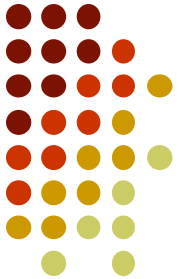
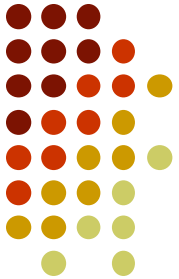


CHƯƠNG 6:
QUẢN LÝ KHÓA

Nội dung



- Xác lập và trao chuyển khóa trong SKC
- Xác lập khóa SKC thông qua sử dụng PKC
- Hạ tầng khóa mật mã công khai (Public Key Infrastructure)
- Giao thức thống nhất khóa DIFFIE-HELLMAN

Quản lý khóa?

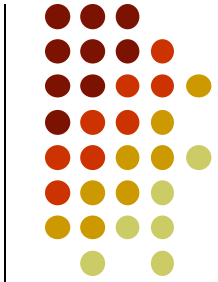
- Khóa là một dạng thông tin đặc thù, then chốt trong mọi hoạt động bảo mật. Vì vậy, cần có những cơ chế, thuật toán đặc biệt để tạo lập và thác tác đối với khóa.
  - **lập khóa**
  - **trao chuyển khóa**
  - **quản lý khóa**
  - **lưu trữ**
  - **phục hồi khóa**

Quản lý khóa?

- **SKC**, vấn đề làm sao xác lập được khóa bí mật chung thông qua một kênh liên lạc công cộng giữa hai cá nhân chưa có gì chung trước đó.
- **PKC,** là sự phức tạp trong quản lý chính khóa công khai, nhằm đảm bảo tính an toàn (chống tấn công sửa đổi, thay thế khóa).

1. XÁC LẬP VÀ TRAO CHUYỂN KHÓA BÍ MẬT TRONG SKC

1.1 Khóa phiên



- Để đảm bảo an toàn, A và B sẽ thường xuyên thay đổi khóa mã mật trong quá trình liên lạc.
- Mỗi phiên liên lạc lại sử dụng một khóa riêng, và vì thế sẽ gọi là **khóa phiên**.
- Hết phiên liên lạc, khóa phiên cũ sẽ hủy, vào phiên mới lại tạo khóa phiên mới.

1.1 Khóa phiên

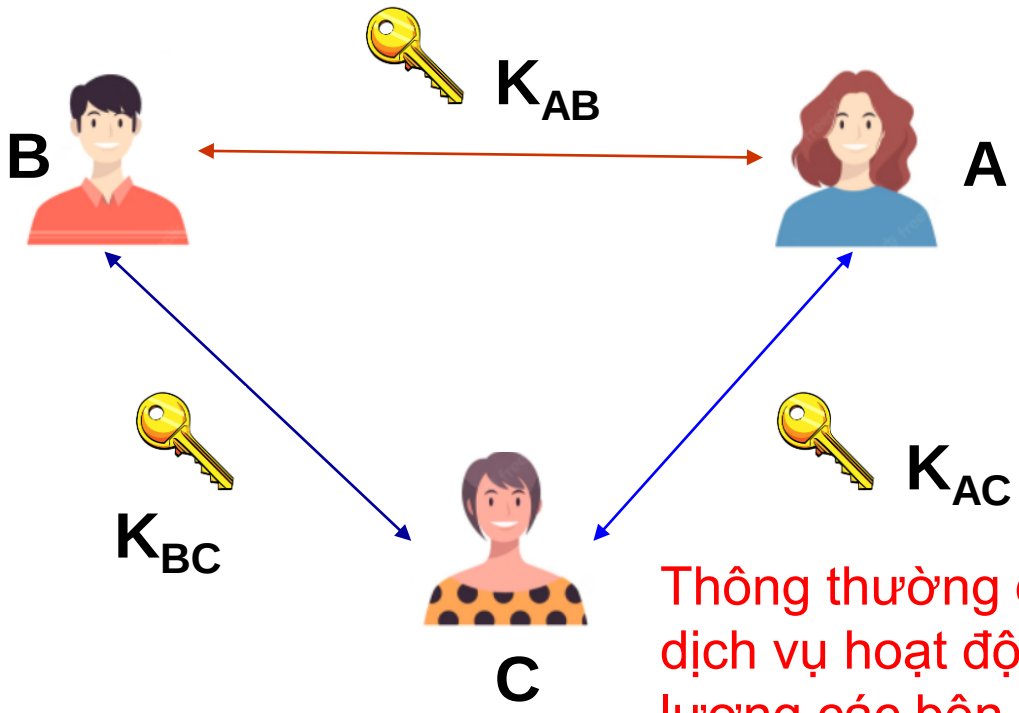
- Ví dụ: để A có thể gửi văn bản m đến B với một khóa phiên tạo riêng cho phiên liên lạc này, có thể kết hợp cả hai việc (tạo khóa phiên và gửi tin mật) trong một bước như sau:

$$A \rightarrow B: \{m\}\, k_s \,||\, \{\, ks \,\}\, k_T$$

- Như vậy khi nhận được, B sẽ lần lượt giải mã phần thứ hai để nhận được khóa phiên ks, rồi dùng nó để giải mã phần thứ nhất để thu được văn bản m.

1.2 Trao chuyển xác lập khóa đối xứng sử dụng người trung gian tin cậy

- Trường hợp 2 bên A và B chưa biết nhau?
- Sử dụng bên thứ ba tin cậy



Thông thường đây là một trung tâm dịch vụ hoạt động có giấy phép và số lượng các bên đăng ký dịch vụ (như A, hay B) có thể rất lớn

1.2 Trao chuyển xác lập khóa đối xứng sử dụng người trung gian tin cậy

- Giao thức Needham-Shroeder

1. A→ C: Alice|| Bob|| r_1
2. C→ A: {Alice || Bob || r_1 || k_s || {Alice || k_s } k_{BC} } k_{AC}
3. A→ B: {Alice || k_s} k_{BC}
4. B→ A: {r_2 }k_s
5. A→ B: {$r_2 - 1$} k_s

1.3 Sự cố mất khóa phiên cũ và giải pháp phòng vệ

- Trường hợp "mất" khóa phiên cũ (E lấy được khóa phiên cũ và mạo danh A)

$$E \rightarrow B: \{ \text{Alice} \,||\, k_s \}\, k_{BC}$$

$$B \rightarrow E: \{ r_2 \}\, k_s$$

$$E \rightarrow B: \{ r_2\text{-}1 \}\, k_s$$

- Giao thức Needham-Shroeder cải tiến (nhãn time T)

1. $A \rightarrow C:\ \text{Alice} \,||\, \text{Bob} \,||\, r_1$
2. $C \rightarrow A: \{\text{Alice} \,||\, \text{Bob} \,||\, r_1 \,||\, k_s \,||\, \{\text{Alice} \,||\, T \,||\, k_s\}\, k_{BC} \}\, k_{AC}$
3. $A \rightarrow B: \{\text{Alice} \,||\, T \,||\, k_s\}\, k_{BC}$
4. $B \rightarrow A: \{r_2\}\, k_s$
5. $A \rightarrow B: \{r_2 - 1\}\, k_s$

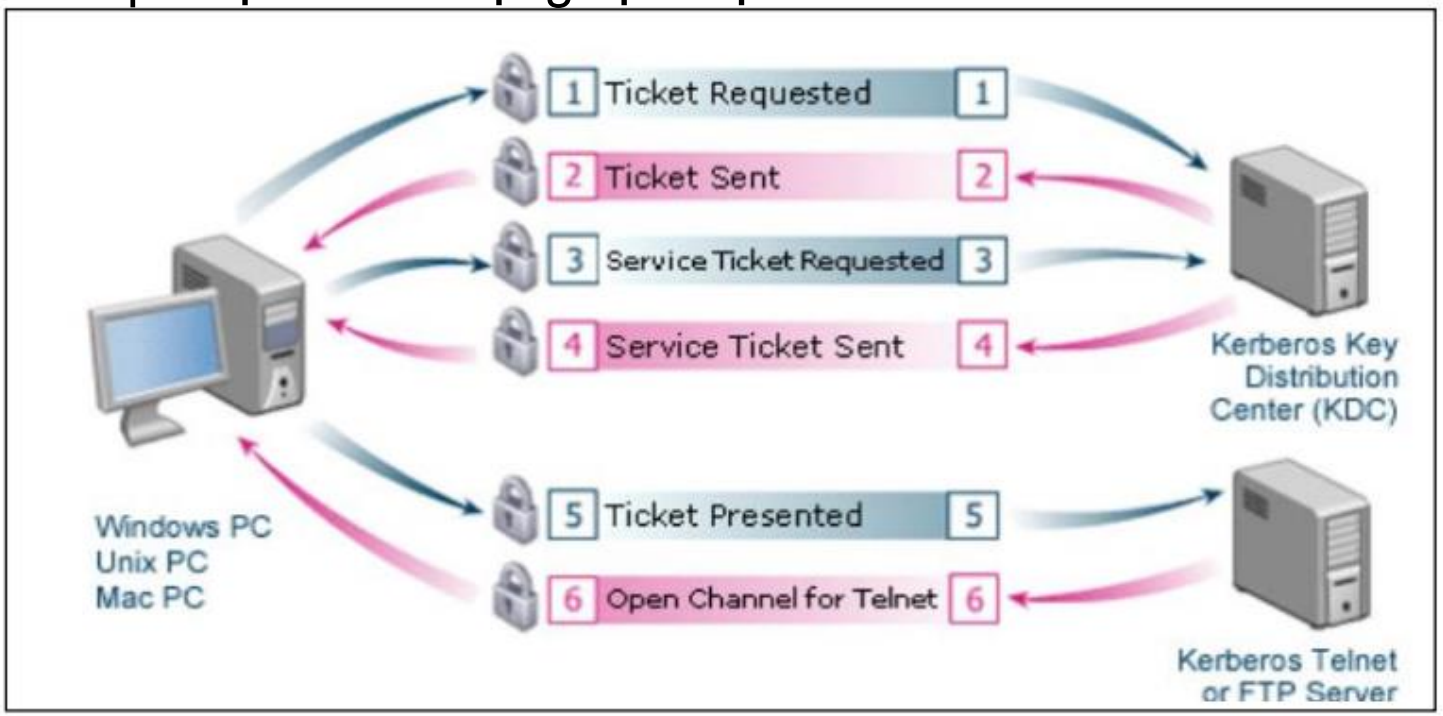
1.4 Giao thức Kerberos

- Giao thức xác thực Kerberos được đề xuất và phát triển bởi đại học MIT từ những năm 80 của thế kỷ trước
- Cơ chế xác thực cho các ứng dụng client-server trên mạng công cộng như Internet
- Cung cấp cơ chế sinh khóa phiên để đảm bảo an toàn cho các kệnh mật, sử dụng mật mã khóa đối xứng, sau khi bước xác thực đã thực hiện xong
- Sử dụng người thứ ba tin cậy trong môi trường mã hóa đối xứng

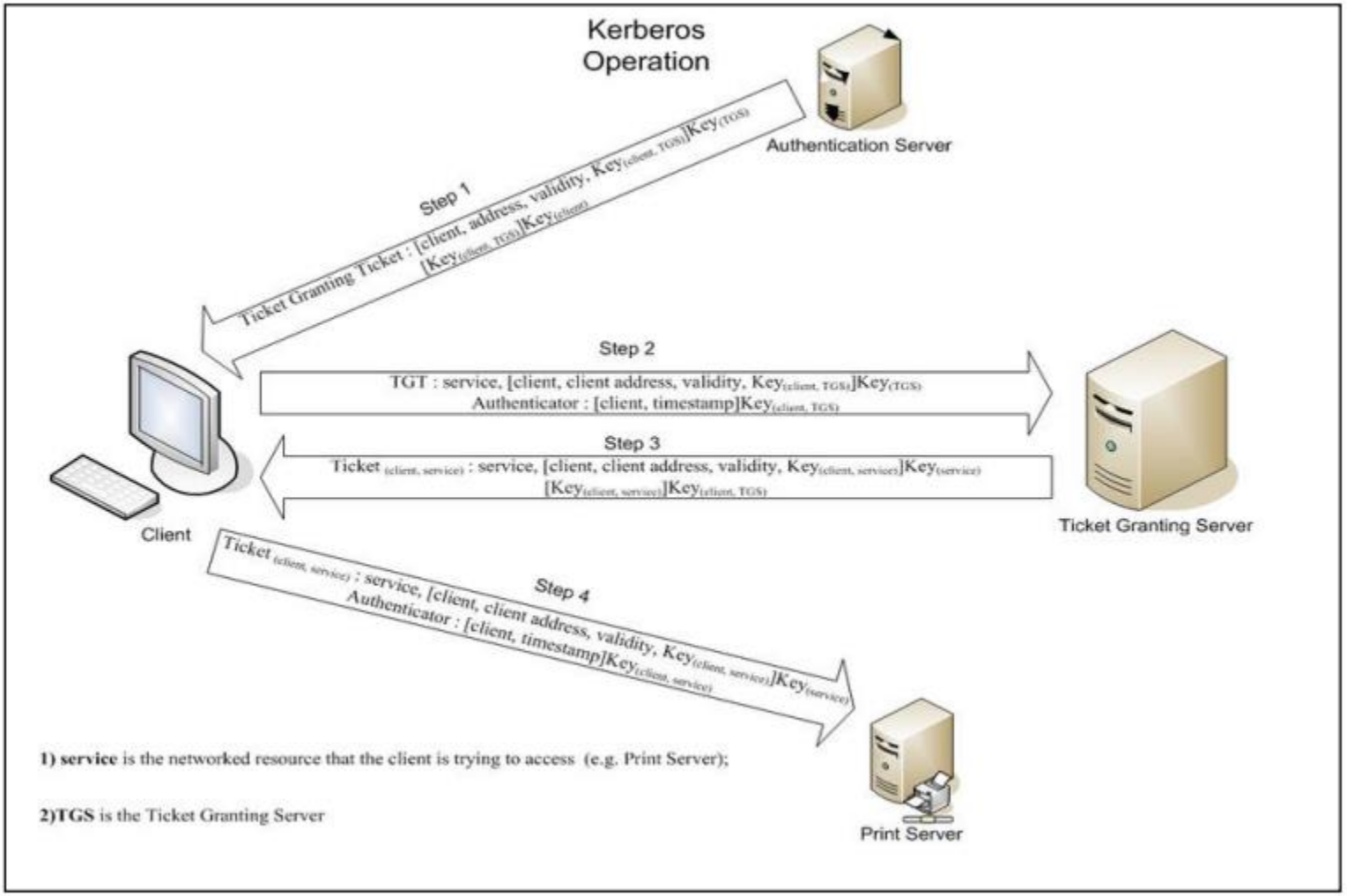
1.4 Giao thức Kerberos

Kerberos thực hiện các quá trình sau để xác thực:

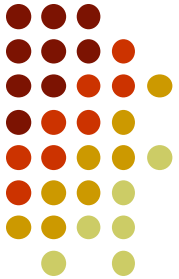
- một user muốn xác thực mình với authentication server (AS)
- sau đó sẽ chứng minh với ticket granting server (TGS) rằng mình đã được xác thực để nhận ticket rồi
- cuối cùng chứng minh với service server (SS) rằng mình đã được chấp nhận để sử dụng dịch vụ rồi.



1.4 Giao thức Kerberos



1.4 Giao thức Kerberos



1. Client gửi một yêu cầu đến AS để yêu cầu dịch vụ.
2. AS kiểm tra bên client có nằm trong cơ sở dữ liệu của mình hay không? Nếu có AS sẽ gửi lại cho bên client 2 gói tin:
   a. Gói tin A: client/TGS session key được mã hóa bởi khóa bí mật của client.
   b. Gói tin B: ticket (chứa ID, địa chỉ mạng client, thời hạn hiệu lực ticket và client/TGS session key) được mã hóa bởi khóa bí mật của TGS.
3. Khi nhận được 2 gói tin trên client giải mã gói tin A để có được session key (TGS). Session key này được sử dụng để giao tiếp với TGS, tuy nhiên client không thể giải mã được gói tin B vì nó được mã hóa bởi khóa bí mật của TGS.
4. Client sau đó sẽ gửi 2 gói tin đến TGS:
   a. Gói tin C: gói tin B (chứa ticket)và ID của dịch vụ yêu cầu.
   b. Gói tin D: ID (client), timestamp, mật mã hóa sử dụng client/TGS session key.

1.4 Giao thức Kerberos

5. Khi nhận được 2 gói tin C và D, TGS sẽ lấy gói tin B ra khỏi C. Giải mã gói tin B sử dụng khóa bí mật của mình:
   a. Gói tin E: ticket (bao gồm ID của client, địa chỉ mạng của client, thời hạn sử dụng session key (client/server)) được mã hóa bởi SS (máy chủ cung cấp dịch vụ).
   b. Gói tin F: session key (client/server) được mã hóa bởi session key (TGS).
6. Khi nhận được 2 gói tin E và F, client sẽ gửi 2 gói tin đến SS:
   a. Gói tin E thu được từ bước trước.
   b. Gói tin F: ID của client, thời điểm yêu cầu và được mã hóa bởi session key (client/server).
7. SS giải mã ticket bằng khóa bí mật của mình và gửi gói tin sau cho client:
   Gói tin H: client/server session key.
8. Client giải mã chứng thực sử dụng client/server session key và kiểm tra timestamp cho phù hợp hay không? Nếu có thì client có thể tin tưởng vào server và sử dụng dịch vụ này.

1.5 Vấn đề sinh khóa

- Một chuỗi số n1, n2, … được gọi là sinh ngẫu nhiên (randomly generated) nếu như với mọi giá trị k, thì không thể đoán trước được giá trị của nk dù trước đó đã quan sát được tất cả các giá trị n1, n2, … nk-1
- Ví dụ: thực hiện phép chọn ngẫu nhiên một trong số 264 giá trị (từ 0 đến 264-1): khả năng đoán được của kẻ thù là gần như bằng 0
- Tuy nhiên: việc sinh số ngẫu nhiên chỉ thông qua các thuật toán → sinh số giả ngẫu nhiên

→Cơ chế giả ngẫu nhiên mật mã, mô phỏng chuỗi ngẫu nhiên thật

2. DÙNG PKC ĐỂ TRAO CHUYỂN KHOÁ BÍ MẬT

- Ý tưởng: A và B, đã có một cách nào đó để biết được khóa công khai của nhau, Khi đó A có thể chủ động tạo khóa phiên ks (sinh số giả ngẫu nhiên) và chuyển qua cho B như sau:

$$A \rightarrow B: \{ \{ \text{Alice} \,||\, k_s \} \, d_A \} \, e_B$$

2. DÙNG PKC ĐỂ TRAO CHUYỂN KHOÁ BÍ MẬT

- Phương án 1:

Giả sử A muốn thiết lập một khoá phiên đối xứng với B.

i) A và B tìm lấy khoá công khai của nhau

ii) A tạo ra một khoá bí mật k_s và vector khởi đầu IV

iii) Alice tạo ra một bản ghi gồm khoá k_s, vector IV, tên của Alice, nhãn thời gian và một số tuần tự (sequence number), rồi mã hoá bản ghi này với khoá công khai của Bob và gửi cho Bob

X= [K, IV, A's ID, timestamp, seq. no.]

A→ B: Y = $E_{Z_B}(X)$

Những thông tin thêm vào này (A's ID, timestamp, seq. no.) dùng để giúp xác thực Alice với Bob và qua đó chống lại replay attack: thông qua việc so sánh nhãn thời gian với thời gian hiện tại, Bob có thể dễ dàng xác định một cuộc liên lạc kiểu trên là hợp lệ hay là một tấn công phát lại.

- Phương án 2: phương án bắt tay ba bước NeedhamSchroeder

A và B cũng có thể xác nhận được sự có mặt của nhau trong thời gian thật thông qua 3 bước sau:

i) A→ B: $E_{Z_B}(R_A, ID_A)$

ii) B → A: $E_{Z_A}(R_A, R_B)$

iii) A → B: $E_{Z_A}(R_B)$

Ở đây R_A, R_B là các số ngẫu nhiên do A, B tạo ra còn ID_A là các thông tin định danh cho A.

Ta có thể thấy rằng sau bước 2, A đã có thể xác mình được rằng đúng phía bên kia đang là B (vì chỉ có như thế thì mới giải mã được và trả về ngay số ngẫu nhiên R_B, kẻ dùng replay attack không thể thoả mãn được yêu cầu, tức là cũng phát lại về đúng các số ngẫu nhiên).

Tương tự, sau bước 3, B đã có thể kiểm tra được rằng đúng phía bên kia đang là A. Tóm lại, bằng cách đó, A và B đã có thể xác thực sự có mặt của nhau tại cùng thời điểm (thời gian thực) và sau đó A chỉ việc gửi khoá phiên sang cho B: A → B: $E_{Z_B}(D_{Z_A}(K))$.

3. Hạ tầng khóa mật mã công khai

- Cần đăng ký khóa công khai đã tạo cho một cơ quan phát hành thẩm quyền (Certificate Authority – CA) →chứng chỉ khóa công khai được phát hành, gắn chặt các thông tin khóa và thông tin người sở hữu

3.1 ISO Authentication Framework - X.509

Version
Serial Number: số giấy chứng nhận do người phát hành, CA đặt ra để phân biệt và lưu trữ các certificate.
Algorithm identifier: thông số về thuật toán mà người phát hành dùng để sinh chữ ký • Algorithm: tên thuật toán • Parameters: các tham số thuật toán
Issuer: Người phát hành ra giấy chứng nhận này (certificate)
Subject: người được chứng nhận Interval of validity: thời hạn sử dụng hợp lệ
Subject's public key: Về khoá công khai của người được chứng nhận • Algorithm: Thuật toán PKC sử dụng với khoá công khai này • Parameters: Các tham số cho thuật toán • Public key: Khoá công khai
Signature: chữ ký của người phát hành

3. Giao thức thống nhất khóa DIFFIE-HELLMAN

- Giao thức này cho phép hai bên A và B có thể xác lập khóa chung mà không cần bên thứ ba tin cậy

A và B thống nhất chọn một số nguyên tố p, một phần tử nguyên thuỷ (primitive element) α, tức là:

$$\{\alpha^0, \alpha^1, \alpha^2, ..., \alpha^{p-1}\} = \{1,2,3 ..., p-1\}$$

1. A chọn một số ngẫu nhiên X_A, $1 \le X_A \le p$. B chọn một số ngẫu nhiên X_B, $1 \le X_B \le p$.

   A giữ bí mật X_A; B giữ bí mật X_B

2. A tính: $Y_A = \alpha^{X_A} \pm p$ và B tính: $Y_B = \alpha^{X_B} \pm p$

   A → B: Y_A

   B → A: Y_B.

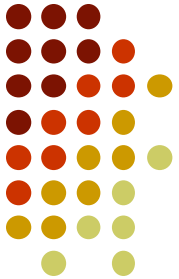
3. A tính:

$$K = (Y_B)^{X_A} \pm p = (\alpha^{X_B})^{X_A} = \alpha^{X_A X_B} \pm p$$

   B tính:

$$K = (Y_A)^{X_B} \pm p = (\alpha^{X_A})^{X_B} = \alpha^{X_A X_B} \pm p$$

3. Giao thức thống nhất khóa DIFFIE-HELLMAN



- Kẻ thù chỉ có thể nghe trộm được Y_A,Y_B truyền qua mạng, để tính được K nó cần phải biết X_A, X_B.
- Dựa vào Y_A tìm X_A là khó: Độ an toàn của hệ thống quyết định bởi tính khó của bài toán tính logarit rời rạc.