TRƯỜNG ĐẠI HỌC NGUYỄN TẤT THÀNH
KHOA CÔNG NGHỆ THÔNG TIN

Bài giảng môn học: AN TOÀN THÔNG TIN

Chương 6:

Xác thực và toàn vẹn thông tin

Số tín chỉ: 2
Số tiết: 30 tiết
(Lý Thuyết)

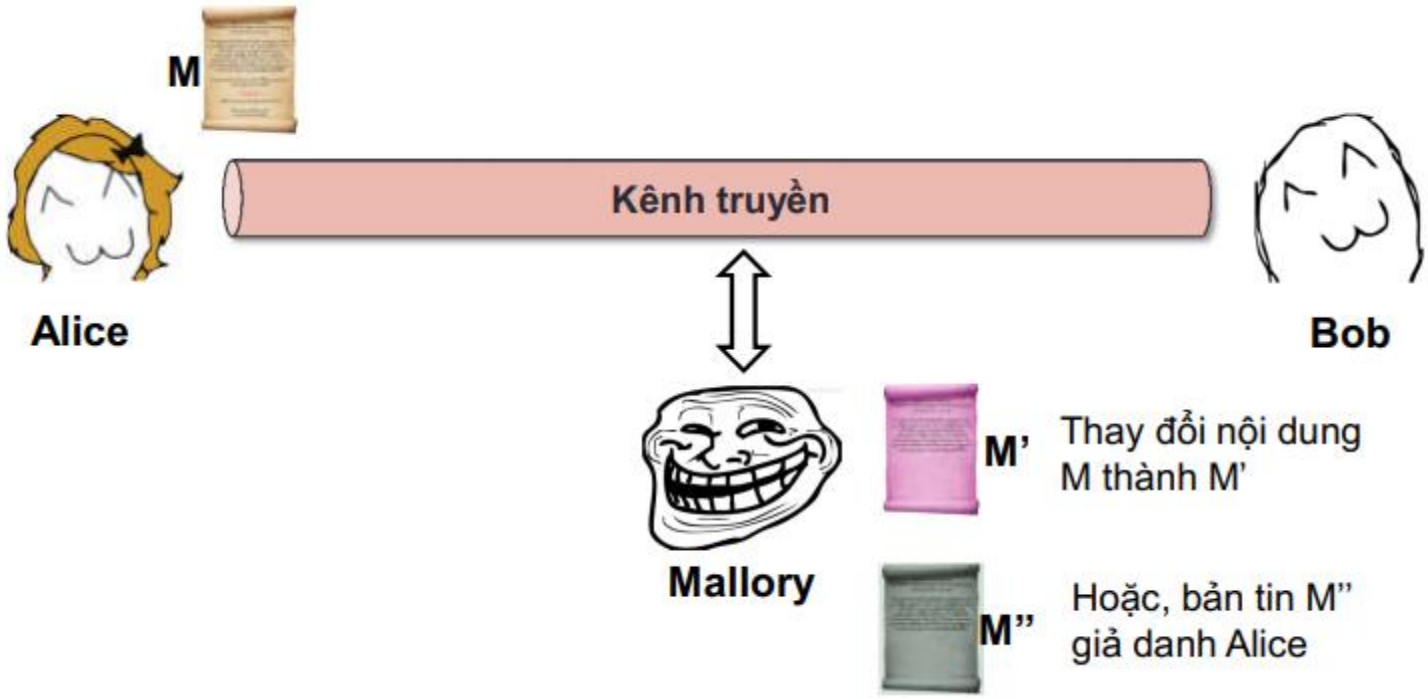
GV: ThS. **Nguyễn Thị Phong Dung**
Email : ntpdung@ntt.edu.vn

Bài 6: Xác thực và toàn vẹn thông tin

- Nhu cầu xác thực thông điệp
- Mã xác thực thông điệp (MAC)
- Xác thực thông điệp dùng RSA
- Hàm băm (hash function)
- Chữ ký số (Digital Signature)
- Chứng thư số (Digital Certificate)
- Hạ tầng khóa công khai (PKI)

Nhu cầu xác thực thông điệp

- **Đặt vấn đề:**
  - *Case 1*: Nếu *Mallory* đánh cắp thông điệp **M** của *Alice*. Sau đó sửa lại thành **M'** rồi gởi cho *Bob*.
  - *Case 2*: *Mallory* tự tạo thông điệp **M'** giả mạo là của *Alice* rồi gởi cho *Bob*

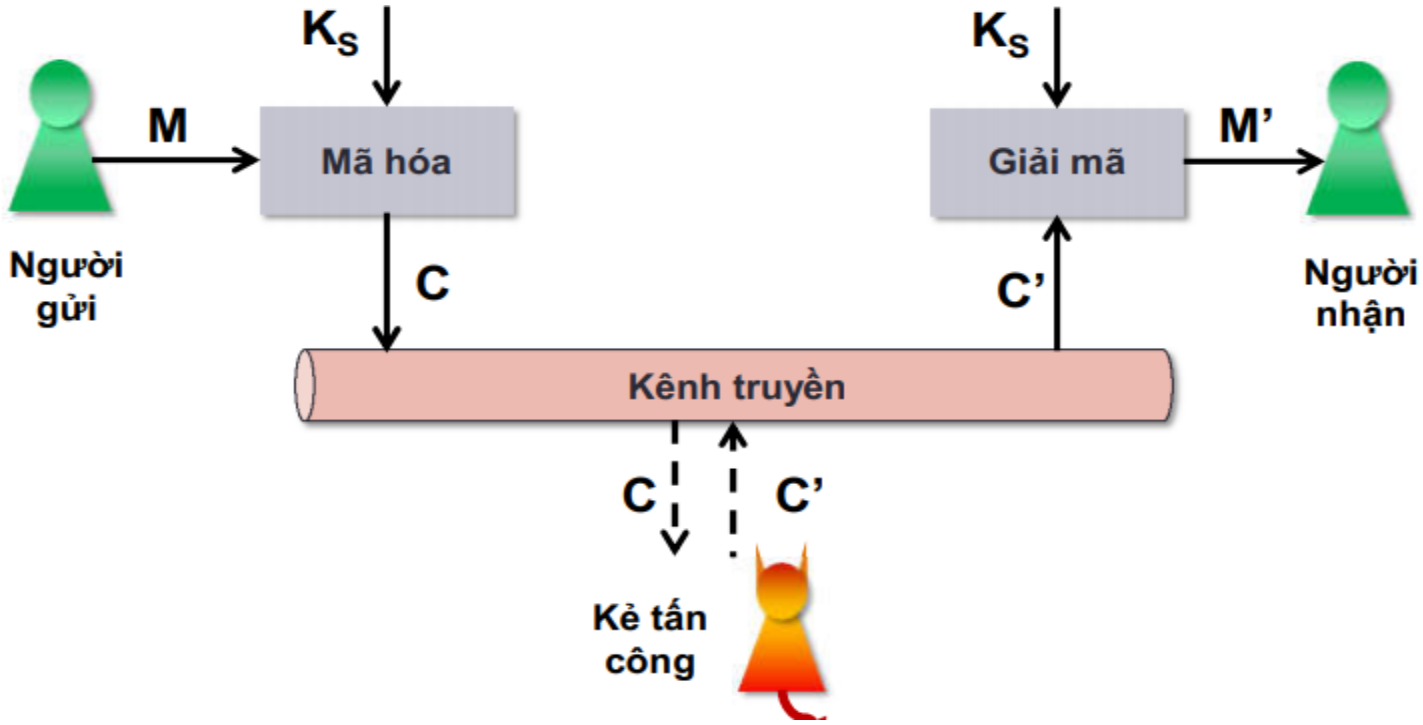


Nhu cầu xác thực thông điệp

- **Nhu cầu xác thực thông điệp:**
  - Kiểm chứng danh tính nguồn phát thông điệp (*Authentication*)
    - Bao hàm cả trường hợp *Alice* phủ nhận bản tin (*Non-repudiation*)
    - Bao hàm cả trường hợp *Bob* tự tạo thông điệp và "vu khống" Alice tạo ra thông điệp này.
  - Kiểm chứng được tính toàn vẹn của thông điệp (*Integrity*) :
    - Nội dung toàn vẹn: bản tin không bị sửa đổi.
    - Bao hàm cả trường hợp *Bob* cố tình sửa đổi.
  - Xác thực thông điệp hỗ trợ phòng chống các dạng tấn công như:
    - Tấn công thay thế (*Substitution*).
    - Tấn công giả danh (*Masquerade*).
    - Tấn công tấn công phát lại (*Reply attack*).

Mật mã khóa đối xứng

- **Có thể dùng mật mã *khóa đối xứng* để xác thực ?**
  - Nguyên lý của mật mã khóa đối xứng:



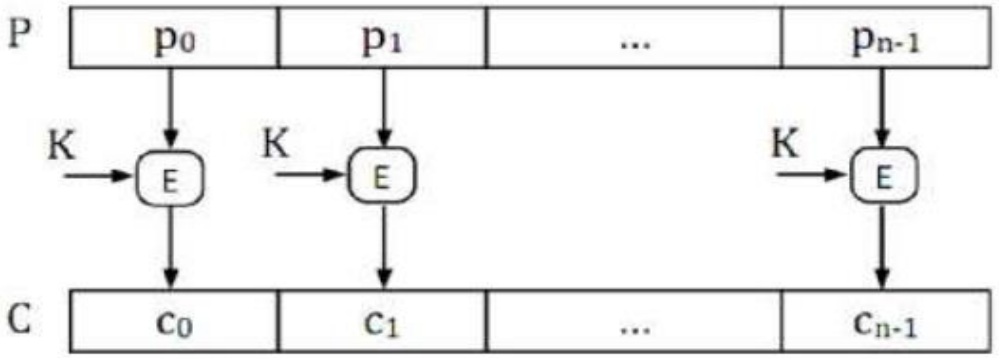
  - ? *Người nhận* có biết **C'** là thông điệp bị thay đổi?
  - ? Mã hóa khóa đối xứng có cung cấp tính năng xác thực?

Mật mã khóa đối xứng

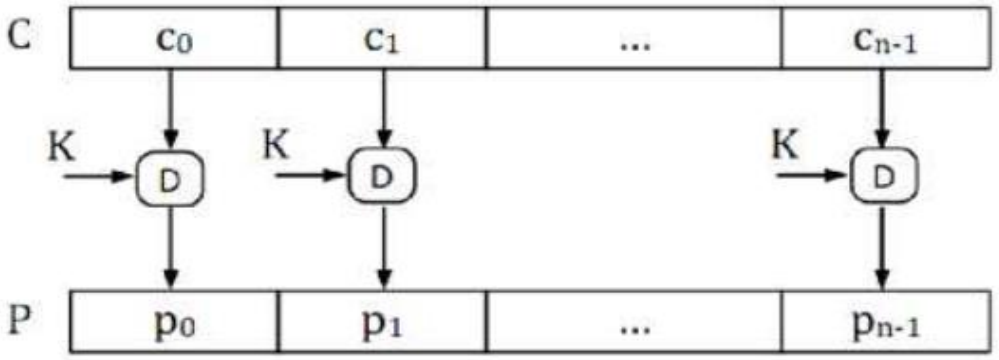
- **Mô hình xử lý mã hóa cho thông điệp:**
  - **Mã hóa dòng** (*Stream Cipher*): từng bit (hay byte) dữ liệu đầu vào được xử lý mã hóa cho đến khi kết thúc luồng dữ liệu.
  - **Mã hóa khối** (*Block Cipher*): dữ liệu được chia thành từng khối, kích thước bằng nhau để mã hóa (và giải mã).
  - Có 2 mô hình xử lý thông tin của mã hóa khối:
    - Mô hình *Electronic Code Book* (**ECB**):
    - Mô hình *Cipher Block Chaining* (**CBC**):

Mật mã khóa đối xứng

- **Mô hình *Electronic Code Book* (ECB):**
  - Thông tin cần mã hóa được chia thành nhiều *khối* ($P_0 \rightarrow P_{n-1}$)
  - Dùng khóa **K** để mã hóa từng *khối* thành phần .
  - Giải mã: dùng khóa **K** giải mã từng khối ($C_0 \rightarrow C_{n-1}$)
  - Nhận xét:
    - Dễ đoán: nếu $C_i = C_j$ thì $P_i = P_j$. => có thể dựa vào các phương pháp thống kê để phá mã.
    - Chỉ thích hợp cho những thông tin ngắn



a) Quá trình mã hóa



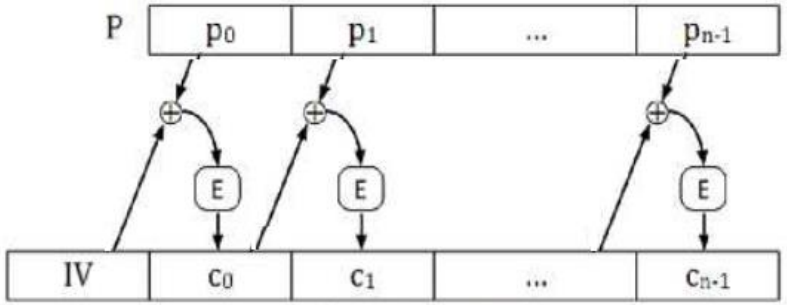
b) Quá trình giải mã

Mật mã khóa đối xứng

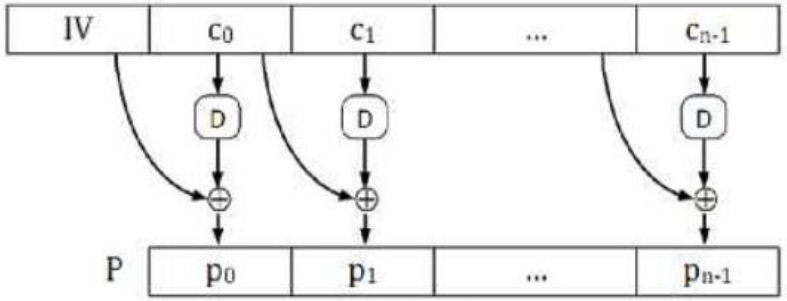
- **Mô hình *Cipher Block Chaining* (CBC):**
  - Khởi tạo khối dữ liệu giả (*Initial Vector* – **IV**)
  - Khối đầu tiên **XOR** với **IV** rồi mã hóa với khóa **K**:

    $C_0 = E(P_0 \oplus IV, K)$
  - Khối tiếp theo **XOR** với *Bản mã của khối trước* rồi mã hóa:

    $C_i = E(P_i \oplus C_{i-1}, K)$
  - Giải mã: dùng khóa **K** và **IV** giải mã khối C_0.
    - từng khối ($C_0 \rightarrow C_{n-1}$)
  - Nhận xét:
    - Nếu $P_i = P_j$ sau khi mã hóa, $C_i \neq C_j$.
    - Giải mã: phải có khóa **K** và khối dữ liệu khởi tạo (*Initial Vector* – **IV**)



a) Quá trình mã hóa



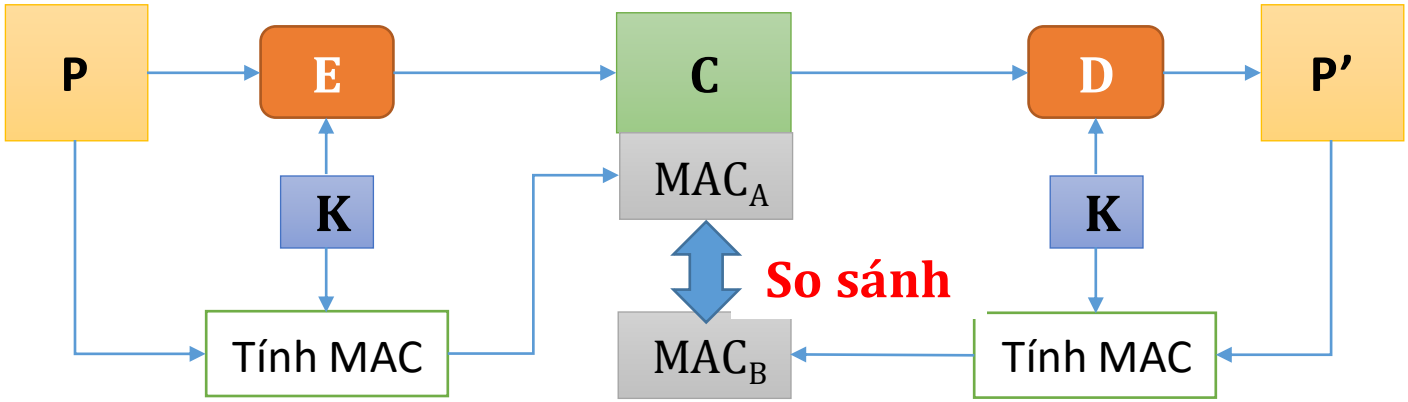
b) Quá trình giải mã

Mã xác thực thông điệp (MAC)

- **Tổng quan về Mã xác thực thông điệp.**
  - Thuật ngữ "***thông điệp***":
    - *Message* = *Plain text* hoặc *Cipher text* (thông điệp = thông tin).
  - Mã xác thực thông điệp (*Message Authentication Code* - **MAC**)
    - Là giải pháp xác thực tính toàn vẹn và nguồn gốc của thông điệp..
    - Bên nhận phát hiện sự thay đổi của thông điệp trên đường truyền.
    - Duy trì tính toàn vẹn và không thể chối bỏ dữ liệu.
  - **MAC** xây dựng dựa trên ***mật mã khóa đối xứng*** (*Pre-share key*)
  - **NIST** (Viện tiêu chuẩn và công nghệ quốc gia của Mỹ):
    - Chuẩn **C-MAC** (*Cipher Message Authentication Code*): sử dụng mã hóa cho *đoạn code dùng xác thực thông điệp*.
    - Chuẩn **H-MAC** (*Keyed-Hash Messasge Authentication Code*) : sử dụng khóa hàm băm cho *đoạn code dùng xác thực thông điệp*.

Mã xác thực thông điệp (MAC)

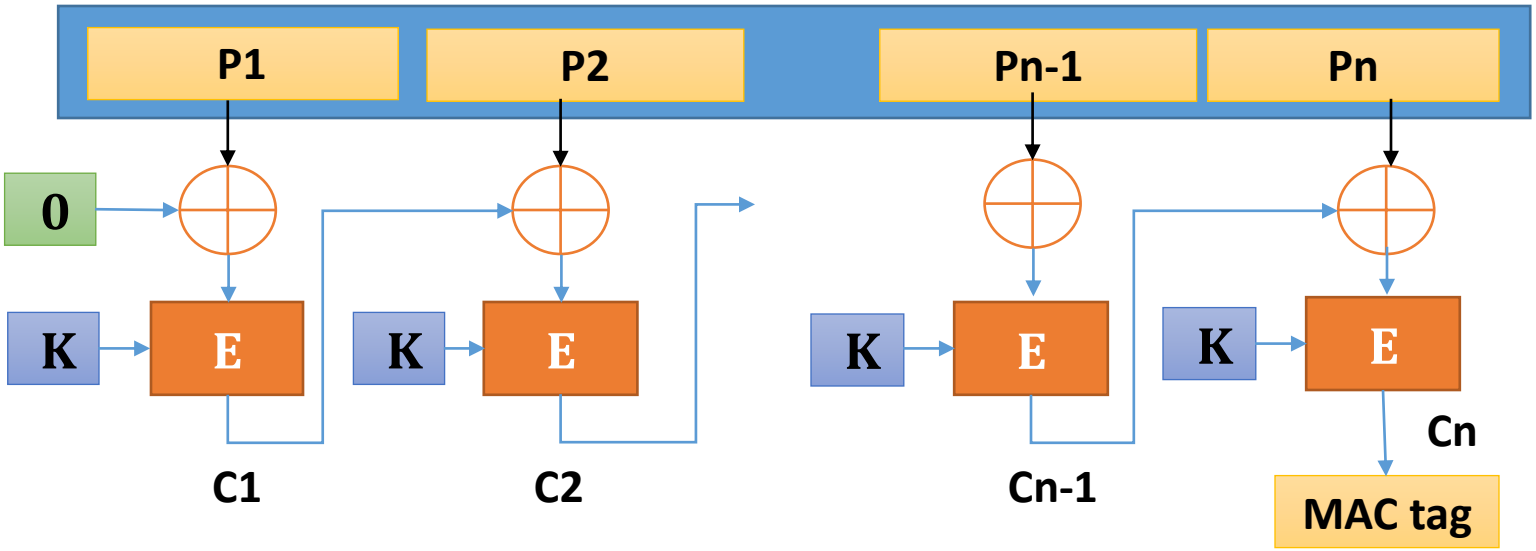
- **Giải thuật chính của MAC:**
  - Bên gởi thực hiện:
    - Tính giá trị $\mathbf{MAC_A}$ dựa trên thông tin **P** và khóa **K**.
    - Gắn (*tag*) $\mathbf{MAC_A}$ vào bản mã **C** trước khi truyền đi.
  - Bên nhận:
    - Giải mã **C** thành **P'**
    - Tính giá trị $\mathbf{MAC_B}$ dựa trên **P'** và khóa **K**.
    - So sánh $\mathbf{MAC_B}$ với $\mathbf{MAC_A}$
    - Nếu $\mathbf{MAC_B = MAC_A}$ => **xác thực.**



Mã xác thực thông điệp (MAC)

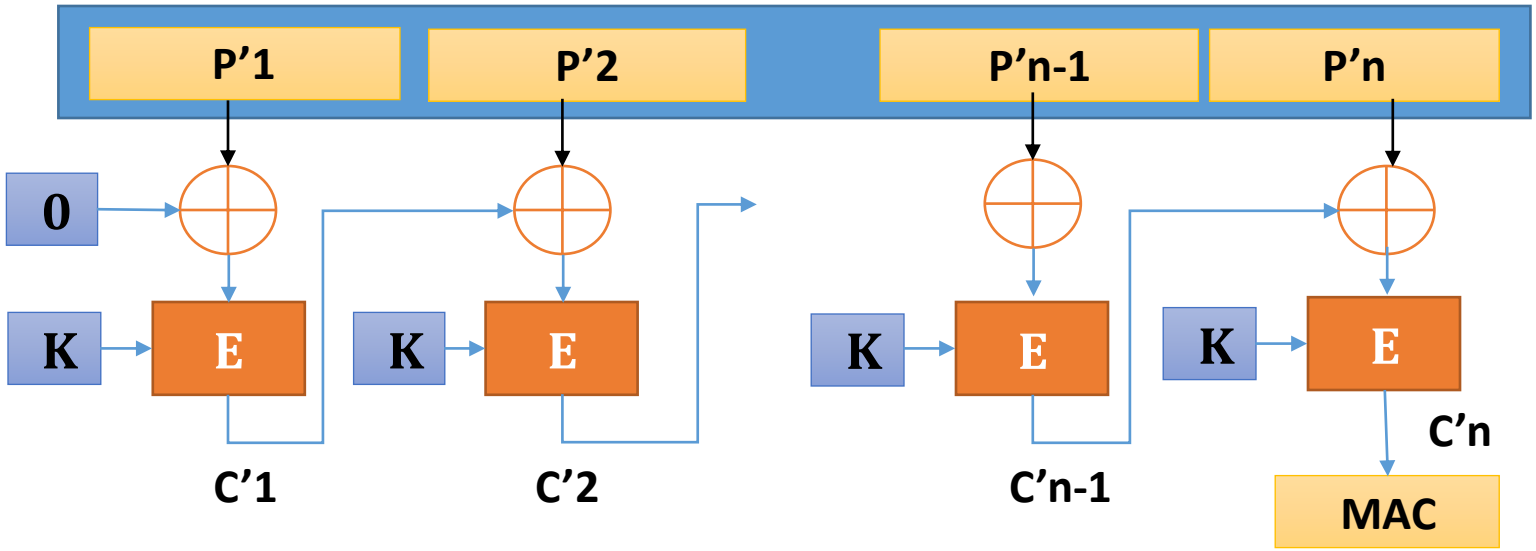
- **CBC-MAC:**
  - ***CBC-MAC*** áp dụng mô hình mã khối **CBC** vào xác thực thông điệp.
  - Bên gởi: tính **MAC** và gán (*tagged*) vào thông điệp gởi:
    - Thông điệp **P** được chia thành **n** khối ($P_1 \rightarrow P_n$)
    - Mã hóa **P** bằng theo mô hình **CBC**, sử dụng *vector* khởi tạo **IV = 0**
    - Lấy khối mã cuối cùng (C_n) làm ***MAC tag*** (gắn vào **C** để gởi đi).

MAC tag: $\mathbf{t = C_n}$



Mã xác thực thông điệp (MAC)

- **CBC-MAC:**
  - Xác thực thông điệp nhận:
    - Giải mã **C** thành **P'** => cần xác thực **P' = P**.
    - Tính ***MAC*** cho **P'** theo mô hình **CBC** với **IV = 0**
    - Kết quả: *MAC tag*: $\mathbf{t' = C'_n}$
    - So sánh *MAC* ***t'*** tính được với ***t*** kèm theo bản mã **C**. Nếu trùng khớp => xác thực.



Mã xác thực thông điệp (MAC)

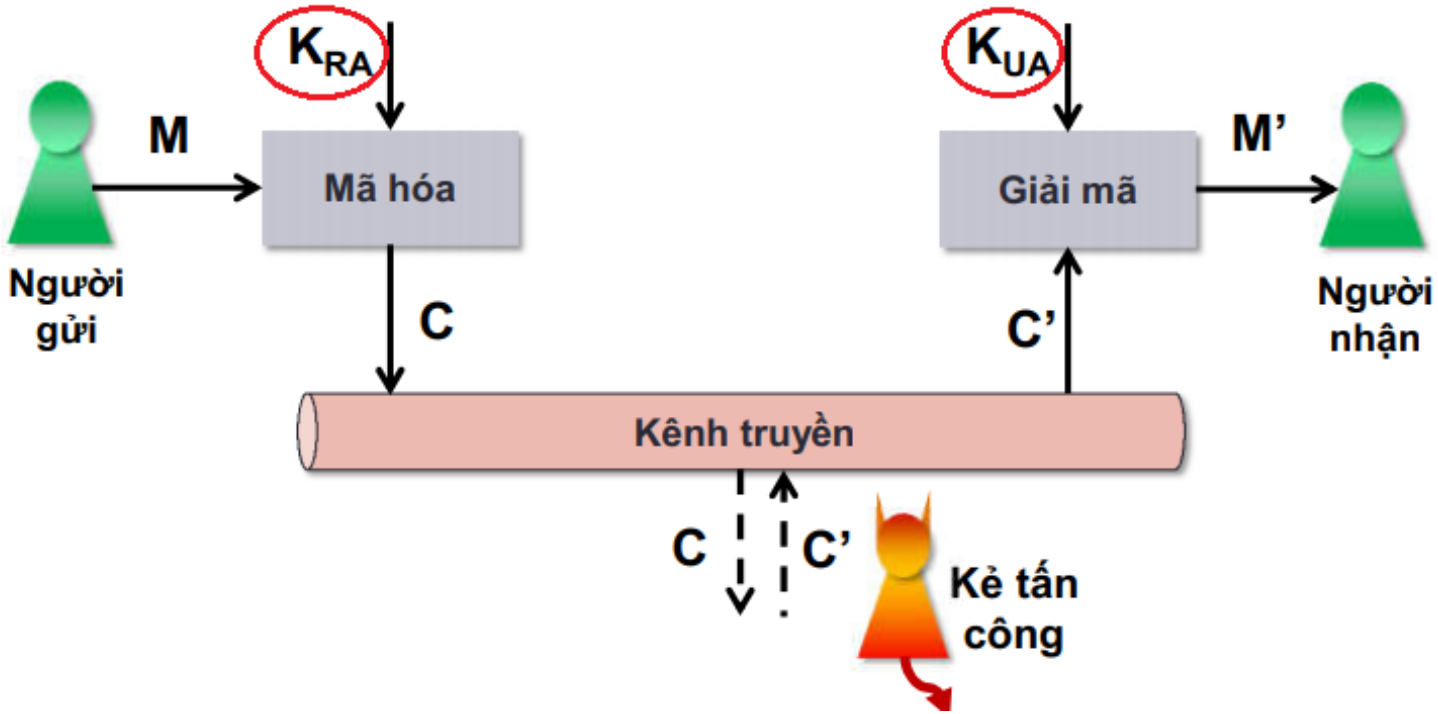
- **Chứng minh tính đúng đắn của CBC-MAC:**
  - *MAC tag* của thông điệp bên gởi: $t = C_n$
  - *MAC tag* của thông điệp bên nhận: $t' = C'_n$
  - $t' = C'_n = E_K(P'_n \oplus C'_{n-1})$ (*xem cách tính MAC bên nhận*)
    - Thay $P'_n = D_K(C_n) \oplus C_{n-1}$ (xem cách giải mã CBC)
  - Được $t' = C'_n = E_K(D_K(C_n) \oplus C_{n-1} \oplus C'_{n-1})$
    - Do tính chất của **XOR**: $A \oplus A = 0$ và $B \oplus 0 = B$
  - Nên: $t' = C'_n = E_K(D_K(C_n) = C_n = t$ $(C_{n-1} \oplus C'_{n-1} = 0)$
  - Kết luận:
    - Nếu $P' = P$ thì $C'_n = C_n \Leftrightarrow t' = t$ => xác thực tính toàn vẹn
    - Nếu $P' \neq P$ thì $C'_n \neq C_n \Leftrightarrow t' \neq t$ => thông điệp nhận không toàn vẹn

Mã xác thực thông điệp (MAC)

- **Nâng cao an toàn của MAC:**
  - Chuẩn **C-MAC** (*Cipher Message Authentication Code*):
    - *Bên gởi*: tính ***MAC***: **t = C_n**
      - ➢ Mã hóa **t** bằng khóa **K2** -> *tagged* vào *thông* điệp gởi (***MAC-tag***).
    - *Bên nhận:* tính ***MAC***: **t' = C'$_n$**
      - ➢ Giải mã ***MAC-tag*** của thông điệp, bên gởi bằng khóa **K2**.
      - ➢ So sánh kết quả giải mã với **t'**. Nếu trùng => xác thực.
  - Chuẩn **H-MAC** (*Keyed-Hash Messages Authentication Code*) :
    - *Bên gởi*: tính ***MAC_t*** **= C_n**
      - ➢ Thực hiện **băm** ***MAC_t*** => **h(t)** -> *tagged* vào *thông* điệp gởi.
    - *Bên nhận*: tính ***MAC_t'*** **= C'$_n$**
      - ➢ Thực hiện **băm** ***MAC_t'*** => **h(t')**
      - ➢ So sánh **h(t')** với **h(t)**. Nếu trùng khớp => xác thực

Xác thực thông điệp dùng RSA

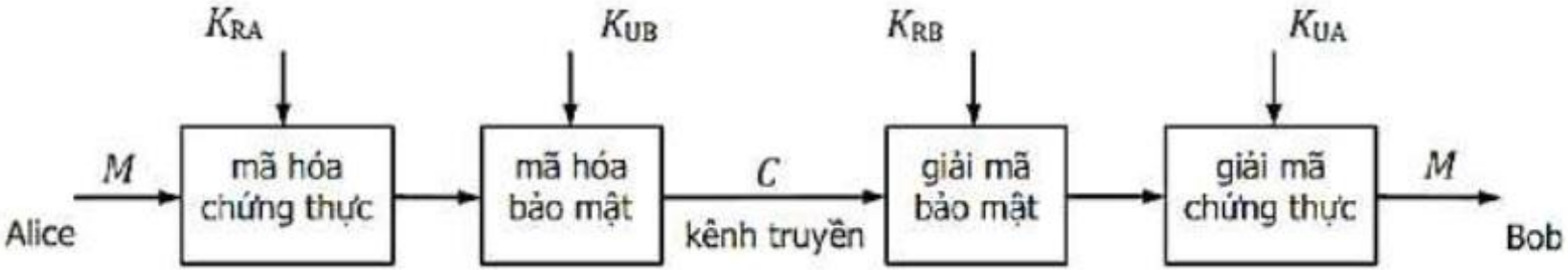
- **Xác thực bằng mật mã *khóa bất đối xứng RSA*:**
  - Hệ mật mã khóa bất đối xứng cung cấp khả năng xác thực thông điệp khi:



  - ? Người nhận có biết **C'** là thông điệp bị thay thế?

Xác thực thông điệp dùng RSA

- **Tính chất của mật mã *khóa bất đối xứng RSA*:**
  - Tính bí mật:
    - Khi mã hóa thông điệp bằng *Public Key*.
    - Giải mã bằng *Private key*.
  - Tính xác thực:
    - Khi mã hóa thông điệp bằng *Private Key*.
    - Giải mã bằng *Public key*.
- **Kết hợp 2 tính chất:**



Hàm băm (hash function)

- **Tổng quan về hàm băm (Hash function):**
  - Vai trò của hàm băm:
    - Nén thông điệp bất kỳ về kích thước cố định.
    - Phát hiện sự thay đổi trên thông điệp.
    - Có thể được sử dụng để tạo chữ ký trên thông điệp.
  - Tính chất của hàm băm:
    - Nếu có ***thông điệp*** (**M**) sẽ dễ dàng tính được ***mã băm*** (**H**) theo một ***hàm băm*** (**h**) nào đó. Ký hiệu: **h = H(M)**.
    - Nếu có *mã băm* (**H**), không thể tính ngược được để cho ra *thông điệp* (**M**) cho dù biết thuật toán băm (**h**).
    - Kích thước mã băm **H** là cố định (tùy theo thuật toán hàm băm).
    - Giá trị của mã băm **H** chỉ phụ thuộc vào **M**
  - Các hàm băm được sử dụng phổ biến:
    - **MD5, SHA-1, SHA-2** (*SHA-256, SHA-384, SHA-512*)

Hàm băm (hash function)

- **Minh họa hàm băm đơn giản:**
  - Dùng thuật toán **XOR** để băm thông điệp
  - Thông điệp (**M**): 1010 1100 0111 1000 1010 1100
  - Băm *8 bit* – chia thông diệp thành các đoạn 8 bit

1010 1100
0111 1000
1010 1100

Mã băm (h): 0111 1000

  - Nếu có thay đổi trên thông điệp

1010 1101 Bit thứ 8 thay đổi 0 => 1
0111 1000
1010 1100

Mã băm mới: 0111 1001 sẽ khác với mã băm cũ

Hàm băm (hash function)

- **Minh họa ứng dụng Hash:**
  - Mật khẩu "**abc**" theo dạng **ASCII**:

    0110 0001 0110 0010 0110 0011
  - Băm *8 bit* theo thuật toán **XOR** ta được mã băm: **0110 0000**
  - Hệ thống lưu trữ mật khẩu "**abc**" dưới dạng mã băm: **0110 0000**
  - Trường hợp *Attacker* lấy được bảng mật khẩu dạng mã băm

    => không thể giải mã băm **0110 0000** thành mật khẩu "**abc**" .

Hàm băm (hash function)

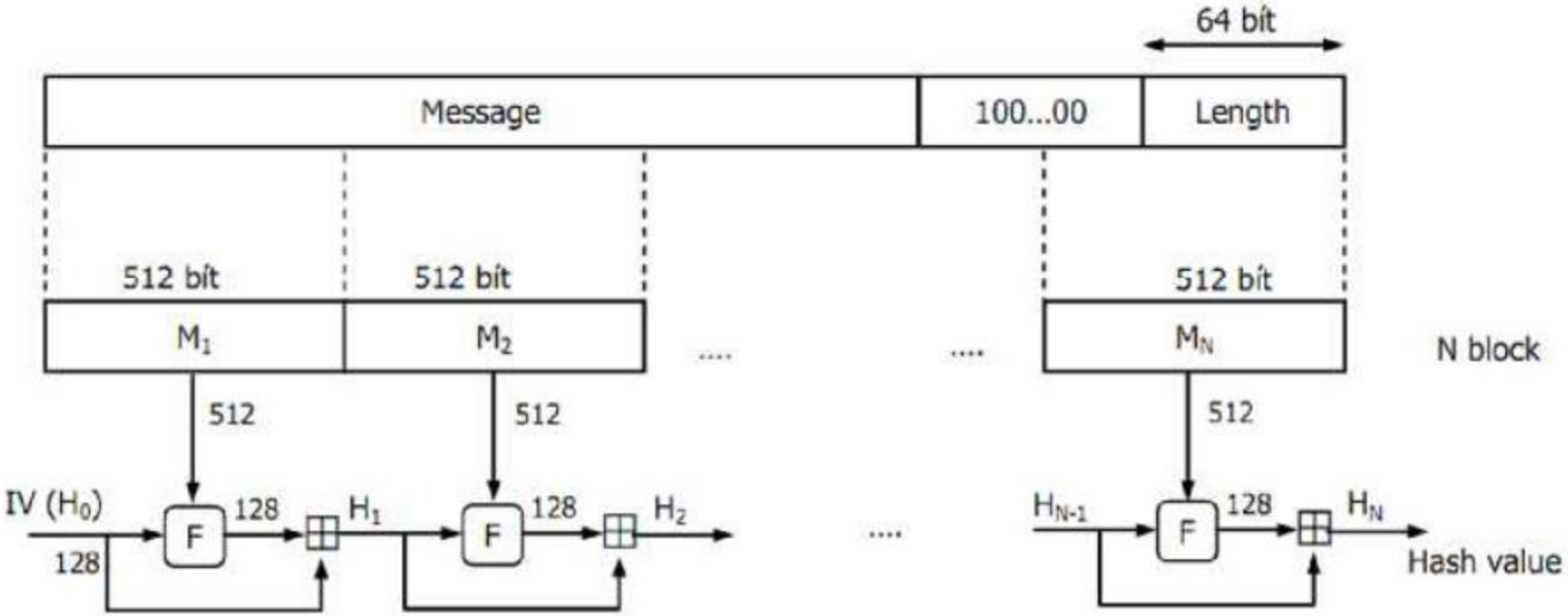
- **Vấn đề xung đột của thuật toán Hash:**
  - Khái niệm xung đột (*collision*) của hàm băm:
    - Thông điệp **X** sau khi băm **H(X)** => được mã băm $\mathbf{h_X}$.
    - Thông điệp **Y** sau khi băm **H(Y)** => được mã băm $\mathbf{h_Y}$..
    - Nếu $\mathbf{h_X} = \mathbf{h_Y}$ gọi là xung đột (*collision*)
  - Tính chống xung đột của thuật toán băm:
    - Chống xung đột yếu: nếu có thể tìm ra 2 thông điệp **X** và **Y** khác nhau mà **H(X) = H(Y)**
    - Chống xung đột mạnh: nếu không thể tìm ra 2 thông điệp **X** và **Y** khác nhau để có **H(X) = H(Y)**

Hàm băm (hash function)

- **Giới thiệu hàm băm MD5:**
  - Tổng quan:
    - **MD5**: *Message Digest version* **5** (tạm dịch: *tóm tắt thông điệp*).
    - Do *R. Rivest* đề xuất vào năm 1992.
    - Đã và vẫn đang được sử dụng rộng rãi.
  - Đặc tính của MD5:
    - Giá trị băm = **128** bit (tương đương **32** ký số Hex).
    - Thông điệp có kích thước tối đa 2^{64} bits.
  - Một vài ví dụ mã băm MD5:
    - Ký tự '**a**' => 0cc175b9c0f1b6a831c399e269772661
    - Chuỗi '**an toan thong tin**' => 5cab4237552af30a8c9d3c52f7f34058
    - Ký tự rỗng "" => d41d8cd98f00b204e9800998ecf8427e

Hàm băm (hash function)

- **Giới thiệu hàm băm MD5:**
  - Lưu đồ giải thuật băm **MD5**:
    - Tương tự mã khối **CBC** với: *Block size* = **512** bits, *IV* = **128** bits.
    - F là thuật toán băm 512 bits => 128 bits ($\mathbf{H_i}$).
    - Mã băm = mã $\mathbf{H_n}$ cuối cùng.



Hàm băm (hash function)

- **Giới thiệu hàm băm SHA-1:**
  - Tổng quan:
    - **SHA**: *Secure Hash Algorithm* (tạm dịch: *thuật toán băm có bảo mật*).
    - **SHA-1** được Hoa kỳ chuẩn hóa: **NIST FIPS 180-1** (1995).
    - Sử dụng trong các giao thức bảo mật thông tin hiện nay: **TLS, SSL, PGP, SSH, S/MIME...**
  - Đặc tính của SHA:
    - Giá trị băm = **160** bit (tương đương **40** ký số Hex).
    - Thông điệp có kích thước tối đa 2^{128} bits.
- **Giới thiệu hàm băm SHA-2:**
  - Tăng cường bảo mật cho ***SHA-1***.
  - Số bit của mã băm thay đổi theo các phiên bản: ***SHA-256, SHA-384, SHA-512***.

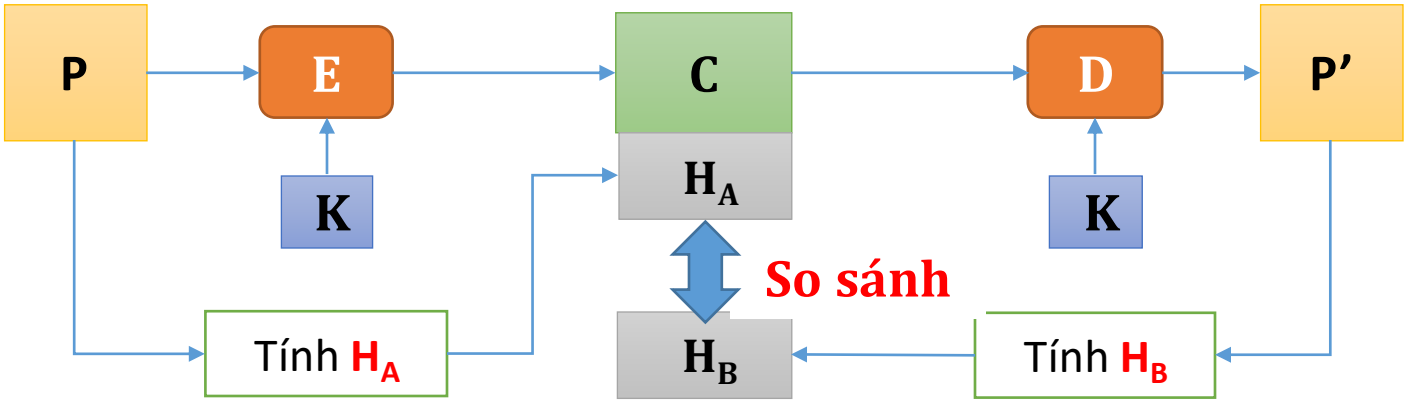
Chữ ký số (Digital Signature)

- **Chữ ký số (*Digital Signature* - DS):**
  - Là thông tin đi kèm theo thông điệp, nhằm hỗ trợ bên nhận:
    - Xác định ***chủ thể, nguồn gốc*** của thông điệp đó.
    - Xác minh tính ***toàn vẹn*** của thông điệp.
  - Công nghệ sử dụng:
    - Dùng hàm băm (***Hash***) cho xác minh tính toàn vẹn.
    - Dùng tính xác thực của mã hóa khóa bất đối xứng (***RSA***).

Đây có phải là "**Chữ ký số**" **?**

Chữ ký số (Digital Signature)

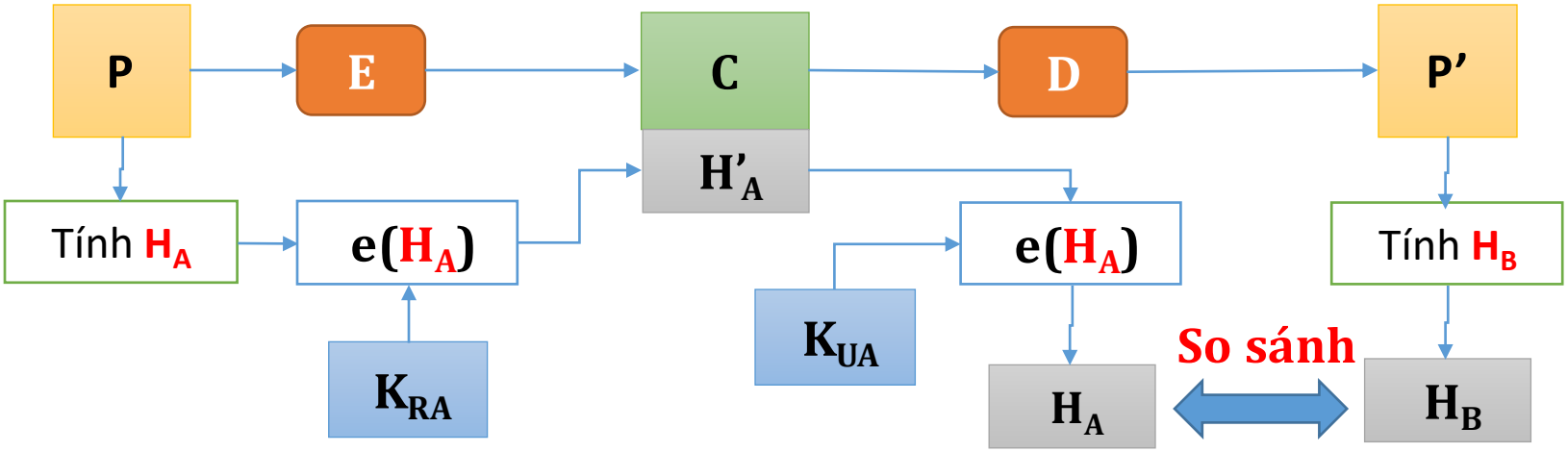
- **Hàm băm: xác minh tính toàn vẹn của thông điệp**



- Bên nhận:
  - Nếu $H_A = H_B$ => thông điệp **M** là toàn vẹn (không bị sửa đổi).
- Yếu điểm của xác thực bằng hàm băm:
  - Nếu *Trudy* đánh cắp **M** trên đường truyền -> sửa đổi **M'** -> tính lại H'_A rồi gởi cho bên nhận.
  - Bên nhận có phát hiện thông điệp **M** đã bị thay đổi?

Chữ ký số (Digital Signature)

- **Chữ ký số: kết hợp hàm băm và mã khóa bất đối xứng:**



- Bên gởi: mã hóa mã băm H_A bằng *Private key* (K_{RA} của bên gởi)
- Bên nhận: dùng *Public key* (K_{UA} của bên gởi) giải mã H'_A.
  - Nếu H_A do ***bên gởi mã hóa*** bằng K_{RA} => $H_A = H_B$. (xác minh)
  - Nếu H_A do ***Trudy*** mã hóa bằng K_{RT} => $H_A \neq H_B$ (không xác minh)

Chữ ký số (Digital Signature)

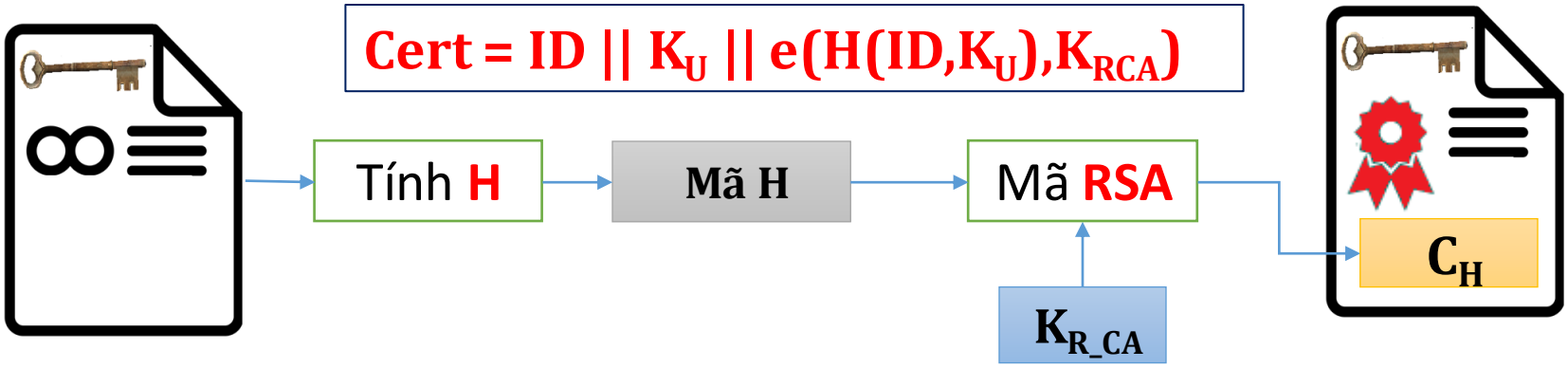
- **Nhận xét về Chữ ký số:**
  - Xét tình huống:
    - Nếu **C** công bố *Public key* $\mathbf{K_{UC}}$ giả mạo là của **A**
    - => **B** nhận $\mathbf{K_{UC}}$ và tin tưởng đó là *Public key* của **A**.
    - => **B** sẽ xác minh thông điệp mã hóa bằng $\mathbf{K_{RC}}$ là của **A**.
  - Nếu chỉ dùng ***Hash*** và ***RSA*** cho *Chữ ký số* là không đủ tin cậy.
  - Giải pháp:
    - Cần có một tổ chức (*Organization*) tin cậy, giúp **B** xác định *Public key* nhận được là của **A** => chống giả mạo.
    - => Dịch vụ ***Chứng thư số*** (*Digital Certificate* - **CA**)

Chứng thư số (Digital Certificate)

- **Khái niệm Chứng thư số (Digital Certificate):**
  - *Digital Certificate* là một *Chứng thư điện tử* do *Cơ quan có thẩm quyền* (*Certificate Authority* - **CA**) cấp cho ***chủ thể*** có nhu cầu.
  - Chứng thư số chứa *thông tin định danh*, *Public key* của ***chủ thể***.
  - Khi các đối tác nhận *Public key* chứa trong *Chứng thư số*, các **CA** tin tưởng sẽ chứng thực *Public key* này là của ***chủ thể***.
  - Quy chuẩn quốc tế của *Digital Certificate* **X.509**.

Chứng thư số (Digital Certificate)

- **Quy trình tạo Chứng thư số theo chuẩn X.509:**
  - Thông tin *ID* và *Public key* của chủ thể (*chứng chỉ chưa được chứng thực*) được chuyển về **CA**.
  - **CA** băm (hash) thông tin *ID* và *Public key* của chủ thể => tạo *mã băm*.
  - Thực hiện mã hóa cho *mã băm* bằng *Private Key* của **CA** => tạo C_H.
  - Gán (tagged) C_H thông tin *ID* và *Public key* của chủ thể => chứng chỉ đã được chứng thực (*Signed Certificate*).

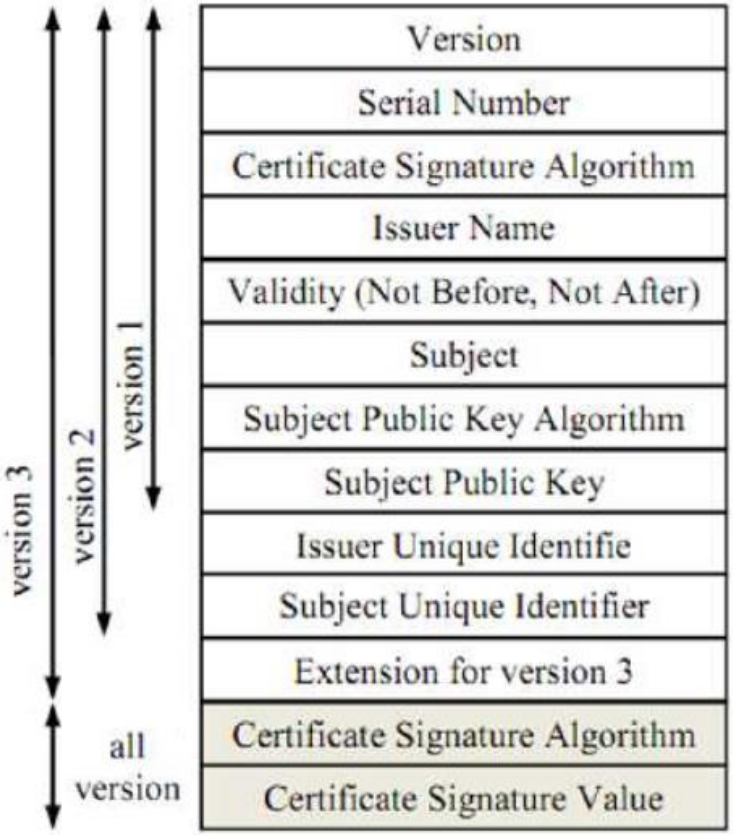


  - Chủ thể nhận *Signed Certificate* => cung cấp cho đối tác khi giao dịch.

Chứng thư số (Digital Certificate)

- **Cấu trúc bên trong một chứng thư số:**

(*xem giải thích ở slide sau)*



Version	Version 3
Serial Number	05:A0:4C
Certificate Signature Algorithm	PKCS #1 SHA-1 With RSA Encryption
Issuer Name	OU = Equifax Secure Certificate Authority; O = Equifax
Validity (Not Before, Not After)	04/01/2006 17:09:06 PM GMT - 04/01/2011 17:09:06 PM GMT
Subject	CN= login.yahoo.com; OU= Yahoo; O= Yahoo! Inc.
Subject Public Key Algorithm	PKCS #1 RSA Encryption
Subject Public Key	30 81 89 02 81 81 00 b5 6c 4f ee ef 1b 04 5d be…
Issuer Unique Identifie	
Subject Unique Identifier	
Extension for version 3	
Certificate Signature Algorithm	PKCS #1 SHA-1 With RSA Encryption
Certificate Signature Value	50 25 65 10 43 e1 74 83 2f 8f 9c 9e dc 74 64 4e…

Chứng thư số (Digital Certificate)

- **Diễn giả cấu trúc bên trong một chứng thư số:**
  - **Version**: phiên bản theo chuẩn X.509
  - **Serial Number**: Số nhận dạng của chứng chỉ số;
  - **Subject**: Thông tin nhận dạng một cá nhận hoặc một tổ chức;
  - **Certificate Signature Algorithm**: Thuật toán tạo chữ ký số.
  - **Certificate Signature Value**: giá trị của chữ ký số.
  - **Issuer name**: tên tổ chức CA cấp phát chứng thư số;
  - **Validity**: ngày cấp và hạn sử dụng chứng chỉ số;
  - **Subject** hay **Issue to**: tên chủ thể sử dụng.
  - **Subject Public Key Algorithm**: thuật toán tạo *Public key* của chủ thể.
  - **Subject Public Key**: giá trị *Public key* của chủ thể.
  - **Thumbprint**: Chuỗi băm tạo từ khóa công khai của chủ thể.

Chứng thư số (Digital Certificate)

- **Hoạt động của Chứng thư số:**
  1. *Server* yêu cầu => **CA** cung cấp *Certificate* cho *Server*.
  2. Khi nhận được yêu cầu giao dịch từ *Client*, *Server* gởi *Cert.* của nó cho *Client*, bao gồm khóa *public* K_{US}
  3. *Client* xác minh *Cert.* (*ID* và *Public key*):
     - ✓ Dùng K_{UCA} (*Public key* của ***CA***) giải mã ***Signature*** trên *Cert* => **h1**.
     - ✓ Băm *Cert.* (*ID* và *Public key*) => **h2** => so sánh với **h1**.
  4. Nếu *Cert.* xác minh đúng, *Client* sẽ giao dịch với *Server*:
     - ✓ Mã hóa dữ liệu bằng khóa K_{uS}
     - ✓ *Server* giải mã bằng khóa K_{RS}.

Hạ tầng khóa công khai (PKI)

- **Dẫn nhập:**
  - Ưu điểm của Chứng thư số:
    - Giải quyết được những tồn tại của Chữ ký số.
    - Hỗ trợ các giao thức mạng "chống giả mạo Public Key"
  - Nhược điểm:
    - Có quá nhiều nhà cung cấp (*Certificate Authority* - **CA**)
    - => *Client* dễ bị lừa bởi những CA giả mạo.
    - => thiếu tính nhất quán của các *Cert*.
    - => sự tin cậy, hạn sử dụng… của các *Cert*. là khác nhau nếu chúng được cấp bởi các CA khác nhau.
  - Giải pháp: xây dựng "*Hạ tầng khóa công khai*" (*Public Key Infrastructure* – **PKI**)

Hạ tầng khóa công khai (PKI)

- **Public Key Infrastructure – PKI**
  - Khái niệm **PKI**:
    - Hạ tầng khóa công khai là một tập các phần cứng, phần mềm, nhân lực, chính sách và các thủ tục để ***tạo***, ***quản lý***, ***phân phối***, ***sử dụng***, ***lưu trữ*** và ***thu hồi*** các chứng chỉ số (*Digital Certificate*)
  - Cấu trúc **PKI**:
    - Certificate Authority (CA): Cơ quan cấp và kiểm tra chứng chỉ số;
    - Registration Authority (RA): Bộ phận kiểm tra thông tin nhận dạng của người dùng theo yêu cầu của CA;
    - Validation Authority (VA): Cơ quan xác nhận thông tin nhận dạng của người dùng thay mặt CA;
    - Central Directory (CD): Là nơi lưu danh mục và lập chỉ số các khóa;
    - Certificate Management System: Hệ thống quản lý chứng chỉ;
    - Certificate Policy: Chính sách về chứng chỉ;

Cám ơn !