TRUYỀN VÀ BẢO MẬT THÔNG TIN

Bài 8:

Mã chứng thực thông điệp, hàm băm

VŨ THỊ TRÀ

Nội dung

- Chứng thực thông điệp với checksum
- Mã hóa MAC
  - Hàm băm : BT sinh nhật;
  - Hàm băm : MD5 và SHA-1; HMAC;
- Chữ ký điện tử
- Một số ứng dụng khác

Chứng thực thông điệp với Checksum

Bài toán thay đổi nội dung thông điệp

Nếu Trudy can thiệp sửa đổi bản mã thì bản giải mã sẽ là một chuỗi bít vô nghĩa, và người nhận biết được là dữ liệu đã bị thay đổi. Ta có hai kết luận sau về tính chứng thực của mã hóa đối xứng và mã hóa khóa công khai:

- **KL**: Trudy không thể tìm ra một bản mã C_T, sao cho khi Bob giải mã bằng khóa K_{AB} (hay khóa K_{UA} với mã khóa công khai) cho ra bản rõ P_T có ý nghĩa theo ý muốn của Trudy.

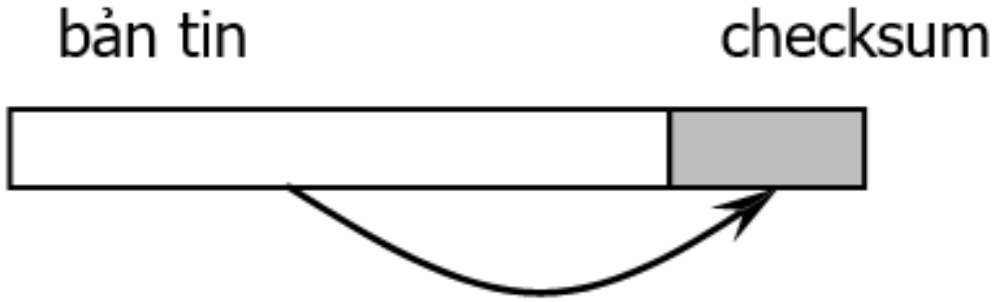
V/ v chứng thực thông điệp

Trong thực tế có nhiều loại dữ liệu mà các bít gần như là ngẫu nhiên. Chẳng hạn như dữ liệu hình ảnh bitmap hay âm thanh. Ngoài ra đối với máy tính, việc nhận dạng ra thế nào là dãy bít có ý nghĩa là một công việc khó khăn. Do đó, chúng ta hầu như chấp nhận rằng bất cứ dãy bít nào cũng có thể có ý nghĩa. Lúc này các phương pháp mã hóa đối xứng và mã hóa công khai không thể bảo đảm tính chứng thực.

Để giải quyết vấn đề này, mã hóa phải vận dụng khái niệm *redundancy* của lĩnh vực truyền số liệu, tức là thêm vào một ít dữ liệu (*checksum*) để biến bản tin, từ dãy bít ngẫu nhiên, trở thành dãy bít có cấu trúc.

Nhiễu và checksum

Trong quá trình truyền số liệu, do tác động *nhiễu* của môi trường, bản tin lúc đến đích có thể bị sai lệch so với bản tin ban đầu trước khi truyền. Để phát hiện nhiễu, một đoạn bít ngắn gọi là *checksum* được tính toán từ dãy bít của bản tin, và gắn vào sau bản tin để tạo *redundancy,* và được truyền cùng với bản tin đến đích.



Phương pháp kiểm lỗi checksum CRC (Cyclic Redundancy Check)

Một đoạn bít ngắn được chọn làm số chia, lấy dãy bít của thông điệp chia cho số chia này, phần dư còn lại được gọi là giá trị checksum CRC. Phép chia này khác phép chia thường ở chỗ dùng phép XOR thay cho phép trừ. Giả sử thông điệp là 10101011 và số chia là 10011, ta có:

```
10101011 | 10011
10011    | 1011
 11001
 10011
  10101
  10011
    110
```

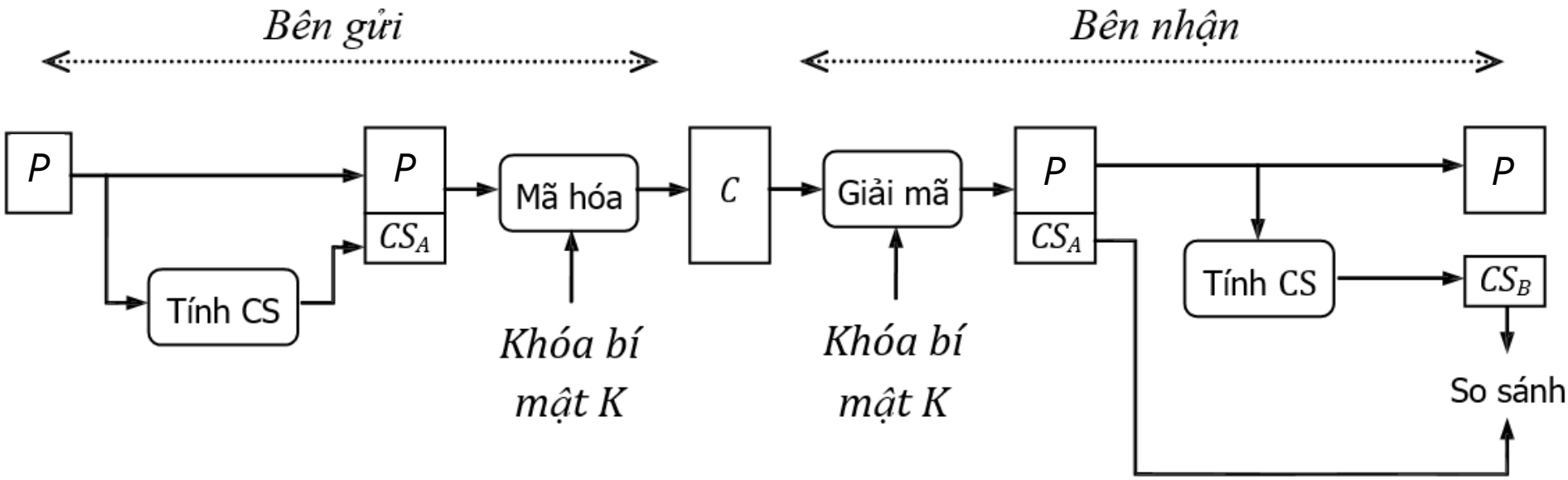
Phương pháp kiểm lỗi checksum CRC (Cyclic Redundancy Check)

Giá trị CRC là phần dư 0110 (ít hơn 1 bít so với số chia) được gửi kèm thông điệp đến người nhận. Người nhận cũng thực hiện phép tính CRC như vậy. Nếu giá trị CRC người nhận tính được trùng khớp với CRC của người gửi thì có nghĩa là thông điệp không bị lỗi trong quá trình truyền dữ liệu. Trong phương pháp CRC không khó để tìm ra hai dãy bít khác nhau mà *có cùng* CRC. Có nghĩa là có thể xảy ra lỗi mà không phát hiện được. Tuy nhiên xác suất ngẫu nhiên xảy ra lỗi trên đường truyền mà làm cho dãy bít truyền và dãy bít nhận có cùng giá trị CRC là rất thấp.

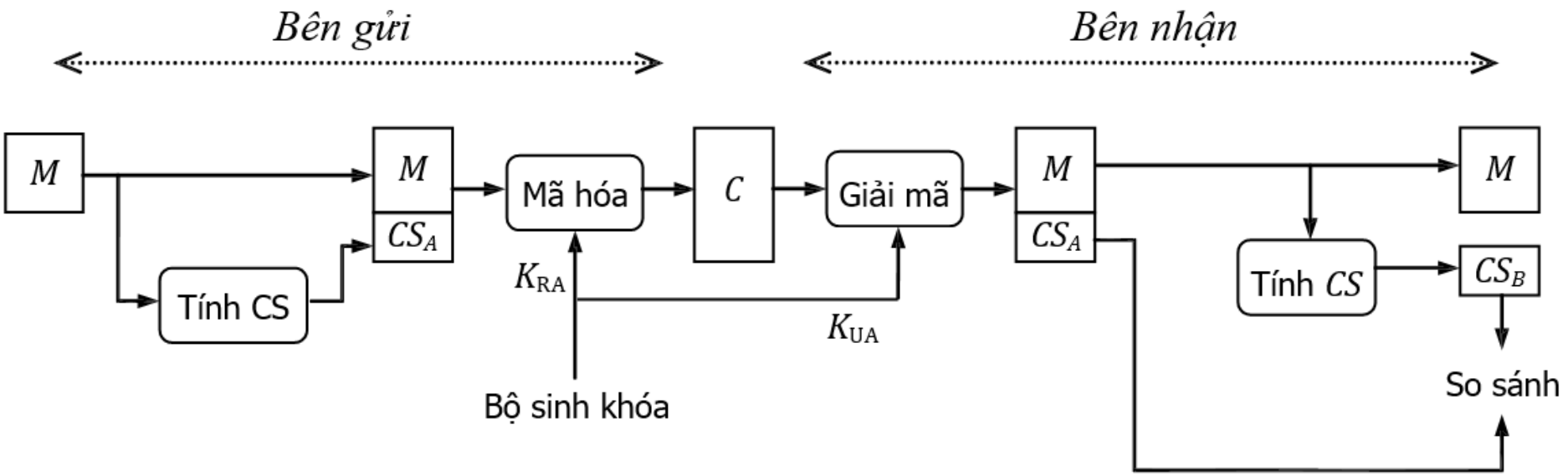
Cơ chế checksum và chứng thực thông điệp

- Nếu áp dụng cơ chế checksum vào chứng thực thông điệp, người gởi có thể tính checksum từ dãy bít của thông điệp, sau đó nối checksum này vào dãy bít đó. Như vậy chúng ta được một dãy bít có cấu trúc. Sau đó tiến hành mã hóa đối xứng hay mã hóa khóa công khai trên dãy bít mới. Vì kích thước của checksum là ngắn nên cũng không ảnh hưởng lắm đến tốc độ mã hóa và băng thông sử dụng.

Mô hình chứng thực mã hóa đối xứng có dùng checksum



Mô hình chứng thực mã hóa khóa công khai có dùng checksum



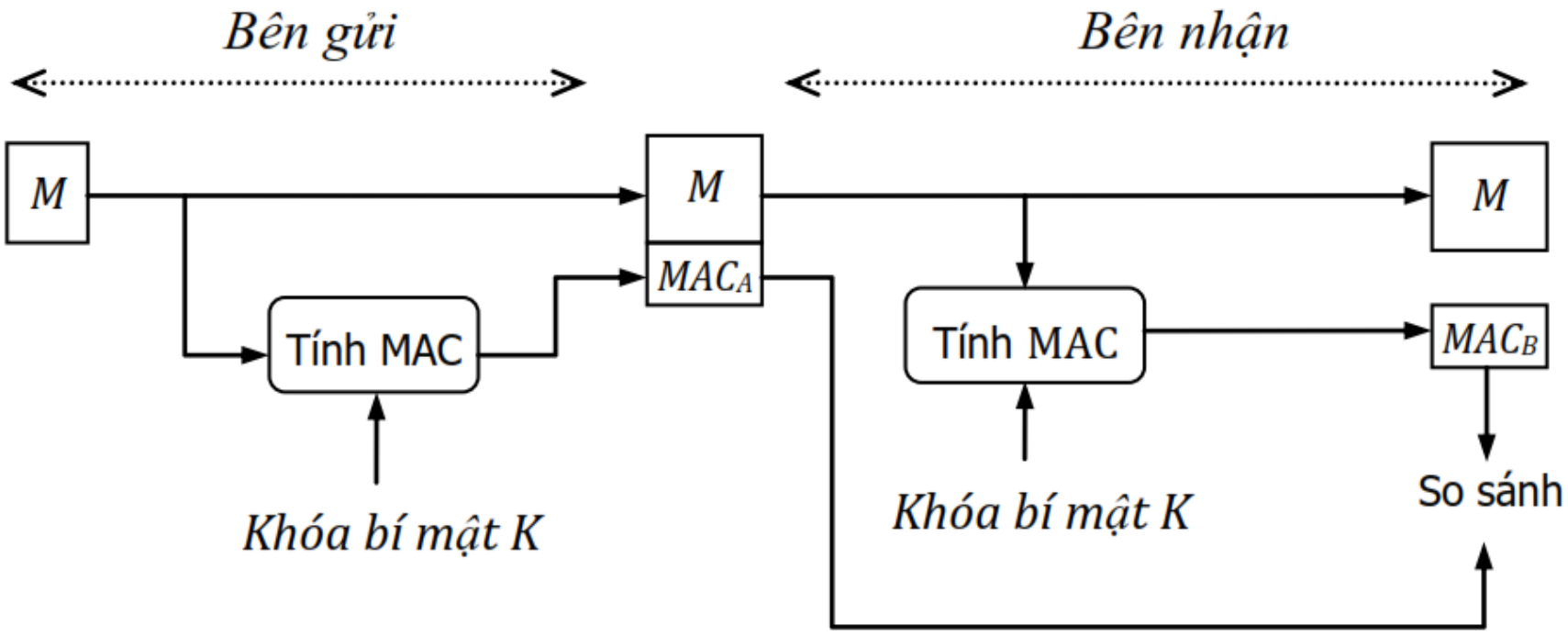
- ✓ *Nếu Trudy sửa bảng mã C, thì bản giải mã của Bob, ký hiệu M_T và CS_T, sẽ mất đi tính cấu trúc. Nghĩa là checksum CS_B mà Bob tính được từ M_T không giống với CS_T.*
- ✓ *Nếu hàm checksum có độ phức tạp cao thì xác suất để $CS_B = CS_T$ là rất thấp.*

Mã chứng thực thông điệp MAC

Mã chứng thực thông điệp (MAC)

- Mã chứng thực thông điệp (MAC) có thể coi là một dạng checksum của mã hóa, được tính theo công thức MAC = C(M, K), trong đó:
    1) M là thông điệp cần tính MAC
    2) K là khóa bí mật được chia sẽ giữa người gởi và người nhận
    3) C là hàm tính MAC
- Vì MAC có khóa K bí mật giữa người gởi và người nhận nên chỉ có người gởi và người nhận mới có thể tính được giá trị MAC tương ứng.

Mô hình mã chứng thực thông điệp (MAC)



- ✓ *Mô hình không đảm bảo tính bảo mật.*
- ✓ *Để có tính bảo mật, M và MAC_A cần được mã hóa trước khi truyền đi.*

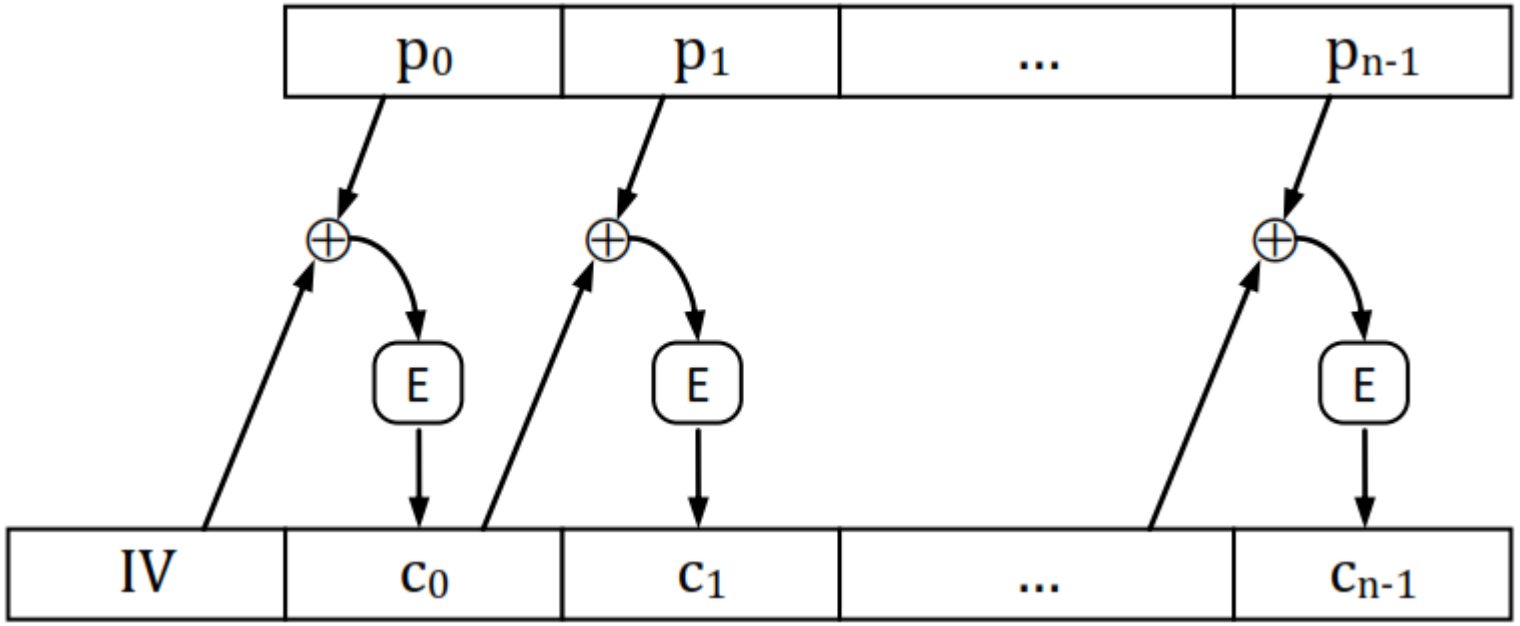
Mô hình mã chứng thực thông điệp (MAC)

- Mã hóa đối xứng cũng có tính chứng thực, như vậy thì tại sao không dùng **mã hóa đối xứng** mà cần dùng **MAC**?

  →*Trong một số trường hợp người ta không cần tính bảo mật mà chỉ cần tính chứng thực, nên sử dụng MAC tiết kiệm được thời gian xử lý hơn.*

- Trong thực tế, người ta hay dùng mô hình CBC và phương pháp DES của mã hóa đối xứng để tính giá trị MAC. Bản mã C_{n-1} được chọn làm giá trị MAC cho bản tin P. Kích thước của MAC là 64 bít, kích thước của khóa K là 56 bít.

Mô hình CBC



Hàm băm – Hash function

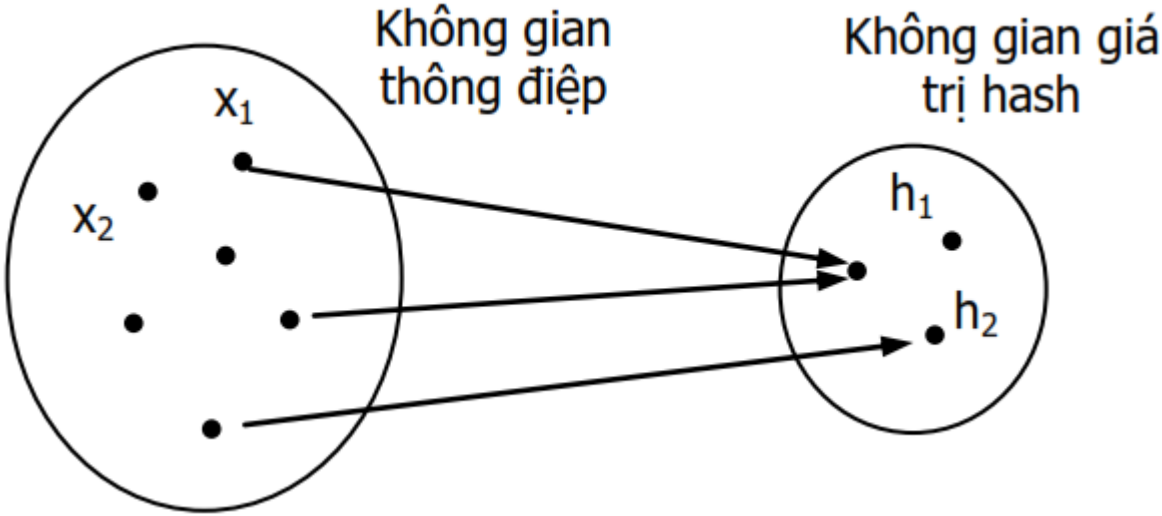
Hàm băm – Hash function

Trong khi phương pháp checksum CRC cho phép hai dãy bít có cùng checksum, thì hàm băm H(x) là một hàm tính checksum mạnh thỏa mãn các yêu cầu sau:

1) H có thể áp dụng cho các thông điệp x với các độ dài khác nhau
2) Kích thước của output h = H(x) là cố định và nhỏ
3) Tính một chiều: với một h cho trước, không thể tìm lại được x sao cho h = H(x) (về mặt thời gian tính toán)
4) Tính chống trùng yếu: cho trước một x, không thể tìm y≠ x sao cho H(x) = H(y)
5) Tính chống trùng mạnh: không thể tìm ra cặp x, y bất kỳ (x≠y) sao cho H(x) = H(y), hay nói cách khác nếu H(x) = H(y) thì có thể chắc chắn rằng x = y.

Hàm băm – Hash function

- Kích thước của input x là bất kỳ còn kích thước của h là nhỏ, ví dụ giả sử kích thước của *x* là *512 bít* còn kích thước của *h* là *128 bít*. Như vậy trung bình có khoảng 2^{384} giá trị *x* mà có cùng giá trị *h*. Việc trùng là không thể loại bỏ. Tính chống trùng của hàm Hash là yêu cầu rằng việc tìm ra hai input *x* như vậy thì phải là rất khó về mặt thời gian tính toán.



Hàm băm

- VD: Xét hai hàm sau: hàm lấy khuôn mặt và hàm lấy dấu vây tay. Có thể thấy hàm lấy khuôn mặt không phải là hàm hash vì chúng có thể tìm ra 2 người giống nhau ở khuôn mặt. Còn hàm lấy dấu vân tay là hàm hash vì trên khắp thế giới không tìm ra hai người giống nhau về dấu vân tay.
- Một yêu cầu nữa của hàm Hash là kích thước của output h không được quá lớn. Nếu kích thước h lớn thì dễ đạt được tính chống trùng tuy nhiên sẽ tốn dung lượng đường truyền.

→ Vậy kích thước của output h cần thiết là bao nhiêu để thực hiện chống trùng có hiệu quả?

Bài toán ngày sinh nhật

- **Bài toán 1**: Giả sử trong phòng có 30 người. Vậy xác suất để có hai người có cùng ngày sinh là bao nhiêu phần trăm?

Nguyên lý chuồng bồ câu Dirichlet phát biểu rằng là cần có 365+1 = 366 người để tìm thấy hai người có cùng ngày sinh với xác suất 100% (để đơn giản, chúng ta bỏ qua năm nhuận). Do đó hầu hết chúng ta sẽ nghĩ rằng với 30 người thì xác suất hai người cùng ngày sinh là nhỏ, chắc chắn nhỏ hơn 50%. Tuy nhiên nếu kiểm tra bằng toán học thì chỉ cần ***23 người*** *là đủ để xác suất lớn hơn 50%. Vì vậy bài toán này còn được gọi dưới tên* ***nghịch lý ngày sinh****.*

Bài toán ngày sinh nhật

$$p(M) = \left(\frac{364}{365}\right)\left(\frac{363}{365}\right)\dots\left(\frac{365-M+1}{365}\right) = \left(1-\frac{1}{365}\right)\left(1-\frac{2}{365}\right)\dots\left(1-\frac{M-1}{365}\right)$$

Xét hàm lũy thừa e^x, chúng ta đã biết một xấp xỉ của e^x khi x nhỏ là $e^x=1+x$. Do đó $p(M)$ có thể viết lại thành:

$$p(M) \approx e^{\frac{-1}{365}}.e^{\frac{-2}{365}}.e^{\frac{-3}{365}}\dots e^{\frac{-M+1}{365}} = e^{-\frac{1+2+3+\cdots(M-1)}{365}} = e^{-\frac{M(M-1)}{2\times365}}$$

Dẫn đến xác suất để tồn tại ít nhất hai người có ngày sinh giống nhau là

$$1-p(M) \approx 1-e^{-\frac{M(M-1)}{2\times365}}$$

Để xác suất này lớn hơn 50% , chúng ta cho biểu thức trên lớn hơn 0.5:

$$1-e^{-\frac{M(M-1)}{2\times365}} \geq \frac{1}{2}$$

$$M(M-1) \geq 2\times365\times log_e 2$$

$M \in (-\infty;-21{,}99) \cup (22{,}99;\infty)$

http://coccoc.com/search/math/

Bài toán ngày sinh nhật

- **Bài toán 2**: Giả sử bạn đang ở trong một căn phòng với M người khác. Hỏi M tối thiểu là bao nhiêu để tồn tại một người có cùng ngày sinh với bạn với xác suất lớn hớn 50% ?

→ *Giải ra M tối thiểu phải có 253 người*

Vấn đề ngày sinh nhật và hàm băm

- Áp dụng vấn đề ngày sinh nhật vào hàm băm, ta thấy rằng tính chống trùng mạnh giống bài toán 1, còn tính chống trùng yếu giống bài toán 2. Giả sử số bít của kết xuất *h* của hàm băm là *n* bít, như vậy số lượng giá trị có thể có của *h* là $N = 2^n$. Giả sử thêm rằng 2^n giá trị băm này đều là *ngẫu nhiên, có khả năng xuất hiện như nhau*. Khi đó, ta có:

$$M(M-1) \geq 2 * 2^n * \log_e 2$$

$$M \geq 2^{n/2}$$

Vấn đề ngày sinh nhật và hàm băm

- Giống như vấn đề ngày sinh nhật, kết quả trên cho thấy, đối với hàm băm chúng ta phải thử khoảng $2^{n/2}$ thông điệp khác nhau để tìm ra hai thông điệp mà có cùng giá trị băm (xác suất lớn hơn 50%). Nếu n=128 thì phải thử khoảng 2^{64} thông điệp, một con số khá lớn, nghĩa là hàm băm này đạt được tính chống trùng mạnh. Do đó việc phá hàm băm cũng khó giống như là việc tấn công vét cạn khóa của mã hóa đối xứng DES.
- Tóm lại có thể phát biểu tính chất chống trùng của hàm băm dưới dạng toán học như sau:

  $\forall x, y$ *nếu* $H(x)=H(y)$ *thì* $x = y$

- Hai hàm băm phổ biến: MD5 và SHA-1

Hàm băm MD5 và SHA-1

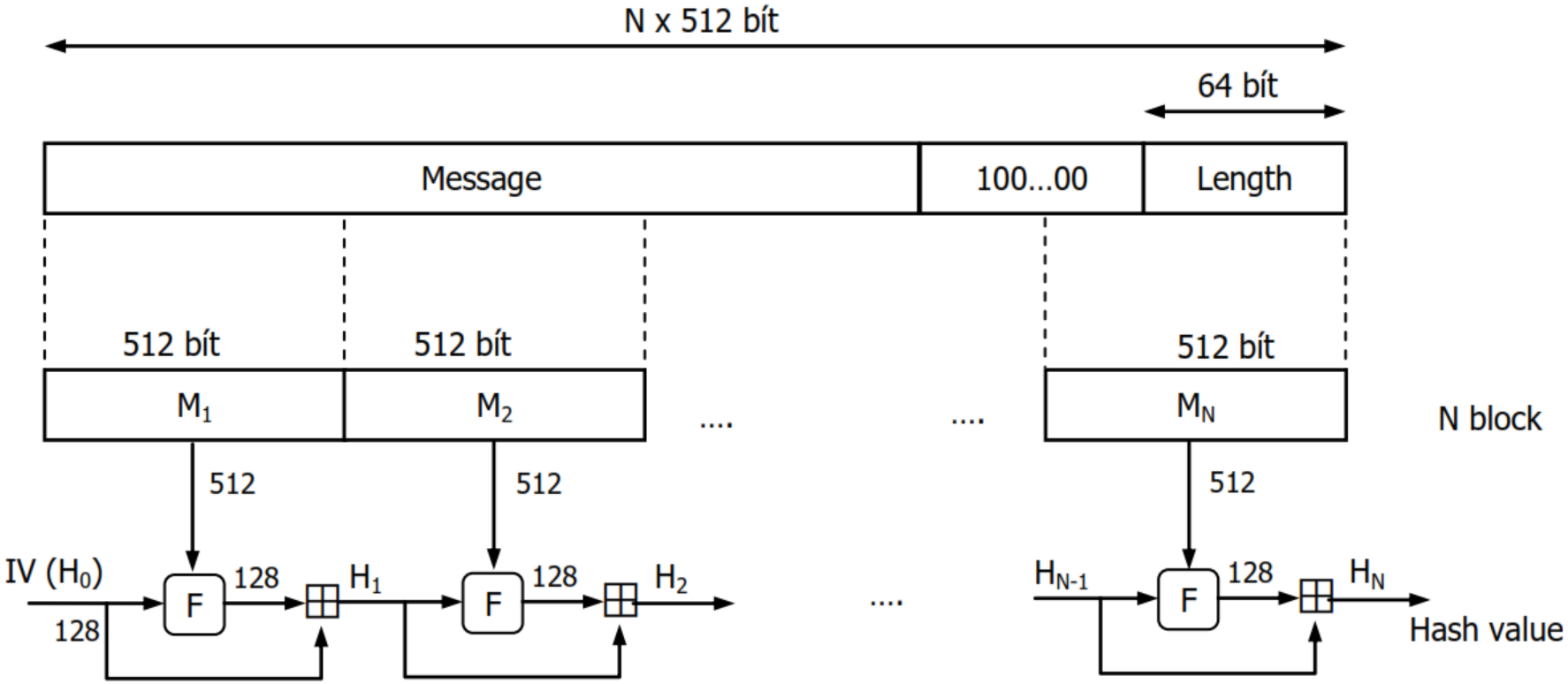
- MD5 được phát minh bởi Ron Rivest, người cũng đã tham gia xây dựng RSA. MD5, viết tắt từ chữ „Message Digest", được phát triển lên từ MD4 và trước đó là MD2, do MD2 và MD4 không còn được xem là an toàn. Kích thước giá trị băm của MD5 là 128 bít, mà chúng ta coi như là an toàn (theo nghĩa không tìm được 2 thông điệp có cùng giá trị băm). Tuy nhiên vào năm 1994 và 1998, một phương pháp tấn công MD5 đã được tìm thấy và một số thông điệp có cùng giá trị băm MD5 được chỉ ra (vi phạm tính chống trùng mạnh). Tuy vậy ngày nay MD5 vẫn còn được sử dụng phổ biến.
- Vì MD5 không còn được xem là an toàn, nên người ta đã xây dựng thuật toán băm khác. Một trong những thuật toán đó là SHA-1 (Secure Hash Algorithm) mà đã được chính phủ Mỹ chọn làm chuẩn quốc gia. SHA-1 có kích thước giá trị băm là 160 bít. Ngày nay còn có ba phiên bản khác của SHA là SHA-256, SHA-384, SHA-512 mà có kích thước giá trị băm tương ứng là 256, 384 và 512 bít.

Hàm băm MD5 và SHA-1

- Tương tự như mã hóa đối xứng, các hàm băm mạnh đều có hiệu ứng lan truyền (*avalanche effect*). Chỉ cần thay đổi 1 bít trong thông điệp đầu vào thì ½ các bít của giá trị băm sẽ thay đổi theo. Điều này làm cho người phá hàm băm không thể thử sai theo kiểu *chosen-plainttext*, nghĩa là không tồn tại cách tấn công nào khác được và buộc phải thử vét cạn 2 thông điệp khác nhau, mà chúng ta đã chứng minh là bất khả thi về mặt thời gian.

Hàm băm MD5

- Kích thước giá trị băm là 128 bít, được dùng để tính giá trị băm của thông điệp có kích thước tối đa là 2^{64} bít



Hàm băm MD5

- Trước tiên thông điệp được thêm dãy bit padding 100….00. Sau đó thêm vào chiều dài (trước khi padding) của thông điệp được biểu diễn bằng 64 bít. Như vậy chiều dài của dãy bít padding được chọn sao cho cuối cùng thông điệp có thể chia thành N block 512 bít M_1, M_2, … , M_N.
- Quá trình tính giá trị băm của thông điệp là quá trình lũy tiến. Trước tiên block M_1 kết hợp với giá trị khởi tạo H_0 thông qua hàm F để tính giá trị hash H_1. Sau đó block M_2 được kết hợp với H_1 để cho ra giá trị hash là H_2 . Block M_3 kết hợp với H_2 cho ra giá trị H_3. Cứ như vậy cho đến block M_N thì ta có giá trị băm của toàn bộ thông điệp là H_N.

Hàm băm MD5

- H_0 là một dãy 128 bít được chia thành 4 từ 32 bít, ký hiệu 4 từ 32 bít trên là abcd. a, b, c, d là các hằng số như sau (viết dưới dạng thập lục phân):

  a = 01234567

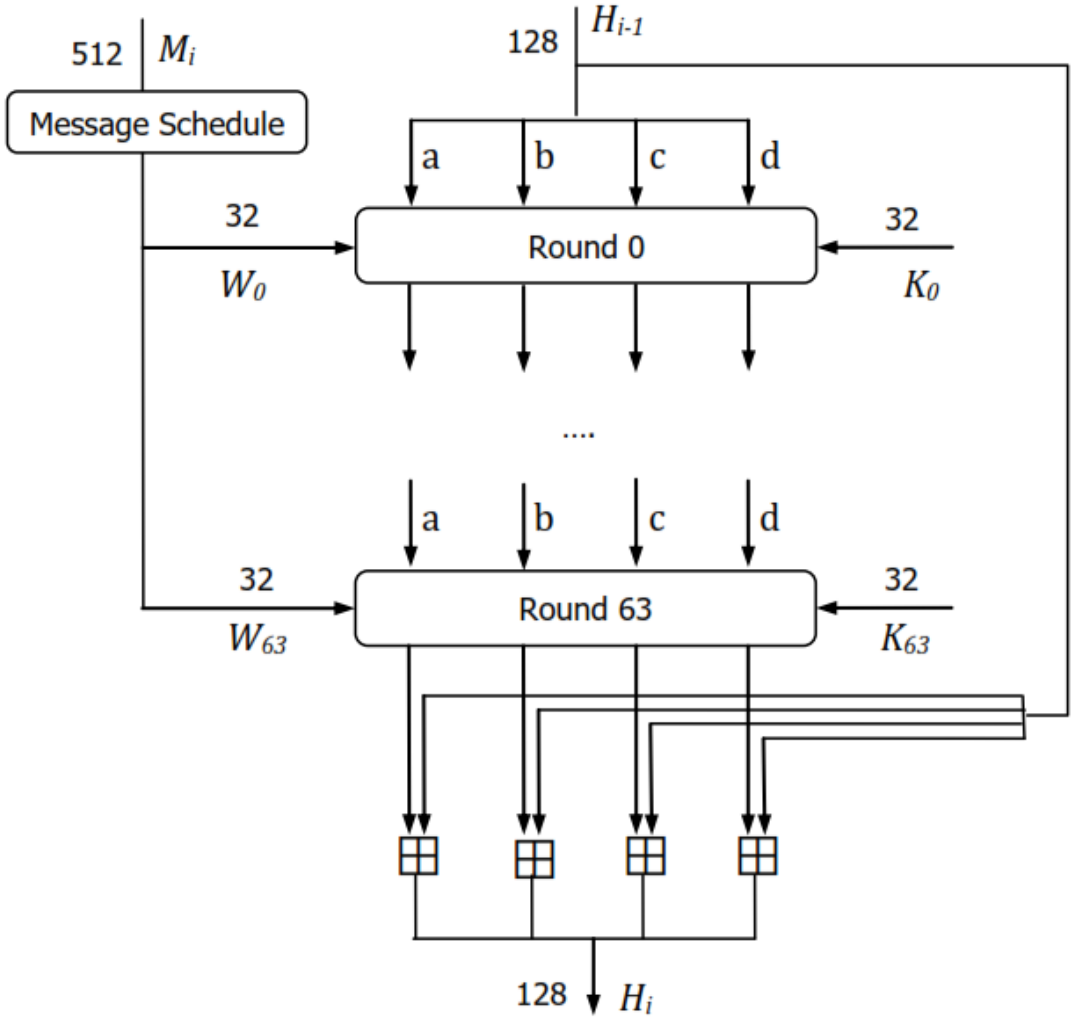
  b = 89abcdef

  c = fedbca98

  d = 76543210

- Tiếp theo ta sẽ tìm hiểu cấu trúc của hàm F.

Cấu trúc của hàm F



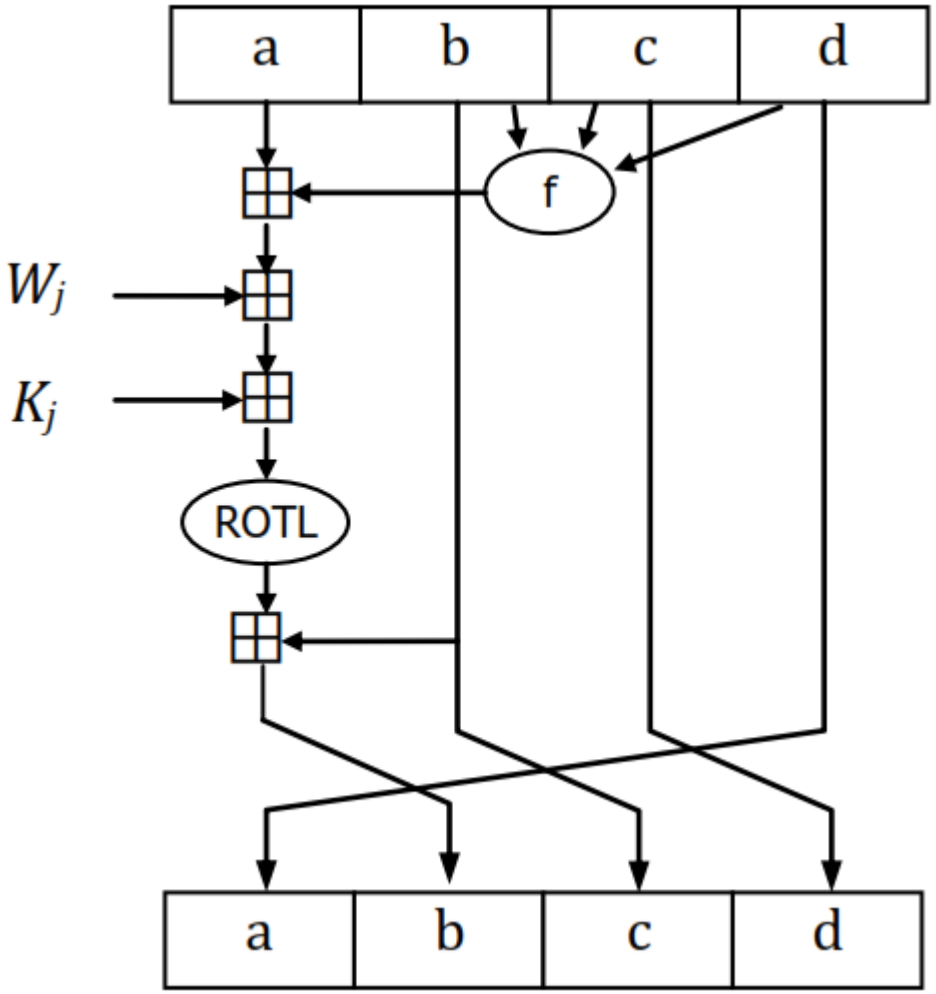
Cấu trúc của hàm F

- Tại mỗi bước lũy tiến, các giá trị abcd của giá trị hash H *i-1* được biến đổi qua 64 vòng từ 0 đến 63. Tại vòng thứ *j* sẽ có 2 tham số là K_j và W_j đều có kích thước 32 bít. Các hằng số K_j được tính từ công thức: K_j là phần nguyên của số $2^{32}abs(sin(i))$ với *i* biểu diễn theo radian.
- Giá trị block M_i 512 bít được biến đổi qua một hàm message schedule cho ra 64 giá trị W_0, W_1,..., W_{63} mỗi giá trị 32 bít. Block M_i 512 bít được chia thành 16 block 32 bít ứngvới các giá trị W_0, W_1, ..., W_{15} (16×32=512). Tiếp theo, 16 giá trị này được lặp lại 4 lần tạo thành dãy 64 giá trị.

Cấu trúc của hàm F

- Sau vòng cuối cùng, các giá trị abcd được cộng với các giá trị abcd của H_{i-1} để cho ra các giá trị abcd của H_i. Phép cộng ở đây là phép cộng modulo 2^{32}.

Cấu trúc 1 vòng



Cấu trúc 1 vòng

Ở đây $b \rightarrow c$, $c \rightarrow d$, $d \rightarrow a$. Giá trị b được tính qua hàm:

$$t = a + f(b, c, d) + W_i + K_i$$

$$b = b + ROTL(t, \mathrm{s})$$

Trong đó:

- Hàm $f(x, y, z)$:

$f(x, y, z) = (x \wedge y) \vee (\neg x \wedge z)$	nếu là vòng 0 đến 15
$f(x, y, z) = (z \wedge x) \vee (\neg \mathrm{z} \wedge y)$	nếu là vòng 16 đến 31
$f(x, y, z) = x \oplus y \oplus z$	nếu là vòng 32 đến 48
$f(x, y, z) = y \oplus (x \vee \neg z)$	nếu là vòng 49 đến 63

- Phép + (hay $\boxplus$) là phép cộng modulo 2^{32}

Cấu trúc 1 vòng

- Hàm *ROTL*(t, *s*): t được dịch vòng trái s bít, với s là các hằng số cho vòng thứ *i* như sau:

i	*s*
0, 4, 8, 12	7
1, 5, 9, 13	12
2, 6,10, 14	17
3, 7, 11, 15	22
16, 20, 24, 28	5
17, 21, 25, 29	9
18, 22, 26, 30	14
19, 23, 27, 31	20
32, 36, 40, 44	4
33, 37, 41, 45	11
34, 38, 42, 46	16
35, 39, 43, 47	23
48, 52, 56, 60	6
49, 53, 57, 61	10
50, 54, 58, 62	15
51, 55, 59, 63	21

Hàm băm SHA-1

- Hàm băm SHA-1 với giá trị băm có kích thước là 160 bít, được dùng để tính giá trị băm của thông điệp có kích thước tối đa là 264 bít.
- Sơ đồ tổng thể của SHA1 cũng tương tự như của MD5, chỉ có điểm khác là kích thước của giá trị hash tại mỗi bước là 160 bít.
  - 160 bít H_0 được chia thành **5 từ 32 bít**
  - Cấu trúc của hàm F: **80 vòng**

HMAC

- Hàm băm cũng có thể dùng để tính MAC bằng cách truyền thêm khóa bí mật K vào hàm băm. Lúc này, giá trị kết xuất được gọi là HMAC.

$$HMAC = H(M||K)$$

Một số ứng dụng của hàm băm

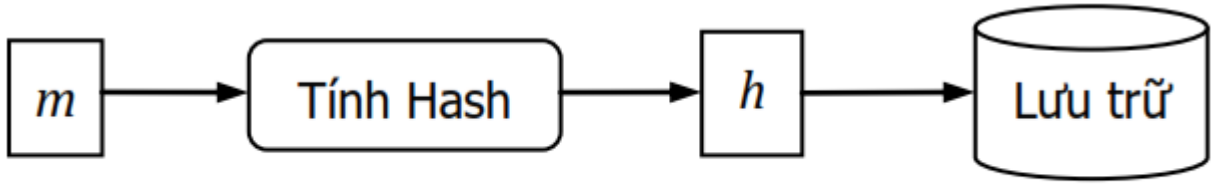
1/ Lưu trữ mật khẩu

- **Tình huống:** Lưu mật khẩu vào CSDL mà không mã hóa

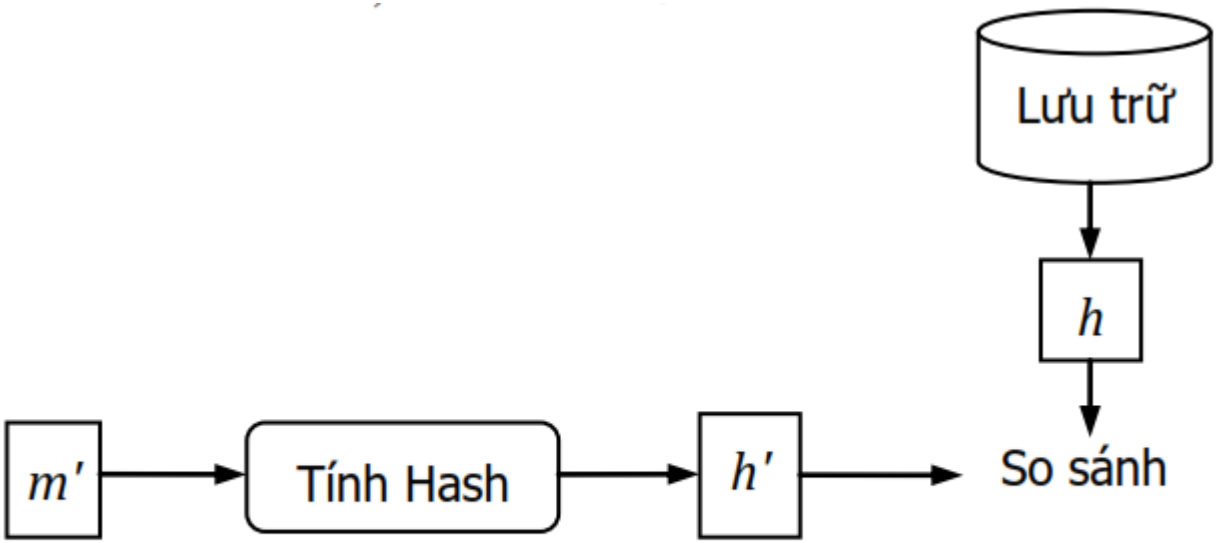
→ Sử dụng hàm băm: giá trị băm của mật khẩu được tính bằng một hàm băm nào đó (MD5 hay SHA-1,…)

  - ✓ Do là hàm 1 chiều -> hacker ko suy lại mật khẩu
  - ✓ Do tính chống trùng -> mật khẩu là duy nhất có giá trị băm tương ứng.

1/ Lưu trữ mật khẩu



a) Lưu trữ mật khẩu



b) Chứng thực mật khẩu, theo tính chống trùng, nếu h'=h thì m'=m

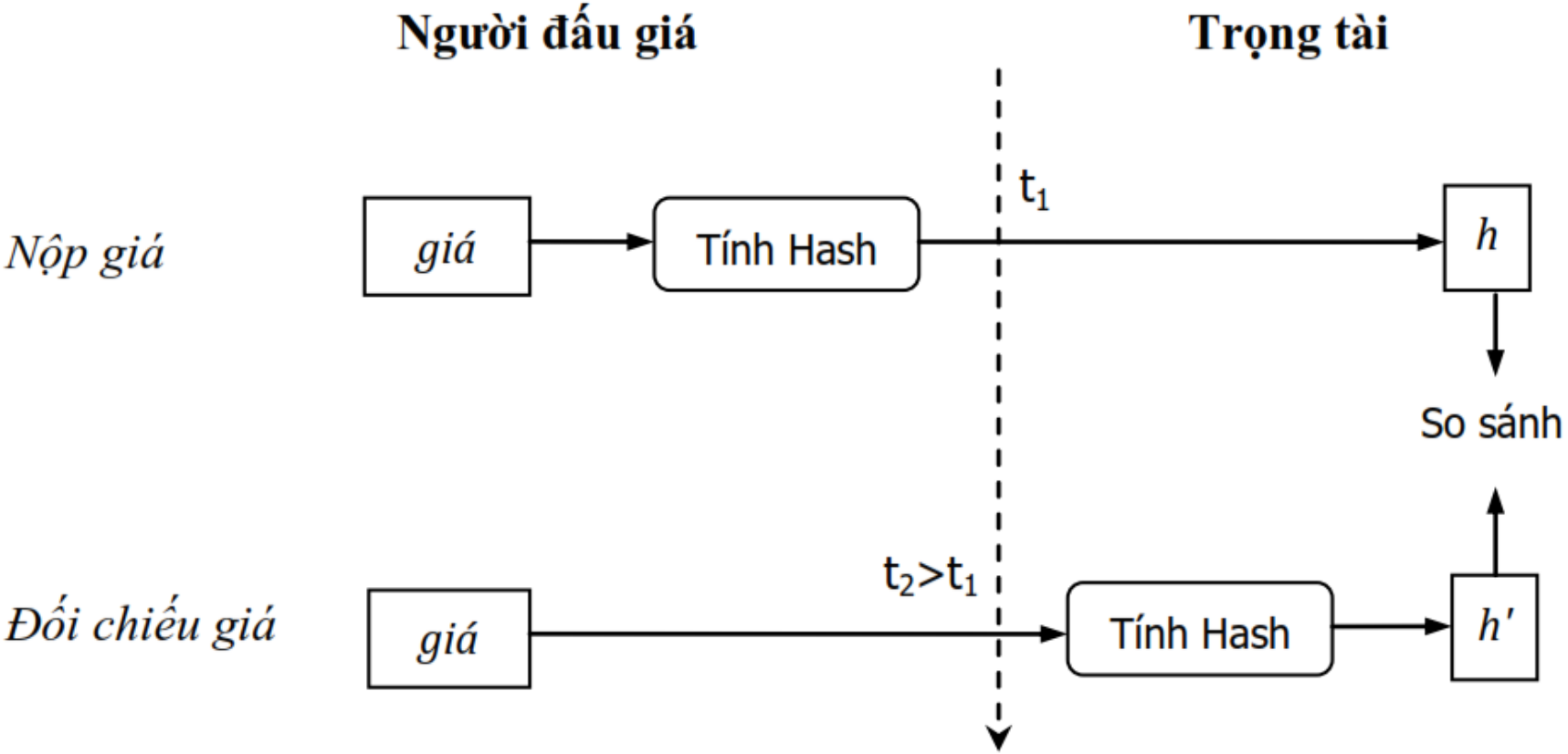
2/ Đấu giá trực tuyến

- **Tình huống:** Đấu giá trực tuyến bằng hình thức đấu giá bít mật. Giá có thể bị lộ bởi trọng tài.

→ *Sử dụng hàm băm: các giá trị băm tương ứng được tính cho mức giá bỏ thầu*

- ✓ *Do là một chiều → tránh lộ giá bởi trọng tài.*
- ✓ *Vì tính chống trùng → giá là duy nhất với giá trị băm tương ứng*

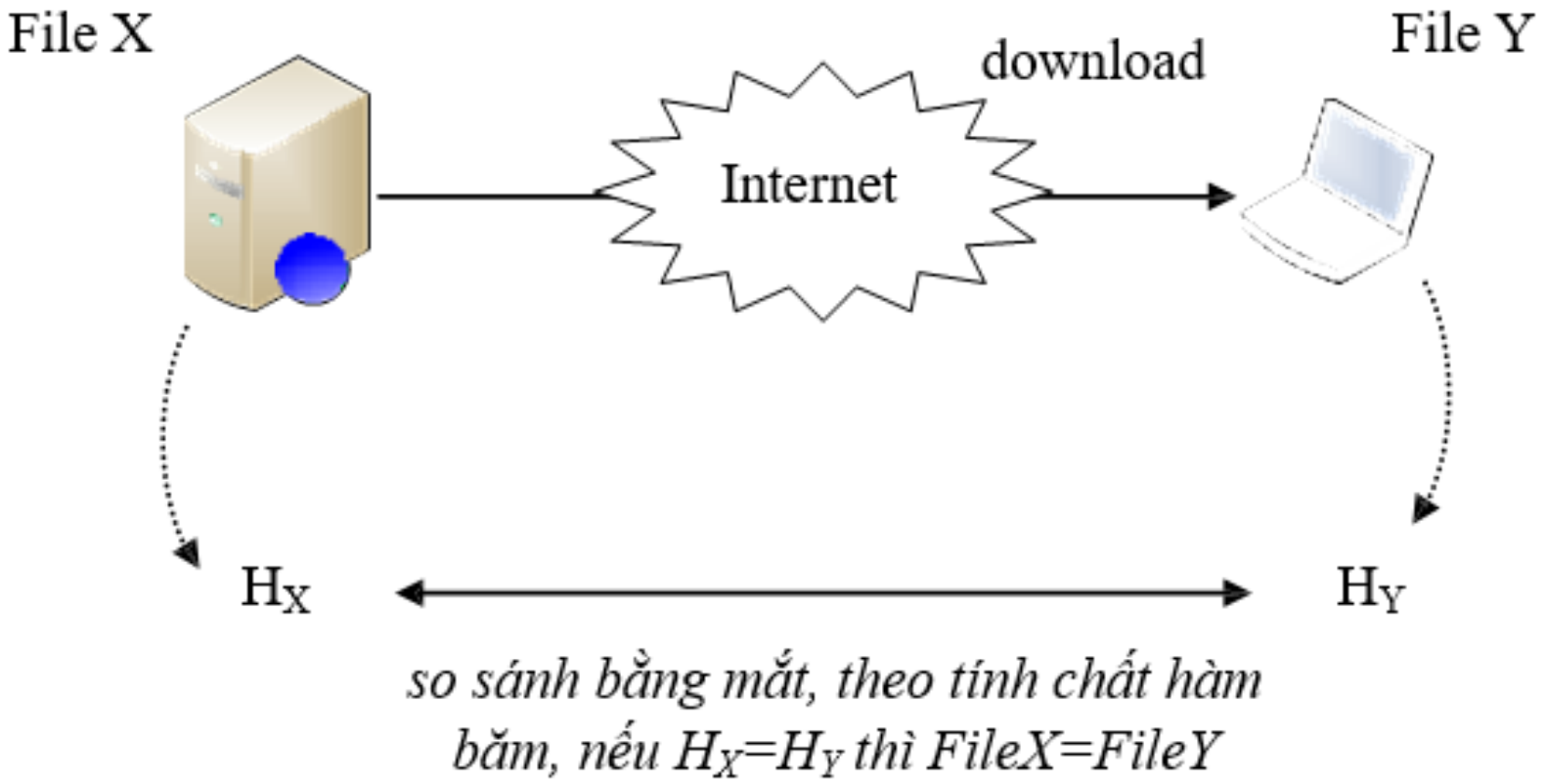
2/ Đấu giá trực tuyến



3/ Tải tệp

- Khi chúng ta download file từ mạng internet, nếu chất lượng mạng không tốt thì có thể xảy ra lỗi trong quá trình download làm cho file tại máy client khác với file trên server.
- Hàm băm có thể giúp chúng ta phát hiện ra những trường hợp bị lỗi như vậy.

3/ Tải tệp

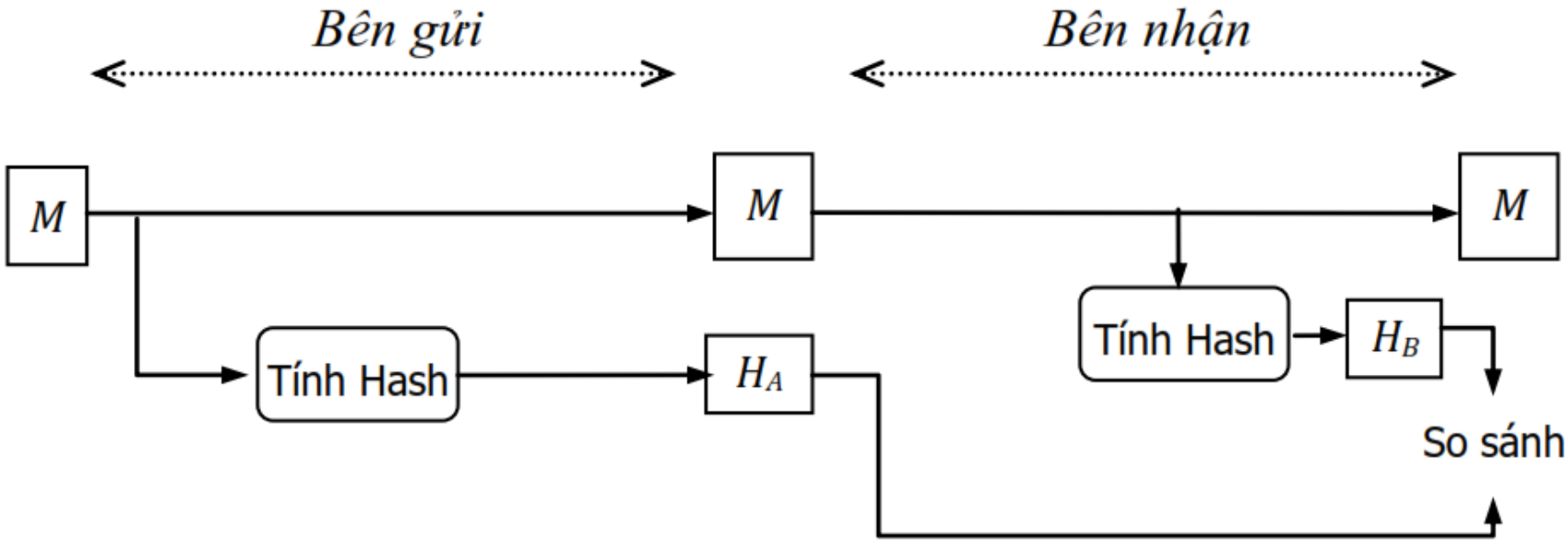

File X
download
File Y
Internet
H_X
H_Y
so sánh bằng mắt, theo tính chất hàm băm, nếu H_X=H_Y thì FileX=FileY

Hàm băm và chữ ký điện tử

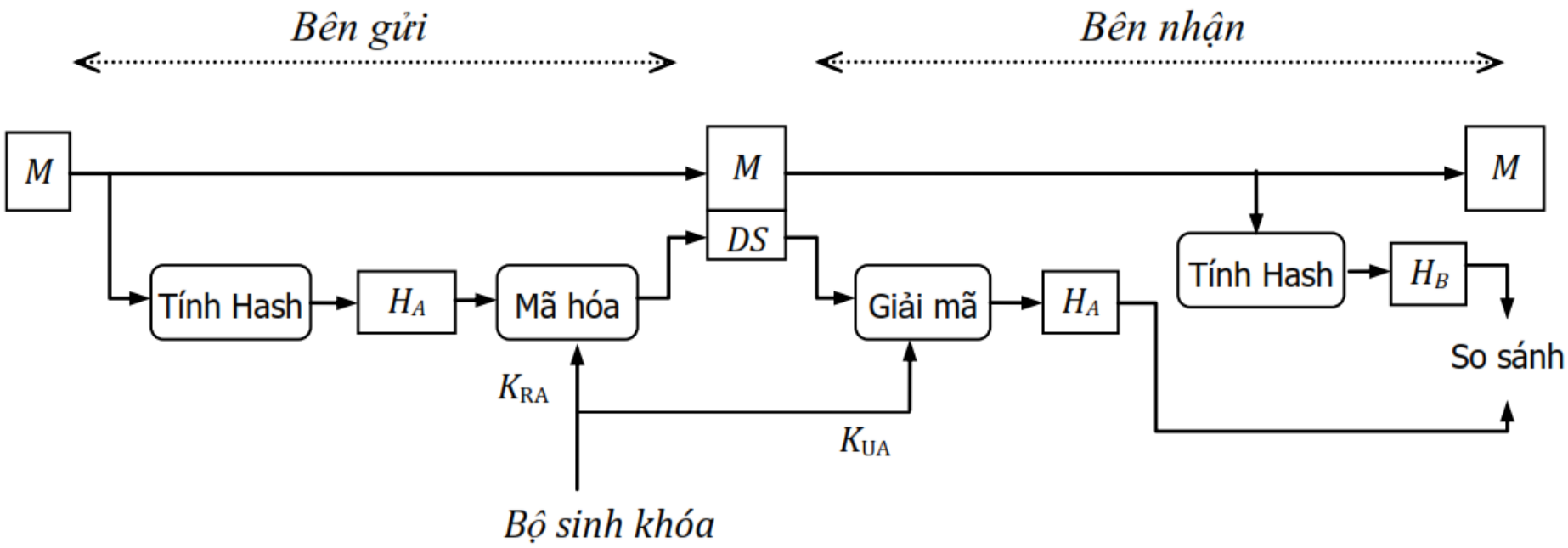
Hàm băm và chữ ký điện tử

- Việc sử dụng khóa bí mật chung cho người gửi và người nhận trong mã chứng thực thông điệp MAC sẽ gặp phải vấn đề tính không từ chối tương tự như mã hóa đối xứng. Dùng hàm băm và mã hóa khóa công khai khắc phục được vấn đề này.

Mô hình đơn giản sử dụng hàm băm



Mô hình chữ ký điện tử



DS: Data signature – chữ ký điện tử

✓ *Sử dụng mã hóa khóa công khai để chứng thực H_A*