

# **Computer Science 425**

## **Distributed Systems**

***CS 425 / CSE 424 / ECE 428***

**Fall 2010**

**Indranil Gupta (Indy)**

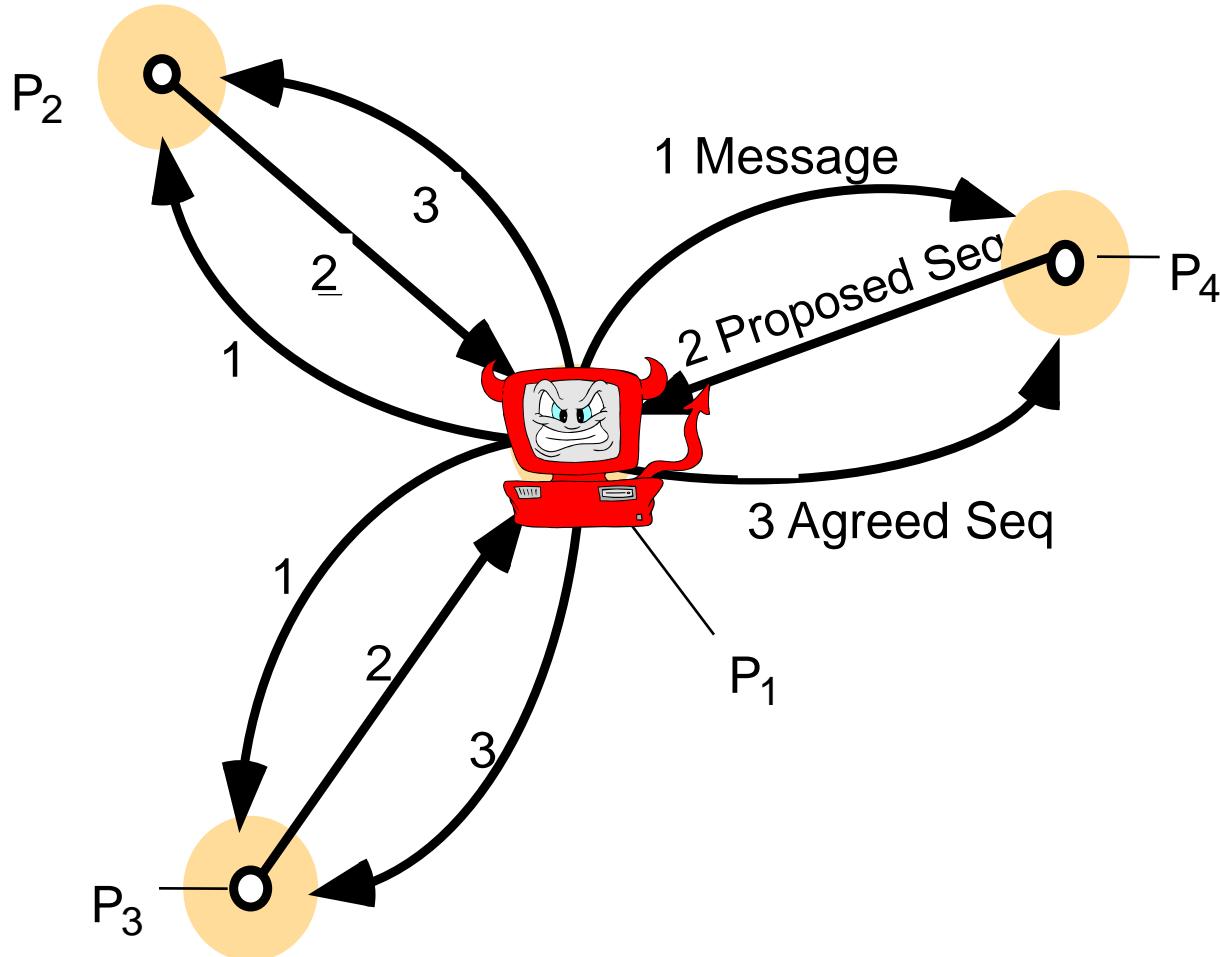
**November 27, 2012**

**Lecture 25**

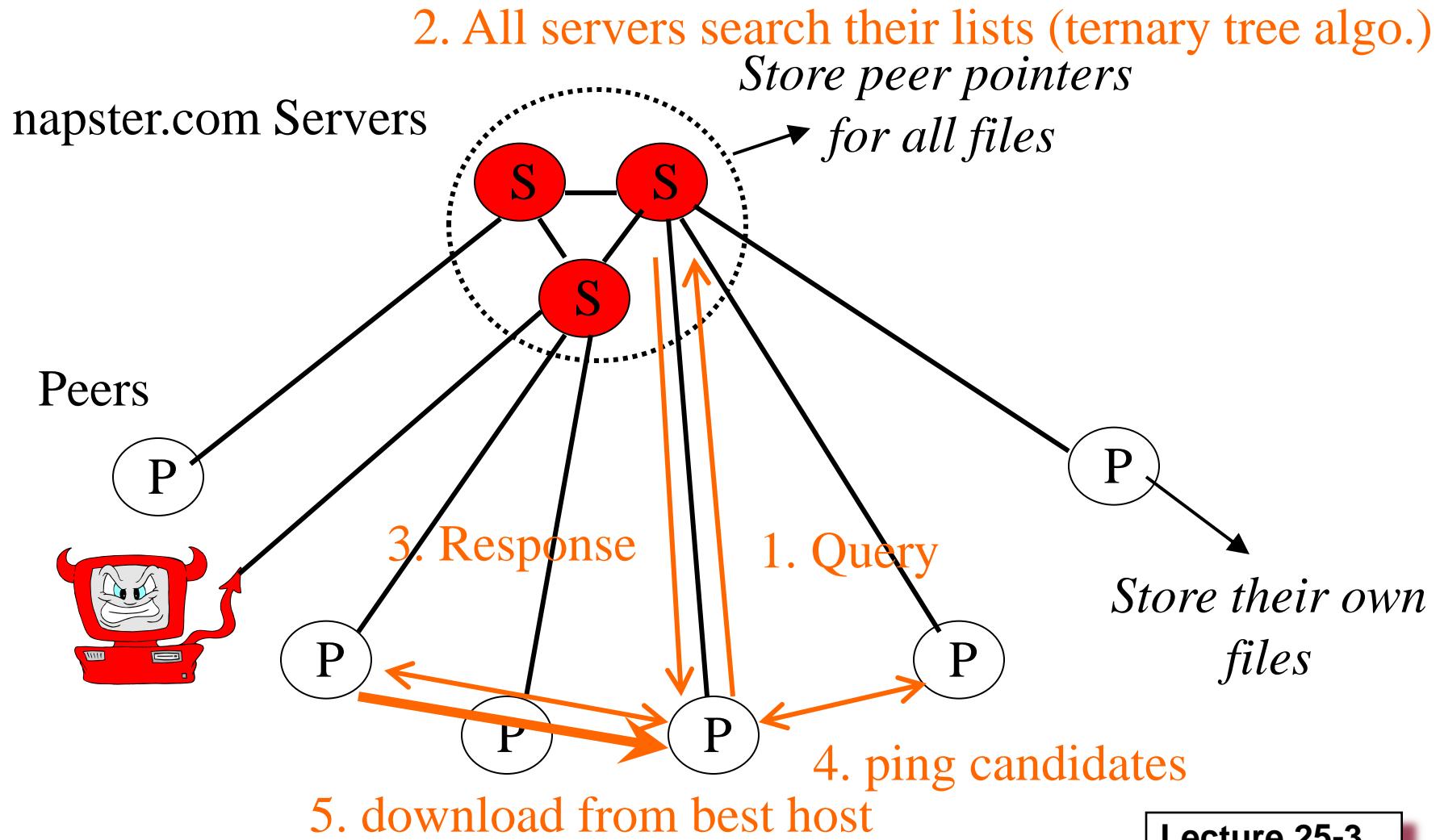
**Security**

**Reading: Chapter 11 (relevant parts)**

# ISIS algorithm for total ordering



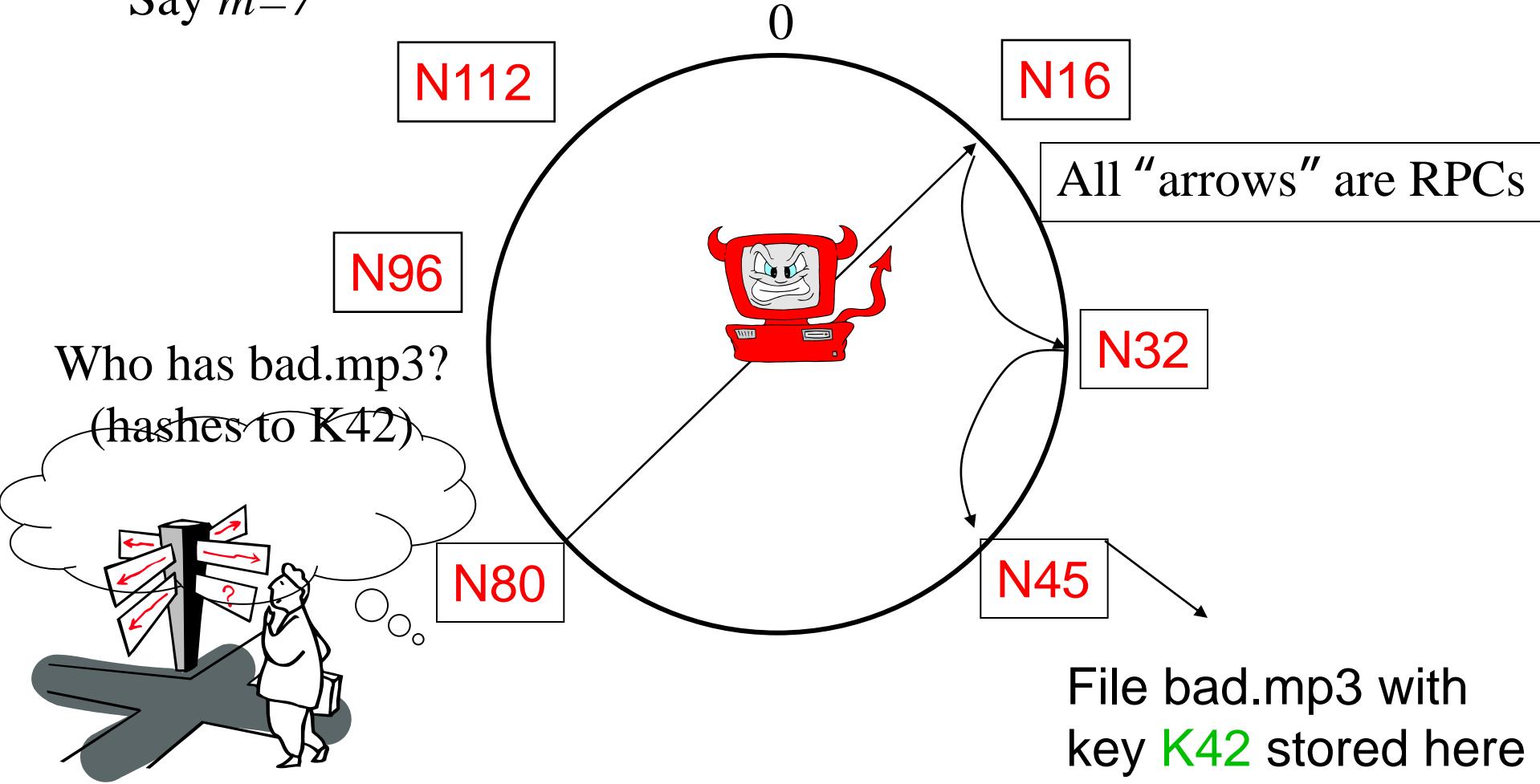
# Napster



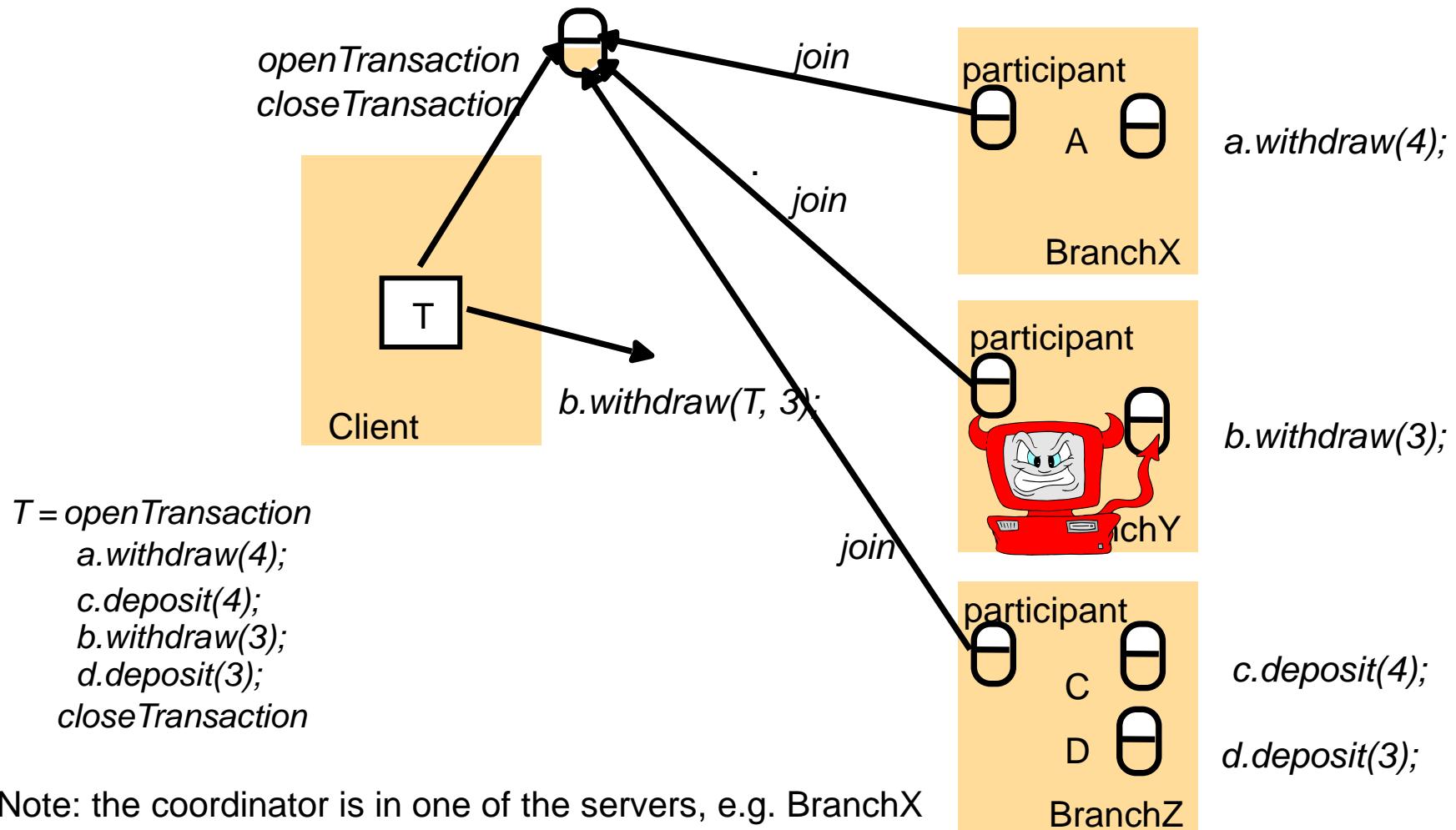
# Chord: client to client

At node  $n$ , send query for key  $k$  to largest successor/finger entry  $< k$   
if none exist, return  $\text{successor}(n)$  to requestor

Say  $m=7$



# Distributed banking transaction



# *Security Threats*

- ❖ **Leakage:** An unauthorized party gains access to a service or data.
  - ❖ Attacker obtains knowledge of a withdrawal or account balance, e.g., via eavesdropping
- ❖ **Tampering:** Unauthorized change of data, tampering with a service
  - ❖ Attacker changes the variable holding your personal checking \$\$ total
- ❖ **Vandalism:** Interference with proper operation, without gain to the attacker
  - ❖ Attacker does not allow any transactions to your account
  - ❖ E.g., DOS=denial of service

# **How Attacks are Carried Out**

## **Attacks on Communication Channel / Network**

- ❖ **Eavesdropping** – Obtaining copies of messages without authority.
- ❖ **Masquerading** – Sending or receiving messages with the identity of another principal (user or corporation). Identity theft.
- ❖ **Message tampering** – Intercepting messages and altering their contents before passing them onto the intended recipient.
- ❖ **Replaying** – Intercepting messages and sending them at a later time.
- ❖ **Denial of Service Attack** – flooding a channel or other resources (e.g., port) with messages.

# ***Addressing the Challenges: Security's CIA***

- ❖ **Leakage:** An unauthorized party gains access to a service or data.
  - Confidentiality : protection against disclosure to unauthorized individuals.
- ❖ **Tampering:** Unauthorized change of data, tampering with a service
  - Integrity : protection against alteration or corruption.
- ❖ **Vandalism:** Interference with proper operation, without gain to the attacker
  - Availability : protection against interference with the means to access the resources.

# **Security Policies & Mechanisms**

- ❖ A Security Policy indicates which actions each entity (user, data, service) is allowed or prohibited to take.
  - ❖ E.g., Only an owner is allowed to make transactions to his account. CIA properties.
- ❖ A Security Mechanism implements and enforces the policy
  - **Encryption and decryption:** transform data to a form only understandable by authorized users, and vice-versa.
  - **Authentication:** verify the claimed identity of a principal, i.e., user, client, service, process, etc.
  - **Authorization:** verify access rights of principal for resource.
  - **Auditing:** make record of and check access to data and resources. Mainly an offline analysis tool, often ex-post.

# *Designing Secure Systems*

- **Need to make worst-case assumptions about attackers:**
  - exposed interfaces, insecure networks, algorithms and program code available to attackers, attackers may be computationally very powerful
  - Typically design system to withstand a known set of attacks (Attack Model or Attacker Model)
  - Tradeoff between security and performance impact
- **Designing Secure Systems**
  - Traditionally done as a layer on top of existing protocols.  
Three phases:
    - Design security protocol
    - Analyze Protocol Behavior when under attacks
    - Effect on overall performance if there were no attacks

# *Familiar Names for Principals in Security Protocols*

---

Alice	First participant
Bob	Second participant
Carol	Participant in three- and four-party protocols
Dave	Participant in four-party protocols
Eve	Eavesdropper
Mallory	Malicious attacker
Sara	A server

---

# Cryptography Notations

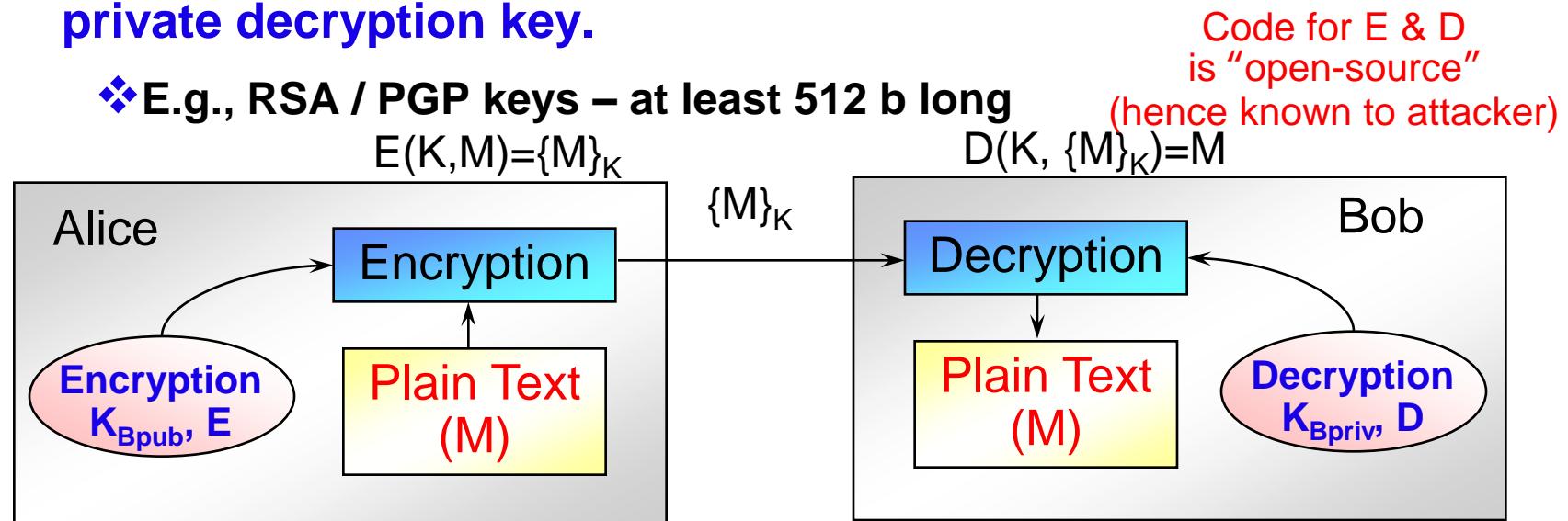
---

$K_A$	Alice's secret key
$K_B$	Bob's secret key
$K_{AB}$	Secret key shared between Alice and Bob
$K_{A\text{priv}}$	Alice's private key (known only to Alice)
$K_{A\text{pub}}$	Alice's public key (published by Alice for all to read)
$\{M\}^K$	(Typical) Message $M$ encrypted with key $K$
$[M]_K$	(Typical) Message $M$ signed with key $K$

---

# Cryptography

- ❖ Encoding (**encryption**) of a message that can only be read (**decryption**) by a key.
- ❖ In **shared key cryptography (symmetric cryptography)** the sender and the recipient know the key, but no one else does.
  - ❖ E.g., DES (Data Encryption Standard) – 56 b key operates on 64 b blocks of data. Notation:  $K_{AB}(M)$ .
  - ❖ How do Alice and Bob get the shared key  $K_{AB}$  to begin with?
- ❖ In **public/private key pairs** messages are encrypted with a published **public key**, and can only be decrypted by a secret **private decryption key**.



# Cryptography

- ❖ Shared versus public/private:
  - ❖ Shared reveals information to too many principles; may need key distribution and repudiation mechanisms
  - ❖ In electronic commerce or wide area applications, public/private key pairs are preferred to shared keys.
  - ❖ Public/private key encrypt/decrypt ops are costly
  - ❖ May use hybrid: pub/pri generates a shared key.
- ❖ Presentation of many next few protocols independent of which keying scheme, viz., shared or pub/priv

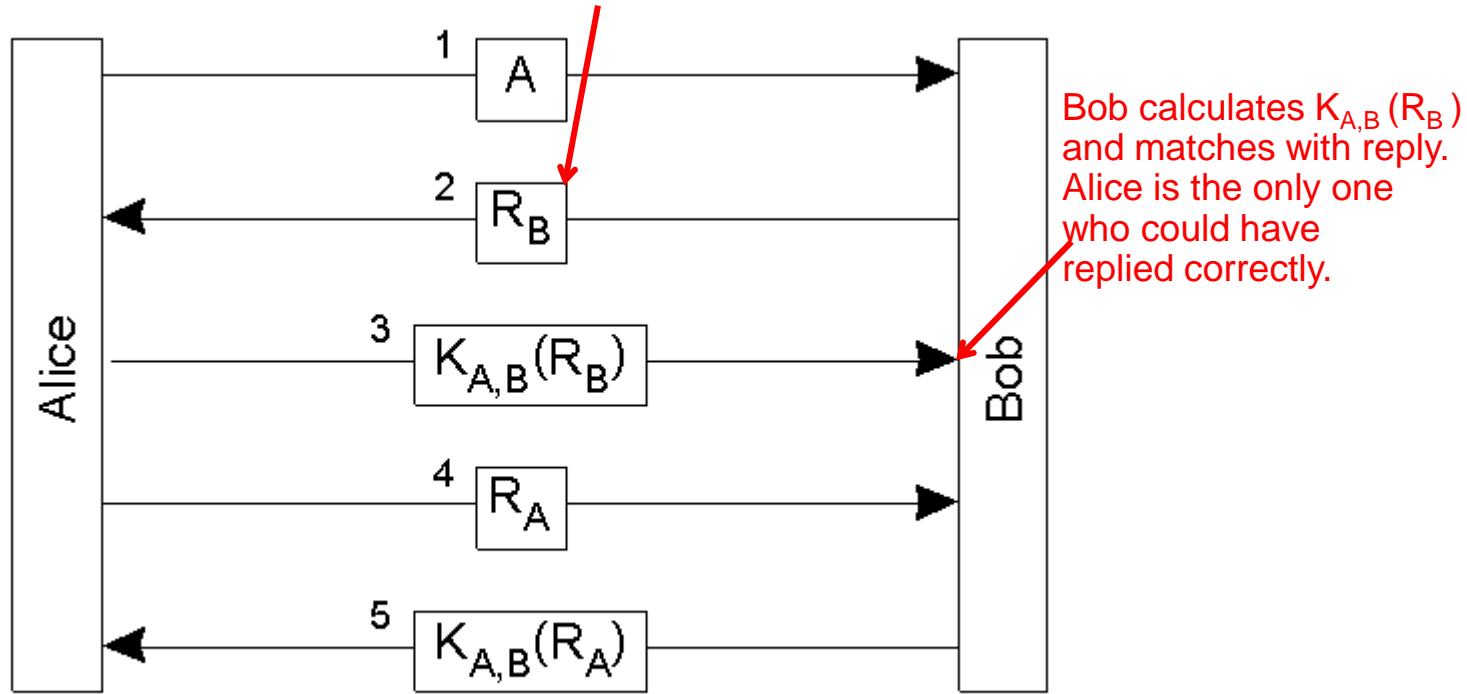
# Authentication

- ❖ Use of cryptography to have two principals verify each others' identities.
  - ❖ Direct authentication: the server uses a shared secret key to authenticate the client.
  - ❖ Indirect authentication: a trusted authentication server (third party) authenticates the client.
  - ❖ The authentication server knows keys of principals and generates temporary shared key (**ticket**) to an authenticated client. The ticket is used for messages in this session.
    - ❖ E.g., Verisign servers

# ***Direct Authentication***

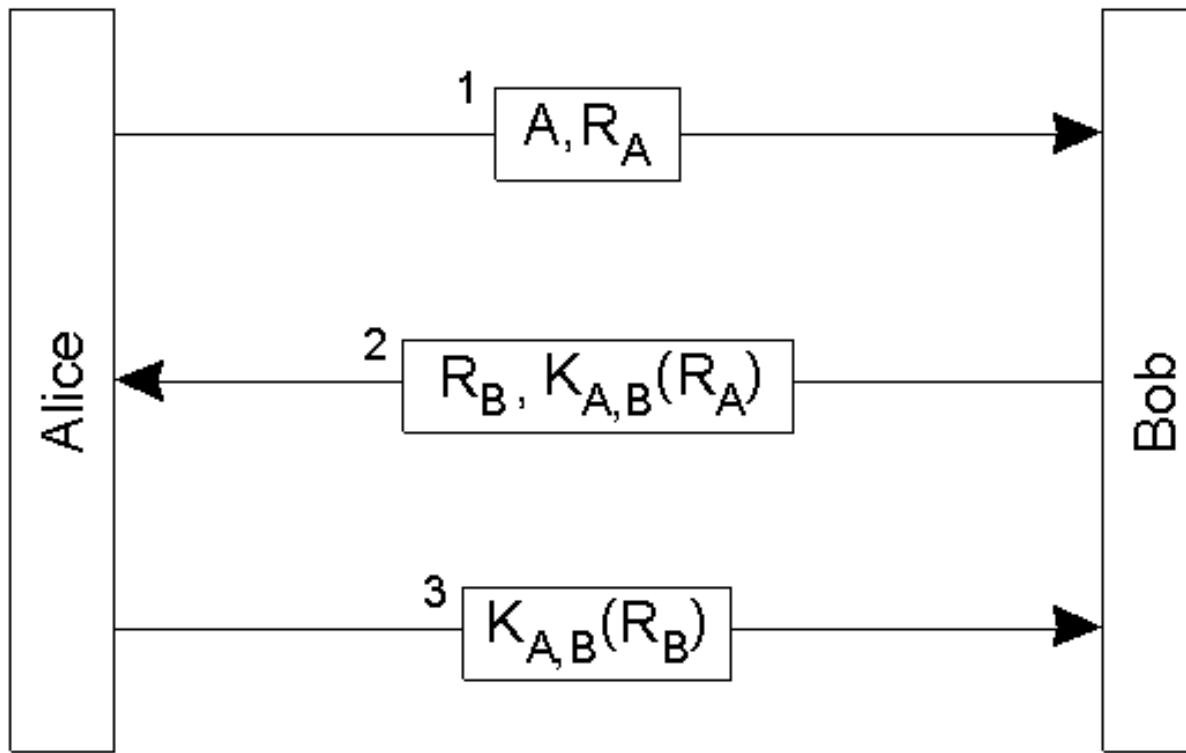
- **Authentication based on a shared secret key.**

“Nonce”=random num,

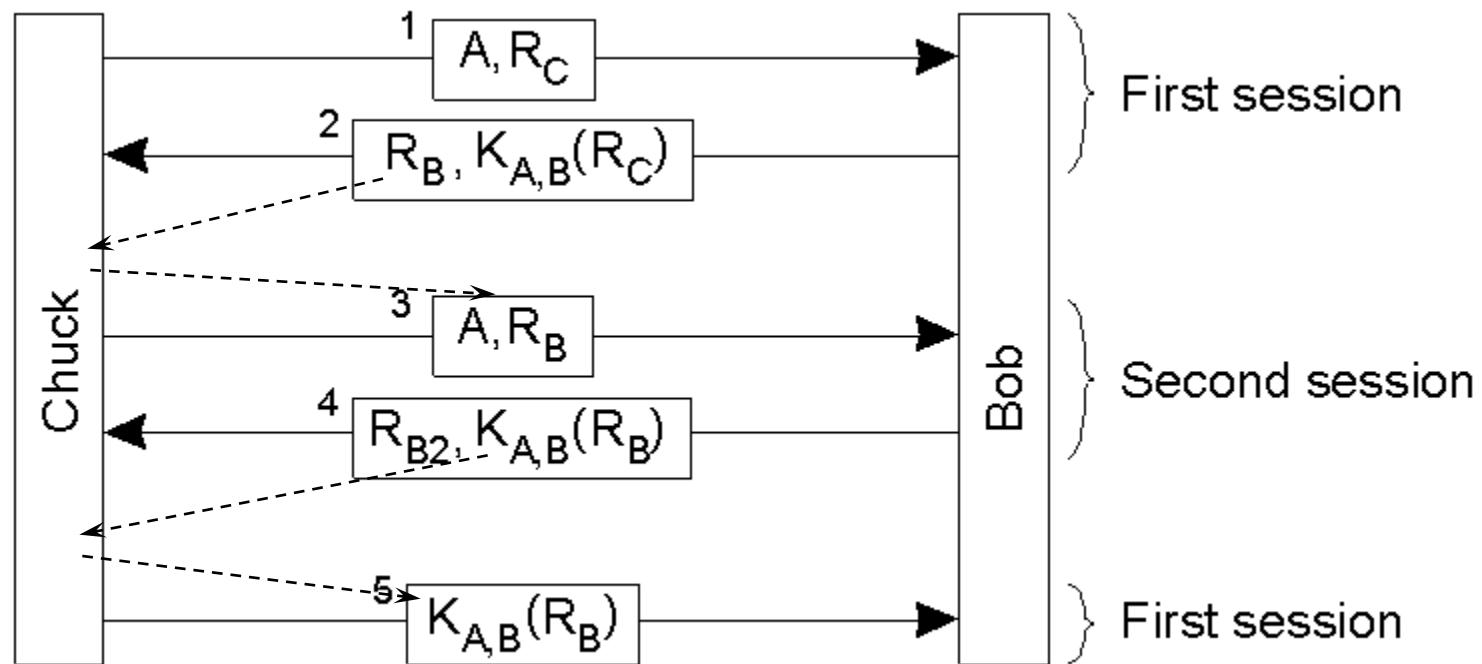


# *“Optimized” Direct Authentication*

- Authentication based on a shared secret key, but using three instead of five messages.



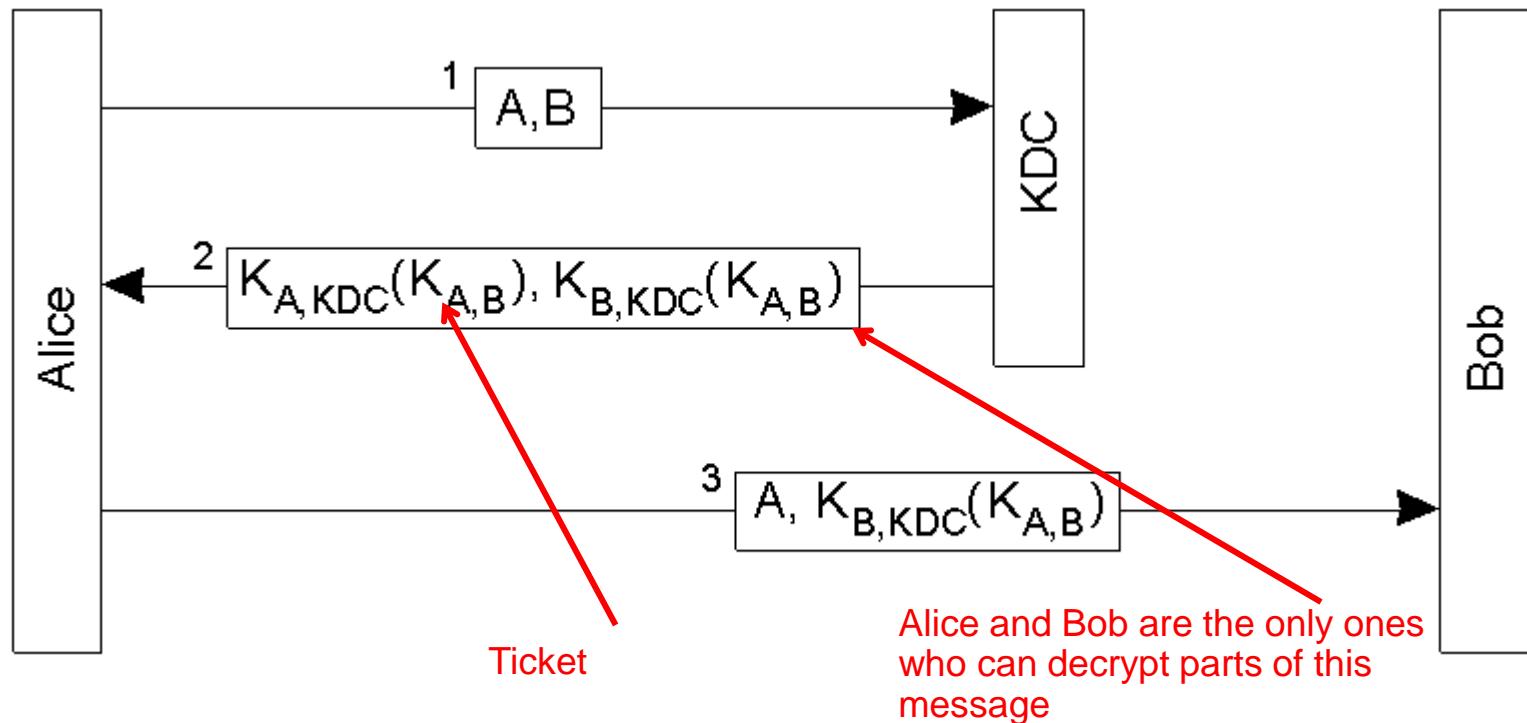
# **Replay/Reflection Attack (with shared keys)**



Steps 1, 2, 5  $\rightarrow$  Chuck is authenticated as Alice

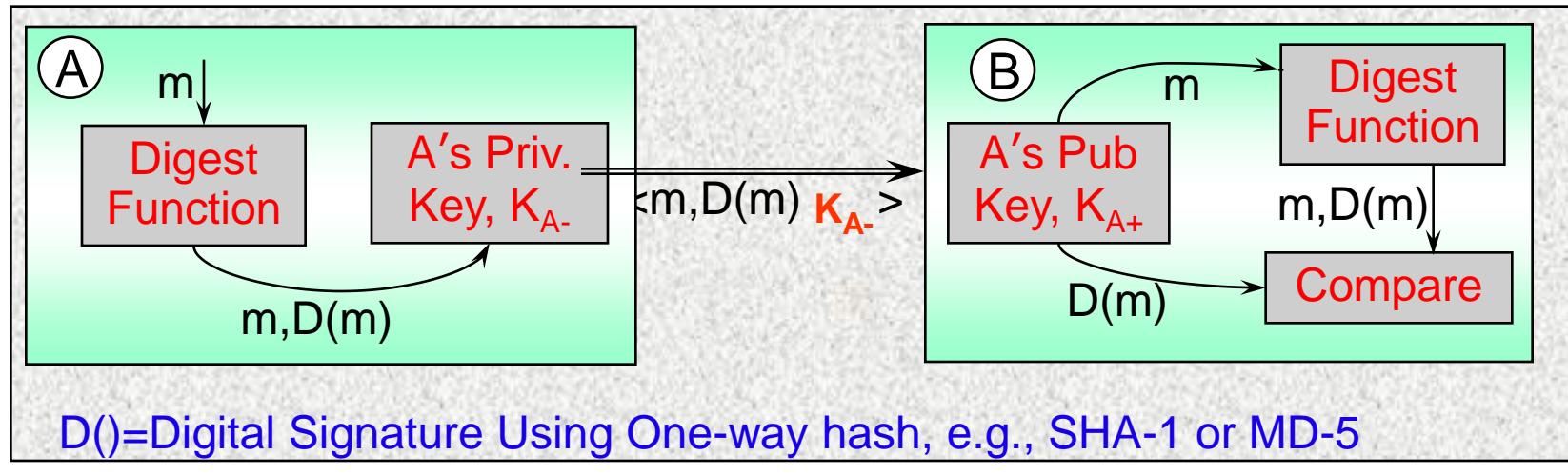
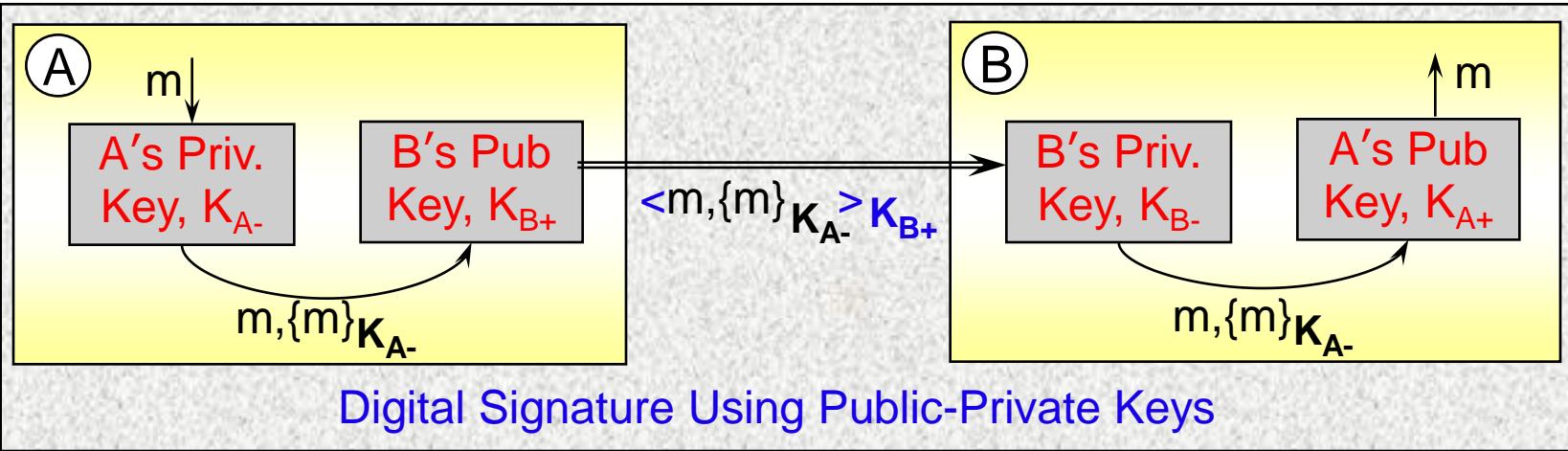
# *Indirect Authentication Using a Key Distribution Center*

- Using a ticket and letting Alice set up a connection to Bob.



# Digital Signatures

- ❖ Signatures need to be authentic, unforgeable, and non-repudiable.

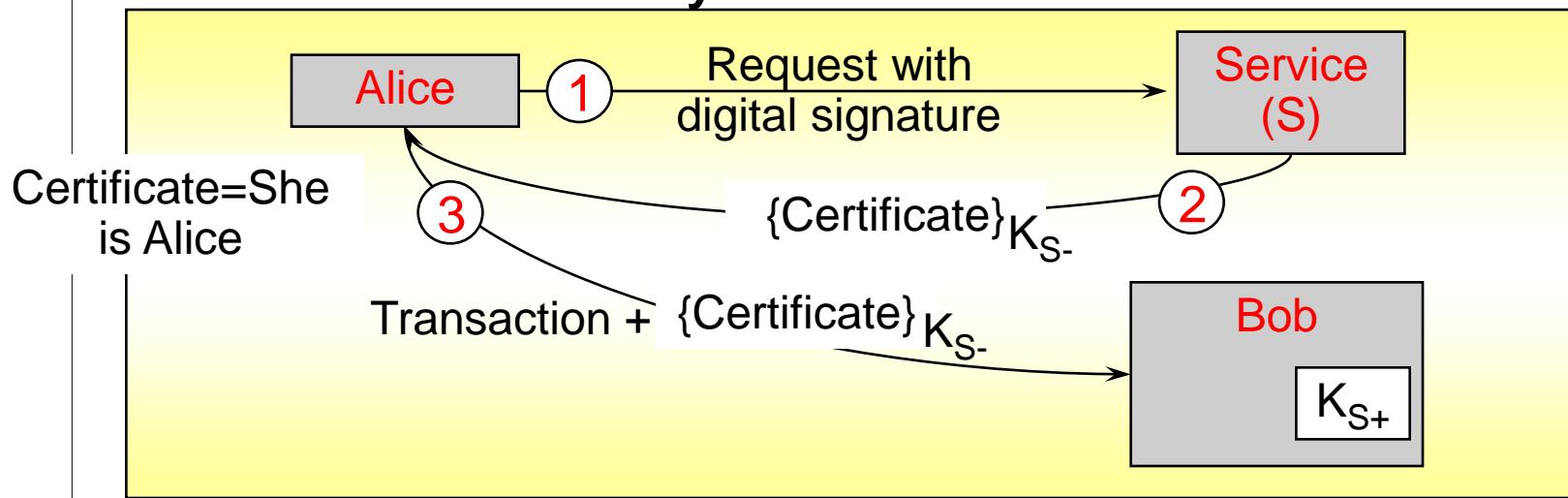


Hashes are fast and have compact output

Lecture 25-20

# Digital Certificates

- ❖ A **digital certificate** is a statement signed by a third party principal, and can be reused
  - ❖ e.g., Verisign Certification Authority (CA)
- ❖ To be useful, certificates must have:
  - ❖ A standard format, for construction and interpretation
  - ❖ A protocol for constructing chains of certificates
  - ❖ A trusted authority at the root of the chain



# **Alice's Bank Account Certificate**

---

1. <i>Certificate type</i>	Account number
2. <i>Name</i>	Alice
3. <i>Account</i>	6262626
4. <i>Certifying authority</i>	Bob's Bank
5. <i>Signature</i>	$\{Digest(field\ 2 + field\ 3)\}_{K_{Bpriv}}$

---

# **Public-Key Certificate for Bob's Bank**

---

1. <i>Certificate type</i>	Public key
2. <i>Name</i>	Bob's Bank
3. <i>Public key</i>	$K_{Bpub}$
4. <i>Certifying authority</i>	Fred – The Bankers Federation
5. <i>Signature</i>	$\{Digest(field\ 2 + field\ 3)\}\ K_{Fpriv}$

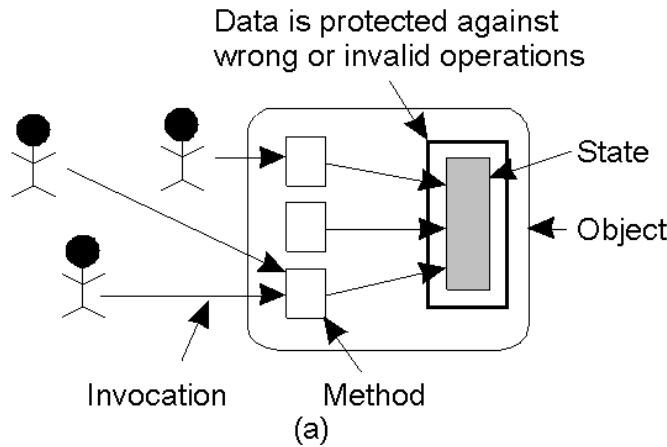
---

(In turn, Fred has a certificate from Verisign, i.e., the root).

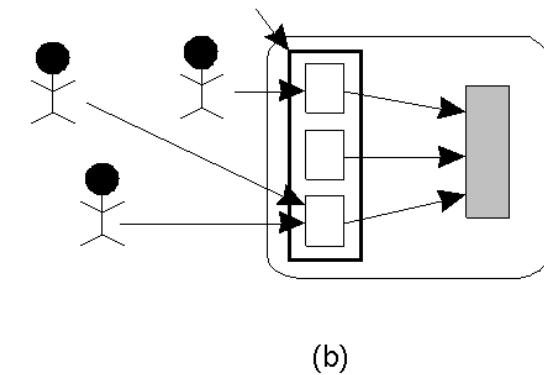
# **Authorization: Access Control**

- ❖ Control of access to resources of a server.
- ❖ A basic form of access control checks <principal, op, resource> requests for:
  - Authenticates the principal.
  - Authorization check for desired op, resource.
- ❖ Access control matrix M (e.g., maintained at server)
  - ❖ Each principal is represented by a row, and each resource object is represented by a column.
  - ❖  $M[s,o]$  lists precisely what operations principal s can request to be carried out on resource o.
  - ❖ Check this before carrying out a requested operation.
  - ❖ M may be sparse.
- ❖ Access control list (ACL)
  - ❖ Each object maintains a list of access rights of principals, i.e., an ACL is some column in M with the empty entries left out.
- ❖ Capability List = row in access control matrix, i.e., per-principal list. May be a signed certificate (verifiable by anyone).

# *Focus of Access Control*

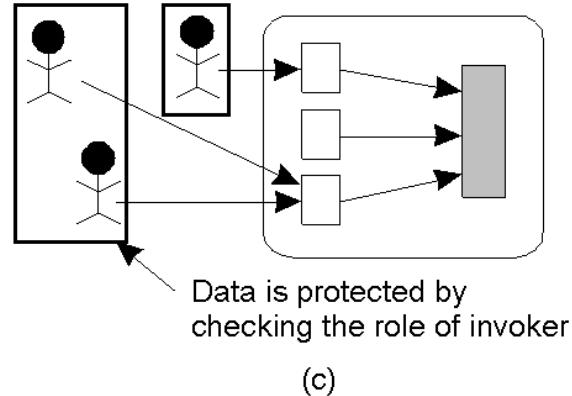


Data is protected against unauthorized invocations



(b)

- **Three approaches for protection against security threats**
- Protection against invalid operations**
  - Protection against unauthorized invocations**
  - Protection against unauthorized users**

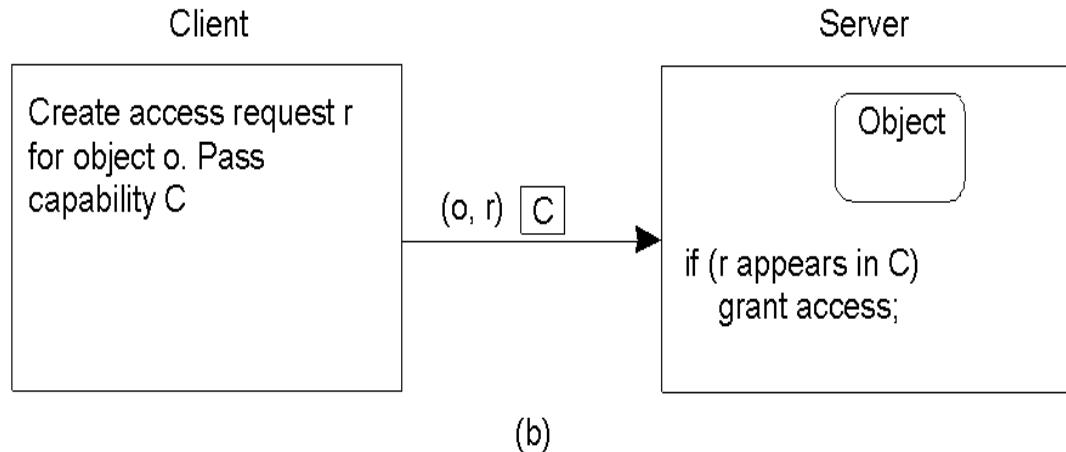
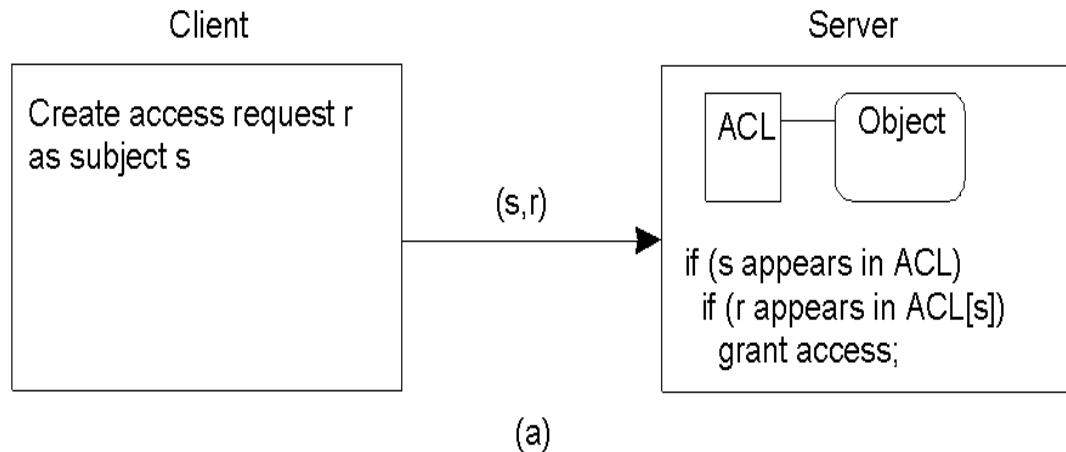


(c)

# ACL and Capability Usage

**Comparison between ACLs and capabilities for protecting objects.**

- a) Using an ACL
- b) Using capabilities.



# Common Web Attacks: XSS

## ❖ XSS = Cross-site Scripting (84% vulnerabilities, according to Symantec in 2007)

- ❖ Most common reported vulnerability to websites
- ❖ Same origin policy: used by sites to enable code within same website to access each other without restrictions, but not across different websites. Browsers often use HTTP cookies for this.
- ❖ XSS exploits/bypasses origin policy
- ❖ Two flavors: (while you're using your favorite bank mybank.com)
  - ❖ (More frequent) Non-persistent: You click on a link that takes you to mybank.com (the real one), but the link contains malicious code. This code executes in your browser with same credentials as mybank.com, e.g., code could send your cookie to attacker who then exploits info from inside it.
  - ❖ Persistent: Attacker uses info from inside your cookie to pretend to be you. E.g., someone adds a script to their myspace profile, and when you visit it, the script executes with your authentication.
- ❖ Prevention
  - ❖ Better cookie handling, e.g., tie cookie to an IP address, or make cookie unavailable to client-side scripts
  - ❖ Disable scripts

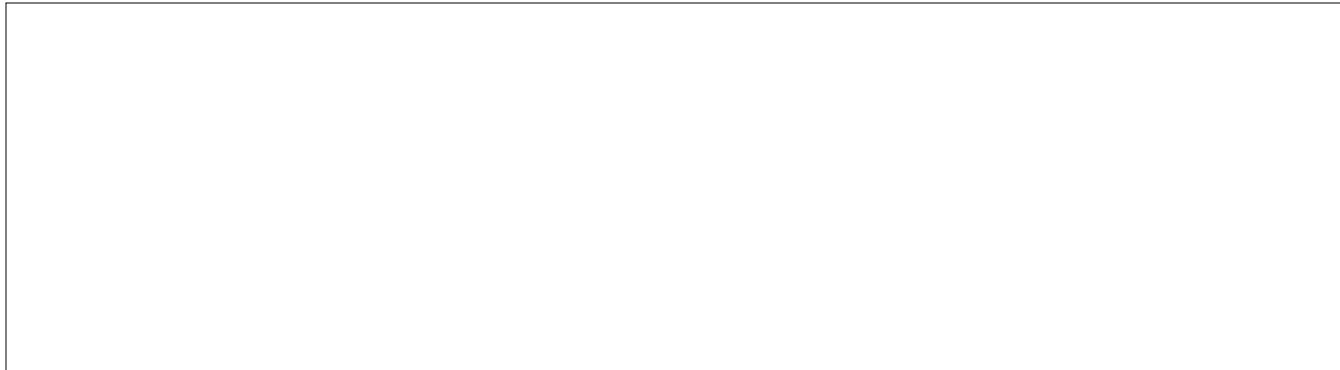
# **Confused Web Attacks: Deputy Attacks**

- ❖ Exploits user's trust in user's deputy (often the browser)
- ❖ Clickjacking: “layer” a malicious site over/under another legitimate site. When user clicks on legitimate site, they’re actually clicking on malicious site.
  - ❖ Prevention: Firefox NoScript feature
- ❖ CSRF (Cross-Site Request Forgery): Different from XSS in many ways: carried out from user’s IP address.
  - ❖ While browsing mybank.com, you open another tab to browse a Google group
  - ❖ Someone there has posted a link <a href=http://mybank.com/transfer?amount=all&to\_account=Alice>Like this page!</a>
  - ❖ You like that page by clicking. Boom!
- ❖ **Prevention**
  - ❖ Better cookie handling, e.g., timeout
  - ❖ Authentication for each operation

# **Announcements**

- **Next two lectures: Byzantine Fault tolerance and Distributed graph processing**
  - Imranul Hoque
  - Abhishek Verma
- **HW4 due next Tue (Dec 4).**
- **MP4 – you should have an original design and have started coding**
- **Midterms (including regrades) + graded HW3's**
  - Please collect them now

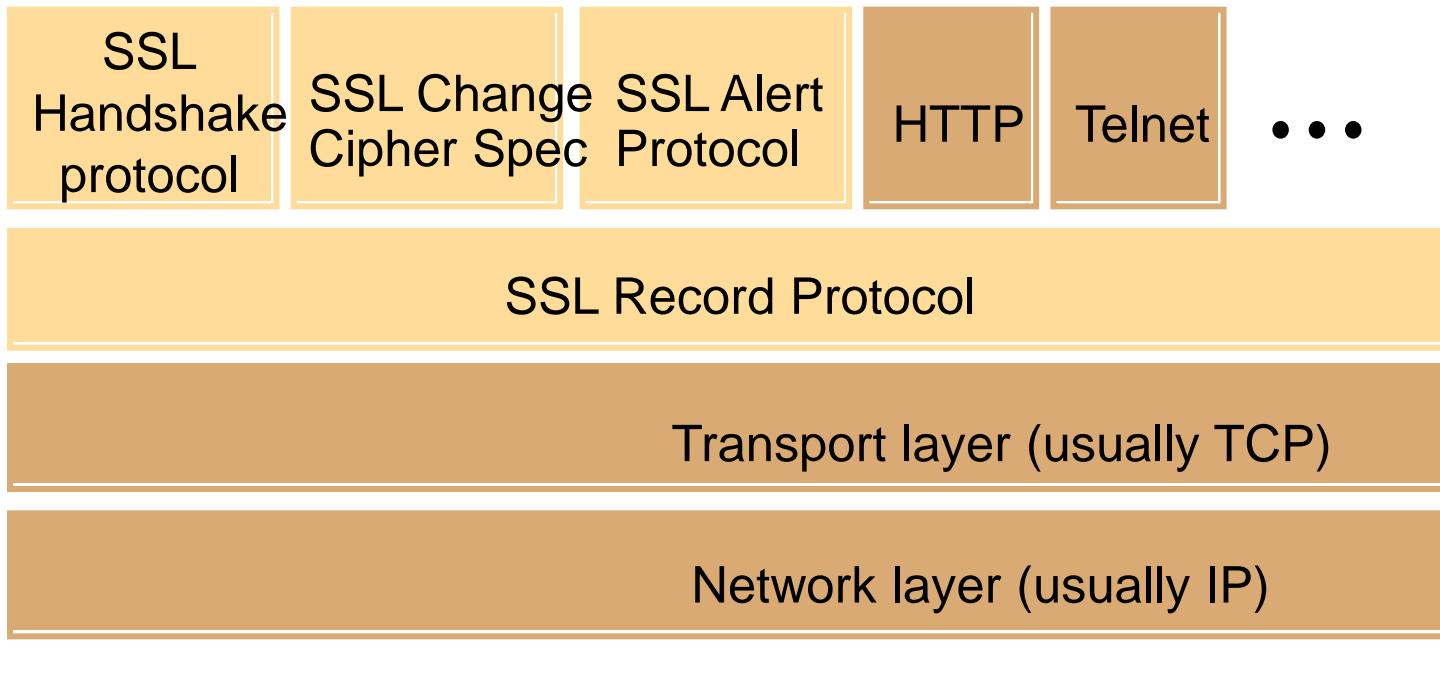
## **Optional Slides (Not Covered)**



# **Secure Socket Layer Protocol**

- ❖ **SSL** was developed by Netscape for electronic transaction security.
- ❖ A protocol layer is added below the application layer for:
  - Negotiating encryption and authentication methods.
  - Bootstrapping secure communication
- ❖ It consists of two layers:
  - The **Record Protocol Layer** implements a secure channel by encrypting and authenticating messages
  - The **Handshake Layer** establishes and maintains a secure session between two nodes.

# ***SSL Protocol Stack***

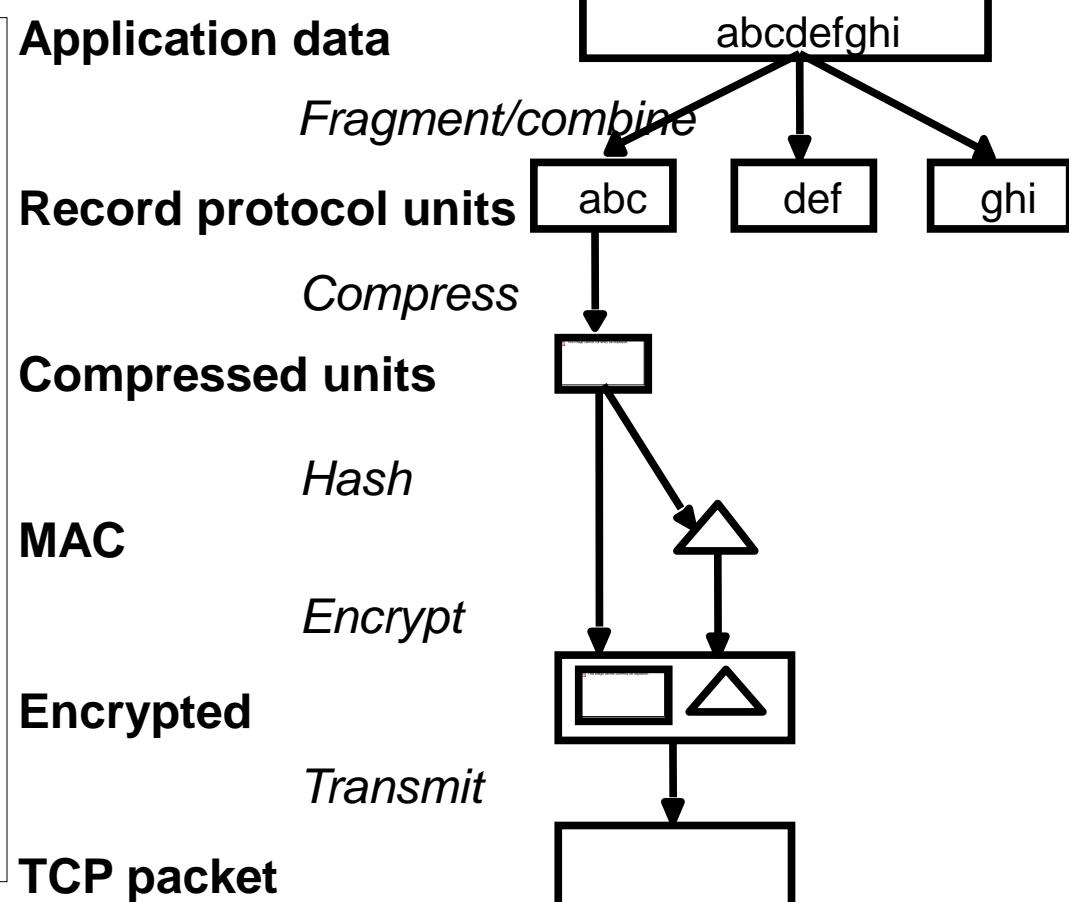


SSL protocols:

Other protocols:

# SSL Record Protocol

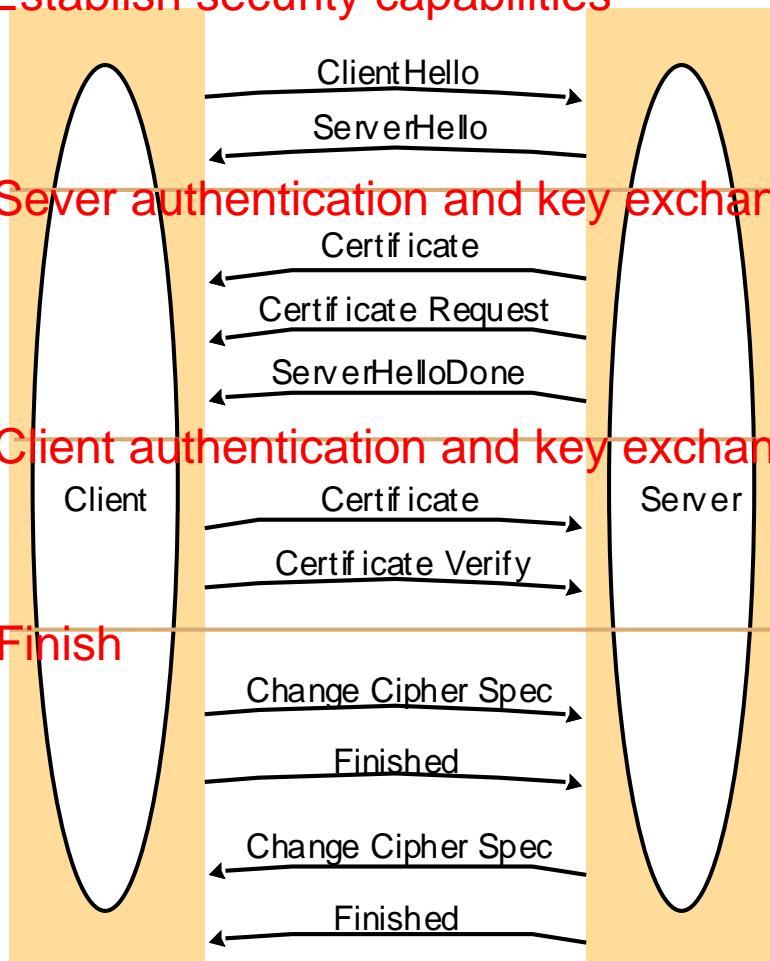
- The record protocol takes an application message to be transmitted,
  - fragments the data into manageable blocks,
  - optionally compresses the data,
  - computes a message authentication code (MAC),
  - encrypts and
  - adds a header.



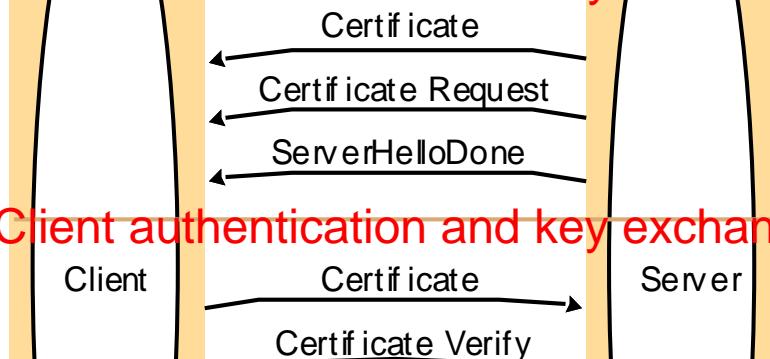
# SSL Handshake Protocol

Cipher suite: a list of cryptographic algorithm supported by the client

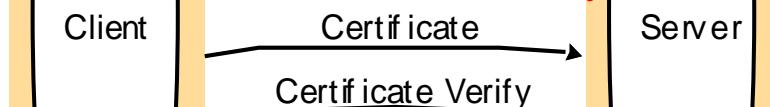
## Phase 1: Establish security capabilities



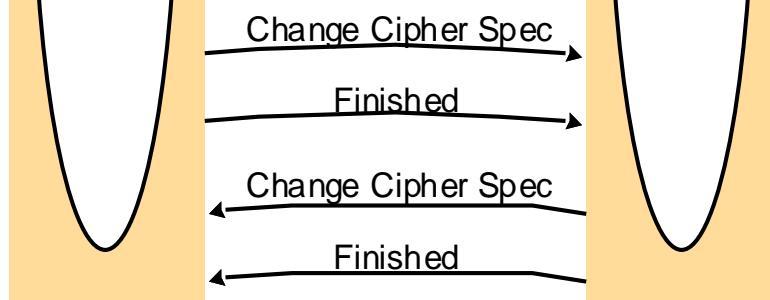
## Phase 2: Server authentication and key exchange



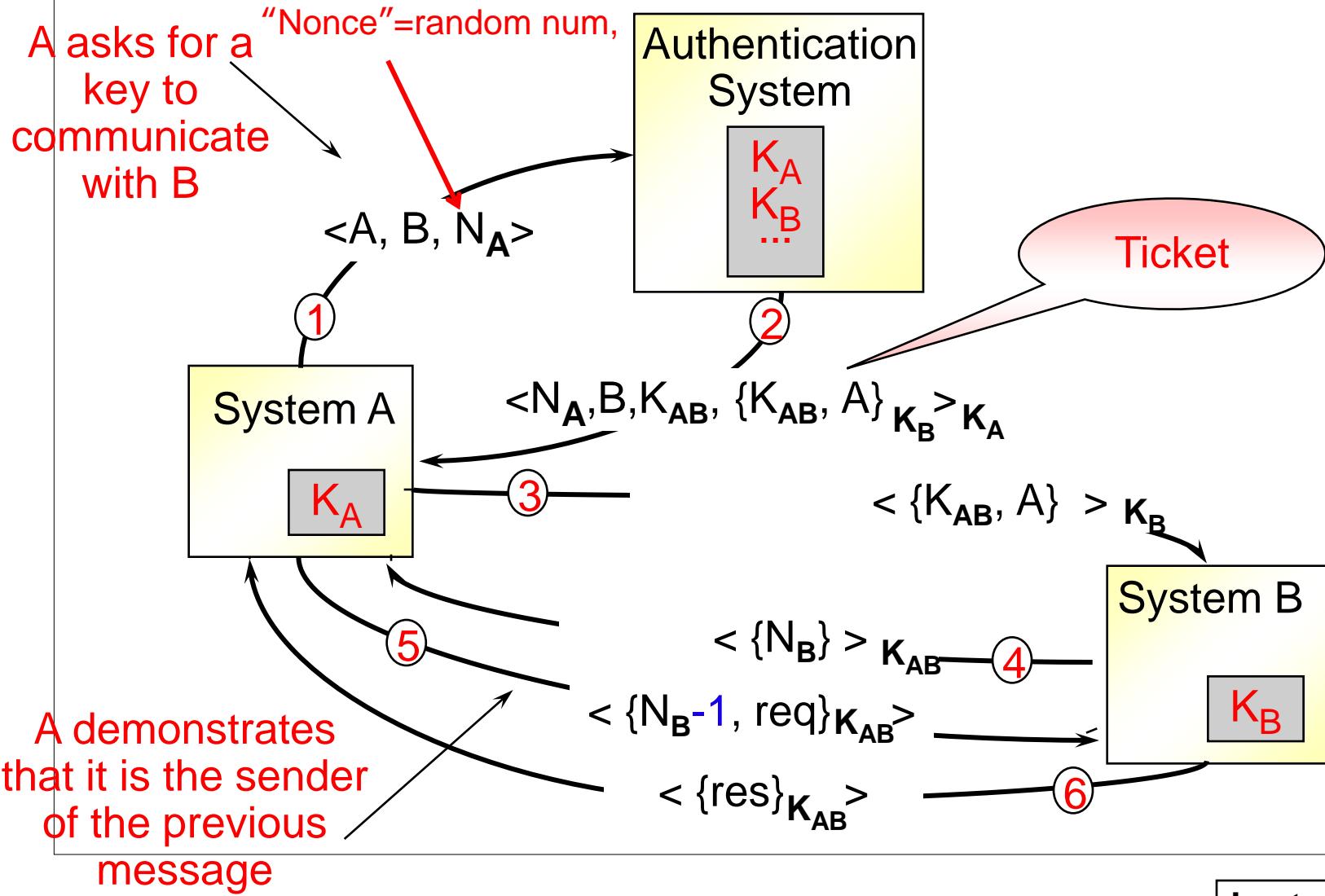
## Phase 3: Client authentication and key exchange



## Phase 4: Finish

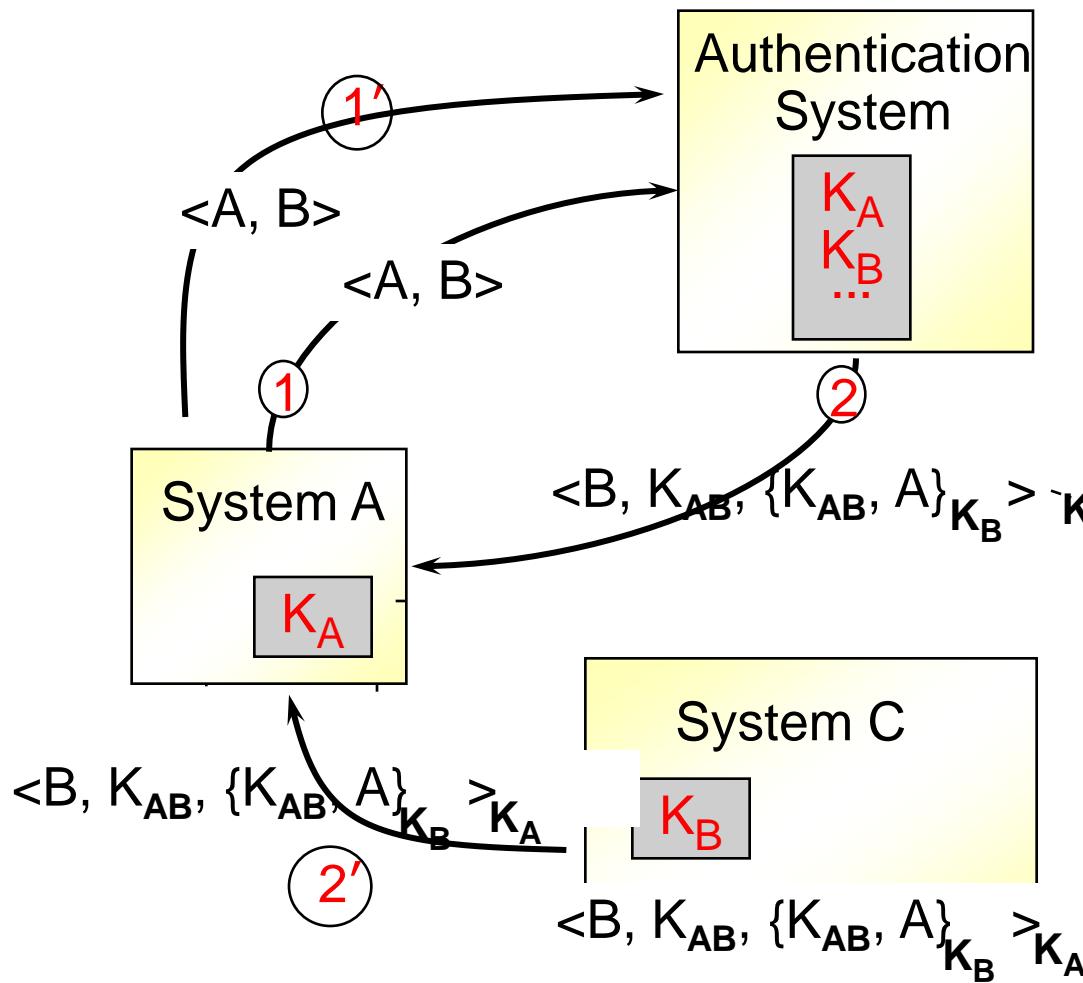


# Needham-Schroeder Authentication



# Why Do We Need Nonce $N_A$ in Message 1?

Because we need to relate message 2 to message 1



Chuck has stolen  $K_B$  and intercepted message 2. It can masquerade as the authentication system.

# Needham–Schroeder Secret-key Authentication Protocol

Variant used for authentication in Windows 2K (slightly modified – nonce added to msg 3)

Header	Message	Notes
1. A->S:	$A, B, N_A$	A requests S to supply a key for communication with B.
2. S->A:	$\{N_A, B, K_{AB}, \{K_{AB}, A\}_{K_B}\}_{K_A}$	S returns a message encrypted in A's secret key, containing a newly generated key $K_{AB}$ and a 'ticket' encrypted in B's secret key. The nonce $N_A$ demonstrates that the message was sent in response to the preceding one. A believes that S sent the message because only S knows A's secret key.
3. A->B:	$\{K_{AB}, A\}_{K_B}$	A sends the 'ticket' to B.
4. B->A:	$\{N_B\}_{K_{AB}}$	B decrypts the ticket and uses the new key $K_{AB}$ to encrypt another nonce $N_B$ .
5. A->B:	$\{N_B - 1\}_{K_{AB}}$	A demonstrates to B that it was the sender of the previous message by returning an agreed transformation of $N_B$ .

# *Kerberos Authentication*

Read section 7.6.2 from text

# Access Control

- ❖ Notion of **protection domain** for a collection of processes:
  - ❖ A protection domain is a set of (object, access rights) pairs kept by a server.
  - ❖ A protection domain is created for each principal when it starts
  - ❖ Unix: each (uid,gid) pair spans a protection domain, e.g., user parts of two processes with same (uid, gid) pair have identical access rights.
  - ❖ Whenever a principal requests an operation to be carried out on an object, the access control monitor checks if the principal belongs to the object's domain, and then if the request is allowed for that object.
- ❖ Each principal can carry a certificate listing the groups it belongs to.
  - ❖ The certificate should be protected by a digital signature.