*Chủ đề 5:*

# Hàm băm mật mã Hash & MAC

# Nội dung

- Định nghĩa hàm băm mật mã
- Cấu trúc của hàm băm mật mã
- Các tính chất của hàm băm mật mã
- Phân loại hàm băm mật mã
- Một số kiến trúc hàm băm phổ biến
  - Hàm băm MD5
  - Các hàm băm SHA
- MAC và HMAC

# Định nghĩa

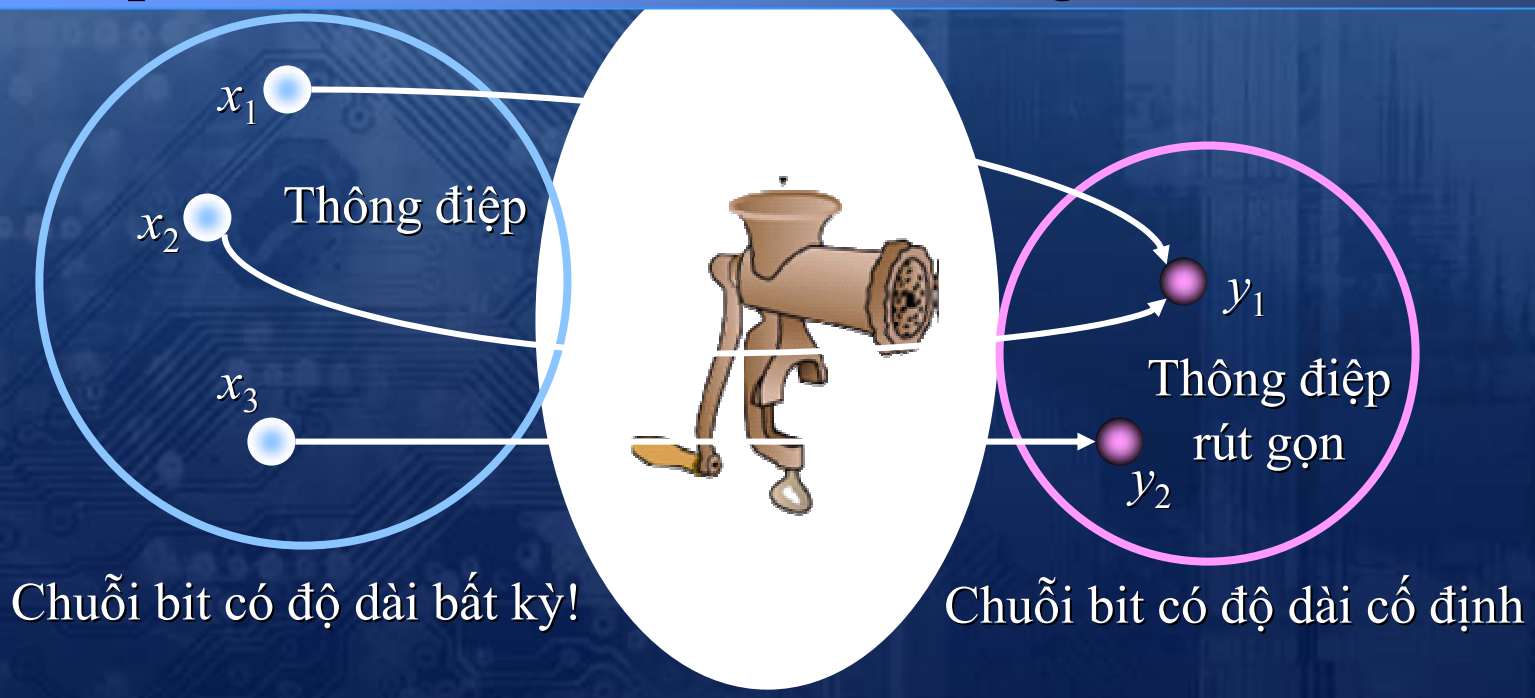
- Hàm băm mật mã là hàm toán học chuyển đổi một thông điệp có độ dài bất kỳ thành một dãy bit có độ dài cố định ( tuỳ thuộc vào thuật toán băm)
- Dãy bit này được gọi là thông điệp rút gọn (message digest) hay giá trị băm (hash value), đại diện cho thông điệp ban đầu
- Hàm băm là nền tảng cho nhiều ứng dụng mã hóa, chữ ký điện tử.
- Các thuật toán phổ biến từ thập niên 1990 đến nay: MD5 và SHA-1

# Cấu trúc của hàm băm

- Cho trước một thông điệp $M$ có độ dài bất kỳ. Tùy theo thuật toán được sử dụng, có thể 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: $M_1, M_2, \ldots M_s$
- 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
  - $H_i = f(H_{i-1}, M_i)$ với $i = 1,2,3, \ldots, s$
- $H_s$ chính là thông điệp rút gọn của $M$ ban đầu

# Ý tưởng chính của hàm băm mật mã

- H là hàm nén mất thông tin (lossy compression function)
- Hiện tượng đụng độ (Collision): $H(x)=H(x')$ với $x \neq x'$
- Kết quả của việc băm "nhìn có vẻ ngẫu nhiên"



# Hàm băm mật mã *H*

- *H* có thể áp dụng trên dữ liệu có kích thước bất kỳ
- Kết quả của *H* là một chuỗi *n*-bit (*n* cố định)
- Dễ dàng tính giá trị *H*(*x*) với *x* bất kỳ
- *H* là hàm một chiều
- *H* an toàn đối với hiện tượng "đụng độ"

# Tính toàn vẹn và tính bí mật

- Tính toàn vẹn (Integrity): người tấn công không thể can thiệp để sửa nội dung thông điệp
- Mã hóa chỉ nhằm đảm bảo tính bí mật, không giúp đảm bảo tính toàn vẹn thông tin
- ➔ Người tấn công có thể sửa đổi nội dung thông điệp đã được mã hóa mà không cần biết nội dung thật sự của thông điệp
- Ví dụ:
  - Trong đấu giá trực tuyến, có thể thay đổi giá đặt của đối thủ mà không cần biết nội dung thật sự của giá đặt

# Tính “một chiều”

- Hàm băm được xem là hàm một chiều khi cho trước giá trị băm, không thể tái tạo lại thông điệp ban đầu, hay còn gọi là “tiền ảnh” (“pre-image”)
- Nếu tìm ra được một phương pháp tấn công cho phép xác định “tiền ảnh” tương ứng với một giá trị băm cho trước thì thuật toán băm sẽ không còn an toàn nữa.
- Cách tấn công nhằm tạo ra một thông điệp khác với thông điệp ban đầu nhưng có cùng giá trị băm gọi là tấn công “tiền ảnh thứ hai” (“second pre-image attack”)

# Tính "một chiều"

- Hàm $H$ rất khó bị biến đổi ngược
  - Cho trước chuỗi bit ngẫu nhiên $y \in \{0,1\}^n$, rất khó tìm ra được chuỗi bit $x$ sao cho $H(x)=y$
- Ví dụ:
  - Brute-force: Với mỗi giá trị $x$, kiểm tra $H(x)=y$
  - SHA-1 cho kết quả là chuỗi gồm 160-bit
    - Giả sử phần cứng cho phép thực hiện $2^{34}$ phép thử trong một giây
    - Có thể thực hiện $2^{59}$ phép thử trong một năm
    - Cần $2^{101}$ ($\sim 10^{30}$) năm để biến đổi ngược SHA-1 với giá trị ngẫu nhiên $y$ cho trước

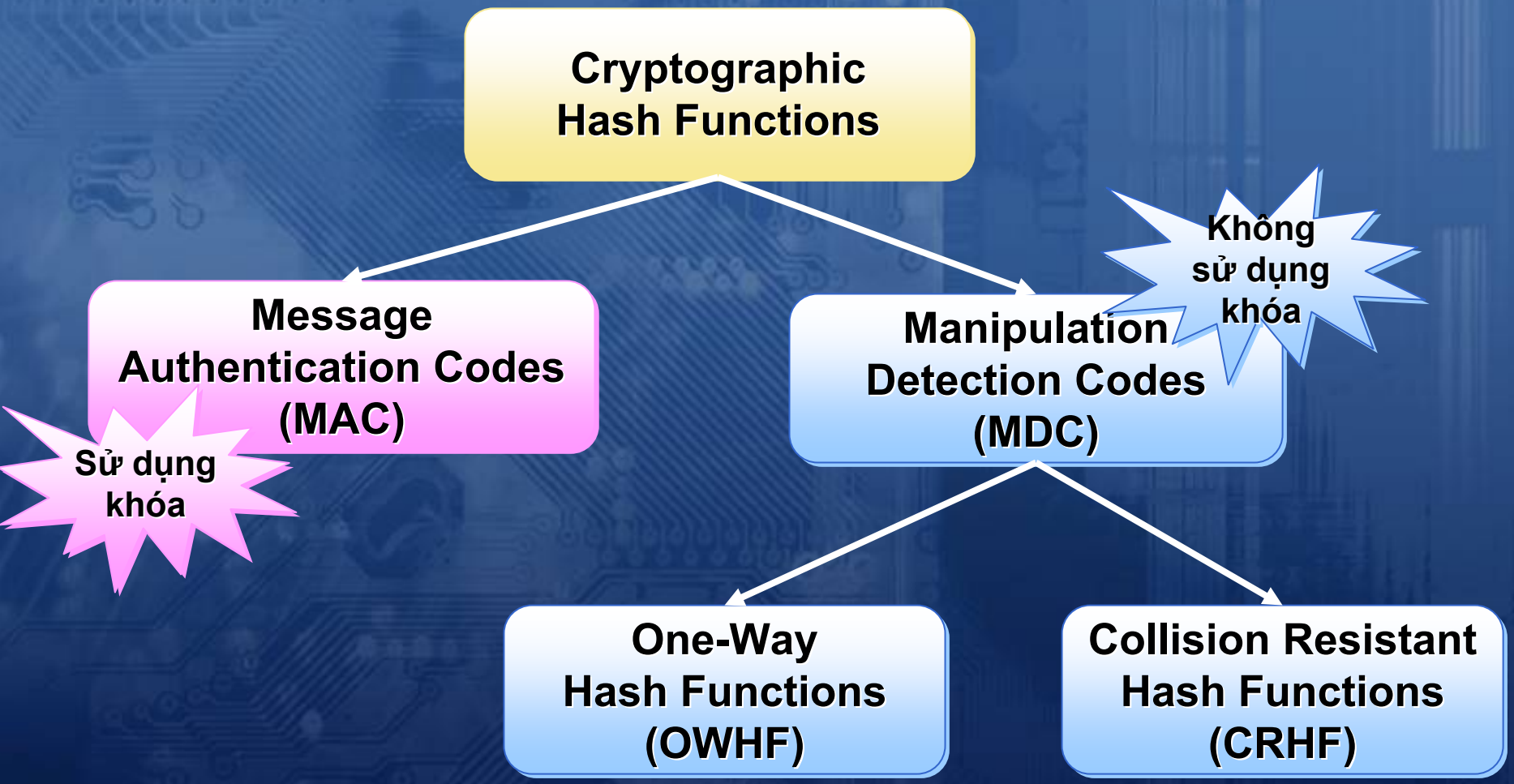
# Tính an toàn đối với hiện tượng đụng độ

- Rất khó có thể tìm được $x$, $x'$ sao cho $H(x)=H(x')$
- 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 $\sqrt{N}$ phép thử ngẫu nhiên để tìm ra một cặp có cùng giá trị

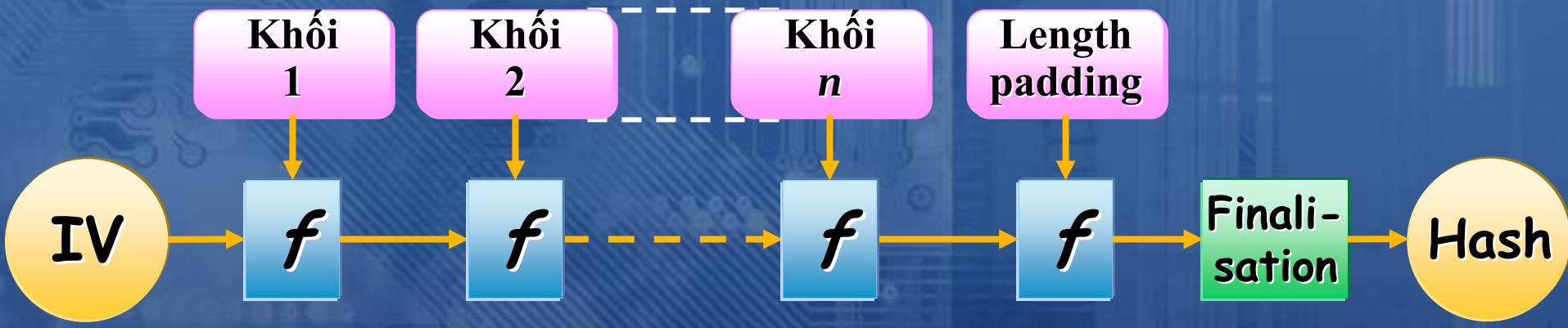
# Tính chất của hàm băm

- An toàn đối với tấn công "tiền ảnh"
  - Preimage resistance
  - cho trước $y$, rất khó tìm được giá trị x sao cho $H(x)=y$
- An toàn đối với hiện tượng đụng độ:
  - rất khó tìm được hai giá trị phân biệt $x$ và $x'$ sao cho $H(x')=H(x)$
- An toàn đối với tấn công "tiền ảnh thứ 2"
  - $2^{nd}$ preimage resistance
  - cho trước $x$ và $y=H(x)$, rất khó tìm được giá trị $x' \neq x$ sao cho $H(x')=H(x)$

# Phân loại hàm băm mật mã



# Cấu trúc Merkle-Damgård




- Tác giả: Ralph Merkle, Ivan Damgård
- Hầu hết các hàm băm đều sử dụng cấu trúc này
- Ví dụ: SHA-1, MD5

# 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.

# MD5

- *Khởi gán các biến:*
  - h0 := 0x67452301
  - h1 := 0xEFCDAB89
  - h2 := 0x98BADCFE
  - h3 := 0x10325476

# MD5

- Hệ số quay trái $R[i]$của mỗi chu kỳ:
  - $R[0..15] := \{7, 12, 17, 22, 7, 12, 17, 22, 7, 12, 17, 22, 7, 12, 17, 22\}$
  - $R[16..31] := \{5, 9, 14, 20, 5, 9, 14, 20, 5, 9, 14, 20, 5, 9, 14, 20\}$
  - $R[32..47] := \{4, 11, 16, 23, 4, 11, 16, 23, 4, 11, 16, 23, 4, 11, 16, 23\}$
  - $R[48..63] := \{6, 10, 15, 21, 6, 10, 15, 21, 6, 10, 15, 21, 6, 10, 15, 21\}$

- Hằng số K[*i*]

  **for** *i* **from** 0 **to** 63

  $K[i] := \text{floor}(\text{abs}(\sin(i + 1)) \times (2 \textbf{ pow } 32))$

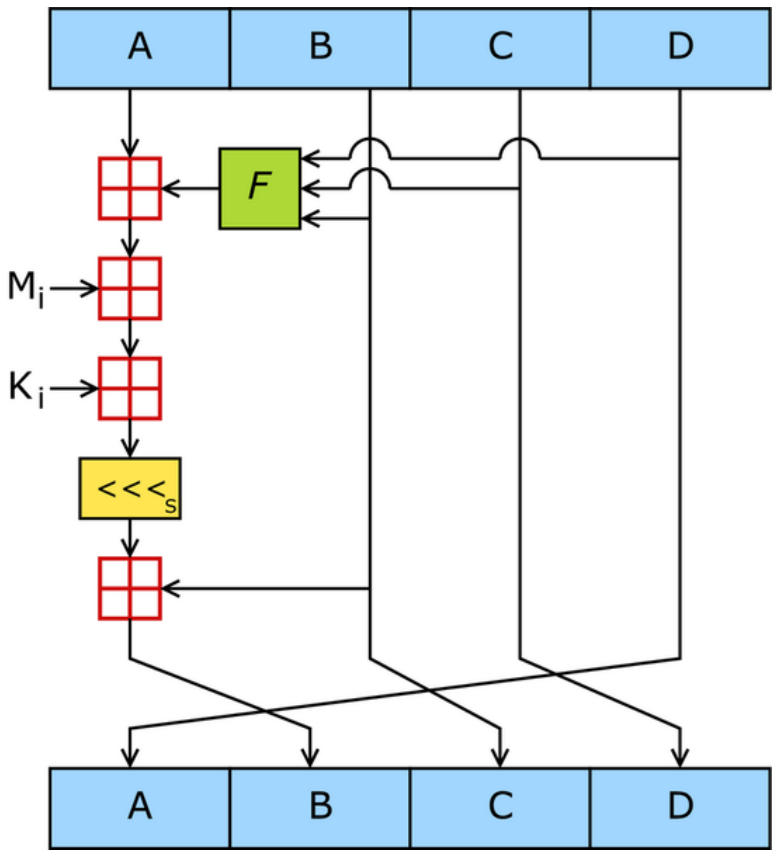
# MD5

- *Tiền xử lý:*
  - Thêm bit 1 vào cuối thông điệp
  - Thêm vào k bit 0 sao cho độ dài thông điệp nhận được đồng dư 448 (mod 512)
  - Thêm 64 bit biểu diễn độ dài dài của thông điệp gốc (giá trị lưu dạng little-endian)

# MD5

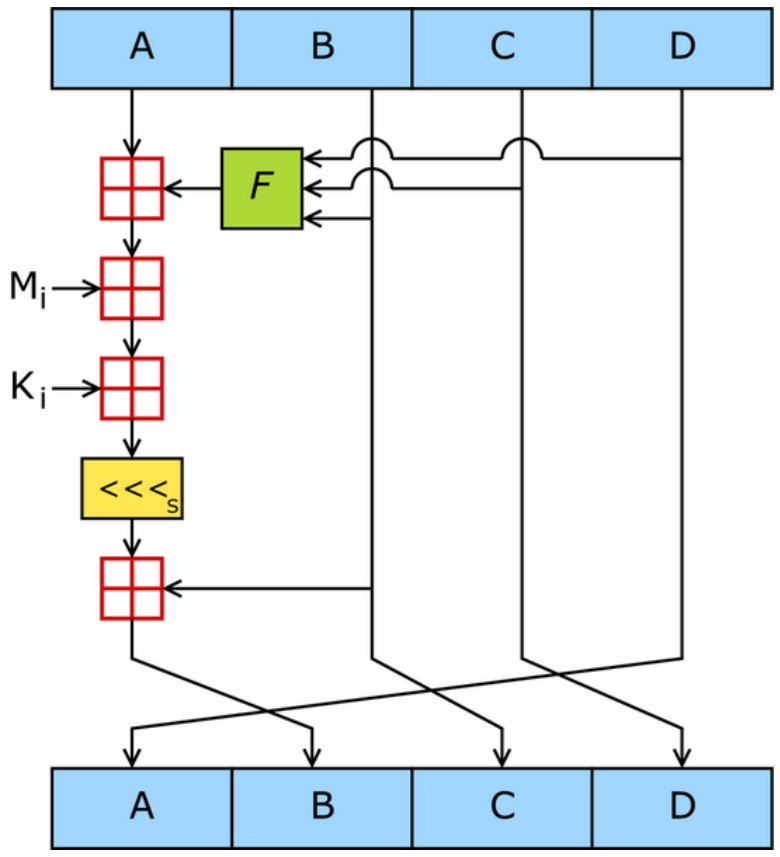
- Chia thông điệp (đã padding) thành các khối 512 bit
- Với mỗi khối 512-bit:
  - Chia thành 16 word (32 bit, little-endian) w[0..15]
  - A= h0, B= h1, C= h2, D= h3
  - 64 chu kỳ xử lý
  - h0+=A, h1+=B, h2+=C, h3+=D
- Kết quả:= h0 | h1 | h2 | h3

# Chu kỳ xử lý trong MD5




- A, B, C, D là 4 word (32 bit) của trạng thái
- *F* là hàm phi tuyến (thay đổi tùy theo chu kỳ)
- <<< *n* là phép quay trái *n* vị trí
- ⊞ phép cộng modulo $2^{32}$.
- $K_t$ là hằng số

# Chu kỳ xử lý trong MD5




**for** $i$ **from** 0 to 63

$f = F[i]\ (B, C, D)$

$g = G[i]\ (i)$

temp = D

D = C

C = B

$B = ((A + f + K[i] + w[g])$

$<<< R[i]) + B$

A = temp

# Chu kỳ xử lý trong MD5

- $0 \le i \le 15$
  - $f := (B \wedge C) \vee ((\neg B) \wedge D)$
  - $g := i$
- $16 \le i \le 31$
  - $f := (D \wedge B) \vee ((\neg D) \wedge C)$
  - $g := (5 \times i + 1)$ **mod** 16
- $32 \le i \le 47$
  - $f := B \oplus C \oplus D$
  - $g := (3 \times i + 5)$ **mod** 16
- $48 \le i \le 63$
  - $f := C \oplus (B \vee (\neg D))$
  - $g := (7 \times i)$ **mod** 16

# SHA1

- Phương pháp Secure Hash Standard (SHS hay SHA1) 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.
- Thông điệp được xử lý theo từng khối 512-bit
- Thông điệp rút gọn độ dài 160-bit

# SHA1

- *Khởi gán các biến:*
  - h0 := 0x67452301
  - h1 := 0xEFCDAB89
  - h2 := 0x98BADCFE
  - h3 := 0x10325476
  - h4 := 0xC3D2E1F0

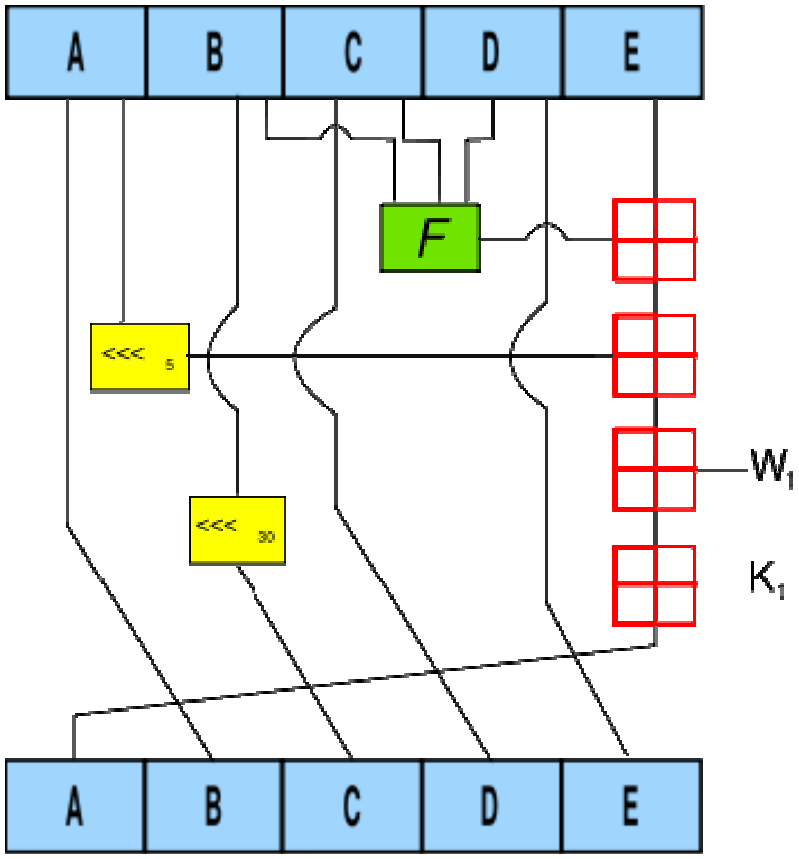
# SHA1

- *Tiền xử lý:*
  - Thêm bit 1 vào cuối thông điệp
  - Thêm vào k bit 0 sao cho độ dài thông điệp nhận được đồng du 448 (mod 512)
  - Thêm 64 bit biểu diễn độ dài dài của thông điệp gốc (giá trị lưu dạng big-endian)

# SHA1

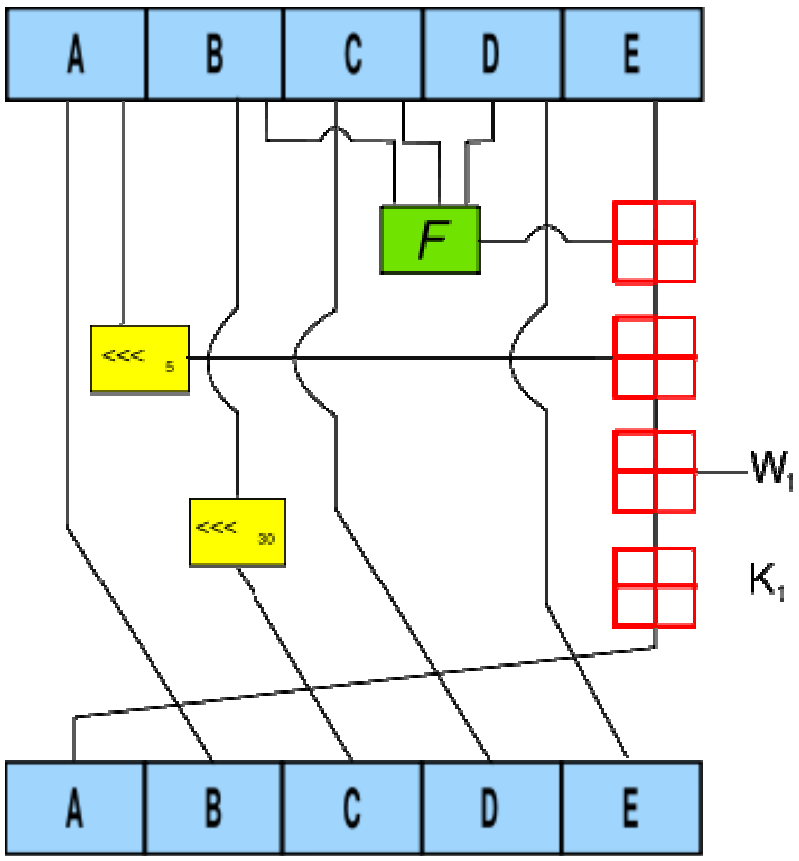
- Chia thông điệp (đã padding) thành các khối 512 bit
- Với mỗi khối 512-bit:
  - Chia thành 16 word (32 bit, big-endian) w[0..15]
  - Mở rộng 16 word (32 bit) thành 80 word (32 bit)
  - $w[i]=(w[i-3] \oplus w[i-8] \oplus w[i-14] \oplus w[i-16]) <<< 1$ với $16 \leq i < 80$
  - A= h0, B= h1, C= h2, D= h3, E= h4
  - 80 chu kỳ xử lý
  - h0+=A, h1+=B, h2+=C, h3+=D, h4+=E
- Kết quả:= h0 | h1 | h2 | h3 | h4

# Chu kỳ xử lý trong SHA1




- *t* là số thứ tự của chu kỳ
- A, B, C, D, E là 5 word (32 bit) của trạng thái
- *F* là hàm phi tuyến (thay đổi tùy theo chu kỳ)
- <<< *n* là phép quay trái *n* vị trí
- ⊞ phép cộng modulo $2^{32}$.
- $K_t$ là hằng số

# Chu kỳ xử lý trong SHA1




**for** $i$ **from** 0 to 79

$f = F[t]$ (B, C, D)

temp = (A <<< 5) + $f$ + E + $K_t$ + w[$i$]

E = D

D = C

C = B <<< 30

B = A

A = temp

# Chu kỳ xử lý trong SHA1

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

$$K_t=\begin{cases}\texttt{0x5a827999}, & 0\le t\le 19\\ \texttt{0x6ed9eba1}, & 20\le t\le 39\\ \texttt{0x8f1bbcdc}, & 40\le t\le 59\\ \texttt{0xca62c1d6}, & 60\le t\le 79\end{cases}$$

# Chu kỳ xử lý trong SHA1

- Công thức của hàm F[t] có thể được viết lại như sau:

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

# Nhóm hàm băm SHA

SHA-256 2002

SHA-384 2002

SHA-1 1994

SHA-224 2004

SHA-512 2002

Secure Hash Standard

# Các thuật toán SHA

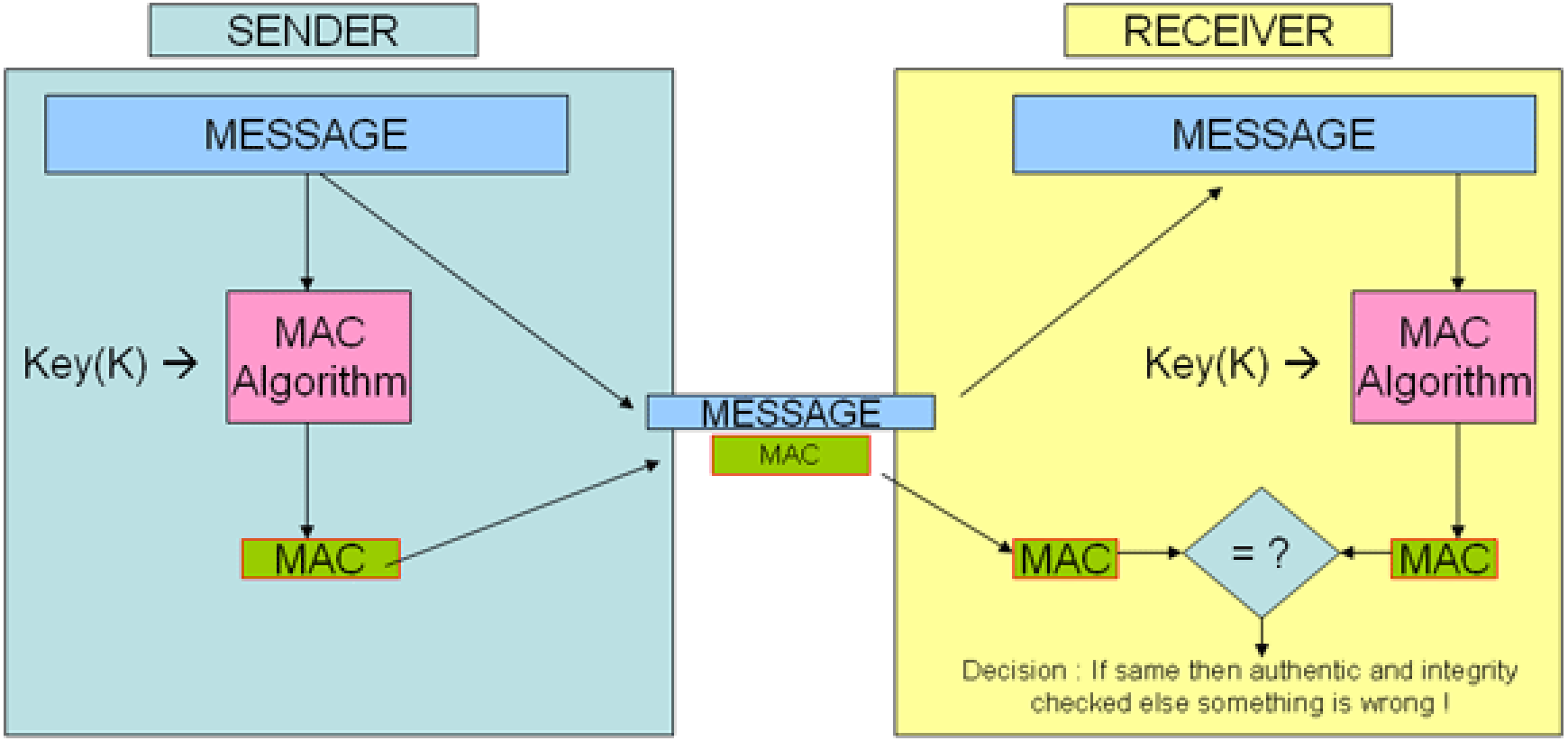
| Thuật toán | Kết quả (bit) | Trạng thái (bit) | Khối (bit) | Thông điệp tối đa (bit) | Word (bit) | # chu kỳ | Thao tác | Đụng độ |
|---|---|---|---|---|---|---|---|---|
| SHA-0 | 160 | 160 | 512 | $2^{64} - 1$ | 32 | 80 | +,and,or, xor,rotl | Có |
| SHA-1 | 160 | 160 | 512 | $2^{64} - 1$ | 32 | 80 | +,and,or, xor,rotl | $2^{63}$ thao tác |
| SHA-256/224 | 256/ 224 | 256 | 512 | $2^{64} - 1$ | 32 | 64 | +,and, or,xor, shr,rotr | Chưa |
| SHA-512/384 | 512/ 384 | 512 | 1024 | $2^{128} - 1$ | 64 | 80 | +,and, or,xor, shr,rotr | Chưa |

# Sử dụng SHA

| Loại ƯD | Sử dụng thông thường | | Suite B | |
|---|---|---|---|---|
| Thuật toán | Đến 2010 | Sau 2010 | Secret | Top Secret |
| SHA-1 | √ | | | |
| SHA-224 | √ | √ | | |
| SHA-256 | √ | √ | √ | |
| SHA-384 | √ | √ | √ | √ |
| SHA-512 | √ | √ | | |

Nguồn: NIST Cryptographic Standards Status Report
April 4, 2006
Bill Burr
Manager, Security Technology Group
NIST
william.burr@nist.gov

# Message authentication code (MAC)




Mục đích: xác định nguồn gốc của thông tin

# MAC và chữ ký điện tử

- Phát sinh MAC và kiểm tra MAC sử dụng chung khóa bí mật (secret key)
- Người gửi và người nhận phải thỏa thuận trước khóa bí mật (giống mã hóa đối xứng)
- Không hỗ trợ việc chống từ chối trách nhiệm (non-repudiation)