AN TOÀN & BẢO MẬT THÔNG TIN

Giảng viên: Ths Phạm Thanh Bình

Bộ môn Kỹ thuật máy tính & mạng

http://dhthuyloi.blogspot.com

Chương 3:

MẬT MÃ KHOÁ CÔNG KHAI VÀ ỨNG DỤNG

- Giới thiệu chung
- Thuật toán mã hoá RSA
- Các hàm Hash và MAC
- Chữ ký số và chứng thực

Bài 3.3 Các hàm Hash và MAC

- Các hàm Hash
- MAC

Các hàm Hash

- Hàm Hash là một công cụ quan trọng trong chứng thực.
- Nhiệm vụ của hàm hash H là tác động lên thông điệp đầu vào M (có kích thước bất kì) để thu được một khối dữ liệu nhỏ H(M) có kích thước cố định ở đầu ra. H(M) được gọi là mã hash của M.
- H(M) đặc trưng cho M, nhưng thường có kích thước nhỏ hơn M, sẽ được sử dụng làm giá trị chứng thực.

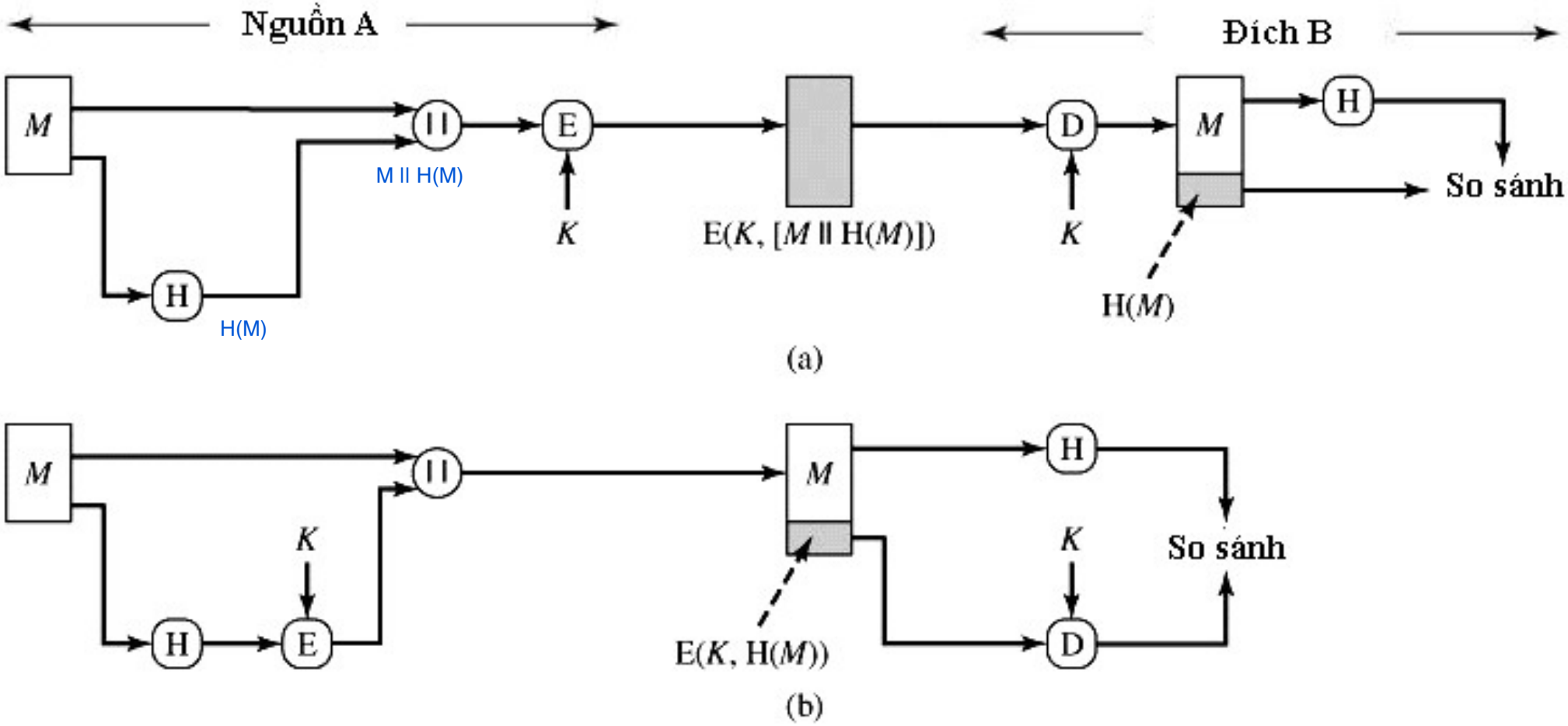
Các đặc điểm của hàm Hash

- H(M) là một hàm của tất cả các bít trong thông điệp M, chỉ cần thay đổi 1 bít của M cũng khiến H(M) bị thay đổi theo.
- Việc tính toán để tìm một giá trị N ≠ M thoả mãn điều kiện H(N) = H(M) là rất khó khăn (hoặc không thể).
- Không thể tính ngược để thu được M từ H(M).

Ứng dụng của mã hash

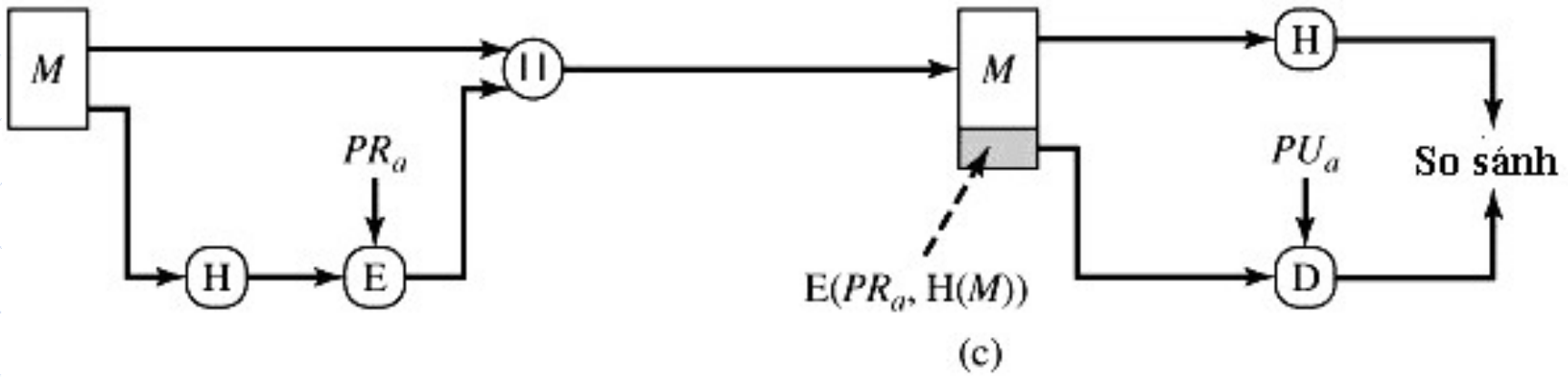
- Mã hash thường được đính kèm và gửi cùng với thông điệp, nhằm mục đích chứng thực cho thông điệp đó là chính xác về mặt nội dung và nguồn gốc.
- Mã hash thường được dùng kết hợp với các mật mã khoá công khai hoặc đối xứng.

Hàm Hash và mật mã đối xứng:

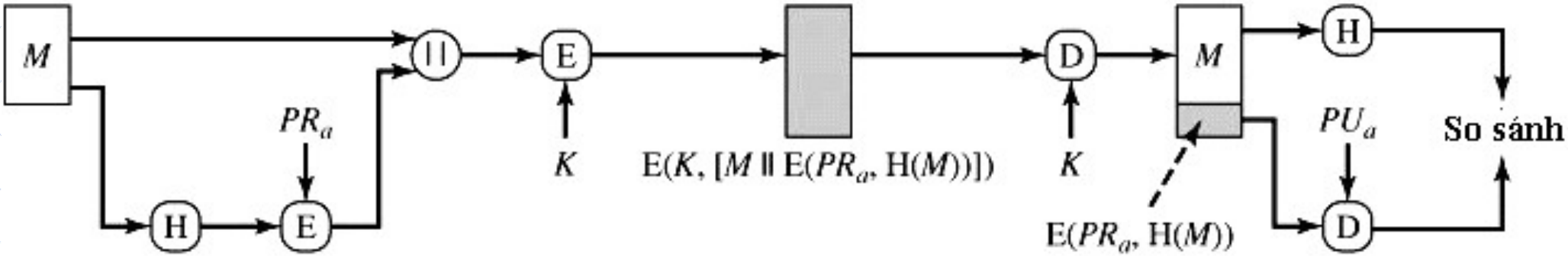


- Mục đích: Đảm bảo rằng thông điệp M đến từ A và không bị thay đổi trên đường truyền (do lỗi đường truyền hoặc bị tấn công)
- Hình (b) giảm được khối lượng tính toán so với hình (a), nhưng không có khả năng giữ bí mật nội dung thông điệp như hình (a).

Hàm Hash và mật mã khoá công khai:



(c)



(d)

- Hình (c): Đảm bảo rằng thông điệp M đến từ A và không bị thay đổi trên đường truyền, nhưng không có khả năng giữ bí mật nội dung thông điệp.

  Đây chính là cơ sở của chữ kí số.
- Hình (d) có thêm khả năng bảo mật cho thông điệp.

Ví dụ về các hàm hash đơn giản:

Ví dụ 1:

- Giả sử thông điệp đầu vào được chia thành N khối, mỗi khối dài n bít
- Mã hash dài n bít được tính bằng cách XOR các bít tương ứng của tất cả các khối:

$$H_i = b_{i1} \oplus b_{i2} \oplus \ldots \oplus b_{iN}$$

Trong đó:

H_i là bit thứ i của mã hash, $1 \le i \le n$.

b_{ij} là bit thứ i của khối thứ j.

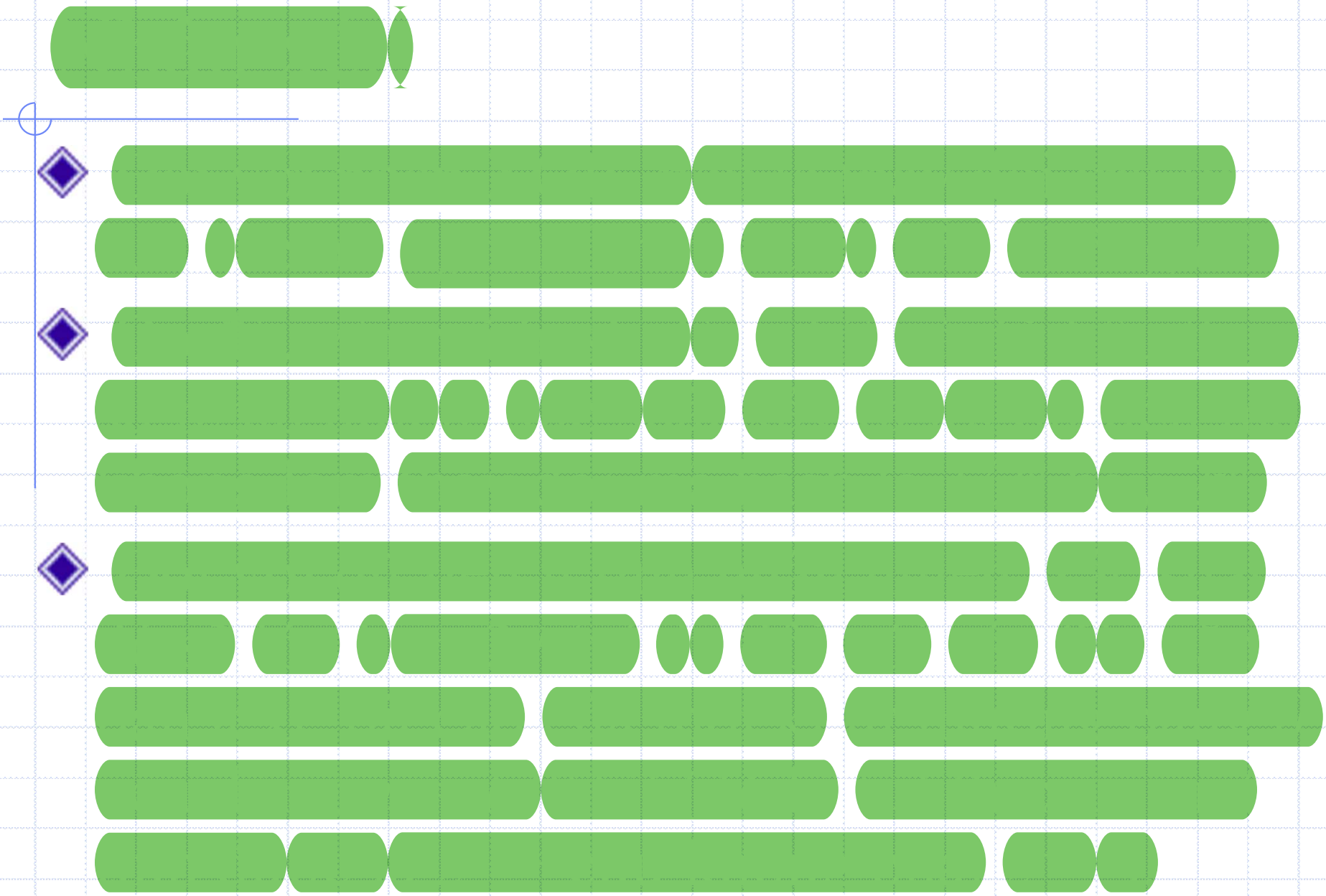
Bài tập 1:

- Nhập một chuỗi kí tự từ bàn phím, chia chuỗi thành từng khối *8* bít (ứng với 1 kí tự) rồi tính mã hash của chuỗi theo phương pháp đã nêu ở Ví dụ 1.

Bài tập 2:

- Nhập một chuỗi kí tự từ bàn phím, chia chuỗi thành từng khối *64* bít (ứng với *8* kí tự) rồi tính mã hash của chuỗi theo phương pháp đã nêu ở Ví dụ 1.

(Nếu độ dài chuỗi không phải là bội số của *8* thì có thể chèn thêm các kí tự quy ước.)

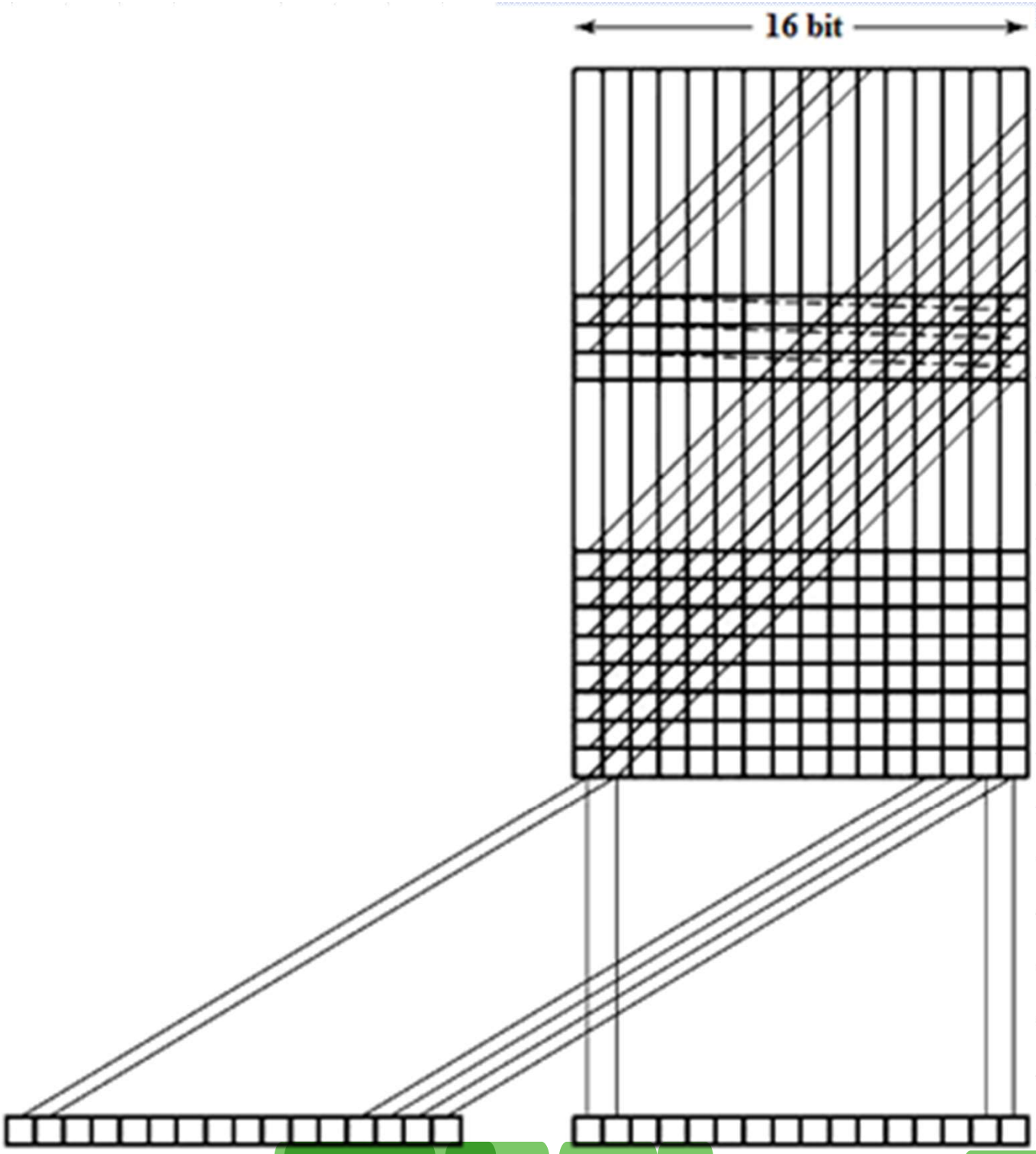


Ví dụ 2:

Giả sử thông điệp đầu vào được chia thành N khối, mỗi khối dài n bít. Để cải thiện tính ngẫu nhiên của đầu vào so với Ví dụ 1, ta sẽ thực hiện phép quay phải trước khi tiến hành XOR:

- Giá trị hash ban đầu được đặt bằng 0
- Thực hiện *N* vòng lặp (*i* từ *0* → *N-1*), mỗi vòng làm các công việc sau:

  - Quay phải một bít giá trị hash hiện tại
  - XOR giá trị hash với khối dữ liệu thứ *i*


16 bit
XOR với 1 bit quay phải
XOR mọi khối 16 bit

Bài tập:

- Nhập một chuỗi kí tự từ bàn phím, chia chuỗi thành từng khối *8* bít (ứng với 1 kí tự) rồi tính mã hash của chuỗi theo phương pháp đã nêu ở Ví dụ 2.

Nhận xét:

- Hai hàm hash đơn giản nói trên có thể được áp dụng để phát hiện lỗi dữ liệu.
- Tuy nhiên chúng ít được dùng trong bảo mật vì độ an toàn không cao. Nếu biết thông điệp ban đầu và mã hash có thể dễ dàng sinh ra một thông điệp mới có cùng mã hash với thông điệp ban đầu.
- Trên thực tế người ta sử dụng các thuật toán hash có độ phức tạp lớn hơn nhiều như SHA, WHIRLPOOL, MD5…

= mã hash: kẻ tấn công đã tìm được 1 khoá khác khoá ban đầu nhưng thông điệp lại giống với thông điệp ban đầu

Bẻ khoá mật mã: tìm ra được khoá. cách giải mã giúp kẻ bẻ khoá đọc được ND thông điệp

Thuật toán SHA

- Thuật toán SHA (Secure Hash Algorithm) được phát triển bởi National Institute of Standards and Technology (NIST) và công bố thành chuẩn quốc gia Hoa Kỳ (FIPS 180) vào năm 1993
- Phiên bản sửa đổi FIPS 180-1 được phát hành vào năm 1995 và còn được gọi là SHA-1
- Năm 2002, NIST công bố FIPS 180-2, gồm ba phiên bản mới của SHA, là SHA-256, SHA-384 và SHA-512 (ứng với các chiều dài của giá trị hash là 256, 384, và 512 bit)
- Năm 2005, SHA-1 bị phá vỡ bởi một nhóm nghiên cứu tại Đại học Shandong (Trung Quốc), và không còn được sử dụng nữa

So sánh tham số của các thuật toán SHA

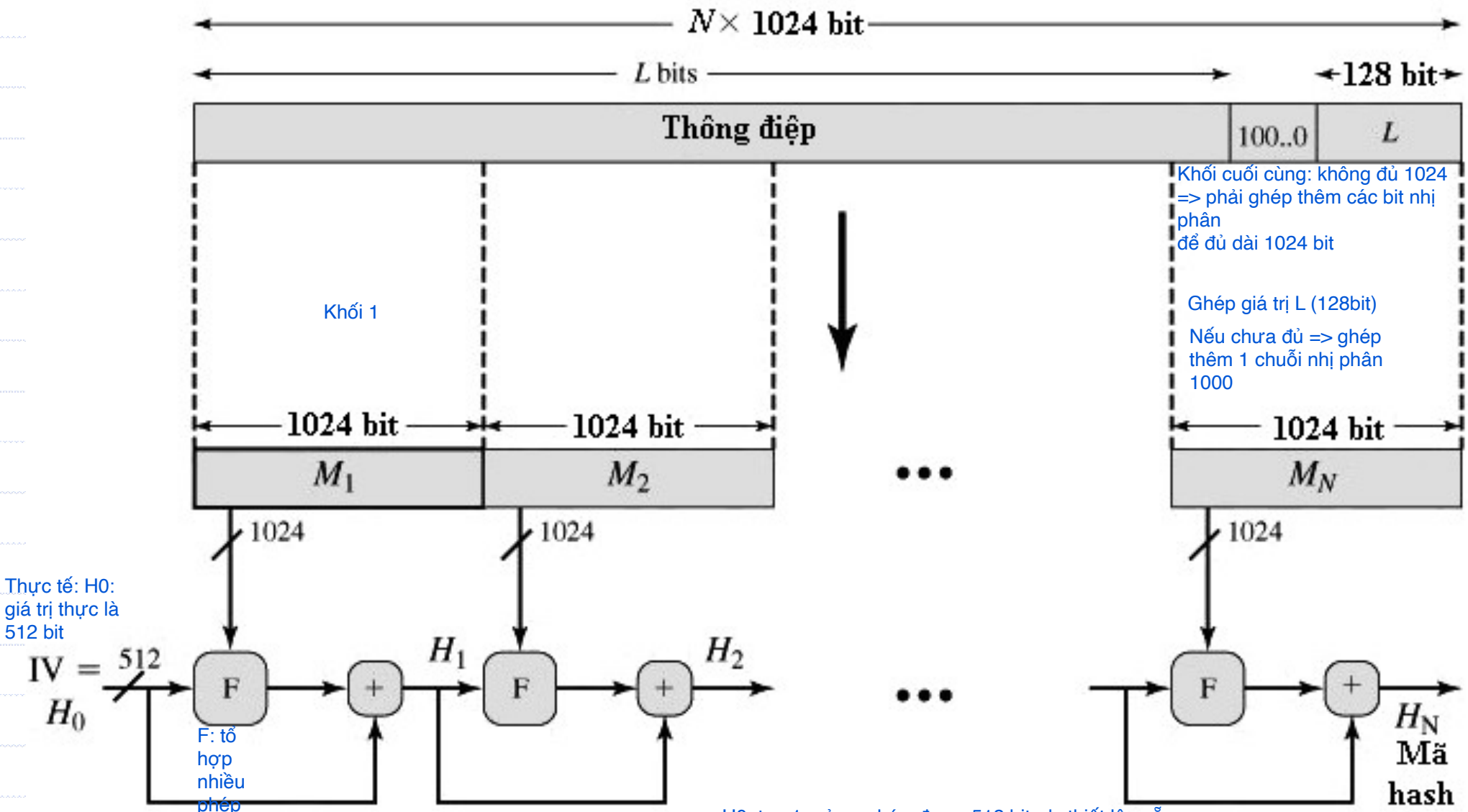
	SHA-1	SHA-256	SHA-384	SHA-512
Kích thước giá trị hash	160	256	384	512
Kích thước thông điệp	$<2^{64}$	$<2^{64}$	$<2^{128}$	$<2^{128}$
Kích thước khối	512	512	1024	1024
Kích thước word	32	32	64	64
Số bước số vòng lặp cần được thực hiện	80	64	80	80
Độ an toàn ($2^{n/2}$) tổng số lượng phép tính mà kẻ tấn công phải làm để có thể tạo ra được 1 thông điệp giả mạo có mã Hash giống thông điệp ban đầu	2^{80}	2^{128}	2^{192}	2^{256}

- Các kích thước được tính bằng bit
- Độ an toàn là số lượng thao tác cần thiết để tấn công vào SHA
- Số bước là số vòng xử lý của hàm tính toán chính (hàm F)

SHA-512

- Thông điệp đầu vào có chiều dài tối đa nhỏ hơn 2^{128} bít, chia thành N khối 1024 bít
- Đầu ra là một giá trị hash 512 bít

đem thông điệp đầu vào chặt ra thành nhiều => đem đi áp dụng thuật toán



Khối cuối cùng: không đủ 1024 => phải ghép thêm các bit nhị phân để đủ dài 1024 bit

Ghép giá trị L (128bit)

Nếu chưa đủ => ghép thêm 1 chuỗi nhị phân 1000

Thực tế: H0: giá trị thực là 512 bit

F: tổ hợp nhiều phép toán phức tạp

- H0: tạo 1 mảng: chứa được 512 bit, dc thiết lập sẵn
- 512 bit dc tương tác với F
- 64bit : H0 + đầu ra F => thu dc H1
- H1 tương tác với khối tiếp theo M2 => lại cộng 64bit và thu được H2
- H2 tương tac với khối M3 => thu được H3
- ….. lặp lại n lần => đầu ra sau phép tương tác thứ n => thu được Hn
- Hn là mã Hash của thông điệp ban đầu

Quá trình thực hiện:

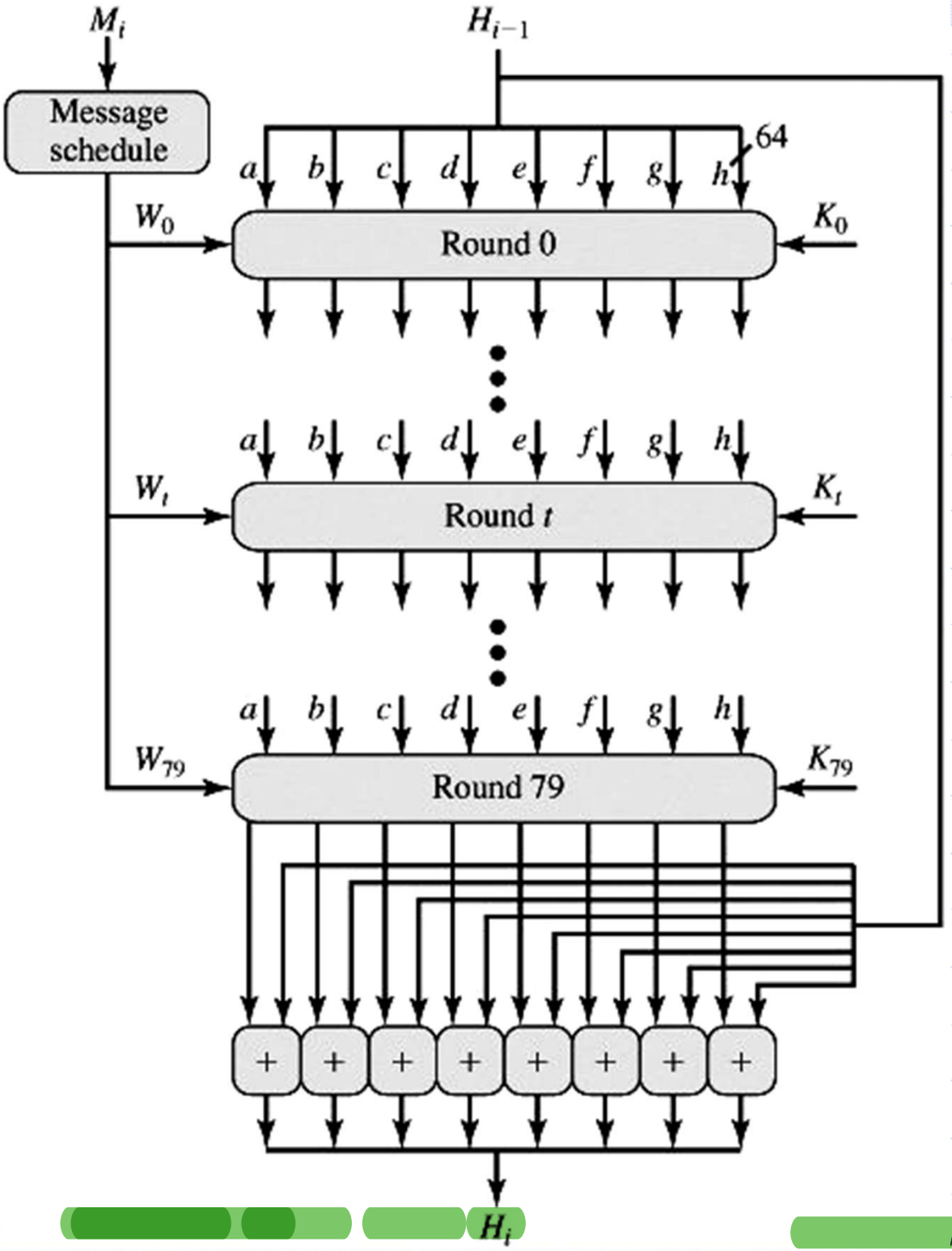
- **Bước 1: Nối các bit đệm**. Thông điệp được độn thêm các bit sao cho chiều dài của nó khi chia cho chiều dài khối 1024 thì dư 896
- **Bước 2: Nối giá trị chiều dài**. Một giá trị 128 bit nhị phân được nối thêm vào cuối thông điệp. Khối này là một số nguyên không dấu, nó chứa chiều dài thực của thông điệp ban đầu (128+896=1024)
- Kết quả của hai bước đầu là thu được thông điệp có chiều dài là bội số của 1024 bit. Tổng chiều dài của thông điệp mới này là $N \times 1024$ bit. Các khối của thông điệp được kí hiệu là $M_1, M_2, ..., M_N$.

- **Bước 3: Khởi tạo bộ đệm giá trị hash**. Một bộ đệm 512 bit được sử dụng để lưu trữ các kết quả trung gian và cuối chu trình sẽ là kết quả của hàm hash. Bộ đệm gồm tám thanh ghi 64-bit (*a*, *b*, *c*, *d*, *e*, *f*, *g*, *h*). Các thanh ghi này được khởi tạo với các số nguyên (thập lục phân) 64-bit tương ứng như sau:

```
a = 6A09E667F3BCC908
b = BB67AE8584CAA73B
c = 3C6EF372FE94F82B
d = A54FF53A5F1D36F1
e = 510E527FADE682D1
f = 9B05688C2B3E6C1F
g = 1F83D9ABFB41BD6B
h = 5BE0CDI9137E2179
```

- **Bước 4: Xử lý thông điệp theo các khối 1024-bit (128-word).** Trọng tâm của thuật toán là một module chứa 80 vòng; module này được kí hiệu là F.

  Cấu trúc logic của thuật toán được miêu tả trong hình sau:


M_i
H_{i-1}
Message schedule
64
a b c d e f g h
W_0
Round 0
K_0
a b c d e f g h
W_t
Round t
K_t
a b c d e f g h
W_{79}
Round 79
K_{79}
+ + + + + + + +
H_i

Trong đó:

- W_i là các giá trị 64 bít được lấy ra từ 1024 bít của khối.
- K_i là các hằng số ngẫu nhiên dài 64 bít, chúng được cộng vào mỗi vòng nhằm loại bỏ tính đều đặn của dữ liệu đầu vào.

- **Bước 5: Đầu ra.** Sau khi tất cả *N* khối 1024-bit đã được xử lý, đầu ra của giai đoạn thứ *N* là một thông điệp sắp xếp (message digest) hay mã hash, là một giá trị 512-bit.

MAC

- MAC (Message Authentication Code - Mật mã chứng thực thông điệp) là một kĩ thuật mã hoá nhằm biến đổi thông điệp đầu vào M (có kích thước bất kì) thành một khối dữ liệu nhỏ có kích thước cố định ở đầu ra.
- Khối dữ liệu đầu ra đó được gọi là mã MAC của thông điệp M. Nếu M bị thay đổi thì mã MAC của nó cũng sẽ thay đổi theo.

- Giống như mã hash, mã MAC thường được đính kèm và gửi cùng với thông điệp, nhằm mục đích chứng thực cho thông điệp đó là chính xác về mặt nội dung và nguồn gốc.
- Sự khác biệt cơ bản của MAC với các hàm hash là: Kĩ thuật mã hoá của MAC có sử dụng mật khoá K:

để tính được hàm Mac phải biết khoá K
điều này k cần với mã Hash

$$MAC = C(K, M)$$

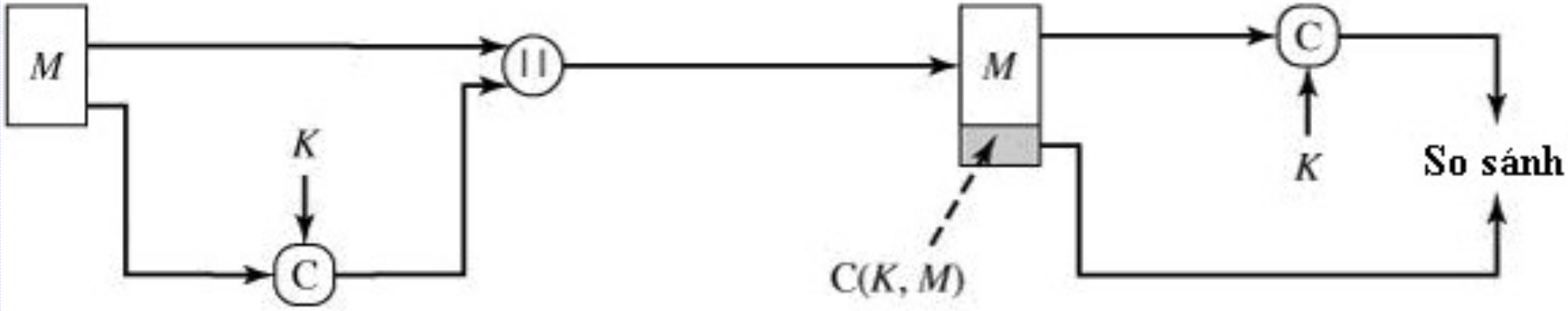
M = Thông điệp đầu vào.

C = Hàm MAC.

K = Mật khoá chia xẻ giữa bên gửi và bên nhận.

MAC = Mã chứng thực thông điệp (Mã MAC).

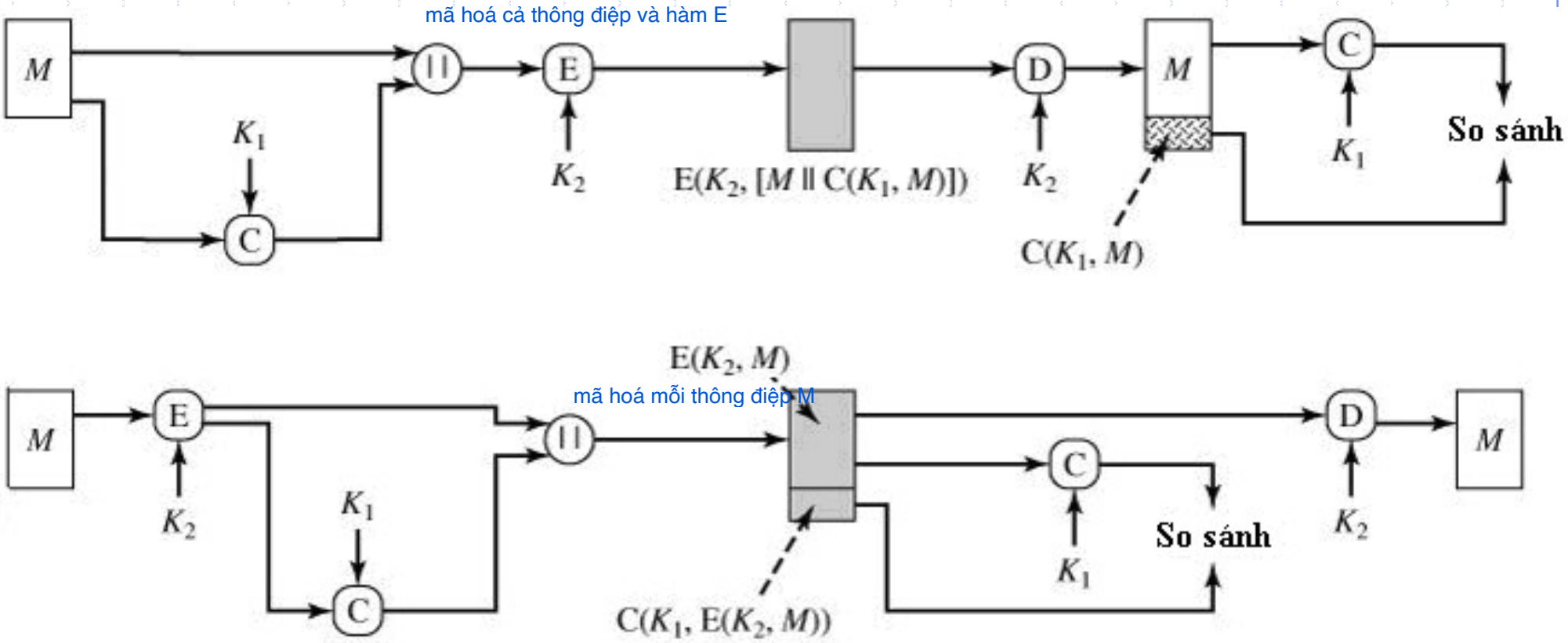
- Hàm Hash: Hacker sửa thông điệp trên đường truyền
- MAC; Hacker không thể sửa dc thông điệp M nhưng k sửa dc mã MAC do không có khoá
=> chỉ cần 1 mình mã MAC là có thể xác định ND thông điệp
- Mã Hash bắt buộc phải phối hợp với mật mã



M || K(C)

- Nếu có kẻ tấn công thay đổi thông điệp *M*, hắn sẽ không thể tạo ra mã *MAC* phù hợp vì không biết mật khoá *K*.
- Người nhận sẽ phát hiện ra sự giả mạo khi giá trị *MAC* nhận được và giá trị *MAC* mà họ tính toán từ *M* có sự khác nhau.

Một số cách dùng khác của MAC



Hết Phần 3_2