in Z_{p-1}$:

$\gamma = g^r \bmod p$ và $\delta = (x - a*\gamma)* r^{-1} \bmod (p-1)$

**** Kiểm tra chữ ký:***

$Ver_k(x,\gamma,\delta)$ = đúng $\Leftrightarrow$ $h^{\gamma} * \gamma^{\delta} \equiv g^{x} \bmod p$. (E2)

2/. Chú ý: Nếu chữ ký được tính đúng, kiểm thử sẽ thành công vì

$h^{\gamma} * \gamma^{\delta} \equiv g^{a\gamma} * g^{r*\delta} \bmod p \equiv g^{(a\gamma + r*\delta)} \bmod p \equiv g^{x} \bmod p$.

Do $\delta = (x - a * \gamma) * r^{-1} \bmod (p-1)$ nên

$(a * \gamma + r * \delta) \equiv (a * \gamma + r (x - a * \gamma) * r^{-1}) \equiv (a * \gamma + x - a * \gamma) \equiv x \bmod (p-1)$.

3.1.3.3. Ví dụ

1/. An gửi cho Thu bản rõ x = 112

**** Tạo cặp khóa (bí mật, công khai)* (a, h):**

Chọn số nguyên tố p = 463. Đặt P = Z_p^*, A = $Z_p^* \times Z_{p-1}$.

Chọn phần tử nguyên thủy $g = 2 \in Z_p^*$.

Chọn khóa bí mật là $a = 211 \in Z_p^*$.

Tính khóa công khai $h \equiv g^a \bmod p = 2^{211} \bmod 463 = 249$.

Định nghĩa tập khóa: $K = \{(p, g, a, h): h \equiv g^a \bmod p\}$.

Các giá trị p, g, h được công khai, phải giữ bí mật a.

**** Ký số:*** Chọn ngẫu nhiên bí mật $r = 235 \in Z_{p-1}^*$. Khóa ký là (a, r).

Vì $r \in Z_{p-1}^*$, nên tồn tại $r^{-1} \bmod (p-1)$. Cụ thể:

UCLN(r, p-1) = UCLN(235, 462) = 1, nên $r^{-1} \bmod (p-1) = 235^{-1} \bmod 462 = 289$.

Chữ ký trên dữ liệu x = 112 là $(\gamma, \delta) = (16, 108)$, trong đó:

$\gamma = g^r \bmod p = 2^{235} \bmod 463 = 16$

$\delta = (x - a*\gamma) * r^{-1} \bmod (p-1) = (112 - 211 * 16) * 289 \bmod 462 = 108$

2/. Thu nhận được và tiến hành xác thực

**** Kiểm tra chữ ký:*** $Ver_k(x,\gamma,\delta)$ = đúng $\Leftrightarrow$ $h^{\gamma} * \gamma^{\delta} \equiv g^{x} \bmod p$.

$h^{\gamma} * \gamma^{\delta} = 249^{16} * 16^{108} \bmod 463 = 132$

$g^{x} \bmod p = 2^{112} \bmod 463 = 132$.

Hai giá trị đó bằng nhau, như vậy chữ ký là đúng.

3.2. XÁC THỰC THÔNG ĐIỆP BẰNG HÀM BĂM

3.2.1. Ý tưởng chính của phương pháp xác thực bằng hàm băm

1/. A gửi cho B cặp tin (X, Y), trong đó X là bản tin, Y là đại diện bản tin X, tức là Y= h (X), h là hàm băm.

2/. Khi nhận được (X, Y), B tính lại h(X) = Z.

Nếu Z = Y, thì B chắc chắn rằng X được bảo toàn, không bị sửa đổi trên đường truyền.

3.2.2. Hàm băm MD4

3.2.2.1. Khái niệm "Thông điệp đệm"

"Thông điệp đệm" (Message Padding) là xâu bit có độ dài chia hết cho 512.

"Thông điệp đệm" được lưu trong mảng M = M[1] M[2] … M[N-1].

Trong đó M[i] là xâu bít có độ dài 32 bit, gọi là *word*

N≡0 mod 16. (32 x 16 = 512)

M được xây dựng từ Bản tin gốc a bằng thuật toán:

1. **d** = 447 – (|**a**| mod 512). (= 512 nếu |**a**| mod 512 > 447)
2. Giả sử **l** là kí hiệu biểu diễn nhị phân của |a| mod 2^{64}, tl: | **l** | = 64
3. **M = a** || 1 || 0^d || **l**

* Độ dài của xâu **a** || 1 || 0^d **là** |**a**| + 1 + **d** = 448 mod 512.

* Độ dài của "Thông điệp đệm" **M** là:

448 mod 512 + | **l** | = 448 mod 512 + 64 = 512 mod 512

Chú ý: Vì **M = a** || 1 || 0^d || **l** nên

d = |**M**| - (|**a**| + 1 + **l**) =

512 -(|**a**| + 1 + **64)** = 512 – (|**a**| + 65) = 447 – (|**a**| mod 512)

Ví dụ

Xâu đầu vào là **a** = "ABC", xây dựng M như sau:

a: = "ABC" = "01000001 01000010 01000011". (Chú ý: 'A'= 65).

* Độ dài tính theo bit của xâu ***a :*** |***a***| = 24 bit

=> d = 447 – (|***a***| mod 512) = 423

|***a***| + 1 + d = 24 +1 + 423 = 448 mod 512

* Biểu diễn nhị phân của độ dài xâu **a** là ***l:***

l = |***a***| mod 2^{64} = 24 mod 2^{64} = 24 = 16 + 8 = $(\underbrace{00\ldots\ldots00}_{59\text{ số}}11000)_2$

➔ Độ dài của l là |***l***| = $|\underbrace{00\ldots\ldots00}_{59\text{ số}}11000|$ = 59 +5 = 64

M = **a** || 1 || 0^d || ***l***

➔ M = 01000001 01000010 01000011 || l || $\underbrace{00\ldots\ldots00}_{423\text{ số}}$ || $\underbrace{00\ldots\ldots00}_{59\text{ số}}$11000

M = M[0] M[1] … M[N-1], N ≡ 0 mod 16

M[0] = 01000001 01000010 01000011 10000000

M[1] = M[2] = ….. =M[13] = M[14] = $\underbrace{00\ldots..00}_{32\text{ số}}$

M[15] = 00000000 00000000 00000000 00011000

Trong việc xây dựng M, ta gắn số **l** đơn lẻ vào sau **a,** sau đó thêm tiếp các số 0 vào đủ để độ dài của M đồng dư với 448 modulo 512. Cuối cùng nối thêm 64 bit (chính là |***l***|) chứa biểu diễn nhị phân về độ dài ban đầu của ***x*** (được rút gọn theo modulo 2^{64} nếu cần).

Xâu kết quả M có độ dài chia hết cho 512. Vì thế khi chặt M thành các *word* 32 bit, số *word* nhận được là N sẽ chia hết cho 16.

Mục đích việc tạo ra mảng M – "thông điệp đệm" – là để các hàm băm xử lý trên từng khối (block) 512 bit, tức là 16 *word*, cùng một lúc.

3.2.2.2. Thuật toán

INPUT : Thông điệp là một xâu ***a*** có độ dài ***b*** bit.

OUTPUT: Bản băm có độ dài cố định 128 bit đại diện cho thông điệp gốc a.

1) Tóm tắt thuật toán

Bước 1: **Khởi tạo các thanh ghi**

Có 4 thanh ghi để tính toán nhằm đưa ra các đoạn mã : A, B, C, D. Bản tóm lược của thông điệp được xây dựng như sự kết nối của các thanh ghi. Mỗi thanh ghi có độ dài 32 bit. Các thanh ghi này được khởi tạo giá trị hecxa.

word A:= 67 45 23 01 *word* B: = ef cd ab 89

word C:= 98 ba dc fe *word* D:= 10 32 54 76

Bước 2:

Xử lý thông điệp ***a*** trong 16 khối *word*, nghĩa là xử lý cùng một lúc 16 *word* = 512 bit.

Chia mảng M thành các khối 512 bit, đưa từng khối 512 bit vào mảng T[j]. Mỗi lần xử lý một khối 512 bit. Lặp lại N/16 lần.

2). Thuật toán MD4

Bước 1/. A:= **67 45 23 01** B:= **ef cd ab 89**

C:= **98 ba dc fe** D:= **10 32 54 76**

Bước 2/. **FOR** i:= 0 **TO** N/16 - 1 **DO**

for j:= 0 **to** 15 **do T**[j] = M[16 i + j];

AA :=A; BB :=B;

CC :=C; DD :=D;

Mỗi lần xử lý 16 từ, mỗi từ 32 bit, tl: 512 bit

Bước 3/. **Vòng 1**

Vòng 2

Vòng 3

Bước 4/. A = A + AA; B = B + BB; C = C + CC; D = D + DD;

Gán giá trị cho 4 biến AA, BB, CC, DD bằng 4 thanh ghi A, B, C, D tương ứng.

3). Các phép tính và các hàm dùng trong Thuật toán MD4

**** Các phép toán logic được sử dụng trong ba vòng.***

$X \wedge Y$ là phép toán AND theo bit giữa X và Y

$X \vee Y$ là phép toán OR theo bit giữa X và Y

$X \oplus Y$ là phép toán XOR theo bit giữa X và Y

$\neg X$ chỉ phép bù của X

$X + Y$ là phép cộng theo modulo 2^{32}

$X <<< s$ là phép dịch vòng trái đi s vị trí ($0 \leq s \leq 31$)

**** Ba hàm F, G, H dùng tương ứng trong vòng 1, 2, 3***

Mỗi hàm này là một hàm boolean tính theo bit.

$F(X, Y, Z) = (X \wedge Y) \vee ((\neg X) \wedge Z)$

$G(X, Y, Z) = (X \wedge Y) \vee (X \wedge Z) \vee (Y \wedge Z)$

$H(X, Y, Z) = X \oplus Y \oplus Z$

Ba vòng trong MD4 là hoàn toàn khác nhau. Mỗi vòng gồm một trong 16 *word* trong T được xử lý. Các phép toán được thực hiện trong ba vòng tạo ra các giá trị mới trong bốn *thanh ghi*. Cuối cùng, bốn *thanh ghi* được cập nhật ở bước 4 bằng cách cộng ngược các giá trị lưu trước đó ở bước 2, bước3. Phép cộng này được xác định là cộng các số nguyên dương, được rút gọn theo modulo 2^{32}.

4). Ba vòng "băm"

Vòng 1

1. A = (A + F (B, C, D) + T[0]) <<< 3
2. D = (D + F (A, B, C) + T[1]) <<< 7
3. C = (C + F (D, A, B) + T[2]) <<< 11
4. B = (B + F (C, D, A) + T[3]) <<< 19
5. A = (A + F (B, C, D) + T[4]) <<< 3
6. D = (D + F (A, B, C) + T[5]) <<< 7
7. C = (C + F (D, A, B) + T[6]) <<< 11
8. B = (B + F (C, D, A) + T[7]) <<< 19
9. A = (A + F (B, C, D) + T[8]) <<< 3
10. D = (D + F (A, B, C) + T[9]) <<< 7
11. C = (C + F (D, A, B) + T[10]) <<< 11
12. B = (B + F (C, D, A) + T[11]) <<< 19
13. A = (A + F (B, C, D) + T[12]) <<< 3
14. D = (D + F (A, B, C) + T[13]) <<< 7
15. C = (C + F (D, A, B) + T[14]) <<< 11
16. B = (B + F (C, D, A) + T[15]) <<< 19

Kết quả "băm" **a** *sau khi được xử lý qua vòng 1.*

1. 64B3DA82	5. 3D5E5934	9. 59798D5E	13. 7551AAC6
2. 34D8EB03	6. 489D5140	10 D206302D	14. 789B984F
3. B7BCB118	7. CCD14D6C	11. 753D6134	15. F55A1F31
4. 6D91B115	8. 454D14D6C	12. F52AED08	16. ABA71E22

Vòng 2

1. A = (A + G (B, C, D) + T[0] + 5A827999) <<< 3
2. D = (D + G (A, B, C) + T[4] + 5A827999) <<< 5
3. C = (C + G (D, A, B) + T[8] + 5A827999) <<< 9
4. B = (B + G (C, D, A) + T[12] + 5A827999) <<< 13
5. A = (A + G (B, C, D) + T[1] + 5A827999) <<< 3
6. D = (D + G (A, B, C) + T[5] + 5A827999) <<< 5
7. C = (C + G (D, A, B) + T[9] + 5A827999) <<< 9
8. B = (B + G (C, D, A) + T[13] + 5A827999) <<< 13
9. A = (A + G (B, C, D) + T[2] + 5A827999) <<< 3
10. D = (D + G (A, B, C) + T[6] + 5A827999) <<< 5
11. C = (C + G (D, A, B) + T[10] + 5A827999) <<< 9
12. B = (B + G (C, D, A) + T[14] + 5A827999) <<< 13
13. A = (A + G (B, C, D) + T[3] + 5A827999) <<< 3
14. D = (D + G (A, B, C) + T[7] + 5A827999) <<< 5
15. C = (C + G (D, A, B) + T[11] + 5A827999) <<< 9
16. B = (B + G (C, D, A) + T[15] + 5A827999) <<< 13

Giá trị 5A827999 là một hằng số ở dạng hecxa có độ dài 32 bit

Kết quả "băm" **a** *sau khi được xử lý qua vòng 2.*

1. 558C2E28	5. 558C2E28	9. 31E9FE4A	13. B60A11E6
2. 5A0E08F9	6. 5A0E08F9	10. 6F68E462	14. 2DED6D8E
3. F6A9B390	7. F6A9B390	11. D745F88A	15. A2870B31
4. 7876BC8F	8. 7876BC8F	12. 7050BC10	16. 4384D178

Vòng 3

1. A = (A + H (B, C, D) + T[0] + 6ED9EBA1) <<< 3
2. D = (D + H (A, B, C) + T[8] + 6ED9EBA1) <<< 9
3. C = (C + H (D, A, B) + T[4] + 6ED9EBA1) <<< 11
4. B = (B + H (C, D, A) + T[12] + 6ED9EBA1) <<< 15
5. A = (A + H (B, C, D) + T[2] + 6ED9EBA1) <<< 3
6. D = (D + H (A, B, C) + T[10] + 6ED9EBA1) <<< 9
7. C = (C + H (D, A, B) + T[6] + 6ED9EBA1) <<< 11
8. B = (B + H (C, D, A) + T[14] + 6ED9EBA1) <<< 15
9. A = (A + H (B, C, D) + T[1] + 6ED9EBA1) <<< 3
10. D = (D + H (A, B, C) + T[9] + 6ED9EBA1) <<< 9
11. C = (C + H (D, A, B) + T[5] + 6ED9EBA1) <<< 11
12. B = (B + H (C, D, A) + T[13] + 6ED9EBA1) <<< 15
13. A = (A + H (B, C, D) + T[3] + 6ED9EBA1) <<< 3
14. D = (D + H (A, B, C) + T[11] + 6ED9EBA1) <<< 9
15. C = (C + H (D, A, B) + T[7] + 6ED9EBA1) <<< 11
16. B = (B + H (C, D, A) + T[15] + 6ED9EBA1) <<< 15

Giá trị 6ED9EBA1 là một hằng số ở dạng hecxa có độ dài 32 bit

Kết quả "băm" **a** *sau khi được xử lý qua vòng 3.*

1. 98A7C489	5. F3031C80	9. C02E826B	13. 03477E5E
2. E70B031C	6. 7D7A371B	10. F38DC78B	14. 77509F0A
3. A96B2FFA	7. 1C2487DE	11. E3C7F63B	15.FB3D792D
4. 58BE9F94	8. F7767709	12. 81AB00F	16. 23D73C06

5). Kết quả "băm"

Kết quả ra là đoạn mã có độ dài 128 bit, được thu gọn từ thông điệp **a** có độ dài ***b*** bit. Đoạn mã này thu được từ 4 thanh ghi A, B, C, D: bắt đầu từ byte thấp nhất của thanh ghi A cho đên byte cao nhất của thanh ghi D.

Với **a** = "ABC", Đại diện văn bản a là thông tin trên 4 thanh ghi liên tiếp: A, B, C, D, trong đó:

Thanh ghi A = 6A8CA15F

Thanh ghi B = 671E4A93

Thanh ghi C = 93F85626

Thanh ghi D = 3409907C

Chú ý : A = A + AA = 03477E5E

67452301

=6A8CA15F

3.2.2.3. Ví dụ

Để hiểu được cách xác thực thông điệp bằng hàm băm, ta sẽ xem sét ví dụ sau:

An muốn gửi cho Thu xâu ***a*** = "ABC", An tiến hành tạo đại diện của xâu "ABC" bằng hàm băm MD4 và được giá trị Đại diện văn bản là 4 thanh ghi trên.

An tiến hành gửi cho Thu xâu "ABC" và bản Đại diện trên.

(ABC, 6A8CA15F, 671E4A93, 93F85626, 3409907C)

An ⟶ Thu

Trong đó "ABC" là xâu đầu vào a

6A8CA15F, 671E4A93, 93F85626, 3409907C lần lượt là giá trị của các thanh ghi A, B, C, D.

Thu sau khi nhận được thông điệp của An, sẽ tiến hành xác thực bằng cách cũng tạo đại diện của xâu ***a*** bằng cùng phương pháp MD4. Nếu kết quả ra là bốn thanh ghi có giá trị trùng khớp với Đại diện mà An đã gửi kèm thì bản rõ mà An gửi là đúng, nếu không trùng lặp thì Thu sẽ loại bỏ vì chứng tỏ đã có sự giả mạo hoặc thay thế của ai đó đối với bản rõ.

Giả sử Minh là kẻ gian muốn thay đổi thông tin của An để gửi lại cho Thu, có hai trường hợp xảy ra:

+ Minh biết được bản rõ ***a*** = "ABC" nhưng không nắm được cách An tạo đại diện nên sẽ gửi cho Thu một đại diện khác.

(ABC, 6A8CB15F, 671C4A93, 92F85626, 3409906C)

Minh ⟶ Thu

Thu sau khi nhận được bản tin giả mạo mà Minh gửi sẽ tiến hành xác thực bằng cách băm xâu "ABC" bằng hàm băm MD4, sau đó tiến hành so sánh với các thanh ghi đại diện gửi kèm. Thu có thể thấy rõ sự sai lệch ở từng thanh ghi và đưa ra kết luận thông điệp mà An gửi đã bị thay đổi.

+ Minh không biết được bản rõ ***a*** = "ABC" nhưng lại tình cờ biết được đại diện của ***a*** là 4 thanh ghi A, B, C, D. Minh sẽ gửi Thu một thông điệp với xâu ***a'*** = "MINH" và đại diện trên, mong muốn được Thu chấp nhận.

(MINH, 6A8CA15F, 671E4A93, 93F85626, 3409907C)

Minh ⟶ Thu

Thu sau khi nhận được bản tin giả mạo mà Minh gửi sẽ tiến hành xác thực bằng cách băm xâu "MINH" bằng hàm băm MD4, sau đó tiến hành so sánh với các thanh ghi đại diện gửi kèm. Thu có thể thấy sự sai khác giữa 2 đại diện và đưa ra kết luận thông điệp mà An gửi đã bị thay đổi

3.2.3. Hàm băm MD5

3.2.3.1. Giới thiệu MD5

Hàm băm MD4 (Message Digest 4) được Giáo sư Rivest đề nghị vào năm 1990. Vào năm sau, phiên bản cải tiến MD5 của thuật toán này ra đời. Cùng với hàm băm SHS, đây là ba phương pháp có ưu điểm tốc độ xử lý nhanh nên thích hợp áp dụng trong thực tế đối với các thông điệp dài.

Thông điệp ban đầu x sẽ được mở rộng thành dãy bit X có độ dài là bội số của 512. Một bit 1 được thêm vào sau dãy bit x, tiếp đến là dãy gồm d bit 0 và cuối cùng là dãy 64 bit l biểu diễn độ dài của thông điệp x. Dãy gồm d bit 0 được thêm vào sao cho dãy X có độ dài là bội số 512. Quy trình này được thể hiện:

Thuật toán 3.2.3.1 *Thuật toán xây dựng dãy bit X từ dãy bit x*

$d = (447 - |x|) \bmod 512$

Gọi dãy 64 bit l là biểu diễn nhị phân của giá trị $|x| \bmod 2^{64}$.

$X = x \,||\, 1 \,||\, 0^d \,||\, l$

Đơn vị xử lý trong MD5 là các từ 32 - bit nên dãy X sẽ được biểu diễn thành dãy các từ $X[i]$ 32 bit: $X = X[0]\, X[1] \ldots X[N-1]$ với N là bội số của 16.

Thuật toán 3.2.3.2 *Hàm băm MD5*

```
A = 0x67452301;
B = 0xefcdab89;
C = 0x98badcfe;
D = 0x10325476;
for i = 0        to N/16 –1
for j = 0              to 15
M[j] = X[16i-j]
end for
AA = A
BB = B
```

CC = C

DD = D

Round1

Round2

Round3

Round4

A = A+AA

B = B+BB

C = C+CC

D = D+DD

end for

Đầu tiên, bốn biến *A, B, C, D* được khởi tạo. Những biến này được gọi là *chaining variables.*

Bốn chu kỳ (Round) biến đổi trong MD5 hoàn toàn khác nhau và lần lượt sử dụng các hàm *F, G, H* và *I*. Mỗi hàm có tham số *X, Y, Z* là các từ 32 bit và kết quả là một từ 32 bit.

$$F(X, Y, Z) = (X \wedge Y) \vee ((\neg X) \wedge Z)$$
$$G(X, Y, Z) = (X \wedge Z) \vee (Y \wedge (\neg Z))$$
$$H(X, Y, Z) = X \oplus Y \oplus Z \qquad (3.2.1)$$
$$I(X, Y, Z) = Y \oplus (X \vee (\neg Z))$$

Với quy ước:

X ^ Y là phép toán AND theo bit giữa X và Y

X $\vee$ Y là phép toán OR theo bit giữa X và Y

X $\oplus$ Y là phép toán XOR theo bit giữa X và Y

$\neg$ X chỉ phép bù của X

X + Y là phép cộng theo modulo 2^{32}

X <<< ***s*** là phép dịch vòng trái đi ***s*** vị trí ($0 \leq s \leq 32$)

Định nghĩa các hàm:

FF (A, B, C, D, Mj, s, t_i):
A = B + ((A + F(B, C, D) + Mj + t_i) <<< s)
GG (A, B, C, D, Mj, s, t_i):
A = B + ((A + G(B, C, D) + Mj + t_i) <<< s)
HH (A, B, C, D, Mj, s, t_i):
A = B + ((A + H(B, C, D) + Mj + t_i) <<< s)
II (A, B, C, D, Mj, s, t_i):
A = B + ((A + I(B, C, D) + Mj + t_i) <<< s)

Với Mj là *M*[*j*] và hằng số t_i xác định theo công thức:

$ti = \lfloor 2^{32} \left| \sin(i) \right| \rfloor$, *i* tính bằng radian.

Bảng 3.2.3.1. Bốn chu kỳ biến đổi trong MD5.

Chu kỳ 1	Chu kỳ 2
FF(A, B, C, D, M0 , 7, 0xd76aa478)	GG(A, B, C, D, M1 , 5, 0xf61e2562)
FF(D, A, B, C, M1 ,12, 0xe8c7b756)	GG(D, A, B, C, M6 , 9, 0xc040b340)
FF(C, D, A, B, M2 ,17, 0x242070db)	GG(C, D, A, B, M11, 14, 0x265e5a51)
FF(B, C, D, A, M3 ,22, 0xclbdceee)	GG(B, C, D, A, M0 , 20, 0xe9b6c7aa)
FF(A, B, C, D, M4 , 7, 0xf57c0faf)	GG(A, B, C, D, M5 , 5, 0xd62fl05d)
FF(D, A, B, C, M5 ,12, 0x4787c62a)	GG(D, A, B, C, M10, 9, 0x02441453)
FF(C, D, A, B, M6 , 17, 0xa8304613)	GG(C, D, A, B, M15, 14, 0xd8ale681)
FF(B, C, D, A, M7 ,22, 0xfd469501)	GG(B, C, D, A, M4 , 20, 0xeid3fbc8)
FF(A, B, C, D, M8 , 7, 0x698098d8)	GG(A, B, C, D, M9 , 5,0x21elcde6)
FF(D, A, B, C, M9 ,12, 0x8b44f7af)	GG(D, A, B, C, M14, 9, 0xc33707d6)
FF(C, D, A, B, M10, 17, 0xffff5bbl)	GG(C, D, A, B, M3 , 14, 0xf4d50d87)
FF(B, C, D, A, M11, 22, 0x895cd7be)	GG(B, C, D, A, M8 , 20, 0x455al4ed)
FF(A, B, C, D, M12, 7, 0x6b901122)	GG(A, B, C, D, M13, 5, 0xa9e3e905)
FF(D, A, B, C, M13, 12, 0xfd987193)	GG(D, A, B, C, M2 , 9,0xfcefa3f8)
FF(C, D, A, B, M14, 17, 0xa679438e)	GG(C, D, A, B, M7, 14,0x676f02d9)
FF(B, C, D, A, M15, 22, 0x49b40821)	GG(B, C, D, A, M12, 20,0x8d2a4c8a)

Chu kỳ 3	Chu kỳ 4
HH(A, B, C, D, M5 , 4, 0xfffa3942)	II(A, B, C, D, M0 , 6, 0xf4292244)
HH(D, A, B, C, M8 , 11, 0x8771f6811	II(D, A, B, C, M7 , 10, 0x432aff97)
HH(C, D, A, B, M11, 16, 0x6d9d6122)	II(C, D, A, B, M14, 15, 0xab9423a7)
HH(B, C, D, A, M14, 23, 0xfde5380c)	II(B, C, D, A, M5 ,21, 0xfc93a039)
HH(A, B, C, D, M1 , 4, 0xa4beea44)	II(A, B, C, D, M12, 6, 0x655b59c3)
HH(D, A, B, C, M4 , 11, 0x4bdecfa9)	II(D, A, B, C, M3 ,10, 0x8f0ccc92)
HH(C, D, A, B, M7 , 16, 0xf6bb4b60)	II(C, D, A, B, M10, 15, 0xffeff47d)
HH(B, C, D, A, M10, 23, 0xbebfbc70)	II(B, C, D, A, M1 , 21, 0x85845ddl)
HH(A, B, C, D, M13, 4, 0x289biec6)	II(A, B, C, D, M8 , 6, 0x6fa87e4f)
HH(D, A, B, C, M0 , 11, 0xeaal27fa)	II(D, A, B, C, M15, 10, 0xfe2ce6e0)
HH(C, D, A, B, M3 , 16, 0xd4ef3085)	II(C, D, A, B, M6 , 15, 0xa3014314)
HH(B, C, D, A, M6 , 23, 0x04881d05)	II(B, C, D, A, M13, 21, 0x4e0811al)
HH(A, B, C, D, M9 , 4, 0xd9d4d039)	II(A, B, C, D, M4 , 6, 0xf7537e82)
HH(D, A, B, C, M12, 11, 0xe6db99e5)	II(D, A, B, C, M11, 10, 0xbd3af235)
HH(C, D, A, B, M15, 16, 0xlfa27cf8)	II(C, D, A, B, M2 , 15, 0x2ad7d2bb)
HH(B, C, D, A, M2 , 23, 0xc4ac5665)	II(B, C, D, A, M9 , 21, 0xeb86d391)

3.2.3.2. Nhận xét

Hàm băm MD5 có những ưu điểm cải tiến so với phương pháp MD4 [45]:

+ MD4 chỉ có ba chu kỳ biến đổI, trong khi MD5 được bổ sung thêm chu kỳ thứ tư giúp tăng mức độ an toàn.

+ Mỗi thao tác trong từng chu kỳ biến đổi của MD5 sử dụng các hằng số t_i phân biệt, trong khi MD4 sử dụng hằng số chung cho mọi thao tác trong cùng chu kỳ biến đổi (Trong MD4, hằng số t_i sử dụng trong mỗi chu kỳ lần lượt là 0, 0x5a827999, 0x6ed9eba1).

+ Hàm G ở chu kỳ hai của MD4: $G(X, Y, Z) = ((X \wedge Y) \vee (X \wedge Z) \vee (Y \wedge Z))$ được thay thế bằng $((X \wedge Z) \vee (Y \wedge Z))$ nhằm giảm tính đối xứng.

+ Mỗi bước biến đổi trong từng chu kỳ chịu ảnh hưởng kết quả của bước biến đổi trước đó nhằm tăng nhanh tốc độ của hiệu ứng lan truyền (avalanche).

+ Các hệ số dịch chuyển xoay vòng trong mỗi chu kỳ được tối ưu hóa nhằm tăng tốc độ hiệu ứng lan truyền. Ngoài ra, mỗi chu kỳ sử dụng bốn hệ số dịch chuyển khác nhau.

3.2.4. Hàm băm Secure Hash Standard (SHS)

Hàm băm SHS do NIST và NSA xây dựng được công bố trên Federal Register vào ngày 31 tháng 1 năm 1992 và sau đó chính thức trở thành phương pháp chuẩn từ ngày 13 tháng 5 năm 1993.

Nhìn chung, SHS được xây dựng trên cùng cơ sở với phương pháp MD4 và MD5. Tuy nhiên, phương pháp SHS lại áp dụng trên hệ thống big-endian thay vì little-endian như phương pháp MD4 và MD5. Ngoài ra, thông điệp rút gọn kết quả của hàm băm SHS có độ dài 160 bit (nên phương pháp này thường được sử dụng kết hợp với thuật toán ký DSS).

Tương tự MD5, thông điệp nguồn x sẽ được chuyển thành một dãy bit có độ dài là bội số của 512. Từng nhóm gồm 16 từ-32 bit $X[0]$, $X[1]$,..., $X[15]$ sẽ được mở rộng thành 80 từ-32 bit $W[0]$, $W[1]$, ..., $W[79]$ theo công thức:

$$W[t] = \begin{cases} X[t], & 0 \le t \le 15 \\ X[j-3] \oplus X[j-8] \oplus X[j-14] \oplus X[j-16], & 16 \le t \le 79 \end{cases}$$

(3. 2. 2)

Trong phiên bản cải tiến của SHS, công thức trên được thay bằng:

$$W[t] = \begin{cases} X[t], & 0 \le t \le 15 \\ (X[j-3] \oplus X[j-8] \oplus X[j-14] \oplus X[j-16]) <<< 1, & 16 \le t \le 79 \end{cases}$$

(3. 2. 3)

Tương tự MD5, hàm băm SHS sử dụng bốn chu kỳ biến đổi, trong đó, mỗi chu kỳ gồm 20 bước biến đổi liên tiếp nhau. Chúng ta có thể xem như SHS bao gồm 80 bước biến đổi liên tiếp nhau. Trong đoạn mã chương trình dưới đây, hàm $f[t]$ và hằng số $K[t]$ được định nghĩa như sau:

$$f[t](X,Y,Z)=\begin{cases}(X\wedge Y)\vee((\neg X)\wedge Z), & 0\le t\le 19\\ X\oplus Y\oplus Z, & 20\le t\le 39\\ (X\wedge Y)\vee(X\wedge Z)\vee(Y\wedge Z), & 40\le t\le 59\\ X\oplus Y\oplus Z, & 60\le t\le 79\end{cases}$$

(3.2.4)

$$K[t]=\begin{cases}0x5a827999, 0\le t\le 19\\ 0x6ed9eba1, 20\le t\le 39\\ 0x8f1bbcdc, 40\le t\le 59\\ 0xca62c1d6, 60\le t\le 79\end{cases}$$

(3.2.5)

A = 0x67452301;

B = 0xefcdab89;

C = 0x98badcfe;

D = 0x10325476;

E = 0xc3d2elf0;

for i=0 **to** N/16 –1

for t=0 **to** 15 do

W[t] = X[16 * t-j]

end for

for t=16 **to** 79

W[t] =(W[t-3] xor W[t-8] xor W[t-14] xor W[t-16])<<<1

a = A

b = B

c = C

d = D

e = E

```
for t=0          to 79
TEMP = (a<<<5) + f[t] (b, c, d) + e + W[t] + K[t]
e    = d
d    = c
c    = b <<< 30
b    = a
a    = TEMP
end for
A = A + a
B = B + b
C = C + c
D = D+ d
E = E + e
end for
```

3.2.4.1. Nhận xét

Hàm băm SHS rất giống với MD4 nhưng thông điệp rút gọn được tạo ra có độ dài 160-bit. Cả 2 phương pháp này đều là sự cải tiến từ MD4. Dưới đây là một số đặc điểm so sánh giữa MD5 và SHS:

+ Tương tự như MD5, hàm băm SHS cũng bổ sung thêm chu kỳ biến đổi thứ tư để tăng mức độ an toàn. Tuy nhiên, chu kỳ thứ tư của SHS sử dụng lại hàm f của chu kỳ thứ 2.

+ 20 bước biến đổi trong cùng chu kỳ của hàm băm SHS sử dụng hằng số chung $K[t]$ trong khi mỗi bước biến đổi của hàm băm MD5 lại dùng các hằng số khác nhau.

+ Trong hàm băm MD5, hàm G ở chu kỳ thứ hai của MD4:
$G(X, Y, Z) = ((X \wedge Y) \vee (X \wedge Z) \vee (Y \wedge Z))$ được thay thế bằng $((X \wedge Z) \vee (Y \wedge Z))$ nhằm giảm tính đối xứng. Hàm băm SHS vẫn sử dụng hàm G như trong MD4.

+ Trong MD5 và SHS, mỗi bước biến đổi chịu ảnh hưởng bởi kết quả của bước biến đổi trước đó để tăng nhanh hiệu ứng lan truyền.

Hiện tại vẫn chưa có phương pháp tấn công nào có thể áp dụng được đối với hàm băm SHS. Ngoài ra, do thông điệp rút gọn của hàm băm SHS có độ dài 160 bit nên có độ an toàn cao hơn đối với phương pháp tấn công brute-force (kể cả phương pháp birthday attack) so với hàm băm MD5.

3.2.5. Hàm băm SHA

3.2.5.1. Ý tưởng của các thuật toán hàm băm SHA

Các thuật toán hàm băm SHA gồm 2 bước: tiền xử lý và tính toán giá trị băm.

+ Bước tiền xử lý bao gồm các thao tác:

- Mở rộng thông điệp
- Phân tích thông điệp đã mở rộng thành các khối *m* bit
- Khởi tạo giá trị băm ban đầu

+ Bước tính toán giá trị băm bao gồm các thao tác:

- Làm *N* lần các công việc sau:
  - Tạo bảng phân bố thông điệp (message schedule) từ khối thứ *i*.
  - Dùng bảng phân bố thông điệp cùng với các hàm, hằng số, các thao tác trên từ để tạo ra giá trị băm *i*.
- Sử dụng giá trị băm cuối cùng để tạo thông điệp rút gọn.

Thông điệp *M* được mở rộng trước khi thực hiện băm,nhằm đảm bảo thông điệp mở rộng có độ dài là bội số của 512 hoặc 1024 bit tùy thuộc vào thuật toán. Sau khi thông điệp đã mở rộng, thông điệp cần được phân tích thành *N* khối *m*-bit trước khi thực hiện băm.

Đối với SHA-1 và SHA-256, thông điệp mở rộng được phân tích thành *N* khối 512-bit $M^{(1)}$, $M^{(2)}$,..., $M^{(N)}$. Do đó 512 bit của khối dữ liệu đầu vào có thể được thể hiện bằng 16 từ 32-bit, $M_0(i)$ chứa 32 bit đầu của khối thông điệp i, $M_1^{(i)}$ chứa 32 bit kế tiếp,…

Đối với SHA-384 và SHA-512, thông điệp mở rộng được phân tích thành *N* khối 1024-bit $M^{(1)}$, $M^{(2)}$,..., $M^{(N)}$. Do đó 1024 bit của khối dữ liệu đầu vào có thể được thể hiện bằng 16 từ 64-bit, $M_0^{(i)}$ chứa 64 bit đầu của khối thông điệp *i*, $M_1^{(i)}$ chứa 64 bit kế tiếp,...

Với mỗi thuật toán băm an toàn, giá trị băm ban đầu $H^{(0)}$ phải được thiết lập. Kích thước và số lượng từ trong $H^{(0)}$ tùy thuộc vào kích thước thông điệp rút gọn.

Các cặp thuật toán SHA-224 và SHA-256; SHA-384 và SHA-512 có các thao tác thực hiện giống nhau, chỉ khác nhau về số lượng bit kết quả của thông điệp rút gọn. Nói cách khác, SHA-224 sử dụng 224 bit đầu tiên trong kết quả thông điệp rút gọn sau khi áp dụng thuật toán SHA256. Tương tự SHA-384 sử dụng 384 bit đầu tiên trong kết quả thông điệp rút gọn sau khi áp dụng thuật toán SHA-512.

3.2.5.2. Khung thuật toán chung của hàm băm SHA

Trong hàm băm SHA, chúng ta cần sử dụng thao tác quay phải một từ, ký hiệu là ROTR, và thao tác dịch phải một từ, ký hiệu là SHR.

Hình 3.2.5.1. *Khung thuật toán chung cho hàm băm SHA*

for $i = 1$ **to** N

for $t = 0$ **to** 15

$W_t = M_t(i)$

end for

for $t = 16$ **to** *scheduleRound*

$W_t = \sigma_1(W_{t-2}) + W_{t-7} + \sigma_0(W_{t-15}) + W_{t-16}$

end for

$a = H_0^{(i-1)}$

$b = H_1^{(i-1)}$

$c = H_2^{(i-1)}$

$d = H_3^{(i-1)}$

$e = H_4^{(i-1)}$

$f = H_5^{(i-1)}$

$g = H_6^{(i-1)}$

$h = H_7^{(i-1)}$

for $t = 0$ **to** 63

$T_1 = h + \sum_1(e) + \text{Ch}(e, f, g) + K_t + W_t$

$T_2 = \sum_0(a) + \text{Maj}(a, b, c)$

$h = g$

$g = f$

$f = e$

$e = d + T_1$

$d = c$

$c = b$

$b = a$

$a = T_1 + T_2$

end for

$H_0^{(i)} = a + H_0^{(i-1)}$

$H_1^{(i)} = b + H_1^{(i-1)}$

$H_2^{(i)} = c + H_2^{(i-1)}$

$H_3^{(i)} = d + H_3^{(i-1)}$

$H_4^{(i)} = e + H_4^{(i-1)}$

$H_5^{(i)} = f + H_5^{(i-1)}$

$H_6^{(i)} = g + H_6^{(i-1)}$

$H_7^{(i)} = h + H_7^{(i-1)}$

end for

Mỗi thuật toán có bảng hằng số phân bố thông điệp tương ứng. Kích thước bảng hằng số thông điệp (scheduleRound) của SHA-224 và SHA-256 là 64, kích thước bảng hằng số thông điệp của SHA-384 và SHA-512 là 80.

Trong hàm băm SHA-224 và SHA-256, chúng ta cần sử dụng các hàm:

$$\text{Ch}(x, y, z) = (x \wedge y) \oplus (\neg x \wedge z)$$
$$\text{Maj}(x, y, z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z)$$
$$\sum_0(x) = \text{ROTR}^2(x) \oplus \text{ROTR}^{13}(x) \oplus \text{ROTR}^{22}(x)$$
$$\sum_1(x) = \text{ROTR}^6(x) \oplus \text{ROTR}^{11}(x) \oplus \text{ROTR}^{25}(x)$$
$$\sigma_0(x) = \text{ROTR}^7(x) \oplus \text{ROTR}^{18}(x) \oplus \text{SHR}^3(x)$$
$$\sigma_1(x) = \text{ROTR}^{17}(x) \oplus \text{ROTR}^{19}(x) \oplus \text{SHR}^{10}(x)$$

Trong hàm băm SHA-384 và SHA-512, chúng ta cần sử dụng các hàm sau:

$$\mathrm{Ch}(x,y,z)=(x\wedge y)\oplus(\neg x\wedge z)$$
$$\mathrm{Maj}(x,y,z)=(x\wedge y)\oplus(x\wedge z)\oplus(y\wedge z)$$
$$\sum_0(x)=\mathrm{ROTR}^{28}(x)\oplus\mathrm{ROTR}^{34}(x)\oplus\mathrm{ROTR}^{29}(x)$$
$$\sum_1(x)=\mathrm{ROTR}^{14}(x)\oplus\mathrm{ROTR}^{18}(x)\oplus\mathrm{ROTR}^{41}(x)$$
$$\sigma_0(x)=\mathrm{ROTR}^{1}(x)\oplus\mathrm{ROTR}^{8}(x)\oplus\mathrm{SHR}^{7}(x)$$
$$\sigma_1(x)=\mathrm{ROTR}^{19}(x)\oplus\mathrm{ROTR}^{61}(x)\oplus\mathrm{SHR}^{6}(x)$$

3.2.5.3. Nhận xét

Chuẩn SHS đặc tả 5 thuật toán băm an toàn SHA-1, SHA-224[3], SHA-256, SHA-84 và SHA-512. Bảng 3.2.5.2 thể hiện các tính chất cơ bản của bốn thuật toán băm an toàn.

Sự khác biệt chính của các thuật toán là số lượng bit bảo mật của dữ liệu được băm – điều này có ảnh hưởng trực tiếp đến chiều dài của thông điệp rút gọn. Khi một thuật toán băm được sử dụng kết hợp với thuật toán khác đòi hỏi phải cho kết quả số lượng bit tương ứng. Ví dụ, nếu một thông điệp được ký với thuật toán chữ ký điện tử cung cấp 128 bit thì thuật toán chữ ký đó có thể đòi hỏi sử dụng một thuật toán băm an toàn cung cấp 128 bit như SHA-256.

Ngoài ra, các thuật toán khác nhau về kích thước khối và kích thước từ được sử dụng.

Bảng 3.2.5.2. *Các tính chất của các thuật toán băm an toàn*

Thuật toán	Kích thước (bit)				Độ an toàn[4] (đơn vị: bit)
	Thông điệp	Khối	Từ	Thông điệp rút gọn	
SHA-1	$< 2^{64}$	512	32	160	80
SHA-224	$< 2^{64}$	512	32	224	112
SHA-256	$< 2^{64}$	512	32	256	128
SHA-384	$< 2^{128}$	1024	64	384	192
SHA-512	$< 2^{128}$	1024	64	512	256

3.3. XÁC THỰC THÔNG ĐIỆP BẰNG MÃ XÁC THỰC

3.3.1. Định nghĩa mã xác thực thông điệp

Thông điệp được xác thực bằng **mã xác thực thông điệp** đi theo bản mã. Mã xác thực thông điệp thực chất là một giá trị băm của bản rõ, hoặc bản mã được tính theo thuật toán khác hẳn với thuật toán mã hóa thông điệp, rồi được mã hóa bằng một khóa khác với khóa mã hóa thông điệp. Nếu mã này xác thực nội dung bản rõ, thì nó cũng gián tiếp xác thực khóa đúng.

Mục đích của phần này là phải có được khả năng xác thực ngay cả khi có một đối phương tích cực - Minh là người có thể quan sát các bản tin trong kênh. Mục đích này có thể đạt được bằng cách thiết lập một ‘’khóa riêng’’ K bằng cách để An và Thu chung một khoá bí mật trước khi mỗi bản tin được gửi đi.

Trong mục này ta sẽ nghiên cứu đảm bảo xác thực, chứ không phải các mã đảm bảo độ mật. Trong mã này, khoá sẽ được dùng để tính một mã xác thực cho phép Thu kiểm tra được tính xác thực của thông báo mà anh ta nhận được. Một ứng dụng khác của mã xác thực là kiểm tra các số liệu trong một file lớn có bị can thiệp vào một cách hợp pháp hay không. Nhãn xác thực sẽ được lưu cùng với số liệu: khóa được dùng để tạo và kiểm tra dấu xác thực được lưu một cách tách bạch trong một “vùng” an toàn.

Về nhiều khía cạnh mã xác thực cũng tương tự như một sơ đồ chữ kí hoặc tương tự như một mã xác thực thông báo (MAC). Sự khác biệt chính là sự an toàn của một mã xác thực là không điều kiện trong khi sơ đồ chữ kí và MAC lại được nghiên cứu theo quan điểm độ an toàn tính toán. Cũng vậy, khi một xác thực (hoặc MAC) được dùng, một bản tin chỉ có thể được kiểm tra bởi người nhận hợp pháp. Trong khi đó bất cứ ai cũng có thể xác minh được chữ kí, bằng cách dùng một thuật toán xác minh công khai.

Định nghĩa 3.3.1

Mã xác thực là một bộ 4 (S, A, K, E) thoả mãn các điều kiện sau :

1. S là tập hữu hạn các trạng thái nguồn có thể.
2. A là tập hợp các nhãn xác thực có thể.
3. K là một tập hữu hạn các khoá có thể (không gian khoá).
4. Với mỗi $k \in K$ có một quy tắc xác thực $e_k : S \rightarrow A$

Tập bản tin được xác định bằng $M = S \rightarrow A$

3.3.2. Ý tưởng chính của phương pháp xác thực bằng mã xác thực

Chú ý một trạng thái nguồn tương đương với một bản rõ. Một bản tin gồm bản rõ với một nhãn xác thực kèm theo, một cách chính xác hơn có thể coi đó là một bản tin đã được xác nhận. Một quy tắc xác thực không nhất thiết phải là hàm đơn ánh.

Để phát thông báo (đã được kí). An và Thu phải tuân theo giao thức sau. Trước tiên họ phải chọn một khoá ngẫu nhiên $k \in K$. Điều này được thực hiện một cách bí mật như trong hệ mật khoá bí mật. Sau đó giả sử rằng An muốn gửi một trạng thái nguồn $s \in S$ cho Thu trên kênh không an toàn, An sẽ tính $a = e_k(s)$ và gửi bản tin (s, a) cho Thu. Khi nhận được (s, a) Thu tính $a' = e_K(s)$. Nếu $a = a'$ thì Thu chấp nhận bản tin là xác thực, ngược lại Thu sẽ loại bỏ nó.

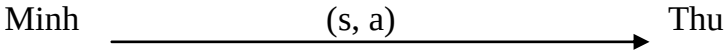
3.3.3. Phương pháp

Ta sẽ nghiên cứu hai kiểu tấn công khác nhau mà Minh có thể tiến hành. Trong cả hai loại này, Minh sẽ là "kẻ xâm nhập vào giữa cuộc".

Giả mạo

Minh đưa ra bản tin (s, a) vào kênh, và hi vọng nó sẽ được chấp nhận. Phương pháp này được mô tả trong hình 3.3.1.

Hình 3.3.1. Việc giả mạo bởi Minh



Thay thế

Minh quan sát một bản tin (s, a) trên kênh, sau đó anh ta biến đổi nó thành (s', a'), trong đó s' = s và hi vọng được Thu chấp nhận như một bản tin xác thực. Bởi vậy anh ta tin sẽ lái được Thu đi tới trạng thái nguồn mới này. Phương pháp này được mô tả như hình 3.3.2

Hình 3.3.2. Phép thay thế của Minh

Ví dụ 3.3.1

Giả sử $K=A=Z$

và $K=Z_3 \times Z_3$

Với mỗi $(i, j) \in K$ và mỗi $s \in S$, xác định

$$e_k(s) = (i.s + j) \bmod 3$$

An gửi cho Thu bản rõ s và khóa k, đồng thời gửi ma trận xác thực (ma trận này tạo bằng tất cả các giá trị $e_k(s)$). Với mỗi khoá $k \in K$ và với mỗi $s \in S$ ta đặt nhãn xác thực $e_k(s)$ vào hàng k và cột s của một ma trận M kích thước K xác suất. Mảng M được mô tả trên hình 3.3.3

Hình 3.3.3. Ma trận xác thực

Khóa \ Bản rõ	0	1	2
(0,0)	0	0	0
(0,1)	1	1	1
(0,2)	2	2	2
(1,0)	0	1	2
(1,1)	1	2	0
(1,2)	2	0	1
(2,0)	0	1	2
(2,1)	1	0	2
(2,2)	2	1	0

Với khóa k = (i, j) = (0, 0) và bản rõ s = 0

Ta có mã xác thực $e_k(s)$ = (i.s + j) mod 3 = (0.0 + 0) mod 3= 0

Khi nhận được bản rõ s và khóa k (khóa bí mật chỉ An và Thu biết) Thu sẽ tiến hành lập ma trận xác thực theo công thức $e_k(s)$ = (i.s + j) mod 3 ((i, j) $\in$ *K* và *s* $\in$ *S*). Sau đó tiến hành so sánh với ma trận mà An đã gửi, nếu trùng khớp thì xác thực đúng, nếu tồn tại một nhãn e_k nào đó không khớp với ma trận xác thực An đã gửI chứng tỏ đã có sự giả mạo hoặc thay thế thông tin trên đường truyền.

**** Tấn công bằng phương pháp "Giả mạo"***

Trước tiên xét cách tấn công giả mạo, Minh sẽ chọn ra một trạng thái nguồn s và cố gắng phỏng đoán một nhãn xác thực "đúng''. Kí hiệu k_0 là khoá đang sử dụng (mà Minh không biết). Minh sẽ thành công trong việc đánh lừa Thu nếu anh ta phỏng đoán $a_0 = e_{k_0}(s)$. Tuy nhiên với bất kì s$\in$S và a$\in$A ta thấy rằng, chỉ có đúng 3 (chứ không phải là 9) quy tắc xác thực k$\in$*K* sao cho $e_k(s)$ =a. Nói cách khác mỗi kí hiệu chỉ xuất hiện 3 lần trong mỗi cột của ma trận xác thực. Bởi vậy dẫn tới Pd_0=1/3.

**** Tấn công bằng phương pháp "Thay thế"***

Giả sử Minh đã quan sát được trên kênh 1 bản tin (0,0). Nhờ đó anh ta đã biết một thông tin nào đó về khoá, anh ta biết rằng :

$$k_0 \in \{(0, 0), (1, 0), (2, 0)\}$$

Bây giờ, giả sử Minh thay bản tin (0,0) bằng bản tin (1,1). Khi đó anh ta sẽ lừa bịp thành công khi và chỉ khi k_0=(0, 1), xác suất để k_0 là khoá bằng 1/3 vì khoá nằm trong tập {(0, 0), (1, 0), (2, 0)}.

Có thể thực hiện một phân tích tương tự đối với bất kì một phép thay thế nào mà Minh tiến hành. Nói chung nếu Minh quan sát một bản tin (s, a) và thấy nó bằng một bản tin bất kì (s', a') trong đó s' $\neq$ s thì anh ta sẽ đánh lừa được Thu với xác suất 1/3. Ta có thể thấy rõ điều này như sau. Việc quan sát được (s, a) sẽ hạn chế khóa và một trong ba khả năng. Trong khi đó với một phép chọn (s', a') chỉ có một khoá chứ không phải ba khoá có thể theo quy tắc a là nhãn xác thực của s'.

KẾT LUẬN

+ Việc đòi hỏi an toàn trong giao dịch cũng như trao đổi thông điệp được đặt lên hàng đầu vì vậy việc xác thực thông điệp là một vấn đề rất quan trọng trong giao dịch hiện nay, đặc biệt là trong giao dịch trực tuyến. Khi nhận được một thông điệp như thư, hợp đồng, đề nghị,…vấn đề đặt ra là làm sao để xác định được đúng đối tác giao dịch và nguồn gốc của thông điệp. Vì vậy đồ án này nghiên cứu một số phương pháp xác thực thông điệp.

+ Kết quả chính của đồ án tốt nghiệp là: Tìm hiểu và nghiên cứu qua tài liệu để hệ thống các vấn đề sau:

1/. Trình bày một số khái niệm cơ bản trong toán học, khái niệm về mã hóa dữ liệu, chữ ký số và hàm băm.

2/. Trình bày tổng quan về xác thực điện tử.

3/. Trình bày một số phương pháp xác thực thông điệp.

TÀI LIỆU THAM KHẢO

1/. Giáo trình an toàn dữ liệu - PGS. TS. Trịnh Nhật Tiến

2/. Mật mã - Lý thuyết và thực hành (D.R SUNSON - Người dịch Nguyễn Bình) (Học viện Kỹ thuật quân sự 1996)

3/. Book_MaHoaVaUngDung (Tài liệu điện tử)

4/. Một số các trang web:

http://vi.wikipedia.org/wiki/X%C3%A1c_th%E1%BB%B1c

http://ictvietnam.net/forum/archive/index.php/t-1578.html