CHƯƠNG III
MÃ HÓA KHÓA CÔNG KHAI

ThS. Nguyễn Cao Đạt
E-mail: dat@hcmut.edu.vn

Tham khảo

[1]. Cryptography and Network Security: chương 4, 8, 9, 10

Nội dung trình bày

- Giới thiệu về trường hữu hạn
- Giới thiệu về lý thuyết số
- Mã hóa khóa công khai và RSA
- Các hệ mã khóa công khai khác
- Quản lý và phân phối khóa

Nội dung trình bày

- **Giới thiệu về trường hữu hạn**
  - Khái niệm về nhóm(group), vòng(ring), trường(field)
  - Phép toán modulo với số nguyên
  - Thuật toán Euclid
  - Trường hữu hạn GF(p)

Nhóm (Group)

- Một tập hợp các phần tử hay các số.
- {G, .}
- Mà với phép toán . thì kết quả cũng thuộc vào tập hợp này(bao đóng).
  - Luật kết hợp: $(a.b).c = a.(b.c)$
  - Có phần tử e: $e.a = a.e = a$
  - Có phần tử nghịch đảo a^{-1}: $a.a^{-1} = e$
- Nếu có tính giao hoán: $a.b = b.a$
  - Thì được gọi là nhóm abelian

Nhóm tuần hoàn

- Định nghĩa lũy thừa trong một nhóm như việc lặp lại của các phép toán nhóm.
  - Ví dụ: $a^3 = a.a.a$
- Và cho phép phần tử e là: $e=a^0$
- Một nhóm là tuần hoàn nếu mỗi phần tử là kết quả của phép toán mũ của vài phần tử cố định.
  - Ví dụ: $b = a^k$ cho vài giá trị a và mọi giá trị b trong nhóm.
  - a được gọi là một generator của nhóm.

Vòng(Ring)

- Một tập các số
- {R, +, .}
- Với hai phép toán(cộng và nhân):
  - Một nhóm abelian với phép toán cộng.
  - Và phép toán nhân thỏa:
    - Bao đóng
    - Kết hợp
    - Phân phối trên phép cộng: `a(b+c) = ab + ac`
- Nếu phép nhân là giao hoán thì được gọi là vòng giao hoán.
- Nếu phép toán nhân có phần tử đơn vị và không có ước số 0 thì được gọi là miền tách rời.

Trường(Field)

- **Một tập các số : {F,+,.}**
  - Với hai phép toán:
    - Nhóm abelian cho phép toán cộng.
    - Nhóm abelian cho phép toán nhân (bỏ qua 0).
    - Một vòng(ring).
- **Có hệ thống phân cấp với các luật**
  - Nhóm(group) -> vòng(ring) -> trường(field)
- **Ví dụ**
  - Số hữu tỉ, số thực, số phức là trường.
  - Số nguyên không phải là trường vì không có phân tử nghịch đảo của phép toán nhân.

Số học modulo

- **Định nghĩa phép toán modulo "`a mod n`" là phần dư khi a chia n.**
- **Thuật ngữ đồng dư: `a = b mod n`**
  - a và b có phần dư như nhau khi chia với n.
  - Ví dụ: 100 = 34 mod 11
- **b được gọi là phần dư (residue) của a mod n**
  - Vì các số nguyên được viết dưới dạng: `a = qn + b`
  - Thường chọn số dư dương nhỏ nhất là residue

    `0 <= b <= n-1`
  - Quá trình này được biết là "**modulo reduction**"
  - `Ví dụ:-12 mod 7 = -5 mod 7 = 2 mod 7 = 9 mod 7`

Ước số

- Một số nguyên b khác 0 chia hết bởi số nguyên a nếu tồn tại số nguyên m: $a = mb$
- $b \mid a$
- Khi đó b được gọi là một ước số của a.
- Ví dụ các ước số của 24 là 1,2,3,4,6,8,12,24

Các phép toán trên số học modulo

- **Thực hiện các phép toán trong phạm vi một tập các số modulo n.**
- Thực hiện các phép toán cộng, nhân và sau đó thực hiện modulo để có kết quả:
  - `a+b mod n = [a mod n + b mod n] mod n`

Số học Modulo

- **Số học modulo với bất kỳ các nhóm số nguyên:** $Z_n = \{0, 1, \ldots, n-1\}$
- **Một vòng giao hoán cho phép cộng**
- **Có phần tử nghịch đảo của phép toán nhân ?**
- **Chú ý**
  - Nếu `(a+b)=(a+c) mod n thì b=c mod n`
  - Nếu `(a.b)=(a.c) mod n`

    thì `b=c mod n` nếu và chỉ nếu `a` và `n` là nguyên tố cùng nhau.

Ví dụ phép cộng trên Z_8

+	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	0
2	2	3	4	5	6	7	0	1
3	3	4	5	6	7	0	1	2
4	4	5	6	7	0	1	2	3
5	5	6	7	0	1	2	3	4
6	6	7	0	1	2	3	4	5
7	7	0	1	2	3	4	5	6

(nghich +: 2 - 6)

x8, module 8 = 1 ?

Ước số chung lớn nhất (GCD)

- **Greatest common divisor(GCD)**
- **Một vấn đề phổ biến trong lý thuyết số**
- **GCD (a,b) là số nguyên dương lớn nhất được chia hết cho cả a và b.**
  - Ví dụ: GCD(60,24) = 12
- **Hai số nguyên tố cùng nhau nếu chúng chỉ có ước chung lớn nhất là 1**
  - Ví dụ: GCD(8,15) = 1
  - 8 và 15 là nguyên tố cùng nhau.

Thuật toán Euclid

- **Một cách hiệu quả để tìm ra GCD(a,b)**
- **Dùng cách tính toán như sau:**

  GCD(a,b) = GCD(b, a mod b)
- **Thuật toán Euclid**

```
EUCLID(a,b)
1. A = a; B = b
2. if B = 0 return  A = gcd(a, b)
3. R = A mod B
4. A = B
5. B = R
6. goto 2
```

Ví dụ GCD(1970,1066)

1970 = 1 x 1066 + 904	gcd(1066, 904)
1066 = 1 x 904 + 162	gcd(904, 162)
904 = 5 x 162 + 94	gcd(162, 94)
162 = 1 x 94 + 68	gcd(94, 68)
94 = 1 x 68 + 26	gcd(68, 26)
68 = 2 x 26 + 16	gcd(26, 16)
26 = 1 x 16 + 10	gcd(16, 10)
16 = 1 x 10 + 6	gcd(10, 6)
10 = 1 x 6 + 4	gcd(6, 4)
6 = 1 x 4 + 2	gcd(4, 2)
4 = 2 x 2 + 0	gcd(2, 0)

Trường Galois(Galois Field – GF)

- Các trường hữu hạn đóng một vai trò quan trọng trong mật mã.
- **Trường hữu hạn còn được gọi là trường Galois. Ký hiệu là: GF(p^n)**
- **Số phần tử trong một trường hữu hạn phải là hàm mũ của p^n (p là số nguyên tố).**
- Trong thực tế thường dùng các trường sau:
  - GF(p)
  - GF(2^n)

Trường Galois GF(p)

- **GF(p) là một tập các số nguyên {0,1, … , p-1} với các phép toán số học trên modulo số nguyên tố p.**
  - Trường hữu hạn.
  - Có thể thực hiện các phép toán +, -, ., / trên trường GF(p).

Ví dụ phép toán nhân và phần tử nghịch đảo trên z_7 [GF(7)]

×	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	4	6	1	3	5
3	0	3	6	2	5	1	4
4	0	4	1	5	2	6	3
5	0	5	3	1	6	4	2
6	0	6	5	4	3	2	1

w	$-w$	w^{-1}
0	0	—
1	6	1
2	5	4
3	4	5
4	3	2
5	2	3
6	1	6

Thuật toán tìm phần tử nghịch đảo

EXTENDED EUCLID(*m*, *b*)

1. (A1, A2, A3)=(1, 0, *m*);
   (B1, B2, B3)=(0, 1, *b*)
2. **if** B3 = 0
   return A3 = gcd(*m*, *b*); no inverse
3. **if** B3 = 1
   return B3 = gcd(*m*, *b*); B2 = b^{-1} mod *m*
4. Q = A3 div B3
5. (T1, T2, T3)=(A1 – Q B1, A2 – Q B2, A3 – Q B3)
6. (A1, A2, A3)=(B1, B2, B3)
7. (B1, B2, B3)=(T1, T2, T3)
8. **goto** 2

Phần tử nghịch đảo của 550 trên GF(1759)

Q	A1	A2	A3	B1	B2	B3
—	1	0	1759	0	1	550
3	0	1	550	1	–3	109
5	1	–3	109	–5	16	5
21	–5	16	5	106	–339	4
1	106	–339	4	–111	**355**	1

Nội dung trình bày

- Giới thiệu về trường hữu hạn
- Giới thiệu về lý thuyết số
  - Số nguyên tố
  - Định lý Fermat, Euler
  - Kiểm tra số nguyên tố problem -> RSA
  - Logarit rời rạc problem -> El Gamal
- Mã hóa khóa công khai và RSA
- Các hệ mã khóa công khai khác
- Quản lý và phân phối khóa

Nội dung trình bày

- **Giới thiệu về lý thuyết số**
  - Số nguyên tố
  - Định lý Fermat
  - Định lý Euler
  - Kiểm tra số nguyên tố
  - Logarit rời rạc

Số nguyên tố

- Vấn đề nghiên cứu số nguyên tố là trung tâm của lý thuyết số.
- Số nguyên tố là số chỉ chia hết cho 1 và chính nó.
- Tức là nó không thể viết là tích của các số khác.
- Chú ý: 1 là số nguyên tố nhưng nói chung thường được bỏ qua.
- Ví dụ: 2,3,5,7 là số nguyên tố; 4,6,8,9,10 là hợp số.
- Danh sách các số nguyên tố nhỏ hơn 200 là:

  2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67
  71 73 79 83 89 97 101 103 107 109 113 127 131 137
  139 149 151 157 163 167 173 179 181 191 193 197 199

Phân tích thừa số nguyên tố

- Phân tích thừa số một số n là viết nó thành tích của các số khác: n=a x b x c
- **Ví dụ:** $91=7 \times 13$; $3600=2^4 \times 3^2 \times 5^2$
- Chú ý rằng việc phân tích thừa số một số là tương đối khó so với việc nhân các thừa số để tạo nên số đó.
- Phân tích thừa số nguyên tố tổng quát:

$$a = \prod_{p \in \mathrm{P}} p^{a_p}$$

Nguyên tố cùng nhau

- Hai số a, b được gọi là nguyên tố cùng nhau nếu không có ước số chung nào lớn hơn 1.
- Ví dụ: 8 và 15 là nguyên tố cùng nhau.
- Có thể xác định ước số chung lớn nhất của chúng (greatest common divisor –GCD) bằng việc so sánh các thừa số nguyên tố của chúng và dùng số mũ nhỏ nhất.
- Ví dụ: $300=2^1x3^1x5^2$ và $18=2^1x3^2$

  $GCD(18,300)=2^1x3^1x5^0=6$

Định lý Fermat

- $a^{p-1} = 1 \pmod p$ khi p là số nguyên tố và $\gcd(a,p)=1$
- Cũng tương đương với $a^p = a \pmod p$
- Cũng được biết là định lý Fermat nhỏ.
- Được dùng trong mã hóa khóa công khai và kiểm tra tính nguyên tố

Hàm phi Euler ø(n)

- **Là số các số nguyên dương nhỏ hơn n và nguyên tố cùng nhau với n.**
- **Ký hiệu là ø(n)**
- **Ví dụ**
  - Với n=10.
  - Tập các số nguyên dương nhỏ hơn n đầy đủ là: {1,2,3,4,5,6,7,8,9}
  - Tập các số nguyên dương nhỏ hơn n và nguyên tố cùng nhau với 10 là: {1,3,7,9}
  - ø(10) = 4.

Hàm phi Euler ø(n)

- **Để tính toán ø(n) cần phải đếm số phần tử trong tập hợp tương ứng.**
- **Một số trường hợp đặc biệt**
  - Nếu p là nguyên tố: ø(p) = p-1
  - Nếu p, q là nguyên tố (p ≠ q: ø(pq) =(p-1)x(q-1)

p^a = (p-1) p^(a-1)

- **Ví dụ**

  ø(37) = 36

  ø(21) = (3-1)x(7-1) = 2x6 = 12

- **Tính toán ø(n) trong trường hợp tổng quát**

Định lý Euler

- **Tổng quát của định lý Fermat**
- **$a^{\phi(n)} = 1 \pmod n$ với bất kỳ số nguyên a và n sao cho gcd(a,n)=1**
- **Ví dụ**

  $a=3,\ n=10:\ \phi(10)=4$ và $3^4 = 81 = 1 \bmod 10$

  $a=2,\ n=11:\ \phi(11)=10$ và $2^{10} = 1024 = 1 \bmod 11$

Kiểm tra tính nguyên tố

- **Cần tìm ra các số nguyên tố lớn**
- slow **Sàn truyền thống dùng chia thử**
  - Ví dụ chia các số nguyên tố nho hơn căn bậc hai của số đang xem xét.
  - Chỉ làm được khi số nguyên là nhỏ.
- **Hướng thay thế là dùng các kiểm tra tính**
  1st popular (uncertain) **nguyên tố dựa trên thuộc tính của số nguyên tố**
  - Các số nguyên tố thỏa mãn tính chất này.
  - Nhưng một vài hợp số cũng thỏa tính chất này và được gọi là giả nguyên tố.
- **Có thể dùng kiểm tra tính nguyên tố xác định**
  may slow **nhưng chậm hơn**

Thuật toán Miller Rabin

- **Dựa trên định lý Fermat**
- **Thuật toán**

TEST (n) is:

1. Find integers k, q, $k > 0$, q odd, so that $(n-1)=2^k q$
2. Select a random integer a, $1<a<n-1$
3. **if** $a^q \bmod n = 1$ **then** return ("maybe prime");
4. **for** $j = 0$ **to** $k - 1$ **do**
   5. **if** $(a^{2^j q} \bmod n = n-1)$
      then return(" maybe prime ")
6. return ("composite")

Xem xét về xác suất

- **Nếu thuật toán Miller-Rabin trả về "composite" thì số nguyên n chắc chắn là hợp số.**
- **Ngược lại có thể là nguyên tố hay giả nguyên tố**
- **Cơ hội phát hiện số giả nguyên tố trong 1 vòng lặp là: $< {}^{1}/_{4}$**
- **Nếu lặp với số lần kiểm tra là t thì:**
  - $Pr(n, t) = 1-4^{-t}$
  - Ví dụ: Với t=10, ta có: $Pr(n, 10) > 0.99999$

Căn nguyên thủy(Primitive Root)

- **Từ định lý Euler:** $a^{\varnothing(n)} \bmod n = 1$
- **Xem xét:** $a^m = 1 \pmod n$, $GCD(a,n) = 1$
  - Phải tồn tại m nhỏ hơn hay bằng $\varnothing(n)$
  - Một khi hàm mũ của a đến m thì kết quả được lặp lại.
- **Nếu m nhỏ nhất bằng $\varnothing(n)$ thì a được gọi là căn nguyên thủy của n.**
- **Nếu n là số nguyên tố thì hàm mũ của a tạo ra một nhóm (group)** $\bmod p$
- **Các số a như vậy được dùng nhiều trong mã hóa khóa công khai nhưng nó tương đối khó tìm.**

a=2, n=9
phi(n) = 2x3 = 6
thử 6, 3, 2 -> thừa số 6
2^6 = 1 mod 9
2^3 = 8 mod 9
2^2 = 4 mod 9
=> 2 là primitive root của 9

4 -> NO

=> 2^ = modulo
domain = result => song ánh

Logarit rời rạc(Discrete Logarithm)

- **Tìm x sao cho $y = g^x \pmod p$**
- **Được viết như sau:**

$$x = \text{dlog}_g\ y \pmod p$$

- **Nếu g là một căn nguyên thủy của p thì x luôn luôn tồn tại.**

Ví dụ:

$x = \log_3 4 \bmod 13$: không có đáp số

$x = \log_2 3 \bmod 13 = 4$

a=3, p=13
phi(p) = 12
3^12 = 1 mod 13
3^3 = 27 mod 13
=> No primitive root

- **<u>Lưu ý:</u> lũy thừa là tương đối dễ dàng, việc tìm kiếm logarit rời rạc lại là một vấn đề khó khăn.**

Nội dung trình bày

- Giới thiệu về trường hữu hạn
- Giới thiệu về lý thuyết số
- Mã hóa khóa công khai và RSA
  - Các nguyên lý của mã hóa công khai
  - Thuật toán, hiện thực, tính an toàn của hệ mã RSA
- Các hệ mã khóa công khai khác
- Quản lý và phân phối khóa

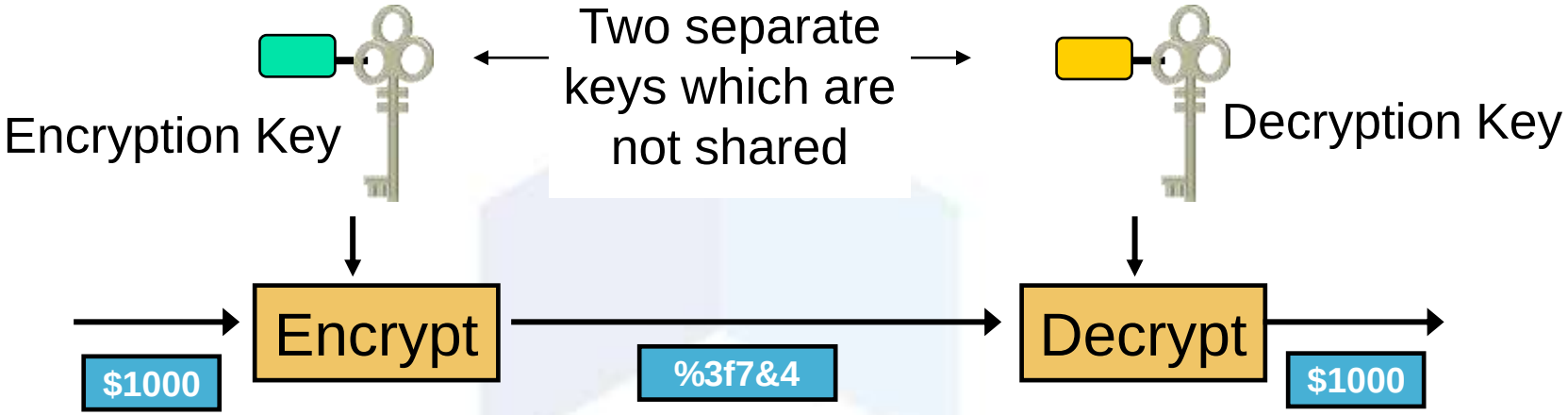
Mã hóa khóa công khai

- **Khuyết điểm của các hệ mã đối xứng**
  - Mã hóa đối xứng dùng một khóa được chia xẻ giữa bên gởi và bên nhận.Nếu khóa này bị tiết lộ, thông tin liên lạc sẽ bị thương tổn.
  - Hệ mã đối xứng xem các bên là như nhau vì vậy không thể bảo vệ bên gửi từ việc giả mạo thông điệp bên nhân và việc tuyên bố thông điệp đã được gửi từ bên gửi.

Mã hóa khóa công khai

- **Mã hóa khóa công khai**
  - Sự phát triển của mã hóa khóa công khai là cuộc cách mạng lớn nhất trong toàn bộ lịch sử 3000 năm của ngành mật mã.
  - Dùng hai khóa gồm một khóa công khai và một khóa riêng.
  - Không đối xứng vì các bên không như nhau.
  - Ứng dụng trên các khái niệm lý thuyết số.
  - Bổ sung chứ không thay thế các hệ mã khóa bí mật.

Mã hóa khóa công khai



- Bổ sung chứ không thay thế các hệ mã khóa đối xứng.
- Khóa thường dùng có chiều dài từ 512 bits đến 4096 bits.
- Tính toán chậm vì chúng dựa trên các phép toán phức tạp.
- Các hệ mã thông dụng gồm RSA, El Gamal, EC, DH.

Vì sao dùng mã hóa khóa công khai ?

- **Giải quyết hai vấn đề khó khăn trong mã hóa đối xứng**
  - **Phân phối khóa:** làm thế nào truyền thông an toàn để truyền khóa bí mật.
  - **Chữ ký số** : Làm thế nào để xác minh thông điệp đến từ một người gửi.
- **Phát minh công khai được Whitfield Diffie và Martin Hellman trình bày tại đại học Stanford vào năm 1976**

Mã hóa khóa công khai

- **Khóa công khai/ hai khóa/mã hóa bất đối xứng liên quan đến việc dùng hai khóa:**
  - Một khóa công khai(**public-key**): Bất kỳ ai cũng được biết . Được dùng để mã hóa thông điệp cần gửi hay xác minh chữ ký của một thông điệp đã nhận.
  - Một khóa riêng(**private-key**): Chỉ người dùng biết. Được dùng để giải mã một thông điệp đã nhận hay ký(tạo) chữ ký trên một thông điệp cần gửi.
- **Là bất đối xứng**
  - Một bên mã hóa hay xác minh thông điệp thì không giải mã hay tạo chữ ký (phía bên kia).

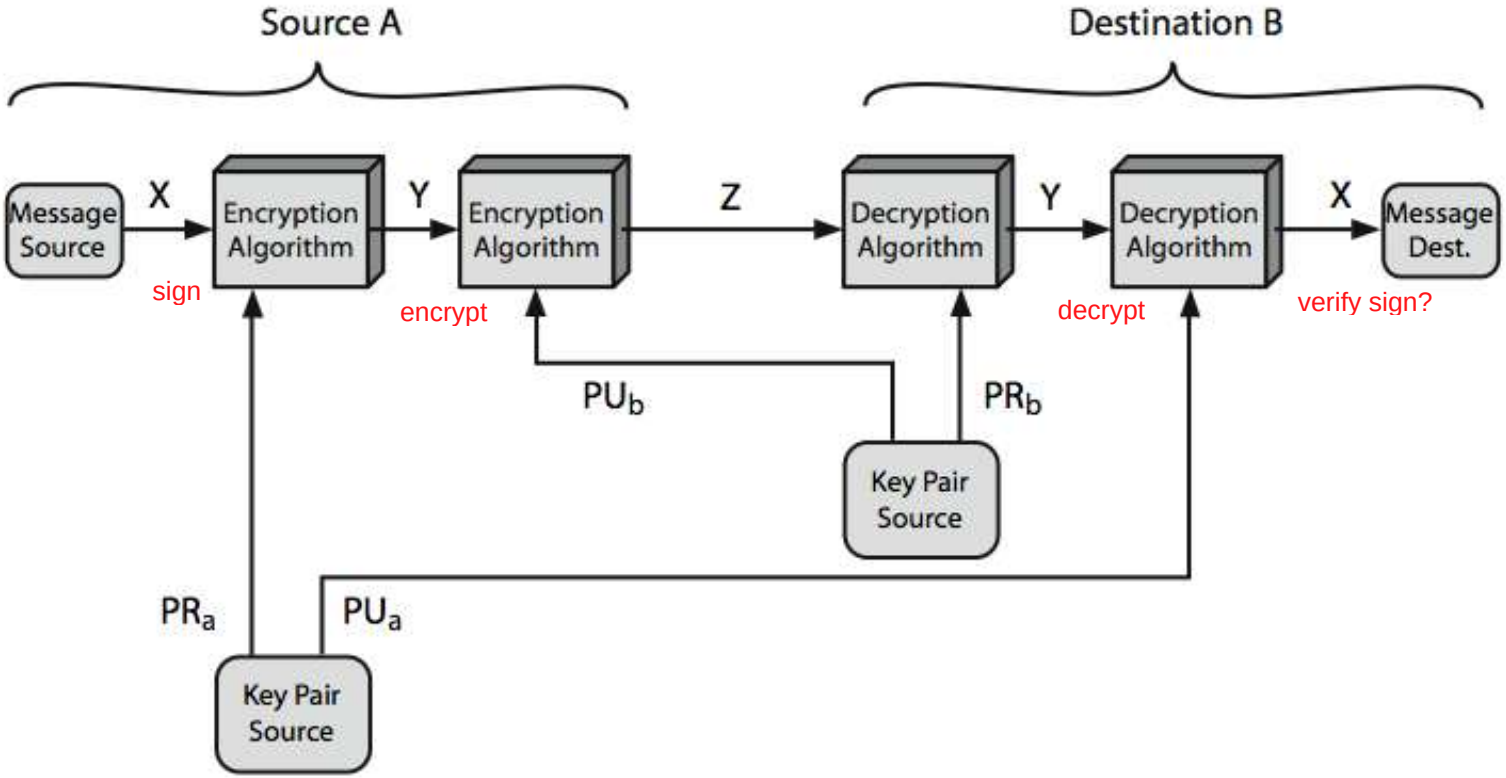
Các đặc điểm của khóa công khai

- Không khả thi để tìm khóa riêng khi chỉ biết thuật toán và khóa công khai.
- Dể dàng tính toàn để mã hóa/giải mã thông điệp khi khóa tương ứng được biết.
- Vai trò của hai khóa là như nhau. Một khóa là khóa mã hóa thì khóa kia là khóa giải mã

Các bài toán ngược

- **Các hệ mã dựa trên việc khó giải các bài toán ngược.**
- **Bài toán phân tích thừa số nguyên tố**
  - Cho 2 số nguyên tố lớn p và q dể dàng để tính toán n = p.q.
  - Nhưng cho n sẽ rất khó để phân tích n thành 2 số nguyên tố lớn p và q.
- **Bài toán logarit rời rạc**
  - Cho $\alpha = g^a \bmod p$. Cho a, g và p tính α là đơn giản.
  - Tuy nhiên cho α, g và p sẽ rất khó để tính toán a.

Ứng dụng của mã hóa khóa công khai



Ứng dụng của mã hóa khóa công khai

- Có thể phân thành 3 loại ứng dụng:
  - Mã hóa / giải mã (bí mật)
  - Chữ ký số (Xác thực)
  - Trao đổi khóa (khóa phiên làm việc)
- Một số thuật toán phù hợp với tất cả loại ứng dụng.
- Một số thuật toán khác chỉ phù hợp cho một hay hai loại ứng dụng cụ thể.

An toàn trong các lược đồ khóa công khai

- Về mặt lý thuyết, tấn công vét cạn là có thể nhưng kích thước khóa được dùng rất lớn (>512bits)
- An toàn dựa trên kích thước khóa đủ lớn để tạo sự khác biệt giữa dể dàng mã hóa/giải mã và khó khăn trong phân tích mã.
- Đòi hỏi dùng các số đủ lớn do đó chúng chậm hơn so với các lược đồ khóa bí mật.

Mã hóa RSA

- Do Rivest, Shamir và Adleman của MIT đưa ra vào năm 1977
- Lược đồ tốt nhất được biết và được sử dụng rộng rãi.
- Dựa trên lũy thừa trong một trường hữu hạn trên các số nguyên modulo một số nguyên tố.
- Dùng các số nguyên lớn(1024 bits)
- An toàn dựa trên bài toán phân tích thừa số nguyên tố lớn.

Tạo khóa trong RSA

- **Mỗi người tạo một cặp khóa.**
- **Chọn ngẫu nhiên 2 số nguyên tố lớn là p, q**
- **Tính toán $n=p.q$**
  - Lưu ý $\phi(n)=(p-1)(q-1)$
- **Chọn ngẫu nhiên một khóa mã hóa e:**
  $1<e<\phi(n)$, $\gcd(e,\phi(n))=1$
- **Giải phương trình sau để tìm ra khóa giải mã d :** $e.d=1 \bmod \phi(n)$ and $0 \leq d \leq \phi(n)$
- **Khoá công khai: PU={e,n}**
- **Khóa riêng: PR={d,n}**

Sử dụng RSA

gcd(M , {p, q}) = 1

- **Để mã hóa một thông điệp M:**
  - Lấy khóa công khai của người nhận là: $PU=\{e,n\}$
  - Tính toán: $C = M^e \bmod n$, $0 \le M < n$
- **Để giải mã thông điệp đã mã hóa:**
  - Dùng khóa riêng: $PR=\{d,n\}$
  - Tính toán: $M = C^d \bmod n$
- **Chú ý rằng thông điệp M phải nhỏ hơn n.**

Vì sao RSA hoạt động

- **Dựa trên định lý Euler:**
  - $a^{\phi(n)} \bmod n = 1$ khi $\gcd(a,n)=1$
- **Trong RSA ta có:**
  - $n=p.q$
  - $\phi(n)=(p-1)(q-1)$
  - $e.d=1 \bmod \phi(n)$ nên $e.d=1+k.\phi(n)$
- **Vậy:**

$$C^d = M^{e.d} = M^{1+k.\phi(n)} = M^1 . (M^{\phi(n)})^k$$
$$= M^1 . (1)^k = M^1 = M \bmod n$$

gcd(C, n) != 1 ????
-> still correct

Ví dụ RSA – Tạo khóa

pre: {p, q, e}
n = p * q
phi = (p - 1) * (q - 1)
de % phi = 1

PU = {e, n}
PR = {d, n}

1. Chọn các số nguyên tố: p=17 & q=11
2. Tính toán $n = pq$ =17 x 11=187
3. Tính toán $\phi(n)=(p-1)(q-1)$=16 x 10=160
4. Chọn e sao cho gcd(e,160)=1: lấy e=7
5. Xác định d: $d.e$=1 mod 160 và d < 160
   Tìm ra được d=23 vì 23x7=161=1x160+1

1. Công bố khóa công khai: PU={7,187}
2. Giữ bí mật khóa riêng: PR={23,187}

Ví dụ RSA – mã hóa/giải mã

- **Thông điệp** $M = 88$ **(88<187)**
- **Mã hóa:**

$$C = 88^7 \bmod 187 = 11$$

C = M^e mod n

- **Giải mã:**

$$M = 11^{23} \bmod 187 = 88$$

M = C^d mod n

Phép toán lũy thừa modulo

- **Dùng thuật toán Square & Multiply**
- **Thuật toán nhanh và hiệu quả.**
- **Chỉ mất $O(\log_2 n)$ phép nhân với số n**
  - eg. $7^5 = 7^4.7^1 = 3.7 = 10 \bmod 11$
  - eg. $3^{129} = 3^{128}.3^1 = 5.3 = 4 \bmod 11$

=
3^2
3^4
3^8
....

thuật toán Square & Multiply

```
modPow(a, b, n)
// compute a^b mod n
// b = b_k b_k1 ... b_0
c = 0; f = 1
for i = k downto 0
    do c = 2 x c
       f = (f x f) mod n
    if b_i == 1 then
       c = c + 1
       f = (f x a) mod n
 return f
```

Mã hóa hiệu quả

- **Nếu e nhỏ thì mã hóa nhanh hơn:**
  - Thông thường chọn: e=65537 (2^{16}-1)
  - Cũng có thể chọn e=3 hay e=17 ???
- **Nhưng nếu chọn e quá nhỏ(e=3) thì có thể bị tấn công.**
- **Nếu chọn e cố định thì phải đảm bảo:**

  $\gcd(e, \phi(n)) = 1$

Tạo khóa RSA

- **Người dùng RSA phải:**
  - Xác định hai số nguyên tố `p, q`
  - Chọn e hay d để tính số còn lại
- **Hai số nguyên tố `p,q` không dể dàng lấy được từ `n=p.q`**
  - Chúng phải đủ lớn
  - Đoán ngẫu nhiên và dùng kiểm tra xác suất
- **`d` là nghịch đảo của `e` nên dùng thuật toán Extended Euclid để tính toán.**

An toàn của RSA

- **Các hướng tiếp cận tấn công RSA:**
  - Tìm kiếm khóa dùng burte force
    - Không khả thi với kích thước các số.
  - Các tấn công toán học
    - Dựa trên tính toán ø(n) từ n
    - Phân tích n ra thừa số
  - Các tấn công thời gian
    - Dựa trên thời gian thực thi của giải mã
  - Các tấn công được chọn bản mã

Bài toán phân tích thừa số

- **Các hướng tiếp cận gồm 3 dạng:**
  - Phân tích $n=p.q$, rồi tính $\varnothing(n)$ và cuối cùng tính d
  - Xác định $\varnothing(n)$ trực tiếp và tính d
  - Tìm d trực tiếp
- **Hiện tại 3 dạng tiếp cận đều tương đương với việc phân tích thừa số**
  - Có một số cải tiến chậm trong nhiều năm qua
    - 05/2005: số có 200 chữ số (663 bits) bị phân tích bởi LS.
  - Cải tiến chủ yếu là cải tiến thuật toán
    - Quadratic Sieve(QS) -> Generalized Number Field Sieve(GHFS) -> Lattice Sieve(LS)
  - Hiện nay kích thước 1024-2048 bits trong RSA vẫn an toàn

Các tấn công thời gian

- **Được phát triển bởi Paul Kocher vào khoảng giữa thập kỷ 90**
  - Khai thác các khoảng thời gian thực hiện trong các phép toán
    - Ví dụ: nhân với số bé hay số lớn.
  - Suy ra kích thước toán hạng dựa trên thời gian thực hiện
- **Biện pháp đối phó**
  - Dùng thời gian tính toán lũy thừa là hằng số
  - Thêm vài độ trễ ngẫu nhiên

Các tấn công được chọn bản mã

- **Tấn công được chọn bản mã(Chosen Ciphertext Attack-CCA)**
- Các tấn công được chọn bản mã khai thác các thuộc tính của RSA để cung cấp thông tin trong việc phân tích mã.
- **Các kẻ tấn công chọn các bản bản mã và lấy được các bản rõ được giải mã ngược lại.**
- **Biện pháp đối phó**
  - Thêm ngẫu nhiên vào bản rõ trước khi mã hóa hay thay đổi bản rõ trước khi mã hóa dùng OASP (Optimal Asymmetric Encryption Padding)

Nội dung trình bày

- Giới thiệu về trường hữu hạn
- Giới thiệu về lý thuyết số
- Mã hóa khóa công khai và RSA
- Các hệ mã khóa công khai khác
  - Trao đổi khóa Diffie-Hellman
  - El Gamal
- Quản lý và phân phối khóa

Trao đổi khóa Diffie-Hellman

- Lược đồ khóa công khai đầu tiên được đề xuất bởi Diffie & Hellman vào năm 1976.
- Là phương pháp cho phép trao đổi công khai một khóa bí mật.
- Được dùng trong một số các sản phẩm thương mại.

Trao đổi khóa Diffie-Hellman

- **Lược đồ phân phối khóa**
  - Không thể dùng để trao đổi thông điệp tùy ý
  - Nó dùng để thiết lập một khóa giữa hai bên
  - Khóa này chỉ được biết bởi hai bên tham gia
  - Giá trị của khóa phụ thuộc vào các bên tham gia
- **Dựa trên phép lũy thừa trên trường hữu hạn**
- **Tính an toàn dựa trên việc khó khăn trong tính toán logarit rời rạc.**

Khởi tạo Diffie-Hellman

- **Các người dùng đồng ý trên các tham số toàn cục:**
  - Số nguyên tố lớn là q
  - a là căn nguyên thủy của q
- **Mỗi người dùng(ví dụ: A) tạo khóa của họ:**
  - Chọn khóa riêng: $x_A < q$
  - Tính toán khóa công khai: $y_A = a^{x_A} \bmod q$
- **Mỗi người dùng có một khóa công khai là y_A**

Trao đổi khóa Diffie-Hellman

- **Chia sẻ khóa phiên làm việc cho A và B là K_{AB}:**

$$K_{AB} = a^{x_A.x_B} \bmod q$$
$$= y_A^{x_B} \bmod q \quad (\textbf{B có thể tính được})$$
$$= y_B^{x_A} \bmod q \quad (\text{A có thể tính được})$$

- **K_{AB} được dùng như một khóa phiên làm việc trong lược đồ mã hóa khóa bị mật giữa A và B.**
- **Nếu A và B sau đó giao tiếp thì họ vẫn dùng cùng khóa phiên trừ phi A hay B chọn một khóa công khai khác.**

Ví dụ về Diffie-Hellman

- **Người dùng Alice và Bob trao đổi khóa**
  - Đồng ý dùng $q=353$ và $a=3$
  - Chọn khóa riêng ngẫu nhiên:
    - A chọn $x_A=97$, B chọn $x_B=233$
  - Tính toán khóa công khai tương ứng:
    - $y_A=3^{97} \mod 353 = 40$ (Al)
    - $y_B=3^{233} \mod 353 = 248$ (B)
  - Trao đổi khóa công khai
  - Tính toán khóa phiên làm việc:
    - $K_{AB}= y_B^{x_A} \mod 353 = 248^{97} = 160$ (A)
    - $K_{AB}= y_A^{x_B} \mod 353 = 40^{233} = 160$ (B)

pre: {q, a. Xa. Xb}
Ya = a^Xa % q
Yb = b^Xb % q

Kab = Yb^Xa % q
Kab =Ya^Xb % q

El Gamal

- **An toàn cũng dựa trên bài toán logarit rời rạc.**
- **Tất cả các thực thể đồng ý dùng:**
  - Số nguyên tố p khoảng 200 chữ số.
  - Và một căn nguyên thủy g
- **Mỗi người tính toán cặp khóa**
  - Chọn một giá trị ngẫu nhiên a làm khóa riêng, a < p, a có số chữsố như p.
  - Tính toán khóa công khai: $\alpha = g^a \bmod p$

Mã hóa/giải mã trong El Gamal

- **B mã hóa thông điệp P và gởi đến A**
  - B chọn k ngẫu nhiên (0< k< p). Tính $\mu = g^k \bmod p$
  - B tính $C = P\alpha^k \bmod p$
  - Gửi (C, μ) đến A
- **A giải mã thông điệp nhận được**
  - A tính $\mu^a = (g^k)^a = (g^a)^k = \alpha^k \bmod p$
  - A tính phần tử nghịch đảo của $\alpha^k \bmod p$:
    $v = (\alpha^k \bmod p)^{-1}$
  - A tính $Cv \bmod p = P\alpha^k (\alpha^k)^{-1} \bmod p = P$

Ví dụ về El Gamal

- p = 13, g =2
- A chọn a =3, tính $\alpha = 2^3 \bmod 13 = 8$
- B mã hóa thông điệp P và gửi đến A:
  - chọn ngẫu nhiên k = 5:
  - Tính $\mu = 2^5 \bmod 13 = 6$
  - Với P = 4. B tính $C = 4.8^5 \bmod 13 = 6$
  - Gửi (C, μ) đến A
- A giải mã thông điệp nhận được
  - A tính $\mu^a \bmod p = 6^3 \bmod 13 = 8$
  - A tính phần tử nghịch đảo của 8 là v =5
  - A tính Cv mod p = 6.5 mod 13 = 4 = P

Nội dung trình bày

- Giới thiệu về trường hữu hạn
- Giới thiệu về lý thuyết số
- Mã hóa khóa công khai và RSA
- Các hệ mã khóa công khai khác
- Quản lý và phân phối khóa
  - Phân phối khóa công khai
  - Phân phối khóa bí mật dựa trên khóa công khai

Quản lý và phân phối khóa

- **Mã hóa khóa công khai giúp giải quyết vấn đề phân phối khóa**
- **Có hai khía cạnh khác biệt:**
  - Phân phối khóa công khai
  - Dùng mã hóa khóa công khai để phân phối khóa bí mật

Phân phối khóa công khai

- **Có nhiều kỹ thuật được đề xuất**
- **Phân loại**
  - Công bố công khai(public announcement)
  - Thư mục công khai(publicly available directory)
  - Cấp quyền khóa công khai(public-key authority)
  - Chứng chỉ khóa công khai(public-key certificates)

Công bố công khai

- **Người dùng phân phối khóa công khai đến từng người nhận hay quảng bá rộng rãi trong cộng đồng.**
  - Ví dụ quảng bá trên trang Web cá nhân
  - Đính kèm khóa công khai trong thông điệp e-mail
- **Khuyết điểm**
  - Dễ dàng giả mạo
  - Bất kỳ ai cũng có thể tạo khóa và tuyên bố đến mọi người hay quảng bá rộng rãi cho đến khi vấn đề giả mạo bị phát hiện

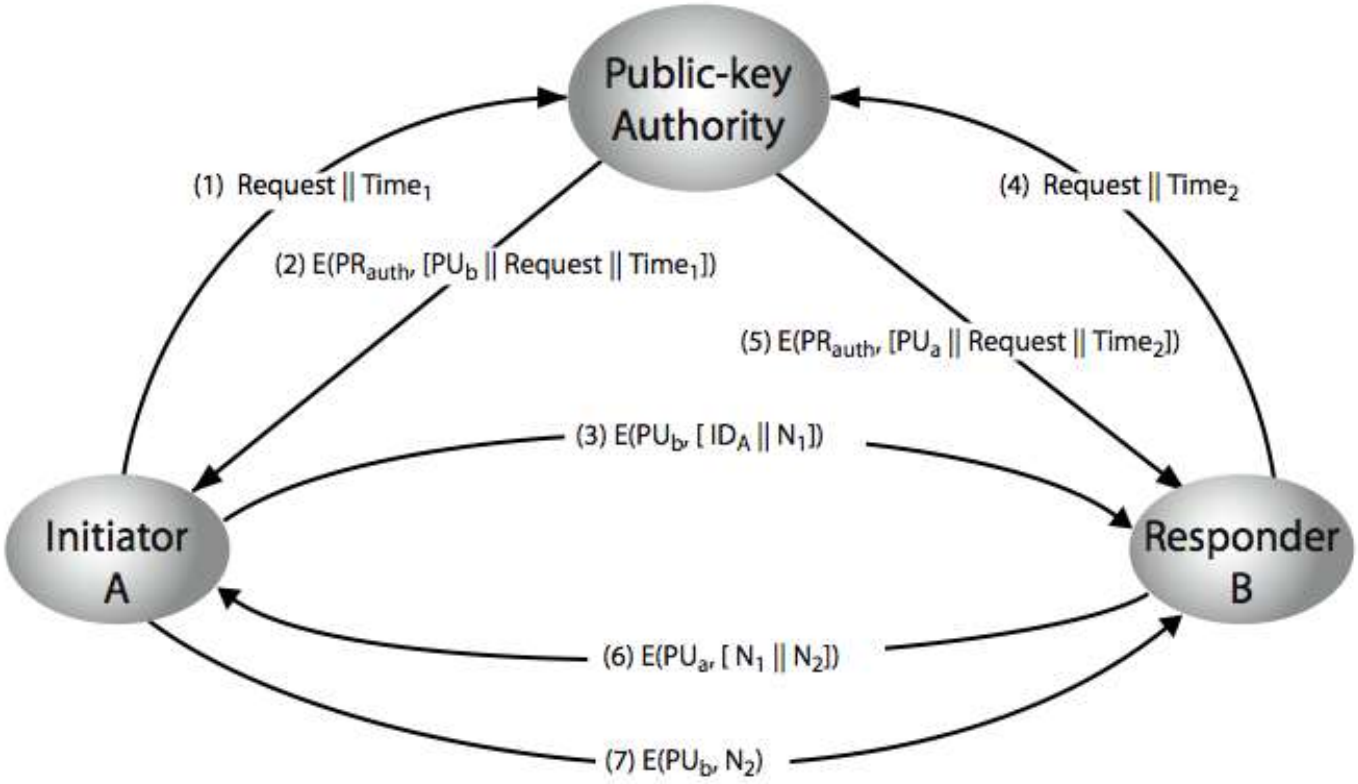
Thư mục công khai

- **Tính an toàn cao hơn bởi việc đăng ký khóa với một thư mục công khai.**
- **Thư mục phải được tin cậy với các thuộc tính:**
  - Chứa các mục {tên,khóa công khai}
  - Người tham gia đăng ký một cách an toàn với thư mục
  - Người tham gia có thể thay thế khóa công khai bất kỳ khi nào.
  - Thư mục được xuất bản định kỳ.
  - Thư mục có thể được truy cập điện tử.
- **Vẫn còn nguy cơ giả mạo.**

Cấp quyền khóa công khai

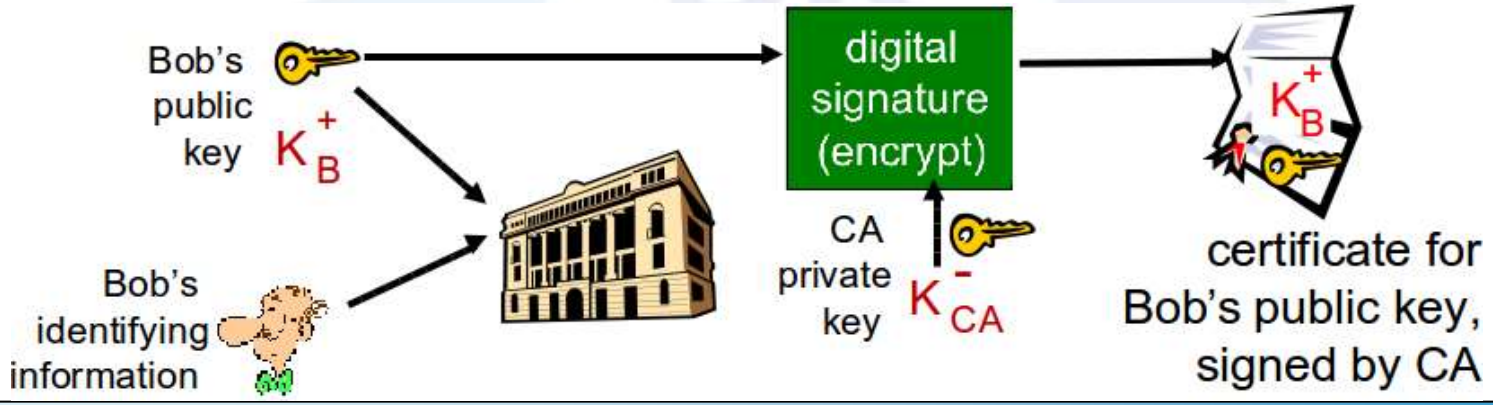
- Cải thiện an toàn.
- Có các đặc tính của thư mục.
- Yêu cầu người dùng phải biết khóa công khai cho thư mục
- Người dùng tương tác với thư mục để có được bất kỳ khóa công khai nào một cách an toàn.
- Yêu cầu truy cập thời gian thực vào thư mục.

Cấp quyền khóa công khai

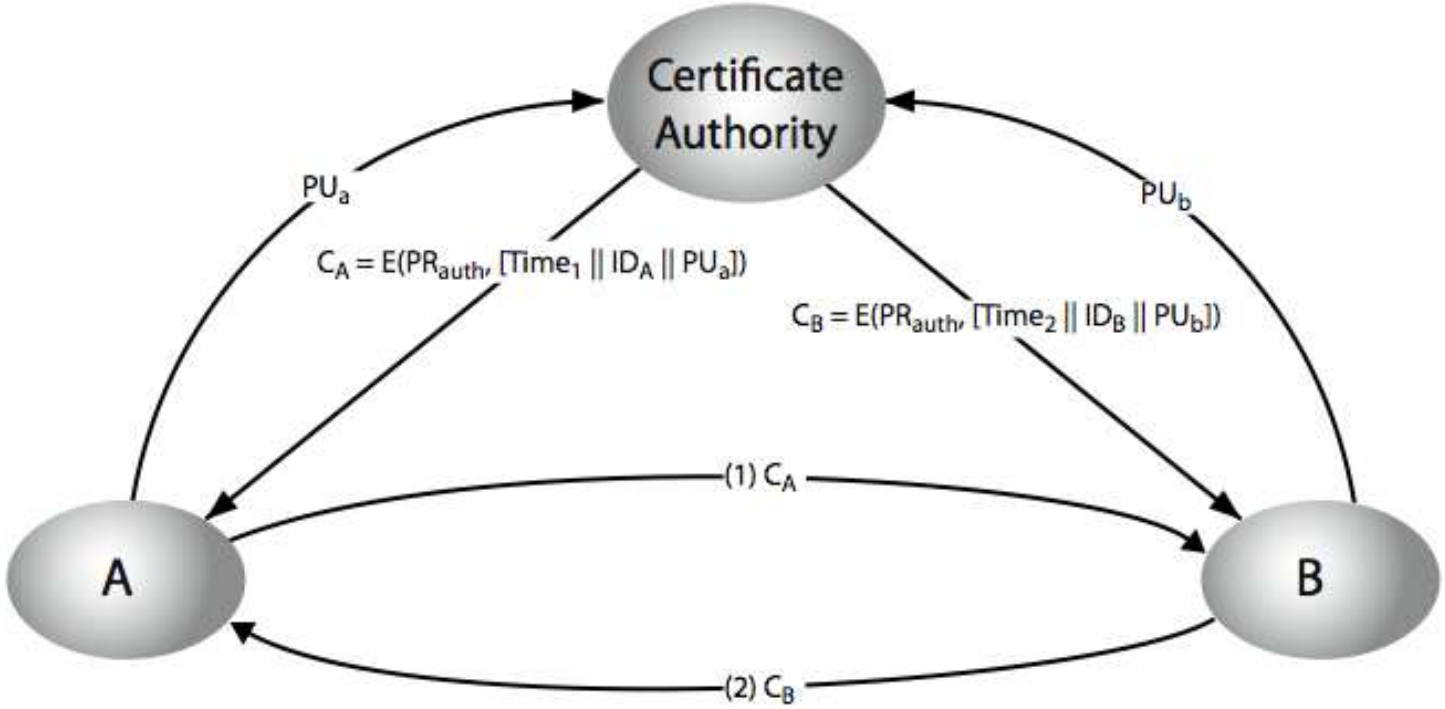


Chứng chỉ khóa công khai

- Chứng chỉ cho phép trao đổi khóa mà không cần truy cập thời gian thực
- Chứng chỉ liên kết nhận dạng với khóa công khai và có các thông tin khác như thời hạn hiệu lực, quyền sử dụng, .., và có chữ ký của một tổ chức đáng tin cậy(Certificate Authority - CA)
- Chứng chỉ có thể được kiểm chứng khi biết khóa công khai của tổ chức CA.



Chứng chỉ khóa công khai



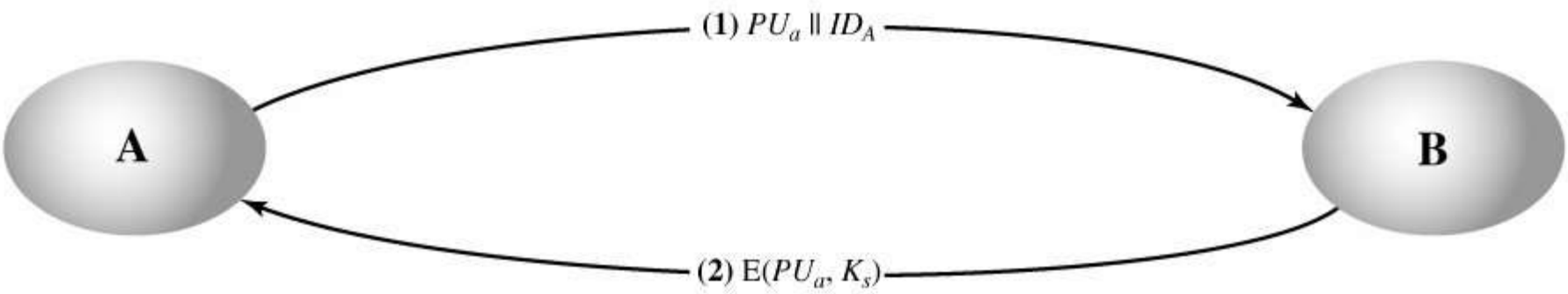
Phân phối khóa bí mật

- Sử dụng các phương pháp trước để có được khóa công khai.
- Khóa công khai có thể sử dụng cho mục đích bí mật hoặc chứng thực.
- Nhưng các thuật toán khóa công khai chậm. Do đó thường dùng mã hóa khóa bí mật để bảo vệ nội dung tin nhắn.
- Cần có một khóa phiên cho mỗi phiên làm việc.

Phân phối khóa bí mật đơn giản

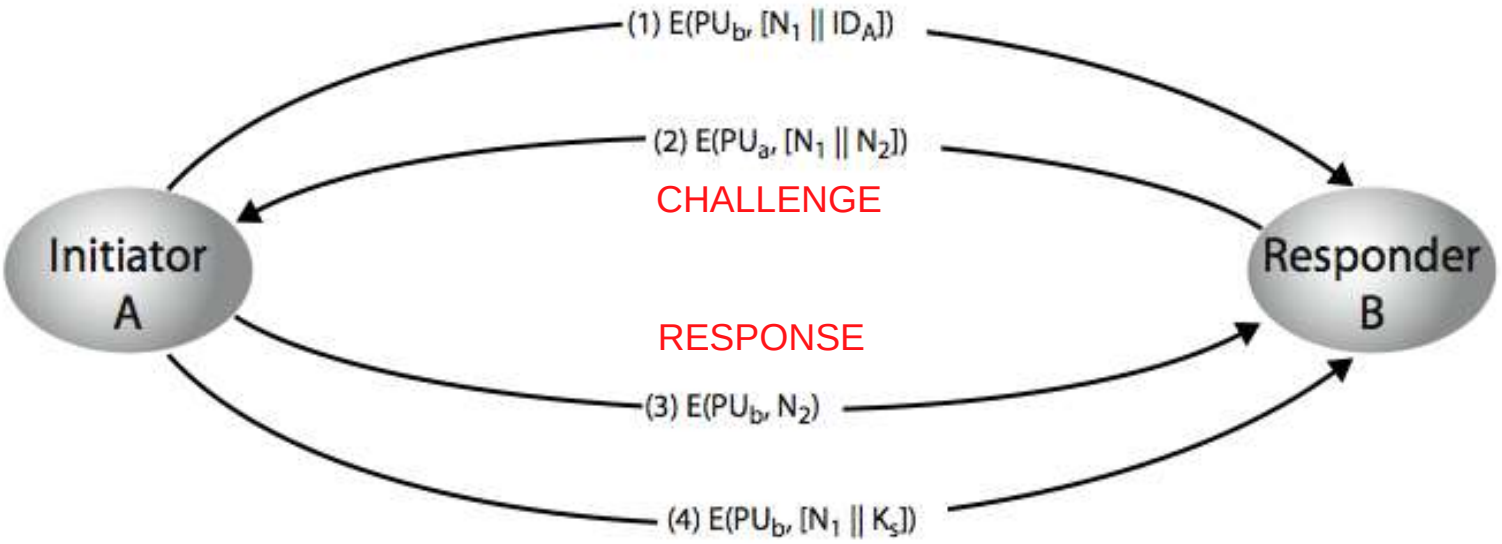
- Đề xuất của Merkle vào năm 1979
- Các thức hoạt động
  - A tạo ra một cặp khóa tạm thời
  - A gửi B khóa công khai và nhận dạng của A
  - B tạo ra một khóa phiên K
  - Gửi K đến A bằng cách mã hóa dùng khóa công khai của A.
  - A giải mã khóa phiên và được khóa phiên K.
- Vấn đề là một đối thủ có thể đánh chặn và mạo danh.
  - Tấn công "man in the middle"

Phân phối khóa bí mật đơn giản



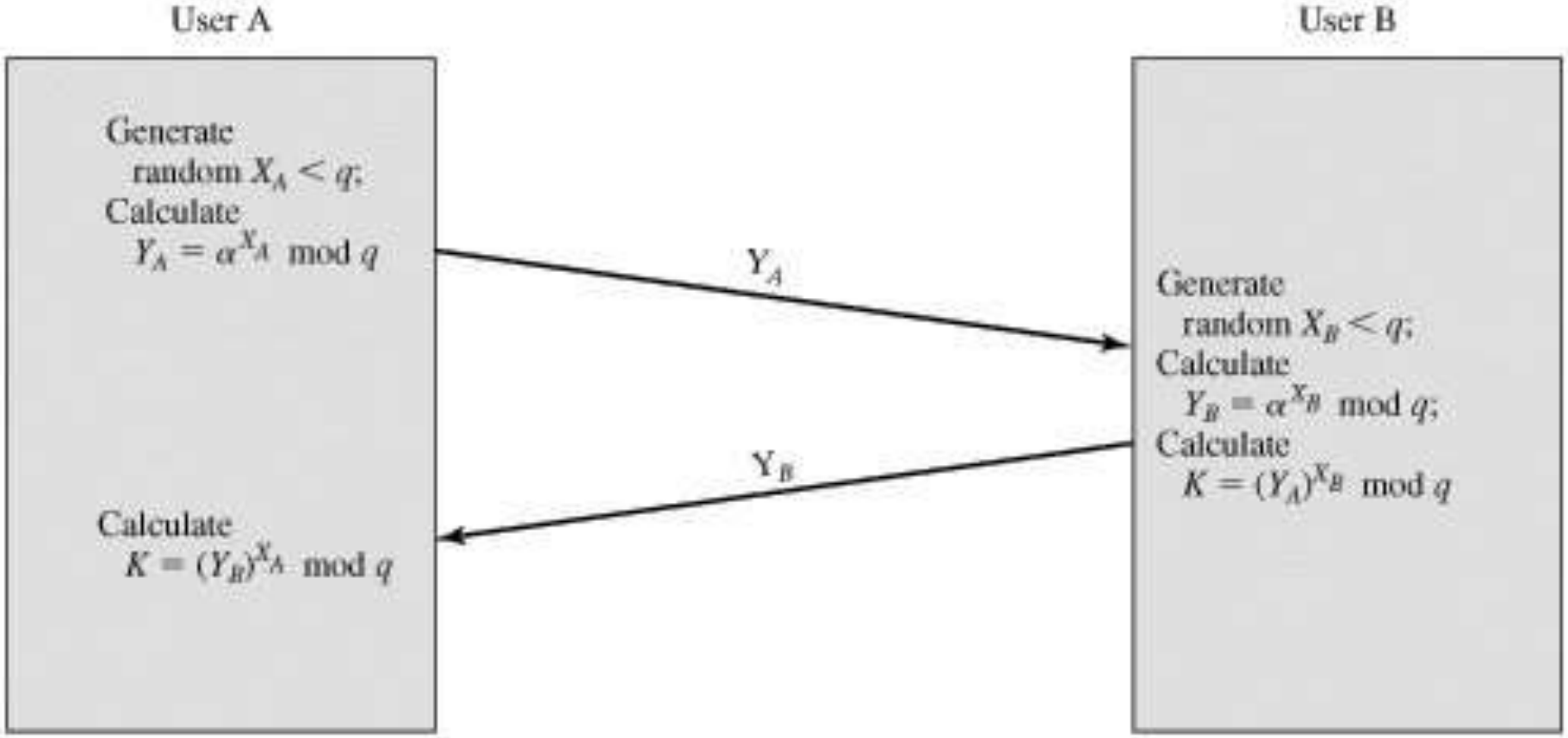
Phân phối khóa bí mật an toàn và xác thực

- Dùng **"challenge and response"**



Các giao thức trao đổi khóa(1/2)

- Dựa trên Diffe-Hellman(D-H)



Các giao thức trao đổi khóa(2/2)

- Các phương thức hoạt động
  - Người dùng tạo ngẫu nhiên các cặp khóa D-H mỗi lần chúng giao tiếp.
  - Người dùng có thể tạo các cặp khóa D-H và cập nhật lên một thư mục và tham khảo đến chúng mỗi lần giao tiếp.
- Cả hai phương thức trên đều có nguy cơ tấn công "man in the middle"
- Vị vậy xác thực các khóa là cần thiết.

Tóm tắt

- Trường hữu hạn đóng vai trò quan trọng trong mật mã. Trường hữu hạn GF(p) được định nghĩa trên mod p.
- Các hệ mã khóa công khai dựa trên các bài toán khó như phân tích thừa số nguyên tố, logarit rời rạc, …
- Hệ mã khóa công khai được sử dụng rộng rãi là RSA.
- Quản lý khóa xem xét cả phân phối khóa công khai và phân phối khóa bí mật.