Chương V

5

QUẢN LÝ VÀ PHÂN PHỐI KHÓA

(KEY MANAGEMENT AND DISTRIBUTION)

Nội dung chính

1. Symmetric-key Distribution
2. Kerberos
3. Symmetric-Key Agreement
4. Public-key distribution

(*Cryptography & Network Security*. McGraw-Hill, Inc., 2007., Chapter 15)

(Cryptography and Network Security: Principles and Practices (3rd Ed.) – Chapter 14)

1. Symmetric-key Distribution

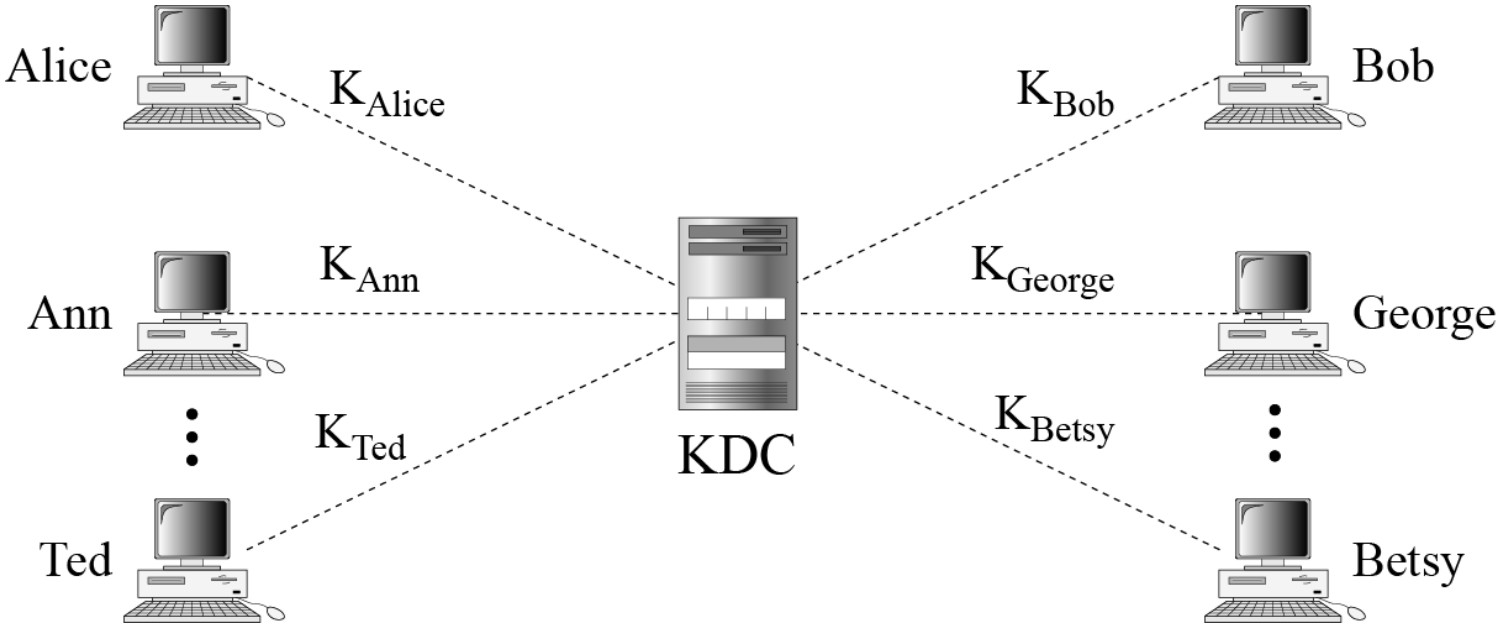
- Mã hóa khóa đối xứng hiệu quả hơn mã hóa khóa bất đối xứng đối với việc mã hóa các thông điệp lớn. Tuy nhiên mã hóa khóa đối xứng cần một khóa chia sẻ giữa hai tổ chức.
- Một người cần trao đổi thông điệp bảo mật với N người, thì người đó cần N khóa khác nhau. Vậy N người giao tiếp với N người khác thì cần tổng số là N*(N-1) khóa

→ số khóa không chỉ là vấn đề, mà phân phối khóa là một vấn đề khác.

→Độ tin cậy của một hệ thống mật mã phụ thuộc vào công nghệ phân phối khóa (*key distribution technique*).

Key-Distribution Center: KDC

- Để giảm số lượng khóa, mỗi người sẽ thiết lập một khóa bí mật chia sẻ với KDC



- Làm thế nào để Alice có thể gửi một thông điệp bảo mật tới Bob

Key-Distribution Center: KDC

- Quá trình xử lý như sau:

1. Alice gửi 1 yêu cầu đến KDC để nói rằng cô ta cần một khóa phiên (session secret key) giữa cô ta và Bob.
2. KDC thông báo với Bob về yêu cầu của Alice
3. Nếu Bob đồng ý, một session key được tạo giữa 2 bên.

- Khóa bí mật này được dùng để chứng thực Alice và Bob với KDC và ngăn chặn Eve giả mạo một trong hai.

Key-Distribution Center: KDC

Flat Multiple KDCs

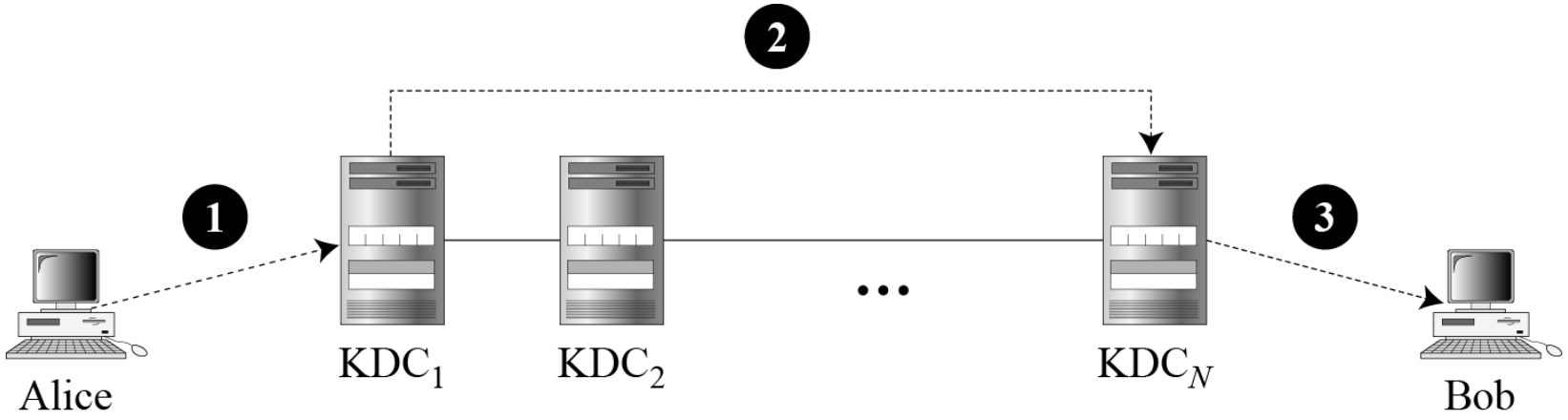
- Khi số lượng người dùng KDC tăng, hệ thống trở nên khó quản lý và một bottleneck sẽ xảy ra.

→ chúng ta có nhiều KDCs, chia thành các domain. Mỗi domain có thể có một hoặc nhiều KDCs

- Alice muốn gửi thông điệp bí mật tới Bob, mà Bob thuộc vào domain khác, thì Alice liên lạc với KDC của cô ta, KDC của Alice tiếp tục liên lạc với KDC trong domain của Bob.
- Hai KDCs như vậy thì được gọi là Flat multiple KDCs

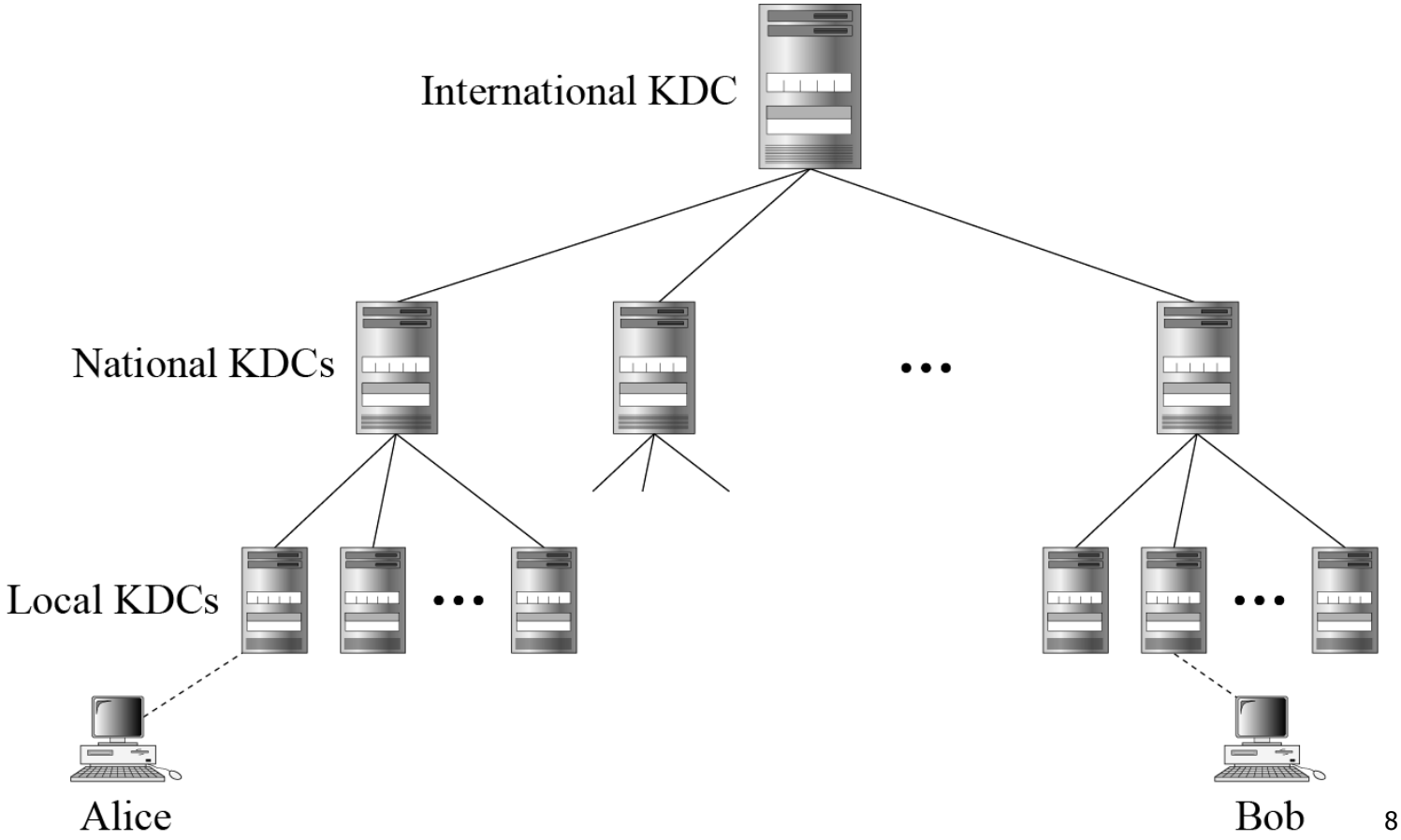
Key-Distribution Center: KDC

Flat Multiple KDCs



Key-Distribution Center: KDC

- **Hierarchical Multiple KDCs**

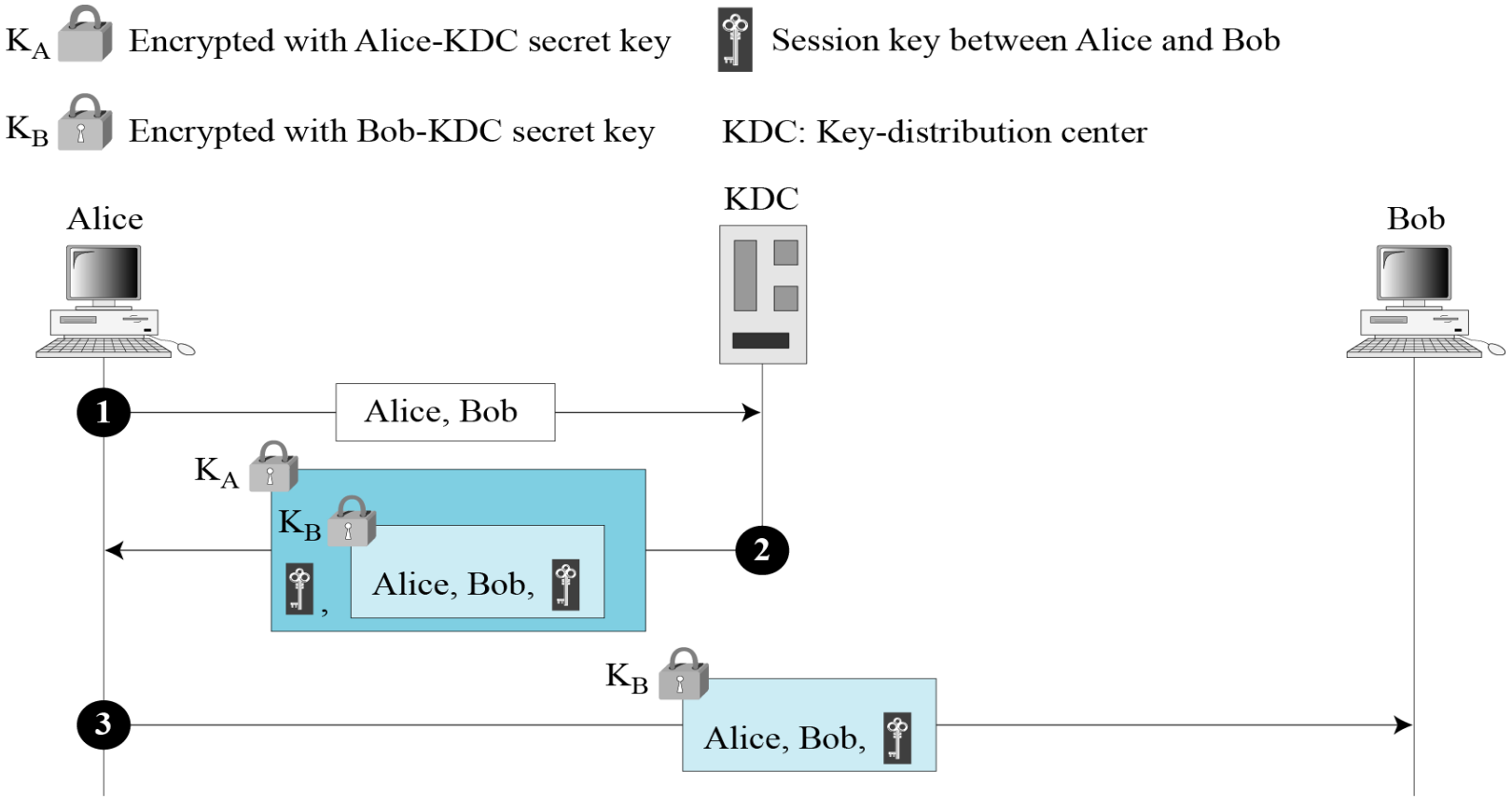


Khóa phiên (Session Keys)

- KDC tạo khóa bí mật cho mỗi thành viên, khóa bí mật này chỉ có thể dùng giữa thành viên và KDC, chứ không dùng giữa hai thành viên
- Nếu muốn dùng giữa hai thành viên, KDC tạo một ***session key*** giữa hai thành viên, sử dụng khóa của họ với trung tâm.
- ***Khóa phiên giữa hai thanh viên chỉ được dùng một lần*** (sau giao tiếp kết thúc thì khóa phiên cũng không còn tác dụng)

Khóa phiên (Session Keys)

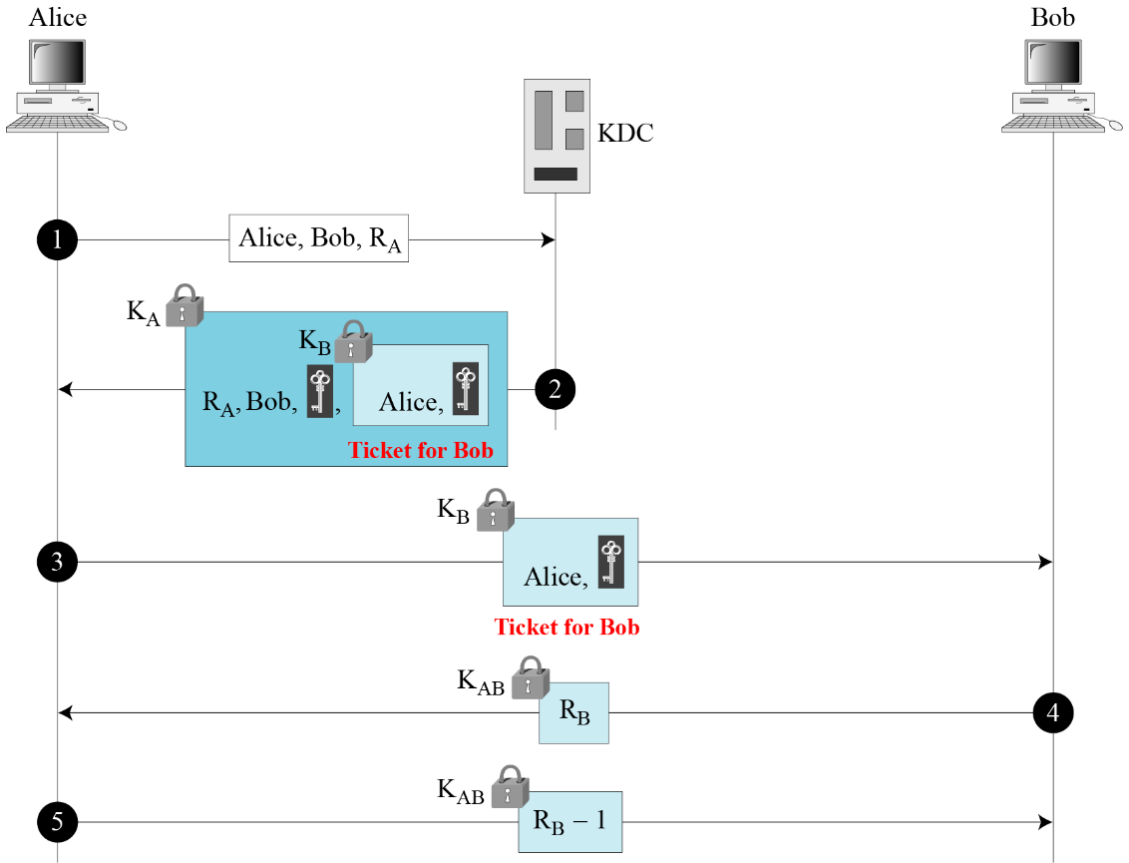
- Một giao thức đơn giản sử dụng một KDC



- Giao thức này có thể bị tấn công phát lại ở bước 3

Khóa phiên (Session Keys)

Giao thức Needham-Schroeder (nền tảng của nhiều giao thức khác)



K_A Encrypted with Alice-KDC secret key

K_B Encrypted with Bob-KDC secret key

K_{AB} Encrypted with Alice-Bob session key

Session key between Alice and Bob

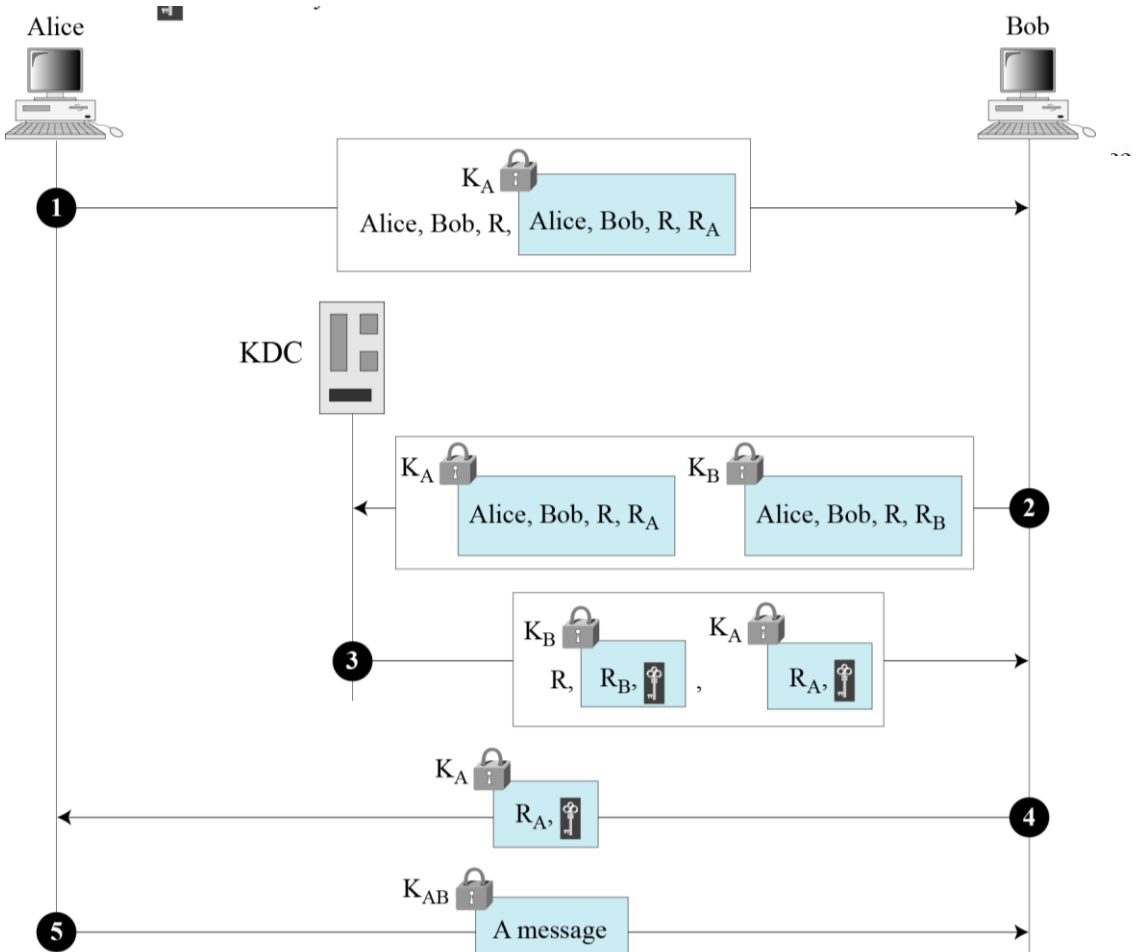
KDC: Key-distribution center

R_A: Alice's nonce

R_B: Bob's nonce

Khóa phiên (Session Keys)

- Giao thức Otway-Rees



K_A Encrypted with Alice-KDC secret key

K_B Encrypted with Bob-KDC secret key

K_{AB} Encrypted with Alice-Bob session key

Session key between Alice and Bob

KDC: Key-distribution center

R_A: Nonce from Alice to KDC

R_B: Nonce from Bob to KDC

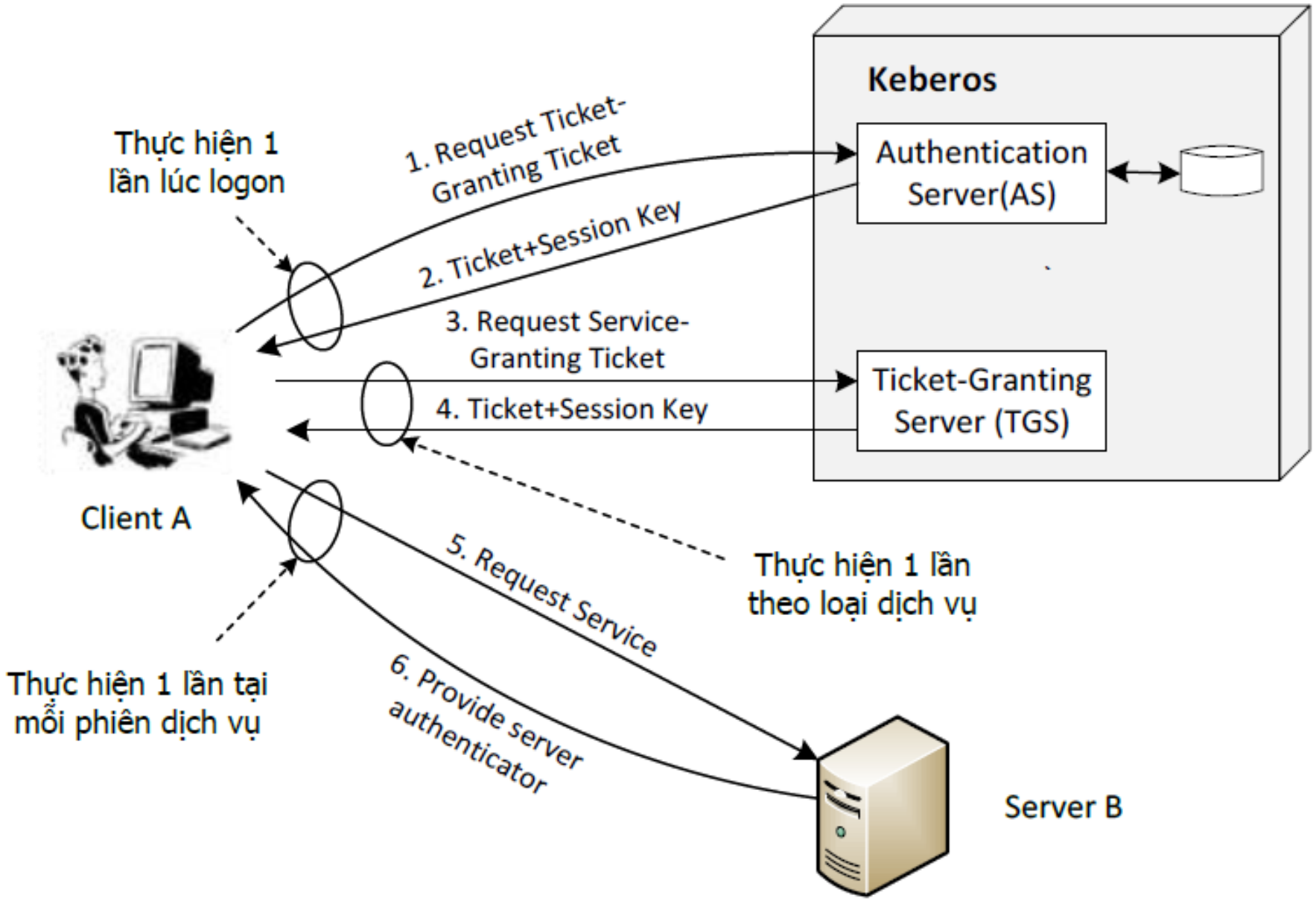
R: Common nonce

2. KERBEROS

- Kerberos là tên của một hệ dịch vụ phân phối (hay cấp phát) khóa phiên (session) cho từng phiên truyền tin bảo mật theo yêu cầu của người dùng trong một mạng truyền tin
- Kerberos là một giao thức chứng thực. Keberos chỉ dựa trên mã hóa đối xứng
- Ra đời cùng thời điểm với KDC, nhưng đã trở nên thông dụng. (Windows 2000 sử dụng cơ chế Kerberos để chứng thực)
- Đầu tiên được thiết kế tại MIT, nó đã qua nhiều phiên bản khác nhau

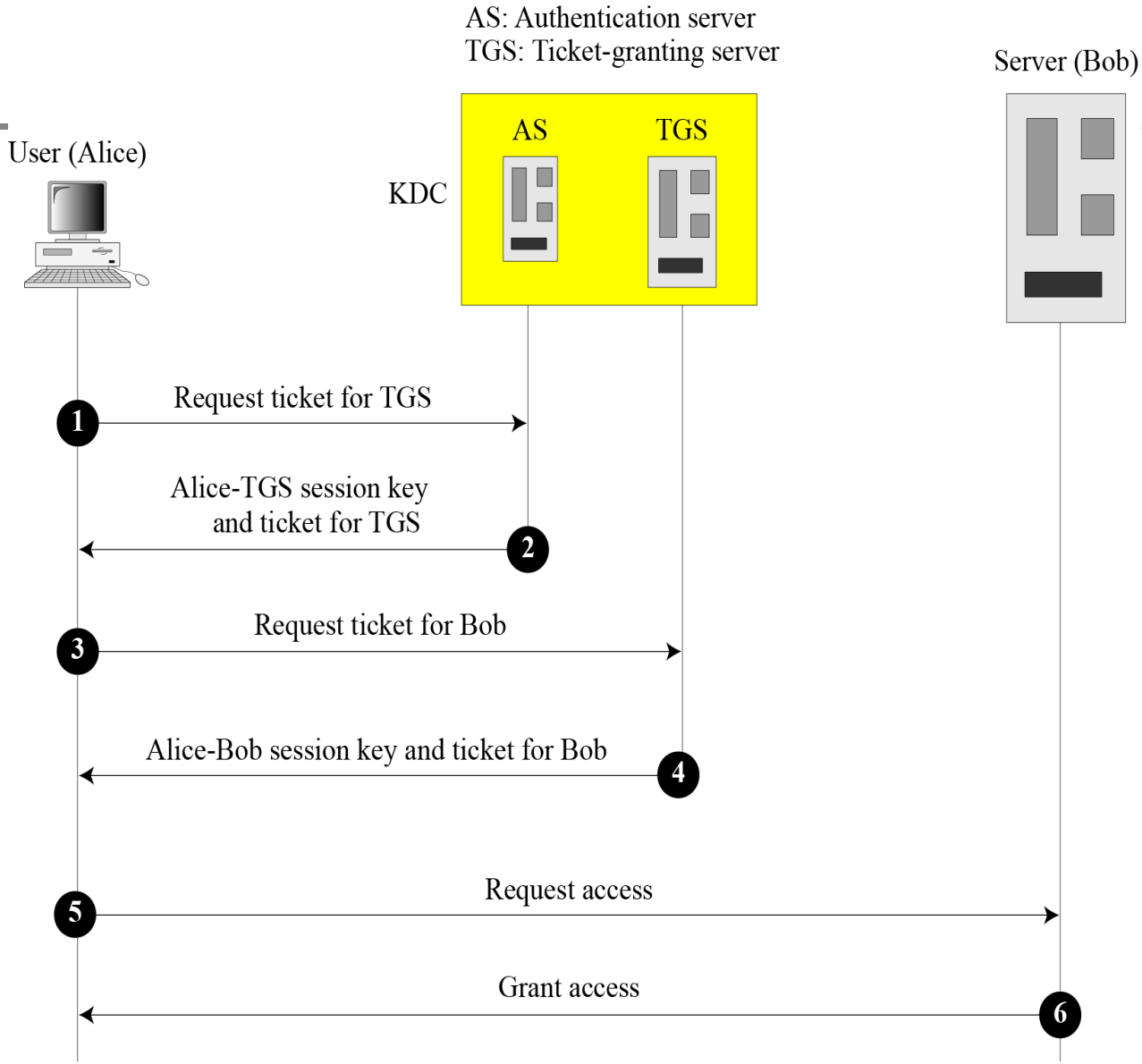
2. KERBEROS

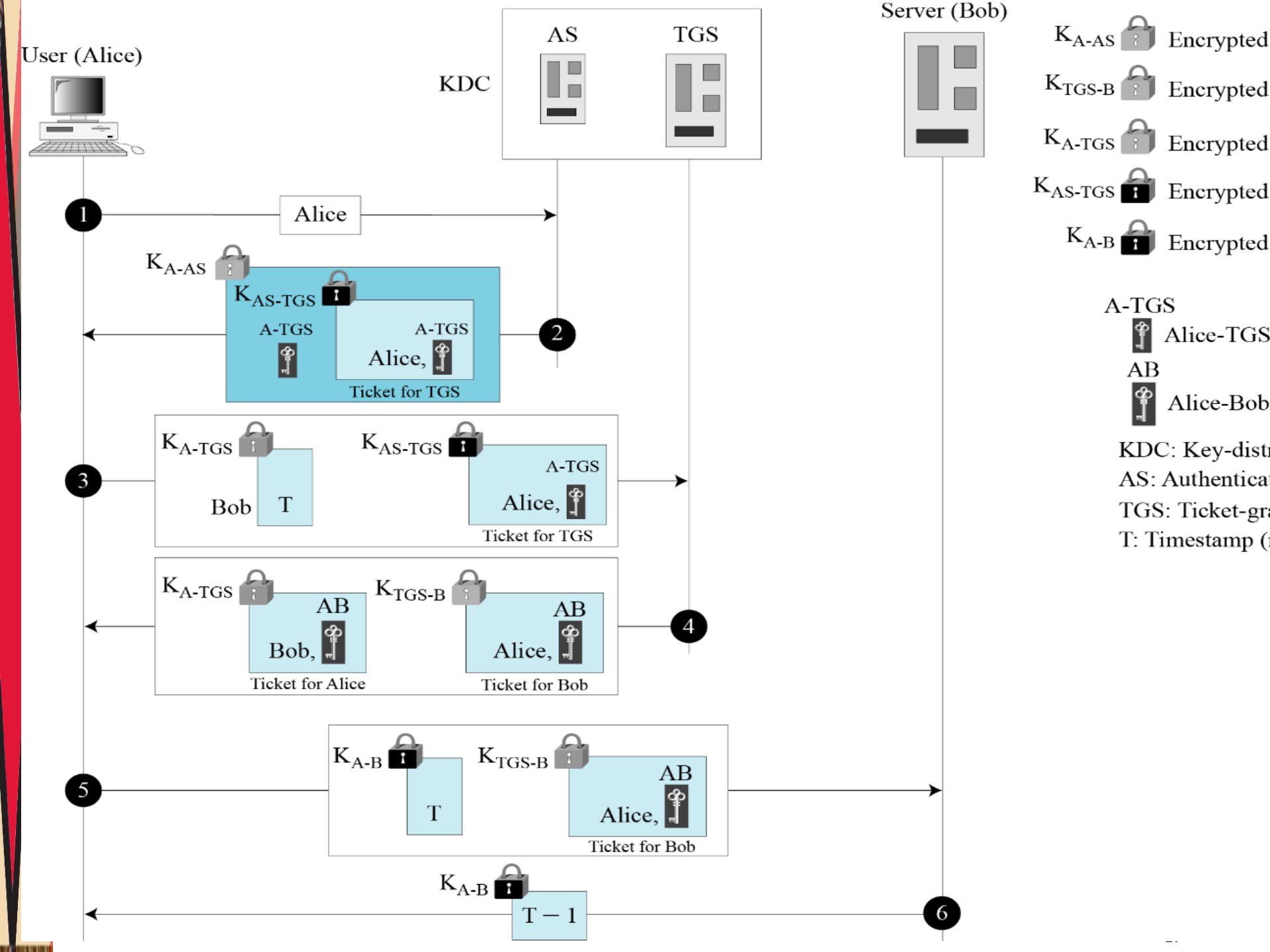
- Mục đích của Keberos là để trao đổi khóa phiên, thông qua đó đảm bảo tính bảo mật và tính chứng thực.
- Do nguyên tắc của Keberos dựa trên KDC nên Keberos cũng kế thừa được những ưu điểm của mô hình KDC như tính phi trạng thái

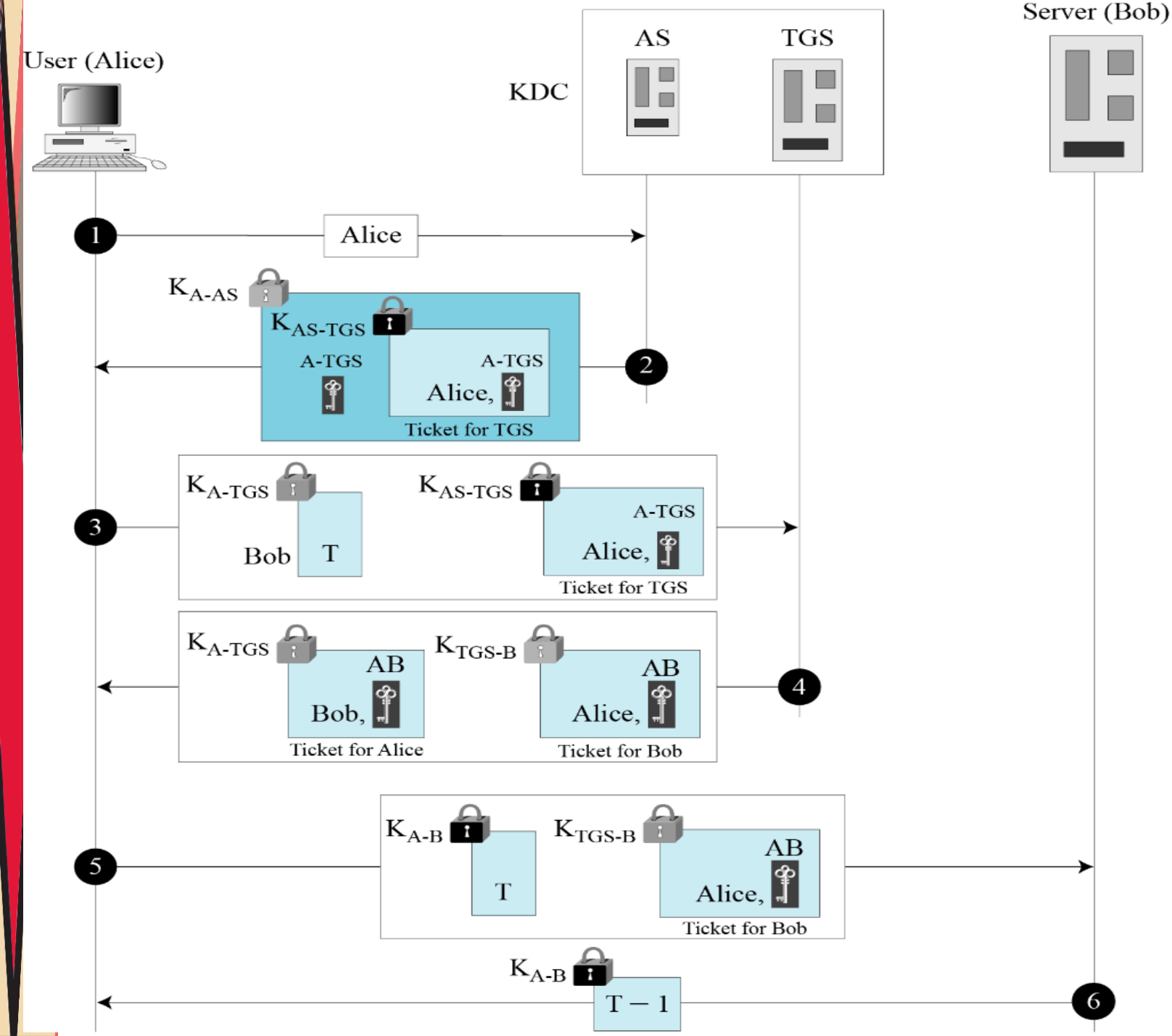


Hình 7-9. Mô hình chứng thực và trao đổi khóa phiên Keberos

Servers


AS: Authentication server
TGS: Ticket-granting server
Server (Bob)
User (Alice)
KDC
AS
TGS
Request ticket for TGS
1
Alice-TGS session key
and ticket for TGS
2
Request ticket for Bob
3
Alice-Bob session key and ticket for Bob
4
Request access
5
Grant access
6


User (Alice)
KDC
AS
TGS
Server (Bob)
K_{A-AS} Encrypted
K_{TGS-B} Encrypted
K_{A-TGS} Encrypted
K_{AS-TGS} Encrypted
K_{A-B} Encrypted
A-TGS
Alice-TGS
AB
Alice-Bob
KDC: Key-distr
AS: Authentica
TGS: Ticket-gra
T: Timestamp (
1
Alice
K_{A-AS}
K_{AS-TGS}
A-TGS
A-TGS
Alice,
Ticket for TGS
2
3
K_{A-TGS}
Bob
T
K_{AS-TGS}
A-TGS
Alice,
Ticket for TGS
4
K_{A-TGS}
AB
Bob,
Ticket for Alice
K_{TGS-B}
AB
Alice,
Ticket for Bob
5
K_{A-B}
T
K_{TGS-B}
AB
Alice,
Ticket for Bob
K_{A-B}
T − 1
6


User (Alice)
KDC
AS
TGS
Server (Bob)
1
Alice
K_{A-AS}
K_{AS-TGS}
A-TGS
A-TGS
Alice,
Ticket for TGS
2
K_{A-TGS}
Bob
T
K_{AS-TGS}
A-TGS
Alice,
Ticket for TGS
3
K_{A-TGS}
AB
Bob,
Ticket for Alice
K_{TGS-B}
AB
Alice,
Ticket for Bob
4
K_{A-B}
T
K_{TGS-B}
AB
Alice,
Ticket for Bob
5
K_{A-B}
T − 1
6
K_{A-AS} Enc
K_{TGS-B} Enc
K_{A-TGS} Enc
K_{AS-TGS} Enc
K_{A-B} Enc
A-TGS
Alic
AB
Ali
KDC: K
AS: Aut
TGS: Ti
T: Times

2. KERBEROS

Trong giao thức Kerberos gồm có:

- Servers
- Operation
- Using Different Servers
- Kerberos Version 5
- Realms

Servers

- Authentication Server (chỉ có 1 AS): là KDC trong giao thức Kerberos. AS có nhiệm vụ cung cấp khóa đối xứng cho trao đổi giữa client A và server TGS
- Ticket-granting server (TGS): đóng vai trò là các KDC, có nhiệm vụ cung cấp khóa đối xứng cho trao đổi giữa client A và server dịch vụ B
- Các người sử dụng A cần đăng ký mật khẩu *K*A của mình với Server AS. Các server dịch vụ B đăng ký khóa bí mật *K*B với Server TGS. Server TGS cũng đăng ký khóa bí mật *K*TGS với Server AS
- Real (data) server (của Bob): cung cấp dịch vụ cho người dùng (Alice)

Using Different Servers

Nếu Alice cần nhận các dịch vụ từ các servers khác, cô ta chỉ cần lặp lại 4 bước sau cùng.

Realms (lãnh địa)

- Kerberos cho phép sự phân bố toàn cục của các AS và TGS, với mỗi hệ thống được gọi là một realm. Người dùng có thể lấy một ticket cho một local server hoặc remote server.

3. Symmetric-Key Agreement

- Alice và Bob có thể tạo ra một session key giữa chúng mà không cần dùng một KDC. Phương pháp tạo session-key này được tham chiếu như một symmetric-key agreement.
- Hai phương pháp
  - Diffie-Hellman Key Agreement
  - Station-to-Station Key Agreement

3.4. Mô hình trao đổi khóa Diffie-Hellman

- Trao đổi khoá Diffie Hellman là sơ đồ khoá công khai đầu tiên được đề xuất bởi Diffie và Hellman năm 1976 cùng với khái niệm khoá công khai.
- Sau này được biết đến bởi James Ellis (Anh), người đã đưa ra mô hình tương tự năm 1970. Đây là phương pháp thực tế trao đổi công khai các khoá mật. Nó thúc đẩy việc nghiên cứu đề xuất các mã khoá công khai. Sơ đồ được sử dụng trong nhiều sản phẩm thương mại.

3.4. Mô hình trao đổi khóa Diffie-Hellman

- Không thể dùng để trao đổi mẩu tin bất kỳ.
- Tuy nhiên nó có thể thiết lập khoá chung.
- Chỉ có hai đối tác biết đến.
- Giá trị khoá phụ thuộc vào các đối tác (và các thông tin về khoá công khai và khoá riêng của họ).
- Dựa trên phép toán lũy thừa trong trường hữu hạn (modulo theo số nguyên tố hoặc đa thức) là bài toán dễ.
- Độ an toàn dựa trên độ khó của bài toán tính logarit rời rạc (giống bài toán phân tích ra thừa số) là bài toán khó.

3.4. Mô hình trao đổi khóa Diffie-Hellman

Giao thức trao đổi khoá giữa A và B:

- A và B thống nhất chọn chung một số nguyên tố q và một phần tử sinh α.

Global Public Elements	
q	prime number
α	$\alpha < q$ *and* α a primitive root of q

3.4. Mô hình trao đổi khóa Diffie-Hellman

Tạo cặp khóa:

User A Key Generation	
Select private X_A	$X_A < q$
Calculate public Y_A	$Y_A = \alpha^{X_A} \bmod q$

User B Key Generation	
Select private X_B	$X_B < q$
Calculate public Y_B	$Y_B = \alpha^{X_B} \bmod q$

3.4. Mô hình trao đổi khóa Diffie-Hellman

- Xác định khóa phiên: Dựa vào sốhọc modulo

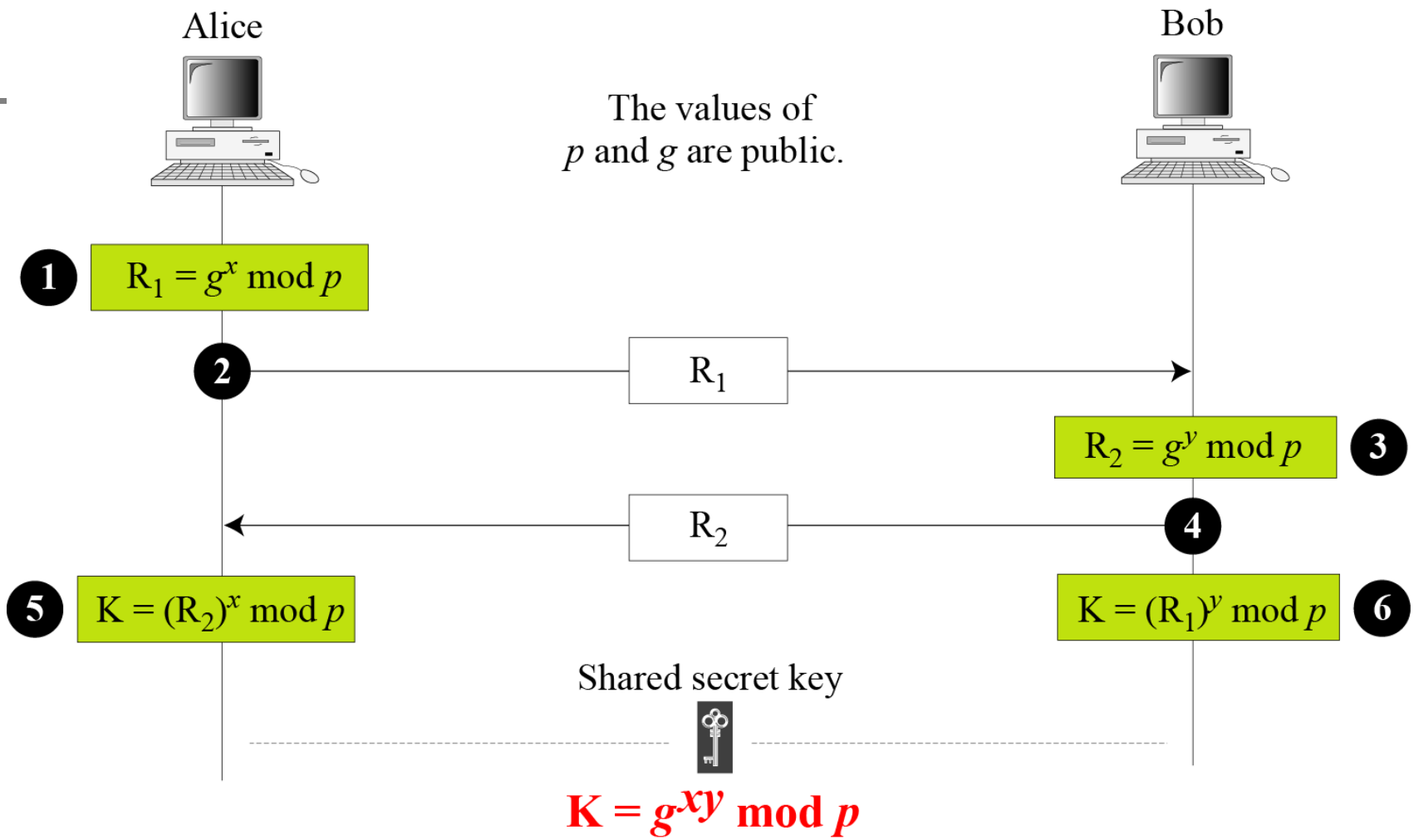
Calculation of Secret Key by User A

$$K = (Y_B)^{X_A} \bmod q$$

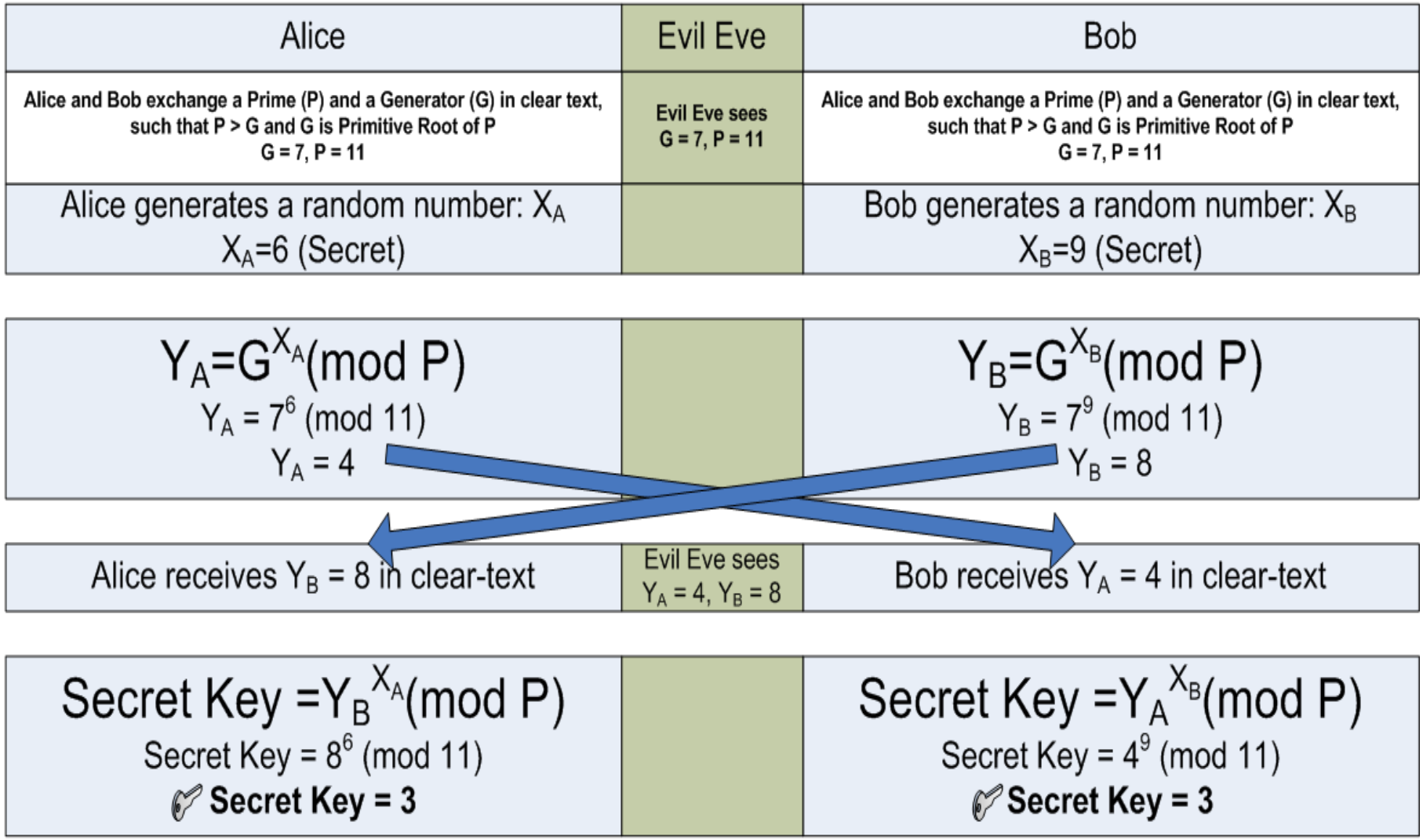
Calculation of Secret Key by User B

$$K = (Y_A)^{X_B} \bmod q$$

3.1 Diffie-Hellman Key Agreement



Diffie Hellman Key Exchange



3.1 Diffie-Hellman Key Agreement

- Ví dụ: Giả sử rằng $g = 7$ và $p = 23$. Các bước như sau:

3.1 Diffie-Hellman Key Agreement

- Bài tập: Cho số nguyên tố q=353 và α=3
- Chọn các khoá mật ngẫu nhiên: A chọn xA=97, B chọn xB=233. Tính khóa công khai và khóa phiên. Sau đó cho nhận xét

3.1 Diffie-Hellman Key Agreement

Bài tập: 1. Alice và Bob thống nhất với nhau chọn số nguyên tố p = 37 và g = 5.

Alice chọn một giá trị ngẫu nhiên bất kỳ $a_A = 7$ và bí mật a_A.
Bob chọn một giá trị ngẫu nhiên bất kỳ $a_B = 5$ và bí mật a_B

3.1 Diffie-Hellman Key Agreement

- Ví dụ 2: Chúng ta dùng một chương trình tạo một số nguyên ngẫu nhiên 521-bit (khoảng 159 chữ số). Chúng ta cũng chọn g, x, và y như sau.

p	7646242985634935721824937659550305074763380967269497489235737728609252356666607554236374233096611800333381061947301309504147387009991780436548785807987581
g	2
x	557
y	273

3.1 Diffie-Hellman Key Agreement

- Giá trị của R_1, R_2, và K là:

R_1	84492028420566550521617294749103509414343369852001266086286363106767361995928082858670080213185929094514021750031997331294583608382194306596602015795535 4
R_2	43526283870920037947074711489558162763638911626211555797512337921856631001143571820839004018187648684175383116534269163026342110672150858962552012885941 43
K	15563800066452229059622582752327076527321804694442367852032040014640650088793665120425742677660832791101715303867456125221315161097658420012040864336177 40

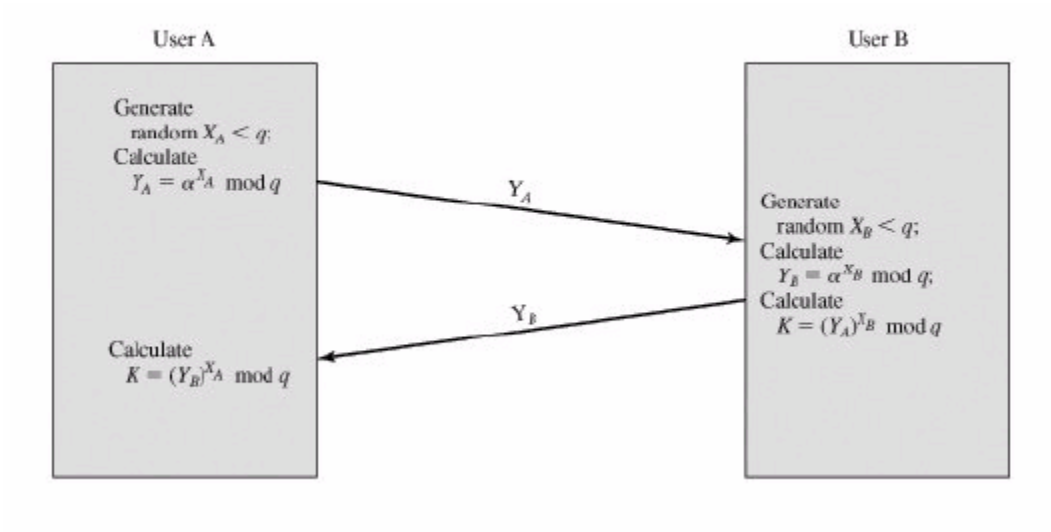
3.1 Diffie-Hellman Key Agreement

Tấn công

q = 353; a = 3; Y_A = 40; Y_B= 248

Vét cạn để tìm 2 khóa bằng nhau (với số nhỏ)

Không chống được tấn công Man-in-the-Middle



3.1 Diffie-Hellman Key Agreement

Bảo mật của Diffie-Hellman:

1.Discrete Logarithm Attack

2.Man-in-the-Middle Attack

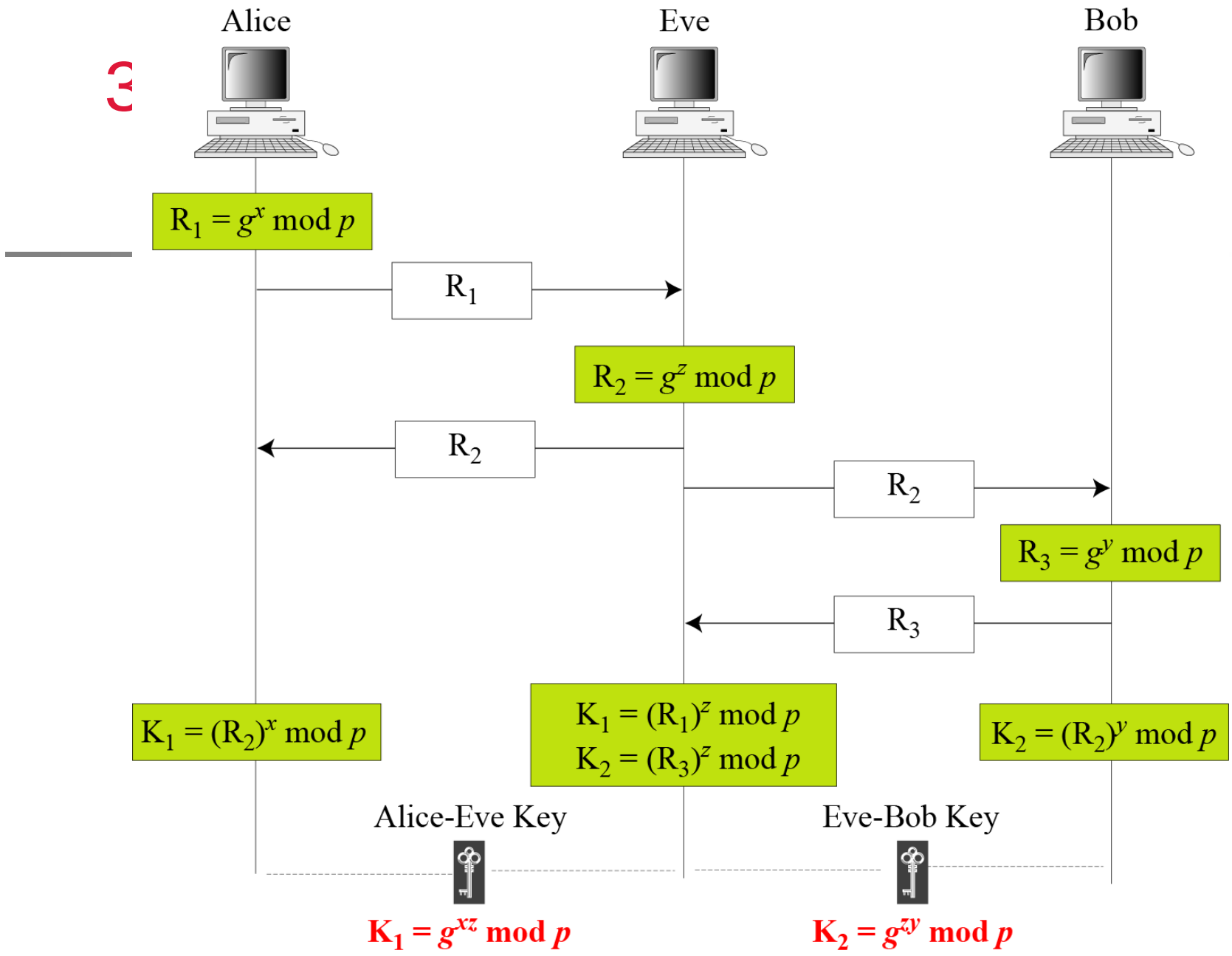
3.1 Diffie-Hellman Key Agreement

Discrete Logarithm Attack

Eve chặn R_1 và R_2. Nếu cô ta tìm ra được x từ $R_1=g^x \bmod p$ và y từ $R_2=g^y \bmod p$ → có thể tính toán được khóa $K=g^{xy} \bmod p$ → Khóa bí mật này không còn bí mật nữa.

Để an toàn, ta nên chọn:

- Số nguyên tố p phải là rất lớn (hơn 300 chữ số)
- Số p phải được chọn sao cho p-1 có ít nhất một thừa số nguyên tố lớn (nhiều hơn 60 chữ số)
- Phần tử sinh phải được chọn từ nhóm <Zp*,x>
- Bob và Alice phải hủy x và y sau khi tính K; x và y chỉ nên sử dụng một lần

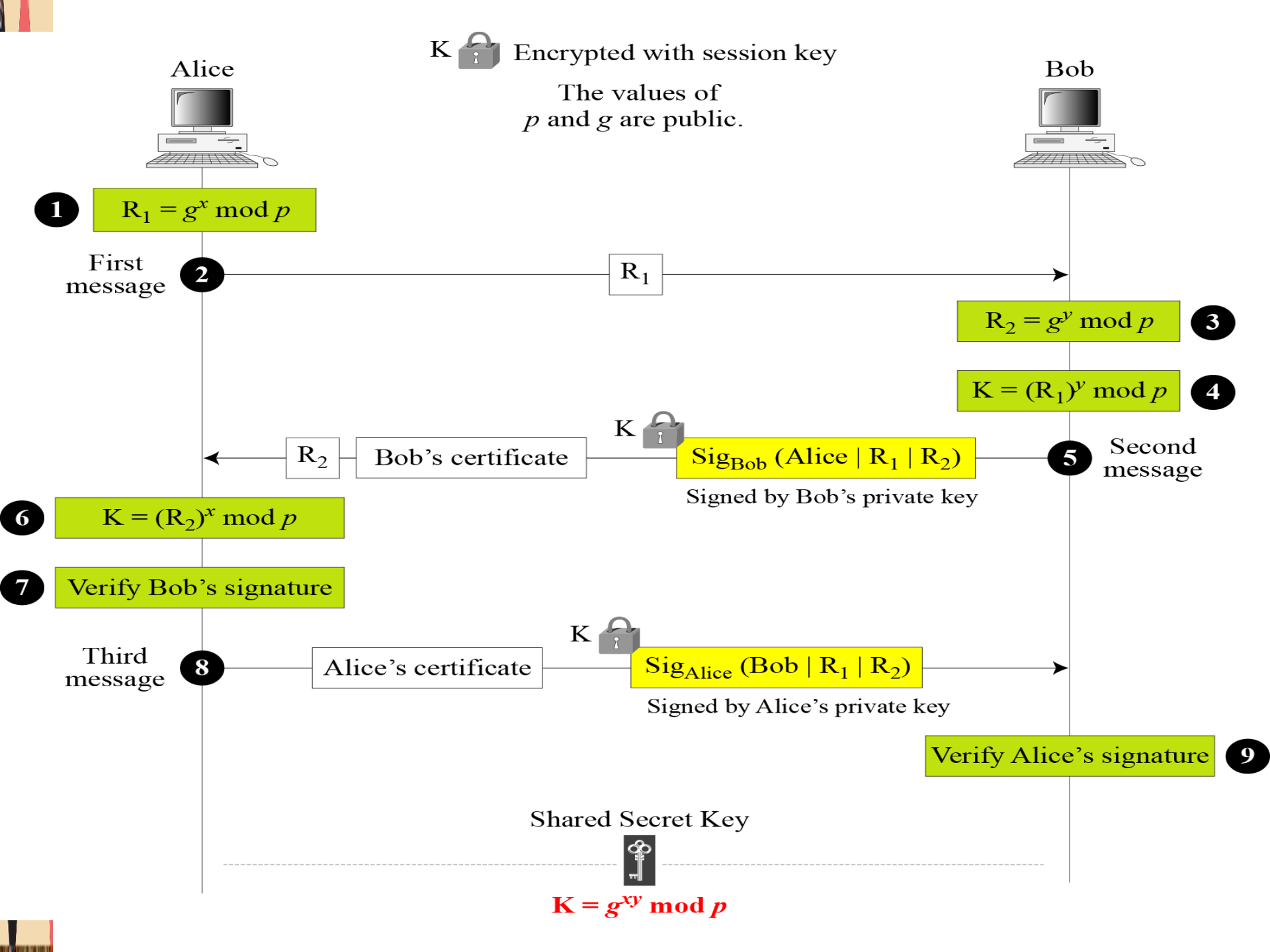

Alice
Eve
Bob
$R_1 = g^x \bmod p$
R_1
$R_2 = g^z \bmod p$
R_2
R_2
$R_3 = g^y \bmod p$
R_3
$K_1 = (R_2)^x \bmod p$
$K_1 = (R_1)^z \bmod p$
$K_2 = (R_3)^z \bmod p$
$K_2 = (R_2)^y \bmod p$
Alice-Eve Key
Eve-Bob Key
$K_1 = g^{xz} \bmod p$
$K_2 = g^{zy} \bmod p$

3.1 Diffie-Hellman Key Agreement

- Giao thức là an toàn đối với việc tấn công thụ động, nghĩa là một người thứ ba dù biết b_A và b_B sẽ khó mà biết được $K_{A,B}$.

3.2 Station-to-Station Key Agreement

- Là một giao thức dựa trên Diffie-Hellman
- Dùng Digtal signature với Public-key Certificates để thiết lập nên session key giữa Alice và Bob


K Encrypted with session key
The values of p and g are public.
Alice
Bob
1
$R_1 = g^x \bmod p$
First message
2
R_1
$R_2 = g^y \bmod p$
3
$K = (R_1)^y \bmod p$
4
K
R_2
Bob's certificate
Sig_{Bob} (Alice | R_1 | R_2)
5
Second message
Signed by Bob's private key
6
$K = (R_2)^x \bmod p$
7
Verify Bob's signature
K
Third message
8
Alice's certificate
Sig_{Alice} (Bob | R_1 | R_2)
Signed by Alice's private key
Verify Alice's signature
9
Shared Secret Key
$K = g^{xy} \bmod p$

3.2 Station-to-Station Key Agreement

- Giao thức này **ngăn chặn được tấn công man-in-the-middle.**
  - Sau khi chặn R_1, Eve không thể gửi R_2 của cô ta cho Alice và giả bộ nó được gửi đến từ Bob bởi vì Eve không thể giả mạo được Private key của Bob để tạo ra Sinature – Signature không thể được thẩm tra bằng public key của Bob được xác định trong Certificate.
  - Cùng cách tương tự Eve không thể giả private key của Alice để ký thông điệp thứ 3 gửi bởi Alice.

4. Hạ tầng khóa công khai (PKI)

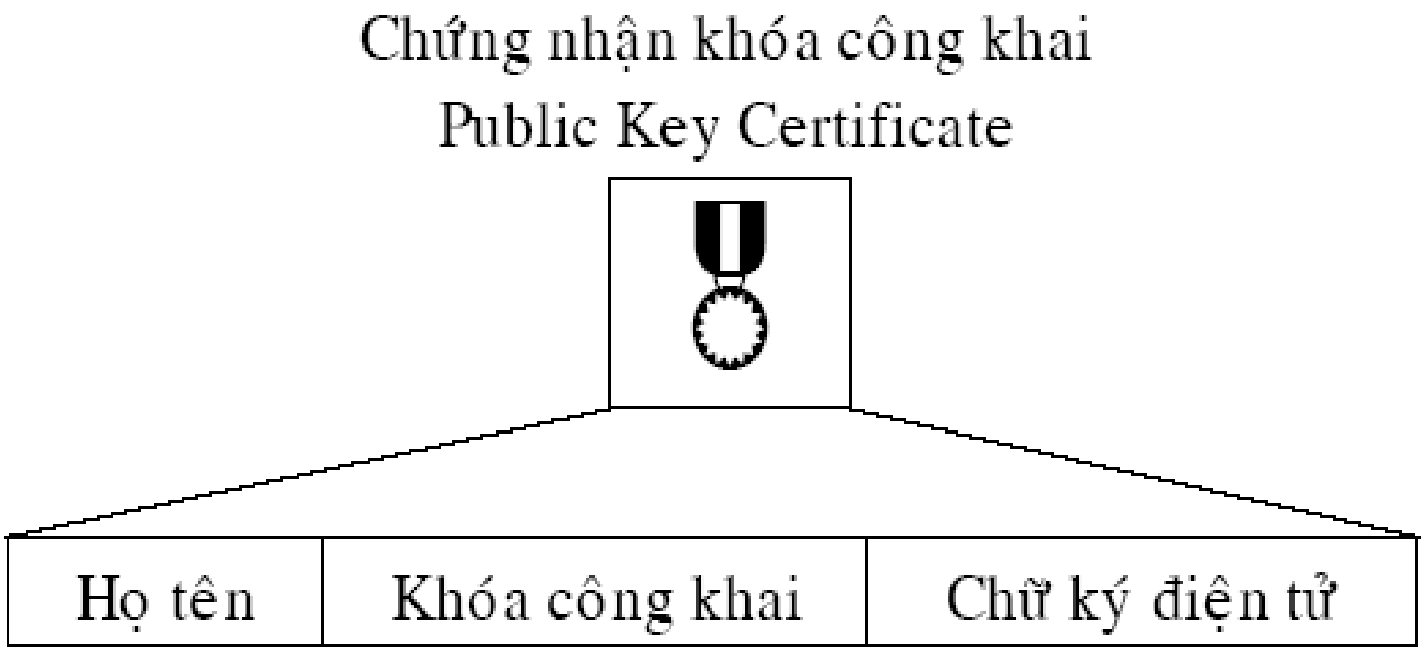
- "*PKI là tập hợp của các công nghệ mật mã, phần mềm, phần cứng chuyên dụng và các dịch vụ cho phép các tổ chức/doanh nghiệp đảm bảo an toàn thông tin liên lạc, định danh và xác thực được người dùng, khách hàng trên các giao dịch qua mạng/Internet*".

<DigiCrypto>

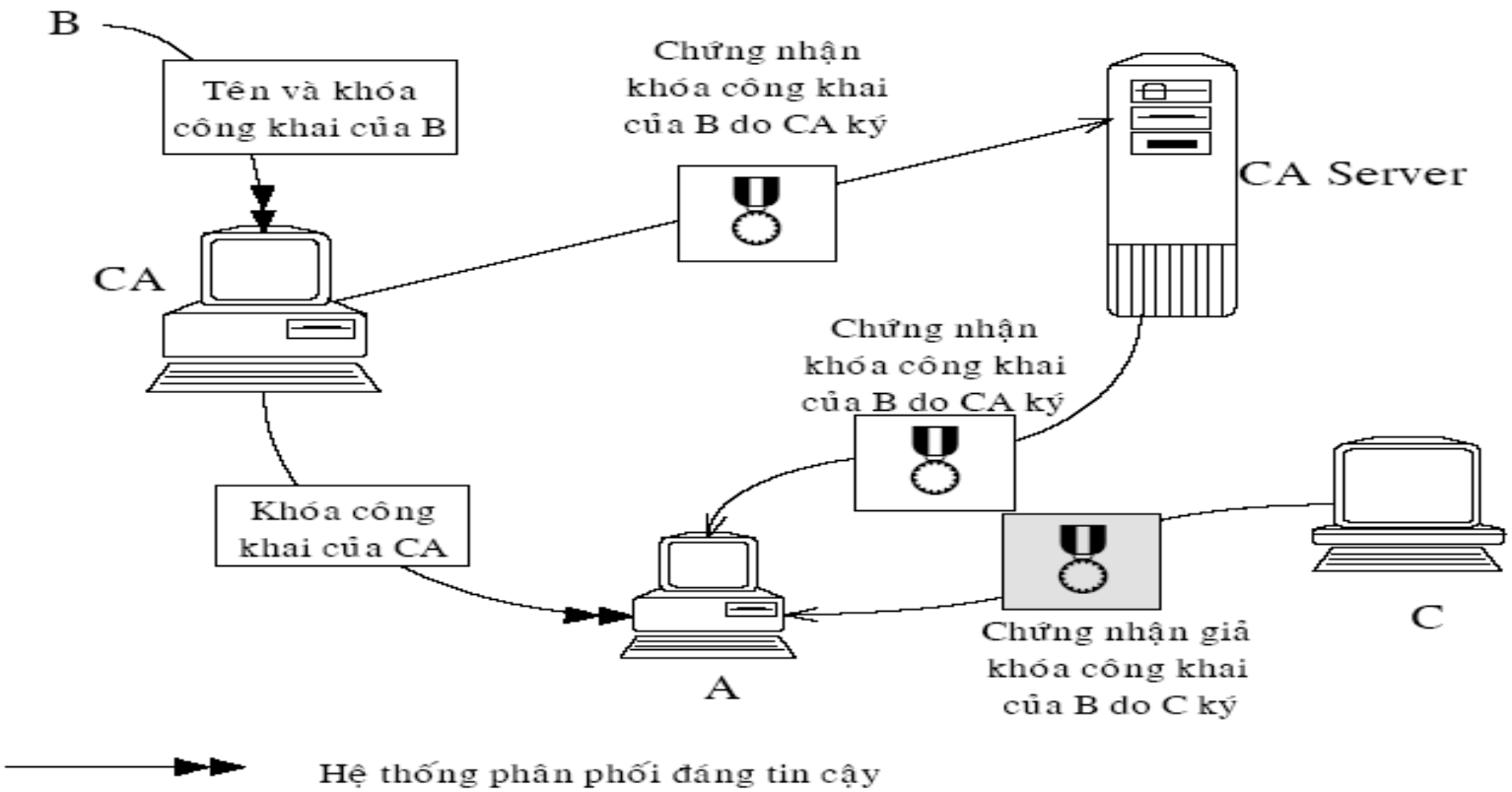
Các thành phần

1. Chứng chỉ khóa công khai: họ tên hoặc định danh của người sở hữu thật sự của khóa, khóa công cộng và chữ ký điện tử giúp xác nhận được tính hợp lệ của hai thành phần này.
2. Hệ thống phân phối khóa tin cậy: sử dụng hệ thống trao đổi thông tin tin cậy để chuyển mã khóa công cộng đến người nhận.

Các thành phần của một chứng nhận khóa công cộng



Mô hình Certification Authority đơn giản



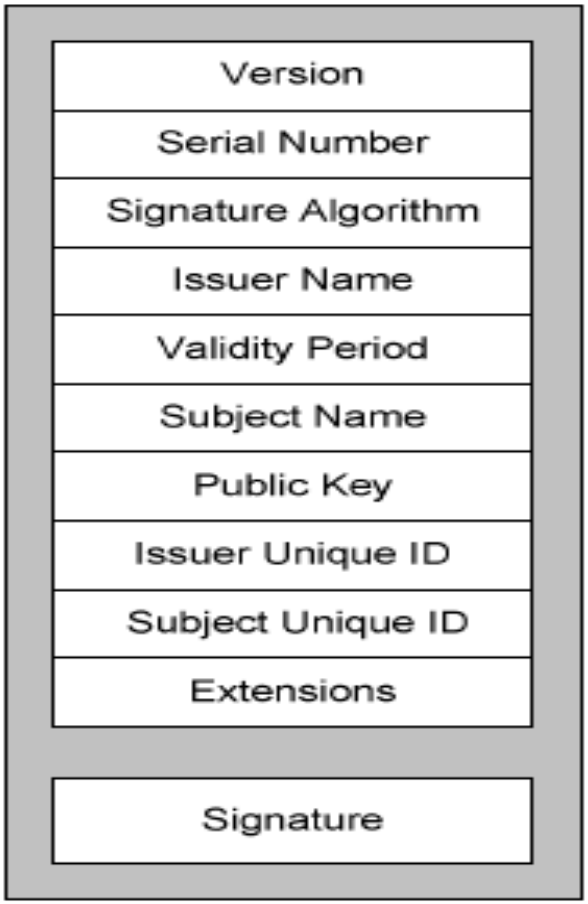
4.1. Các loại giấy chứng nhận khóa công cộng

- Giấy chứng nhận là một tập tin nhị phân có thể dễ dàng chuyển đổi qua mạng máy tính.
- Tổ chức CA áp dụng chữ ký điện tử của nó cho giấy chứng nhận khóa công cộng mà nó phát hành.

Chứng nhận X.509

- Chứng nhận X.509 là chứng nhận khóa công cộng phổ biến nhất.
- Hiệp hội viễn thông quốc tê (ITU) đã chỉ định chuẩn X.509 vào năm 1988 (phiên bản 1).
- Phiên bản 2 (1993) của chuẩn X.509 được phát hành với 2 trường tên nhận dạng duy nhất được bổ sung.
- Phiên bản 3 (1997) của chuẩn X.509 được bổ sung thêm trường mở rộng.

Phiên bản 3 của chuẩn chứng nhận X.509



Mô tả

- Một chứng nhận khóa công cộng kết buộc một khóa công cộng với sự nhận diện của một người (hoặc một thiết bị).
- Khóa công cộng và tên thực thể sở hữu khóa này là hai mục quan trọng trong một chứng nhận.
- Hầu hết các trường khác trong chứng nhận X.509 phiên bản 3 đều đã được chứng tỏ là có ích.

Mô tả một số trường

- *Signature Algorithm*: Thuật toán chữ ký chỉ rõ thuật toán mã hóa được CA sử dụng để ký giấy chứng nhận.
- *Subject Name*: là một X.500 DN (X.500 Distinguised Name – X.500 DN), xác định đối tượng sở hữu giấy chứng nhận mà cũng là sở hữu của khóa công cộng.

(tiếp)

- *Public key*: Xác định thuật toán của khóa công cộng (như RSA) và chứa khóa công cộng được định dạng tuỳ vào kiểu của nó.
- *Extensions*: Chứa các thông tin bổ sung cần thiết mà người thao tác CA muốn đặt vào chứng nhận. Trường này được giới thiệu trong X.509 phiên bản 3.
- *Signature*: Đây là chữ ký điện tử được tổ chức CA áp dụng.

Chứng nhận PGP

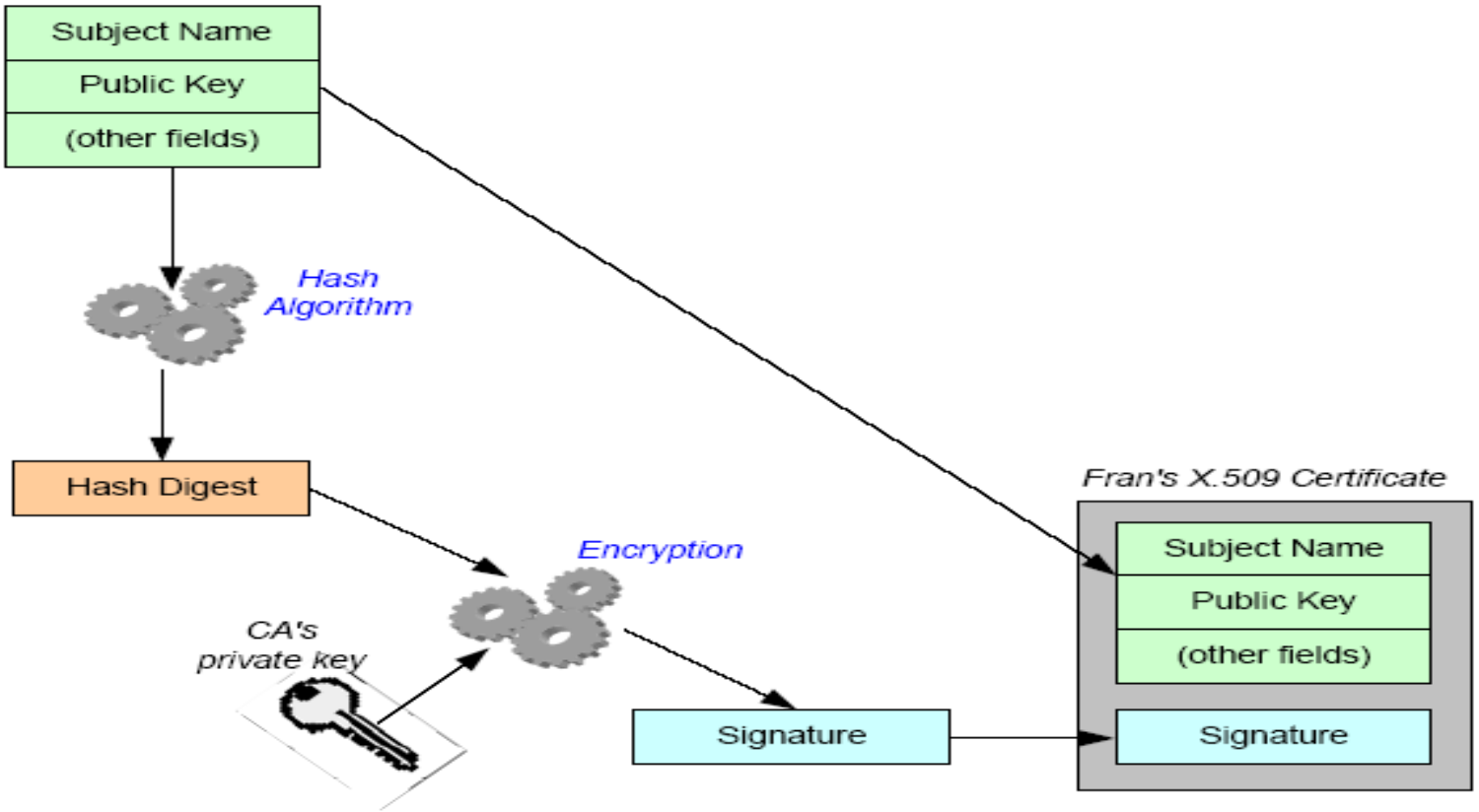
- Giấy chứng nhận X.509 được ký bởi tổ chức CA.
- Trong khi đó, giấy chứng nhận PGP có thể được ký bởi nhiều cá nhân.

➢ Mô hình tin cậy của giấy chứng nhận PGP đòi hỏi phải tin tưởng vào những người ký giấy chứng nhận PGP muốn dùng chứ không chỉ tin tưởng vào CA phát hành X.509.

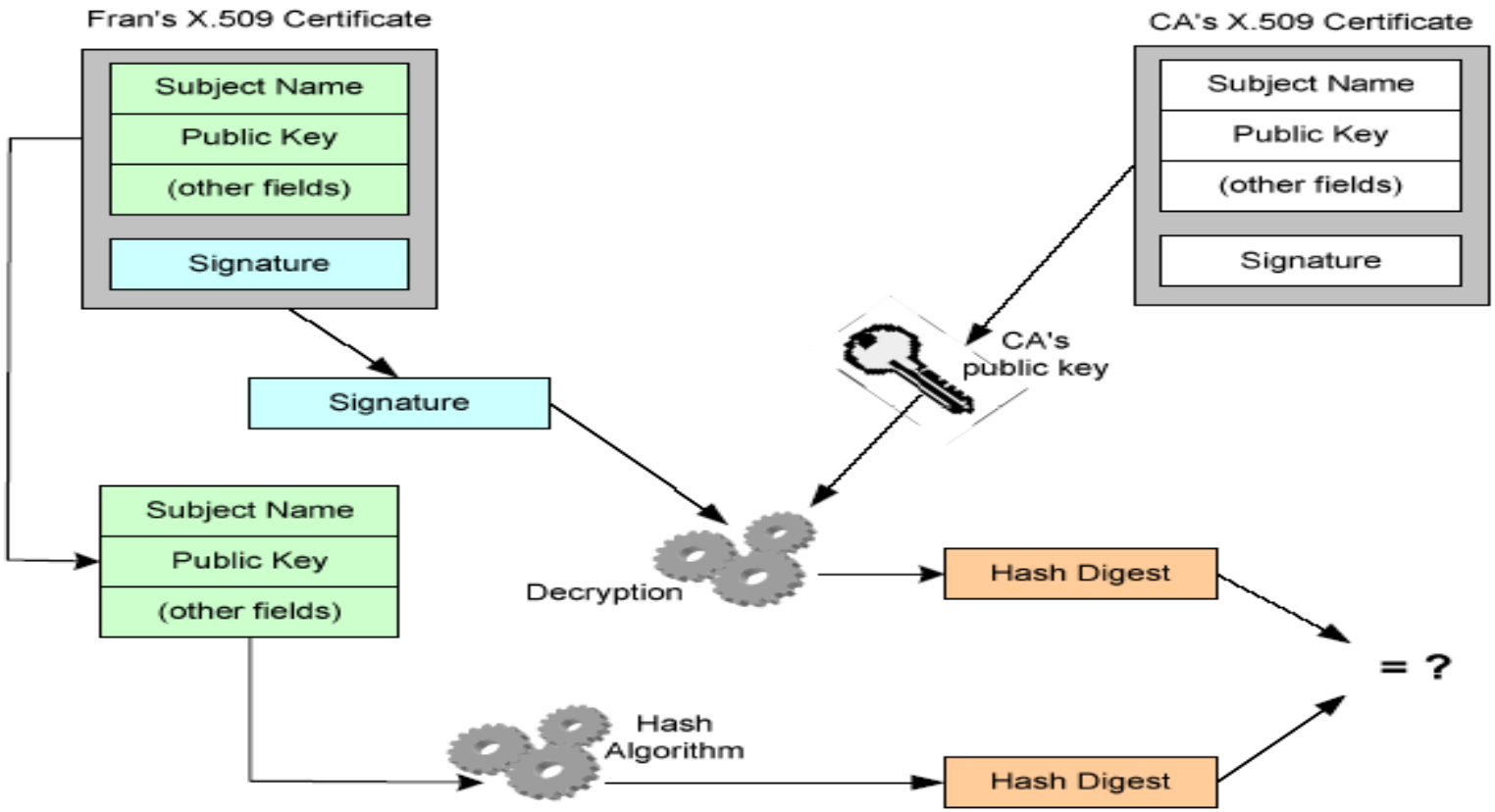
4.2. Sự chứng nhận và kiểm tra chữ ký

- Quá trình chứng nhận chữ ký diễn ra theo hai bước.
  - Đầu tiên, các trường của chứng nhận được băm bởi thuật toán cho trước.
  - Sau đó, kết quả xuất của hàm băm, được mã hóa với khóa bí mật của tổ chức CA đã phát hành chứng nhận này.

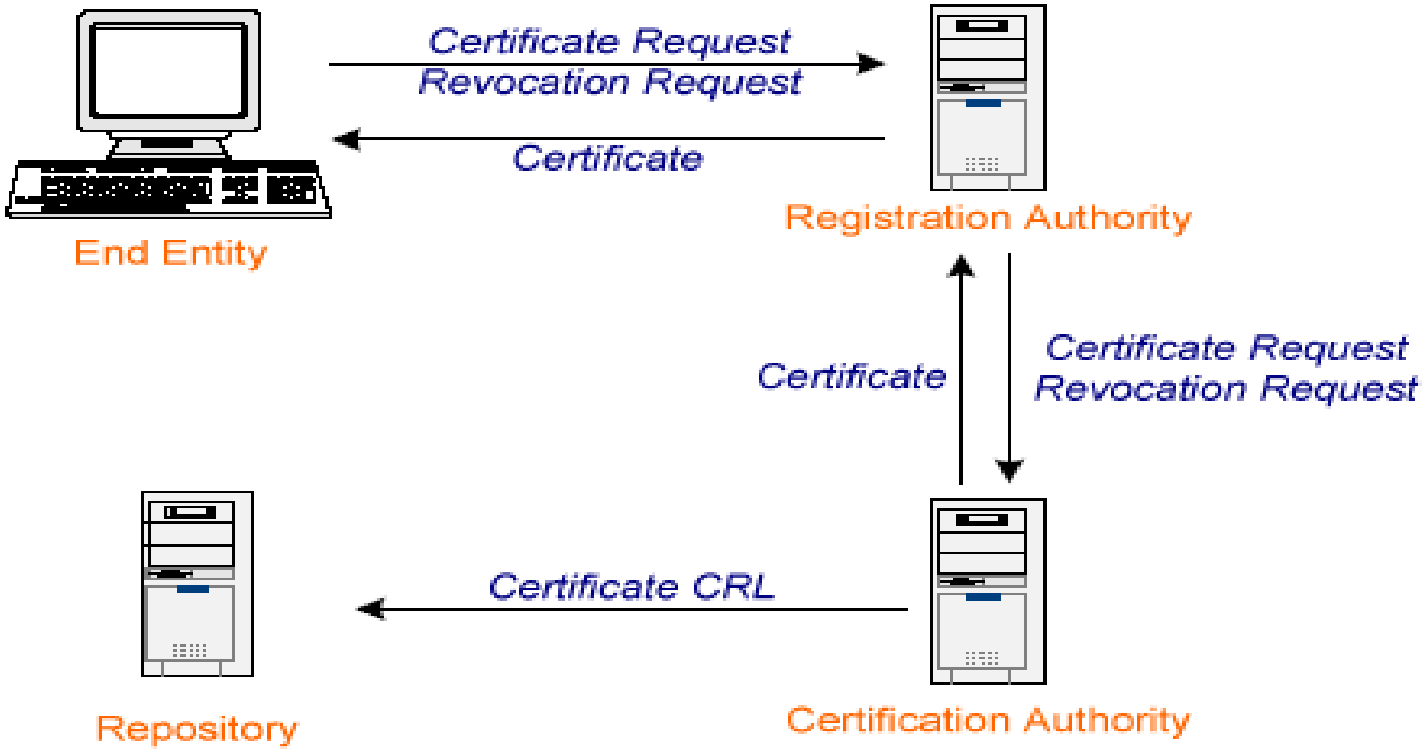
Quá trình ký chứng nhận



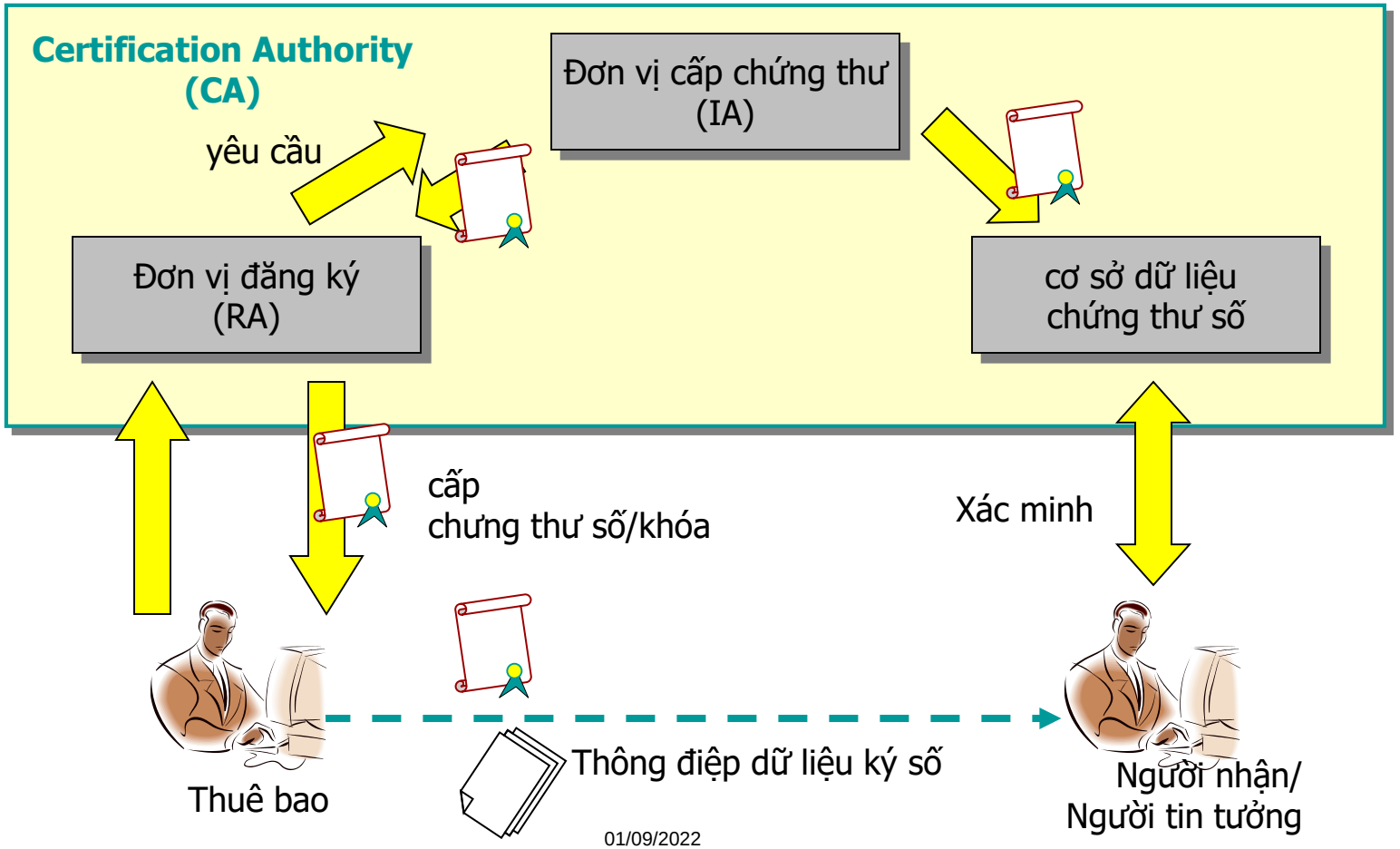
Quá trình kiểm tra chứng nhận



4.3. Các thành phần của một cở sở hạ tầng khóa công cộng



Mô hình cơ bản



4.3.1. *Tổ chức chứng nhận – Certificate Authority (CA)*

- Tổ chức CA là một thực thể quan trọng duy nhất trong X.509 PKI. (Public key Infrastructure).
- Tổ chức CA có nhiệm vụ phát hành, quản lý và hủy bỏ các giấy chứng nhận.

Mô tả

- Để thực hiện nhiệm vụ phát hành giấy chứng nhận của mình, CA nhận yêu cầu chứng nhận từ khách hàng.
- Sau đó, tổ chức CA tạo ra nội dung chứng nhận mới cho khách hàng và ký nhận cho chứng nhận đó.
- Nếu CA có sử dụng nơi lưu trữ chứng nhận thì nó sẽ lưu giấy chứng nhận mới được tạo ra ở đó.

4.3.2. *Tổ chức đăng ký chứng nhận – Registration Authority (RA)*

- Một RA là một thực thể tùy chọn được thiết kế để chia sẻ bớt công việc trên CA.
- Một RA không thể thực hiện bất kỳ một dịch vụ nào mà tổ chức CA của nó không thực hiện được

Mô tả

- Các nhiệm vụ chính của RA có thể được chia thành các loại:
  - Các dịch vụ chứng nhận.
  - Các dịch vụ kiểm tra.

Nhận xét

- Một RA hoạt động như là một xử lý ngoại vi của CA.
- Một RA chỉ nên phục vụ cho một CA. Trong khi đó, một CA có thể được hỗ trợ bởi nhiều RA.

4.3.3. *Kho lưu trữ chứng nhận – Certificate Repository (CR)*

- Một kho chứng nhận là một cơ sở dữ liệu chứa các chứng nhận được phát hành bởi một CA.
- Kho có thể được tất cả các người dùng của PKI.

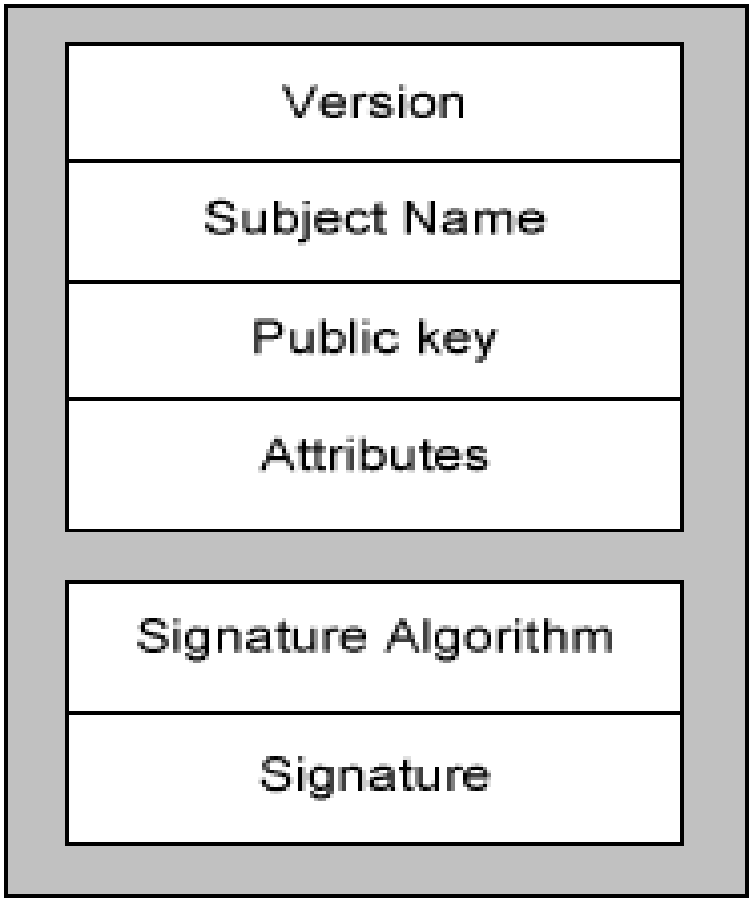
4.4. Chu trình quản lý giấy chứng nhận

- ***Khởi tạo***
- ***Yêu cầu về giấy chứng nhận***
- ***Tạo lại chứng nhận***
- ***Hủy bỏ chứng nhận***
- ***Lưu trữ và khôi phục khóa***

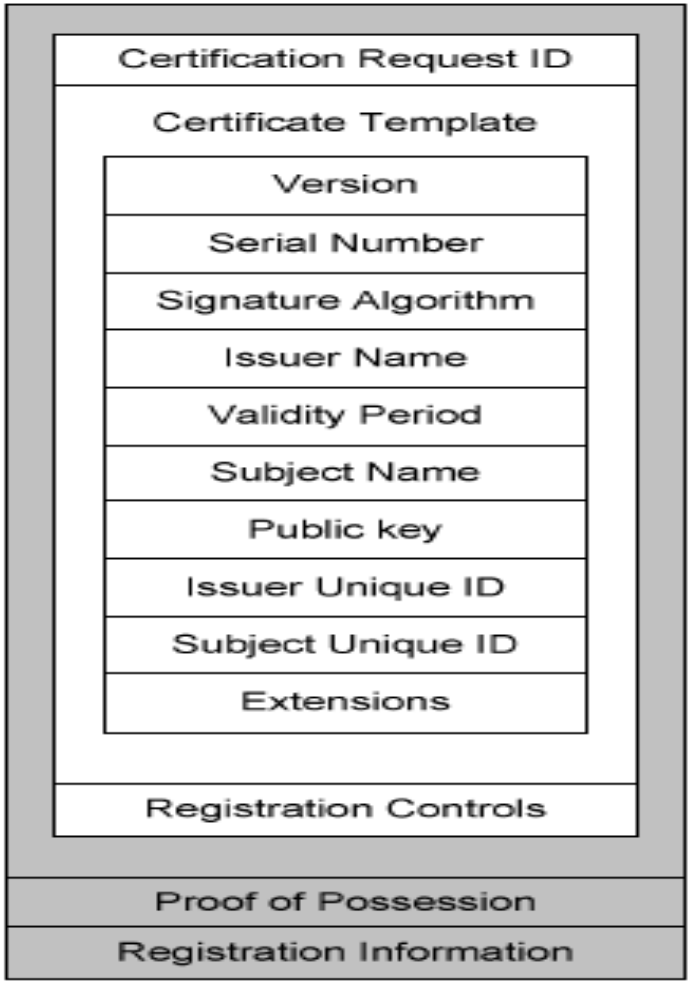
4.4.1. *Yêu cầu về giấy chứng nhận*

- Hầu hết các CA sử dụng một trong hai phương thức tiêu chuẩn của yêu cầu chứng nhận : PKCS #10 và CRMF.

Mẫu yêu cầu chứng nhận theo chuẩn PKCS#10



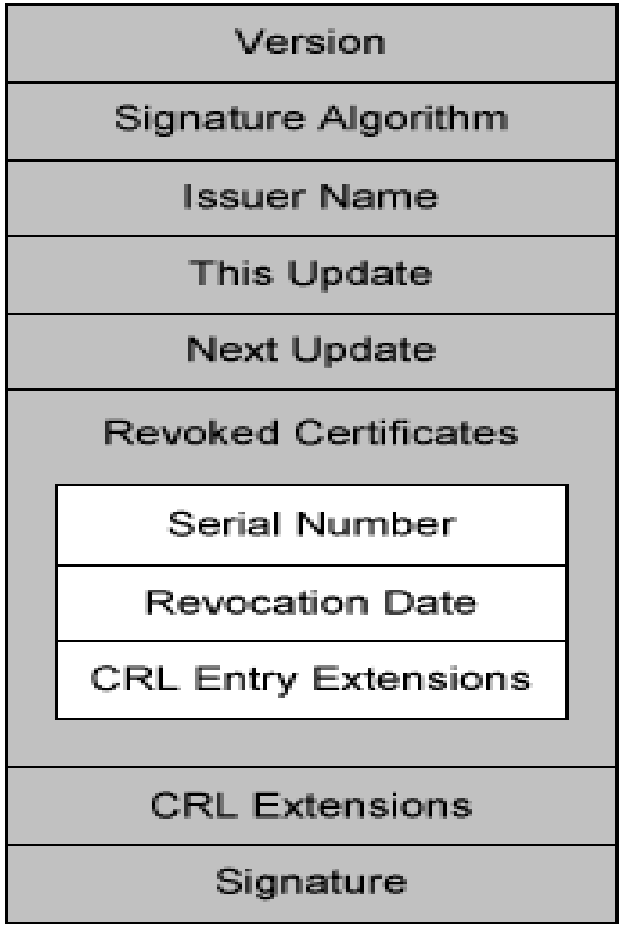
Định dạng thông điệp yêu cầu chứng nhận theo RFC 2511



4.4.2. *Hủy bỏ chứng nhận*

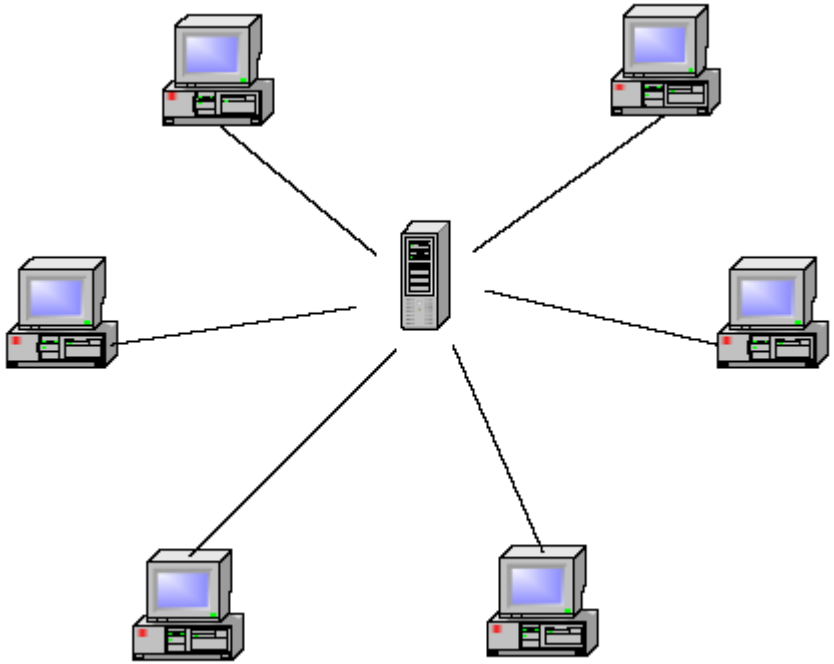
- Certificate Revocation List (CRL) là cách đầu tiên và thông dụng nhất để phổ biến thông tin hủy bỏ.
- CRL chứa thông tin thời gian nhằm xác định thời điểm tổ chức CA phát hành nó.
- CA ký CRL với cùng khóa bí mật được dùng để ký các chứng nhận.

Phiên bản 2 của định dạng danh sách chứng nhận bị hủy

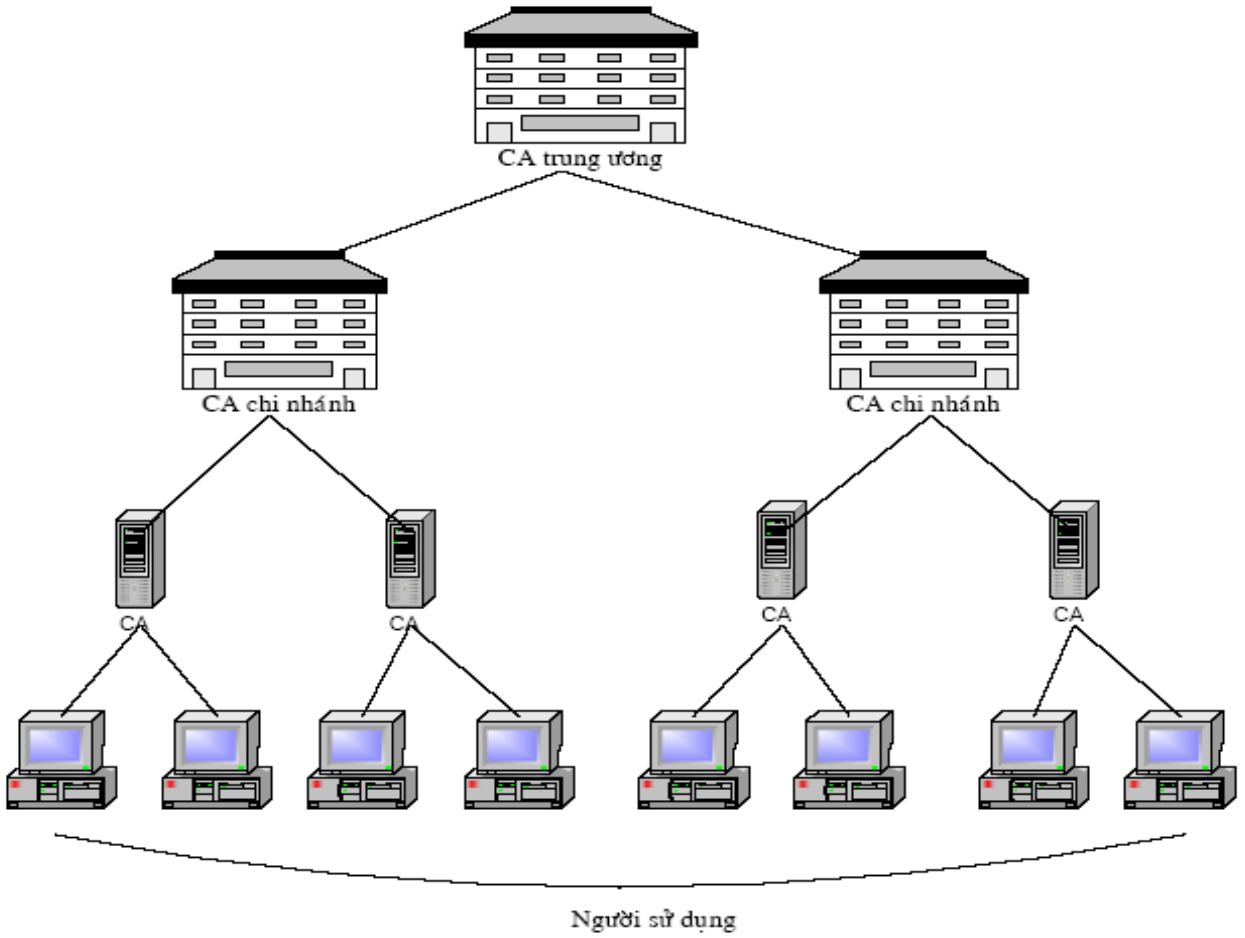


4.5. Các mô hình CA

- Mô hình tập trung



Mô hình phân cấp



Nhận xét

- Tất cả mọi chứng nhận khóa công cộng đều được ký tập trung bởi tổ chức CA và có thể được xác nhận bằng khóa công cộng của CA.
- Khuyết điểm chính của mô hình này là hiện tượng “nút cổ chai” tại trung tâm.

Nhận xét

- Tổ chức CA được phân ra thành nhiều cấp, tổ chức CA ở cấp cao hơn sẽ ký vào chứng nhận khóa công cộng của các tổ chức CA con trực tiếp của mình.
- Một chứng nhận khóa công cộng của người sử dụng sẽ được ký bởi một tổ chức CA cục bộ.

Câu hỏi và bài tập

1. Liệt kê các cách mà secret key có thể được phân phối cho hai bên giao tiếp.
2. Khóa của Distribution Center là gì?
3. Cho biết trách nhiệm của một KDC
4. Session Key là gì và chỉ ra một KDC có thể tạo một session key giữa Alice và Bob
5. Kerberos là gì và tên server của nó; mô tả trách nhiệm của từng Server.
6. Thế nào là tấn công Man-in-the-middle

Câu hỏi và bài tập

7. Giao thức Station-to-Station là gì, cho biết mục đích của nó
8. CA là gì, mối quan hệ của nó vỡi mã hóa khóa công khai

THANKS YOU