Authentication Protocols
(身份鉴别协议)

- Mutual Authentication
- One-way Authentication

Mutual Authentication Protocols (相互鉴别协议)

- used to convince(确信) parties each others and to exchange session keys
- key issues:
  - – confidentiality – to protect session keys
  - – timeliness (及时性)– to prevent replay attacks

Replay Attacks

- a valid(正确的) signed message is copied and later resent
- 三大对策Countermeasures include:
  - use of sequence numbers (generally impractical 需记录)
  - timestamps (needs synchronized clocks)
  - challenge/response (using unique nonce)

Mutual Authentication : Using Symmetric Encryption

- use a two-level hierarchy of keys
- usually with a trusted Key Distribution Center (KDC)
  - each party shares own master key with KDC
  - KDC generates session keys used for connections between parties
  - master keys used to distribute these to them

Mutual Authentication

Needham-Schroeder Protocol

- original third-party key distribution protocol
- session key between A B issued by KDC
- protocol overview:

1. A->KDC: $ID_A \| ID_B \| N_1$

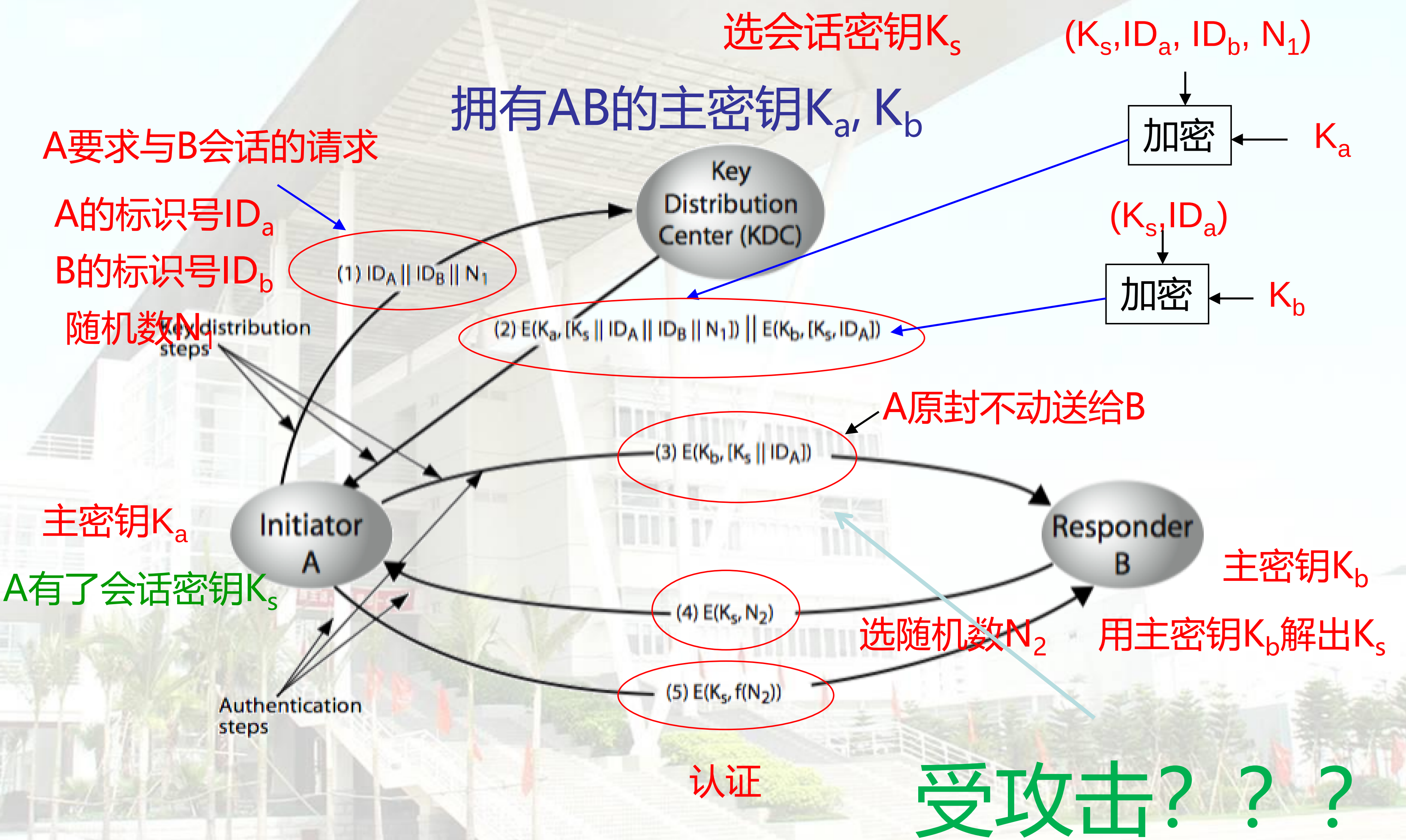
2. KDC -> A: $E_{Ka}[Ks \| ID_B \| N_1 \| E_{Kb}[Ks \| ID_A]]$

3. A -> B: $E_{Kb}[Ks \| ID_A]$

4. B -> A: $E_{Ks}[N_2]$

5. A -> B: $E_{Ks}[f(N_2)]$

Needham-Schroeder Protocol (2)



Mutual Authentication : Needham-Schroeder Protocol (3)

- used to securely distribute a new session key for communications between A & B
- but is vulnerable(易受攻击) to a replay attack if an old session key has been compromised
  - then message 3 can be resent convincing B that is communicating with A
- modifications to address this require:
  - timestamps
  - using an extra nonce

Mutual Authentication: Denning Protocol 改进

- Add timestamps T
- protocol overview:

  1. A->KDC: $ID_A \parallel ID_B$

  2. KDC -> A: $E_{Ka}[Ks \parallel ID_B \parallel T \parallel E_{Kb}[Ks \| ID_A] \parallel T]$

  3. A -> B: $E_{Kb}[Ks \| ID_A \| T]$

  4. B -> A: $E_{Ks}[N_2]$

  5. A -> B: $E_{Ks}[f(N_2)]$

But needs synchronized clocks!!!

Mutual Authentication: Needham-Schroeder Protocol (4) 改进

- using an extra nonce
- protocol overview:

1. A->B: $ID_A \parallel N_a$
2. B-> KDC : $ID_B \parallel N_b \parallel E_{Kb}[ID_A \parallel N_a \parallel T_b]$
3. KDC -> A: $E_{Ka}[ID_B \parallel N_a \parallel Ks \parallel T_b] \parallel E_{Kb}[ID_A \parallel Ks \parallel T_b] \parallel N_b$
4. A -> B: $E_{Kb}[ID_A \parallel Ks \parallel T_b] \parallel E_{Ks}[N_b]$

时间T_b由B的时钟决定,B只检查自身的时间,不存在AB时间同步问题

Using Public-Key Encryption

- need to ensure have correct public keys for other parties
- using a central Authentication Server (AS)
  -鉴别/认证服务器
- various protocols exist using timestamps or nonces

Denning AS Protocol

1. A -> AS: $ID_A \| ID_B$ ： A想与B建立连接

2. AS -> A: $E_{PRas}[ID_A \| PU_a \| T] \| E_{PRas}[ID_B \| PU_b \| T]$

AS私钥 A的公钥 B的公钥 会话密钥

3. A -> B: $E_{PRas}[ID_A \| PU_a \| T] \| E_{PRas}[ID_B \| PU_b \| T] \| E_{PUb}[E_{PRa}[K_s \| T]]$

A的私钥!!!

- note session key is chosen by A, hence AS need not be trusted to protect it
- timestamps prevent replay but require synchronized clocks

Woo-Lam Method (1)

1. A -> KDC: $ID_A \parallel ID_{B:}$ A想与B建立连接
2. KDC -> A: $E_{Kauth}[ID_B \| KU_b]$ ：将B的公钥告诉A
3. A -> B: $E_{KUb}[N_a \| ID_A]$
4. B -> KDC: $ID_B \parallel ID_A \parallel E_{KUauth}[N_a]$
5. KDC -> B: $E_{KRauth}[ID_A \| KU_a] \parallel E_{KUb}[E_{KRauth}[N_a \| [K_s \| ID_B]]]$

   将A的公钥告诉B 会话密钥

6. B -> A: $E_{KUa}[E_{KRauth}[N_a \parallel [K_s \| ID_B] \parallel N_b]$
7. A -> B: $E_{Ks}[N_b]$ 会话密钥

Woo-Lam Modified Method (2)

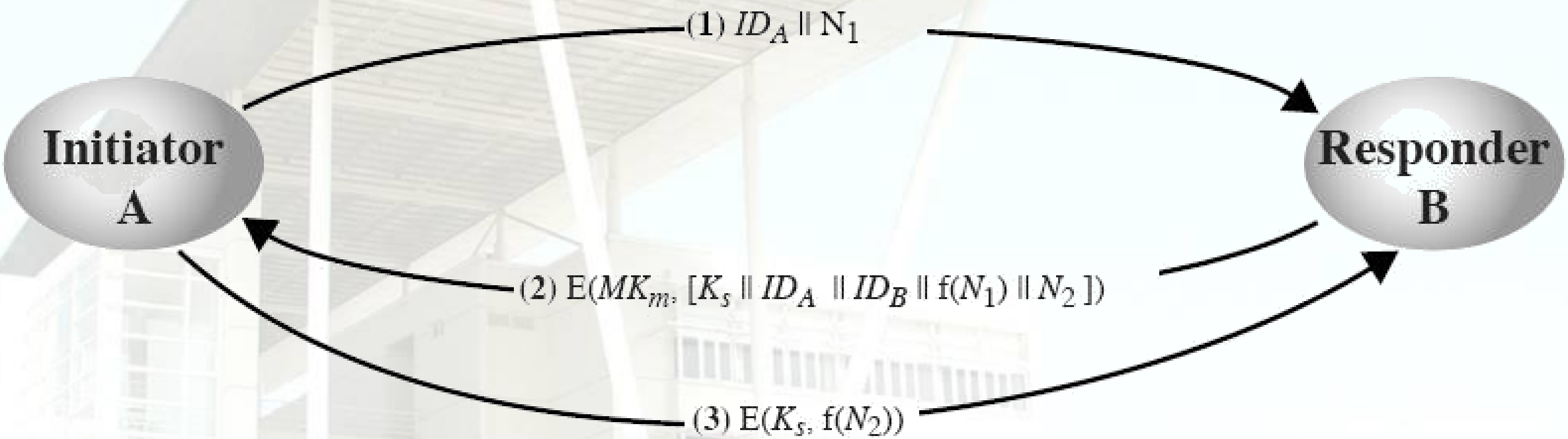
1. A -> KDC: $ID_A \,||\, ID_B$
2. KDC -> A: $E_{Kauth}[ID_B||KU_b]$ 将B的公钥告诉A
3. A -> B: $E_{KUb}[N_a|| ID_A]$
4. B -> KDC: $ID_B \,||\, ID_A \,||\, E_{KUauth}[N_a]$
5. KDC -> B: $E_{KRauth}[ID_A||KU_a] \,||\, E_{KUb}[E_{KRauth}[N_a \,||[K_s||ID_A \,||\, ID_B]]$
6. B -> A: $E_{KUa}[E_{KRauth}[N_a \,||\, [K_s|| ID_A \,||\, ID_B] \,||\, N_b]$
7. A -> B: $E_{Ks}[N_b]$

加入(N_a, ID_A) 唯一标识了A的连接请求

One-Way Authentication

- required when sender & receiver
  are not in communications at same time
    eg. email
- header in clear so can be delivered by email system
- contents of body protected & sender authenticated

decentralized(分散式) key distribution



要求发送方 向接收方提出请求,等待包含会话密钥的响应,才进行通信

Figure 7.11 Decentralized Key Distribution

Using Symmetric Encryption

- refine use of KDC but can’t have final exchange of nonces:

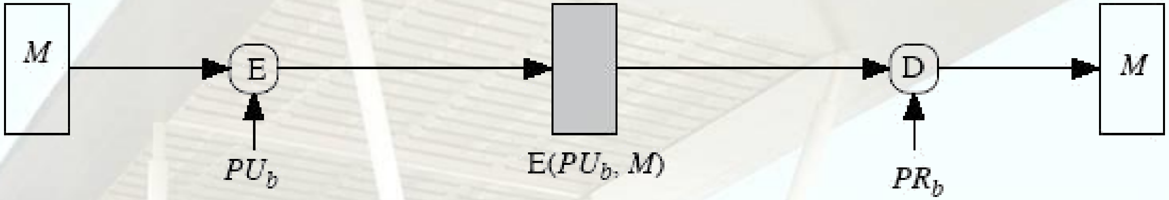
1. A->KDC: $ID_A \| ID_B \| N_1$

2. KDC -> A: $E_{Ka}[Ks \| ID_B \| N_1 \| E_{Kb}[Ks\|ID_A]]$

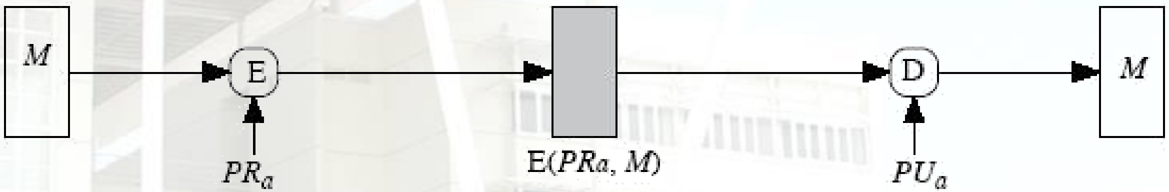
3. A -> B: $E_{Kb}[Ks\|ID_A] \| E_{Ks}[M]$

does not protect against replays

Public-Key Approaches

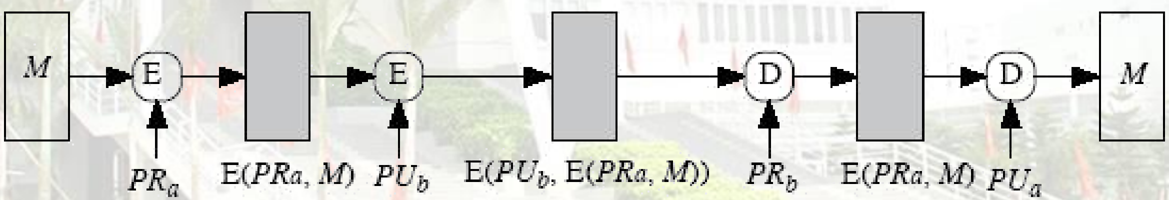


(b) Public-key encryption: confidentiality



(c) Public-key encryption: authentication and signature

要求知道对方公钥



(d) Public-key encryption: confidentiality, authentication, and signature

Public-Key Approaches

if confidentiality is major concern:

A->B: $E_{PUb}[Ks] \| E_{Ks}[M]$

– encrypt session key by public key,

– encrypt message by session key

- if authentication needed, use digital signature with digital certificate:

A->B: $M \| E_{PRa}[H(M)] \| E_{PRas}[T \| ID_A \| PU_a]$

– with message, signature, certificate

Authentication Applications

- authentication functions
- application-level authentication & digital signatures
- Kerberos
  – a private-key authentication service
- X.509
  – a public-key directory authentication service