Distributed Systems

a.y. 2023/2024

Distributed Systems:

Security

- Security policies
- Security mechanisms
- Cryptography
- Communication
- Authorization

Security Threats, Policies, and Mechanisms

- Types of security threats:
  - Interception
  - Interruption
  - Modification
  - Fabrication

Main threats classes ...:

- Leakage
- Tampering
- Vandalism

Threats not defeated by secure channels or other cryptographic techniques

- Denial of service attacks

  Deliberately excessive use of resources to the extent that they are not available to legitimate users

- Trojan horses and other viruses
  - Viruses can only enter computers when program code is imported.
  - But users often require new programs, for example:
    - New software installation
    - Mobile code downloaded dynamically by existing software (e.g. Java applets)
    - Accidental execution of programs transmitted surreptitiously
  - Defences: code authentication (signed code), code validation (type checking, proof), sandboxing.

Threats and forms of attack

- Eavesdropping
  - obtaining private or secret information
- Masquerading
  - assuming the identity of another user/principal
- Message tampering
  - altering the content of messages in transit
    - man in the middle attack (tampers with the secure channel mechanism)
- Replying
  - storing secure messages and sending them at a later date

*

Security Threats, Policies, and Mechanisms

- Types of security mechanisms:
  - Encryption
  - Authentication
  - Authorization
  - Auditing

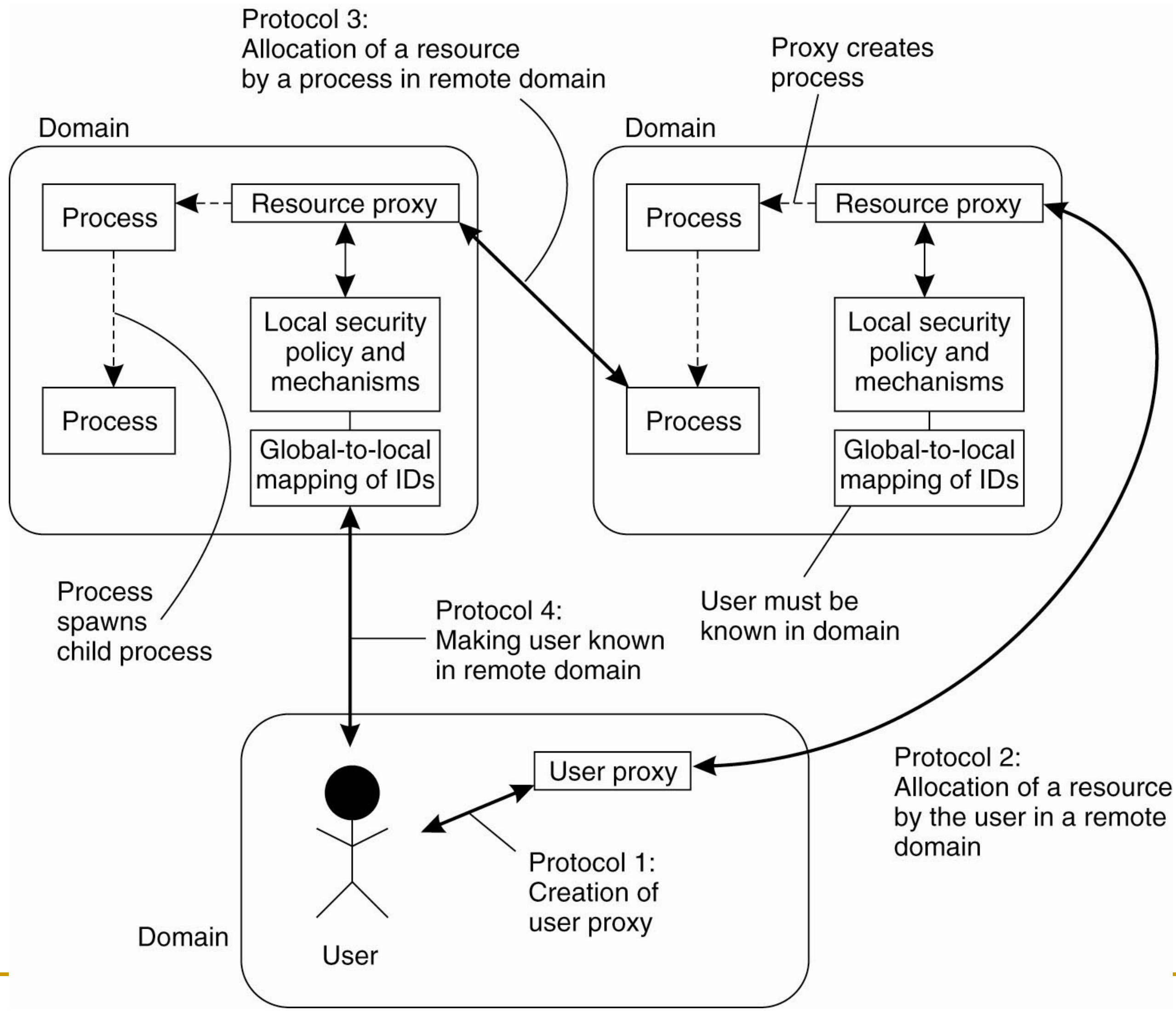
Example: The Globus Security Architecture

1. The environment consists of multiple administrative domains.
2. Local operations are subject to a local domain security policy only.
3. Global operations require the initiator to be known in each domain where the operation is carried out.

Example: The Globus Security Architecture

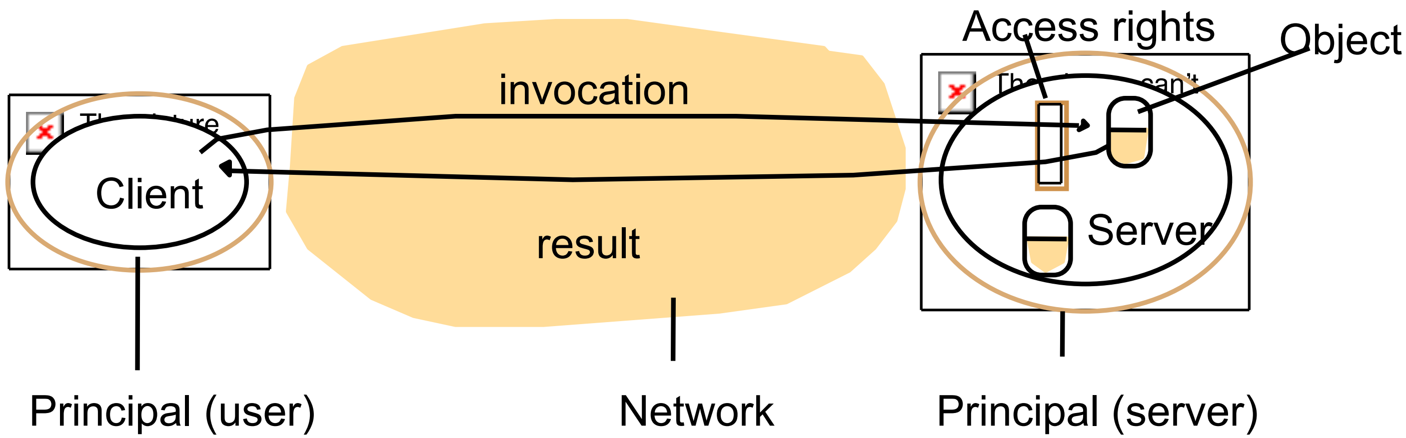
4. Operations between entities in different domains require mutual authentication.
5. Global authentication replaces local authentication.
6. Controlling access to resources is subject to local security only.
7. Users can delegate rights to processes.
8. A group of processes in the same domain can share credentials.

Example: The Globus Security Architecture



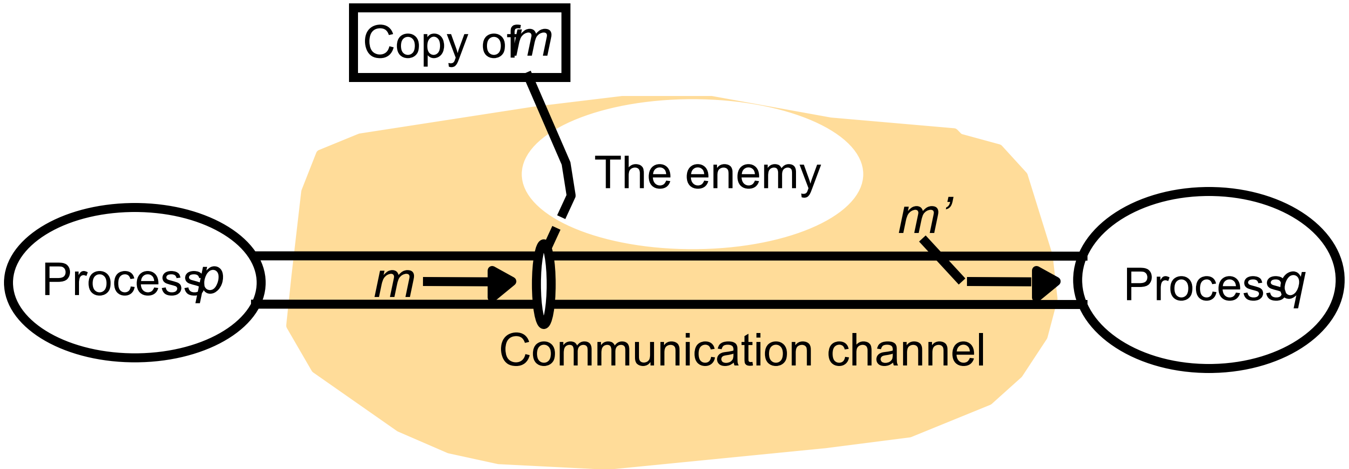
- Basic techniques
  - Cryptographic techniques
  - Secrecy
  - Authentication
  - Certificates and credentials
  - Access control
- Symmetric and asymmetric encryption algorithms
- Digital signatures

Revision: Objects and principals



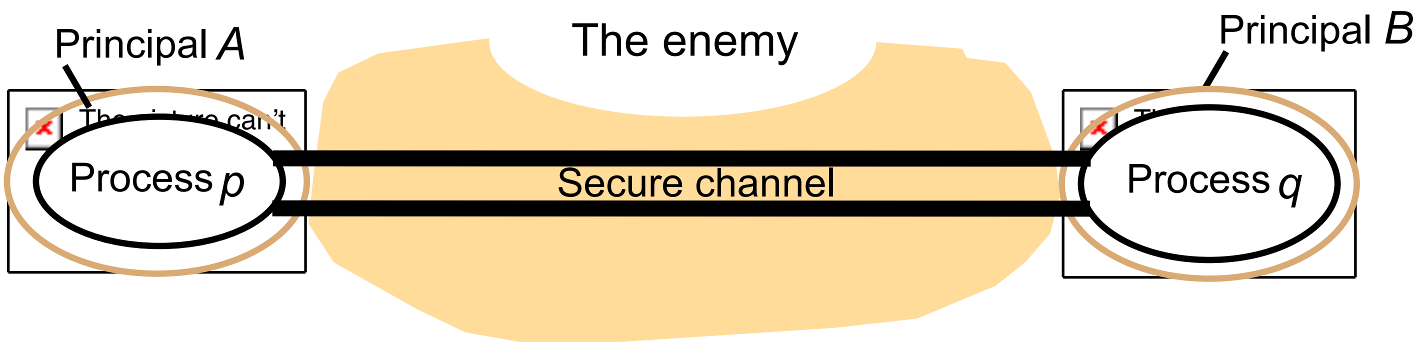
- Object (or resource)
  - Mailbox, system file, part of a commercial web site
- Principal
  - User or process that has authority (rights) to perform actions (identity of principal is important)

Revision: The enemy



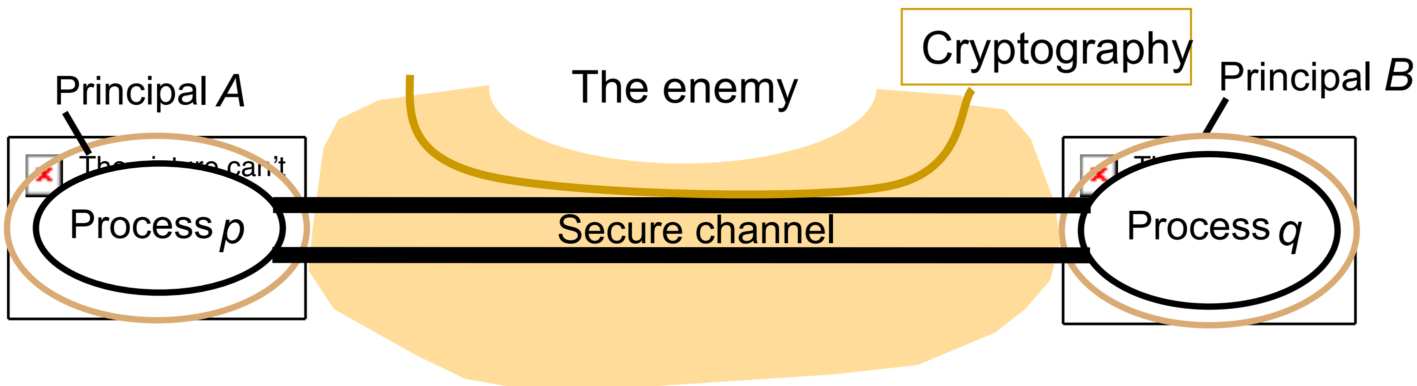
- Attacks
- Enemy (or adversary)
- Threats (to processes, to communication channels, to services)

Revision: Secure channels



- Properties
  - Each process is sure of the identity of the other process
  - Data is private and protected against tampering
  - Protection against repetition and reordering of data

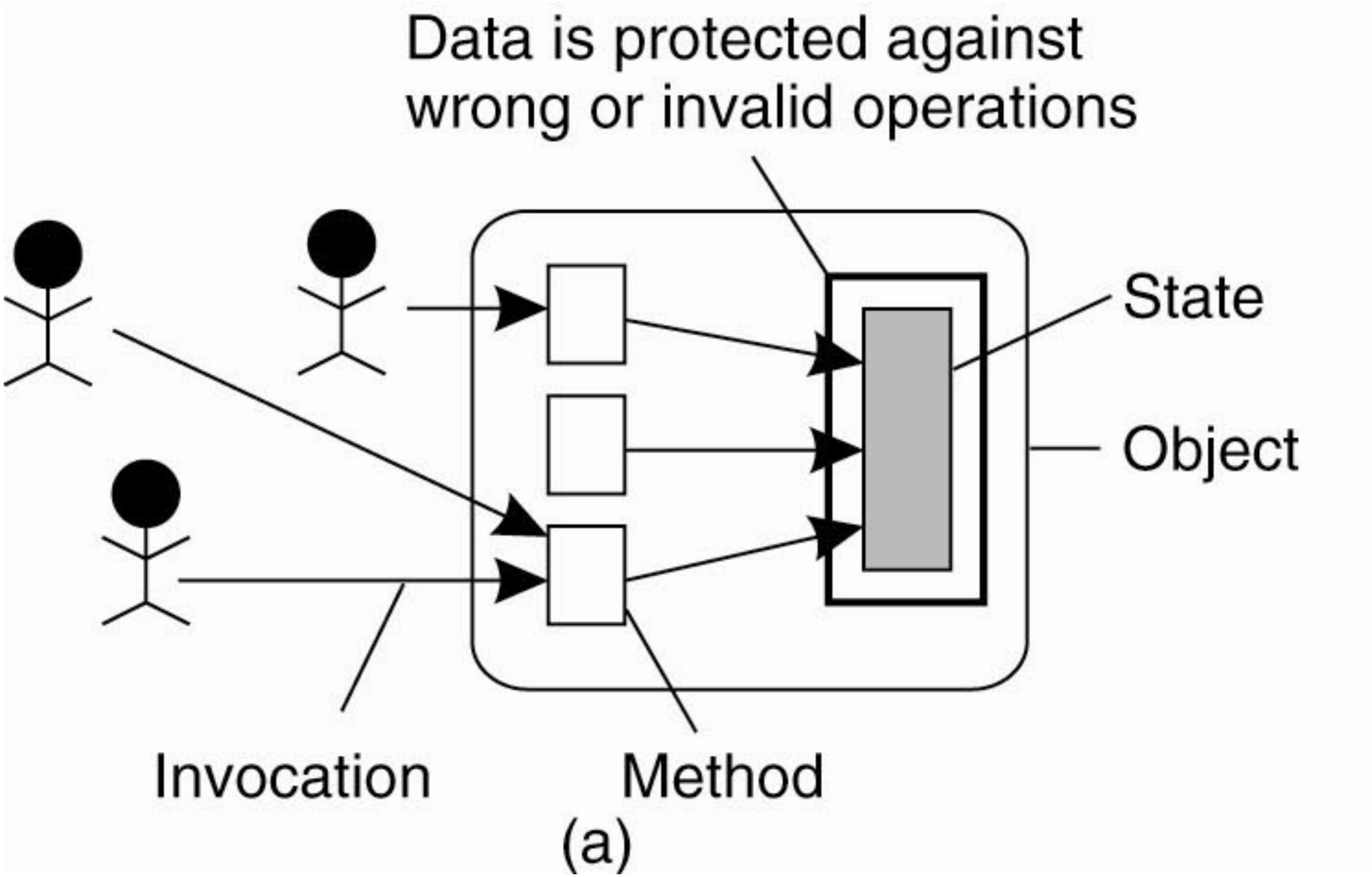
Revision: Secure channels



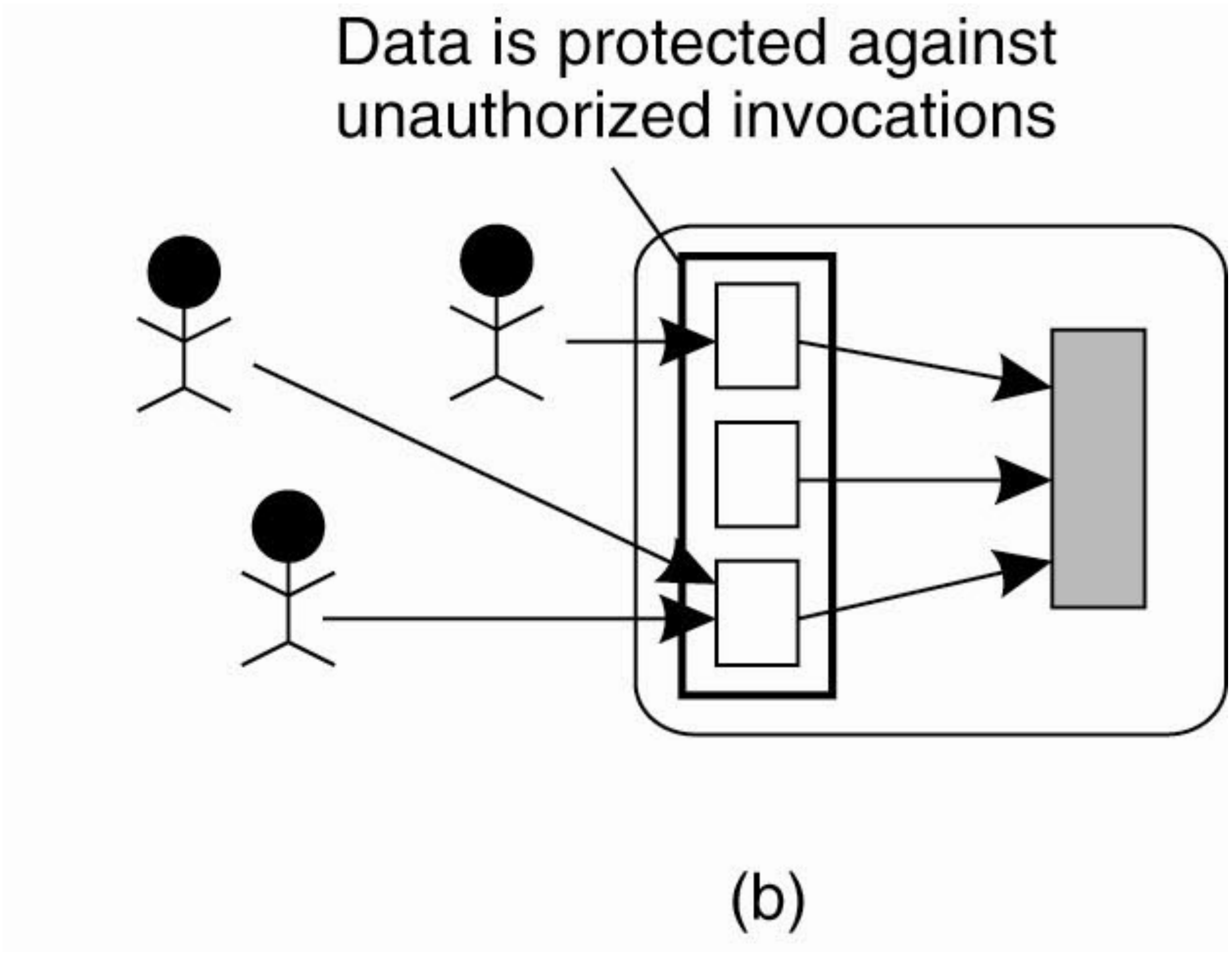
- Employs cryptography
  - Secrecy based on cryptographic concealment
    - *Confusion* and *diffusion*

  - Authentication based on proof of ownership of secrets
    - Conventional shared crypto keys
    - Public/private key pair

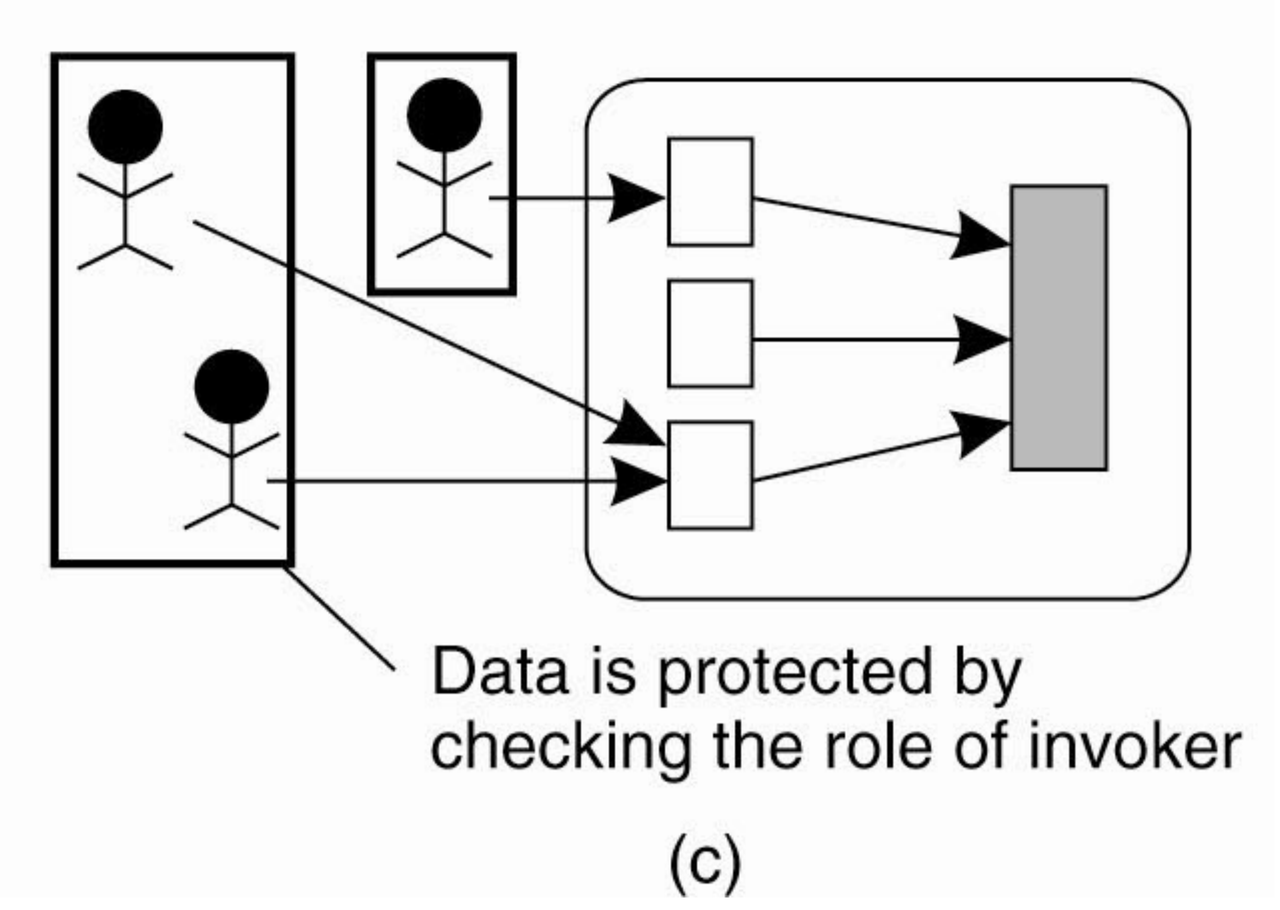
Focus of Control: Protection against invalid operations



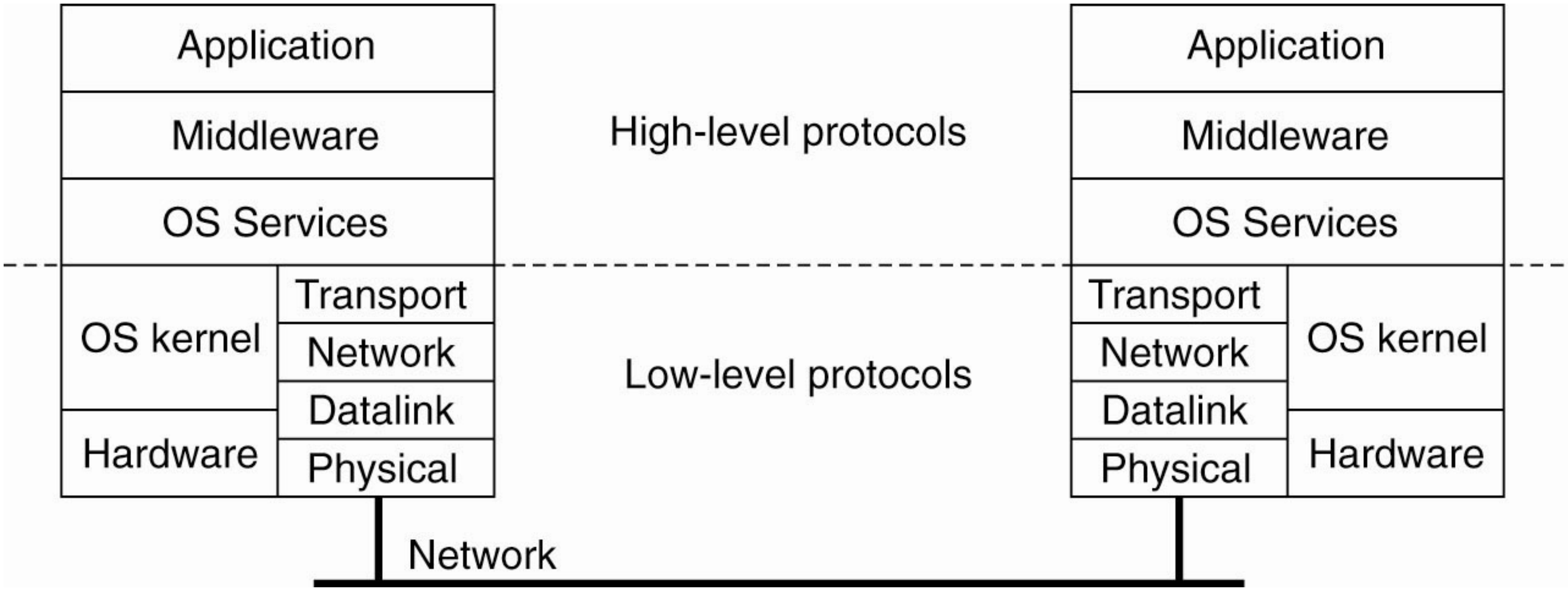
Focus of Control : Protection against unauthorized invocations



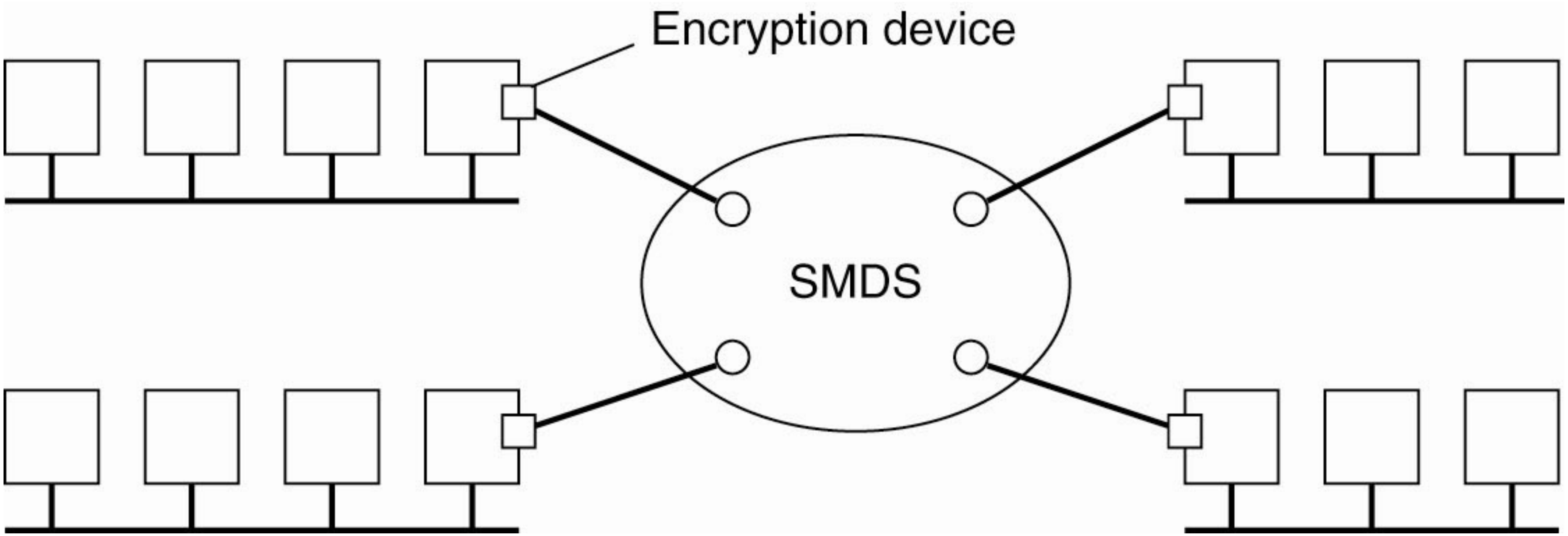
Focus of Control: Protection against unauthorized users.



Layering of Security Mechanisms

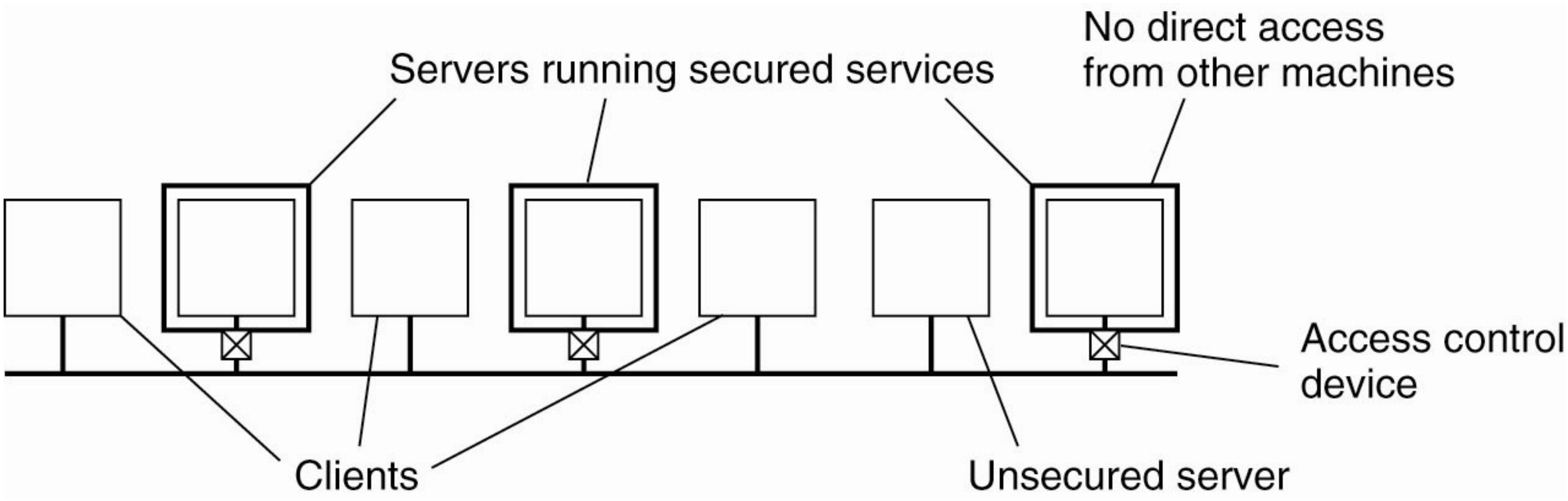


Layering of Security Mechanisms



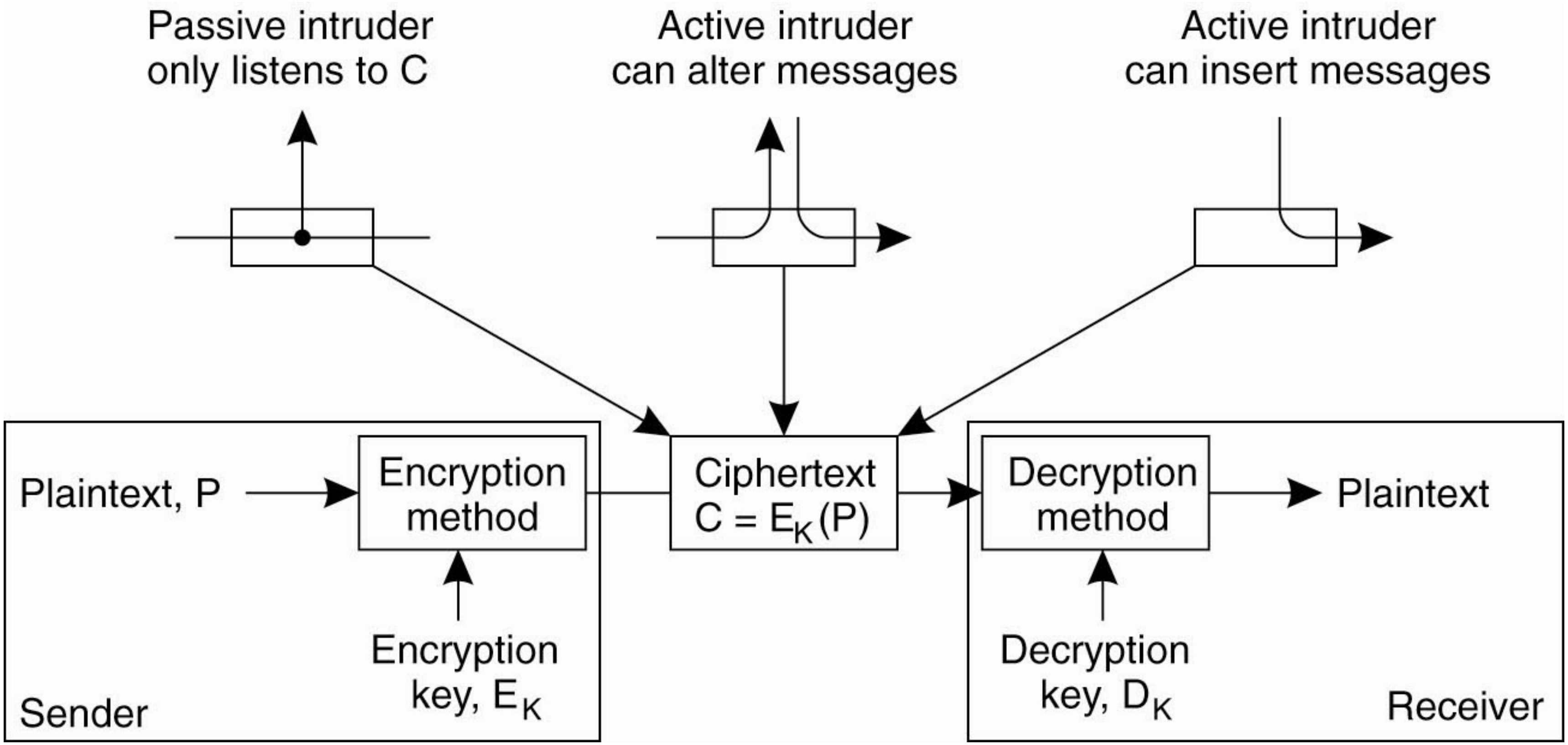
- Several sites connected through a wide-area backbone service.

Distribution of Security Mechanisms



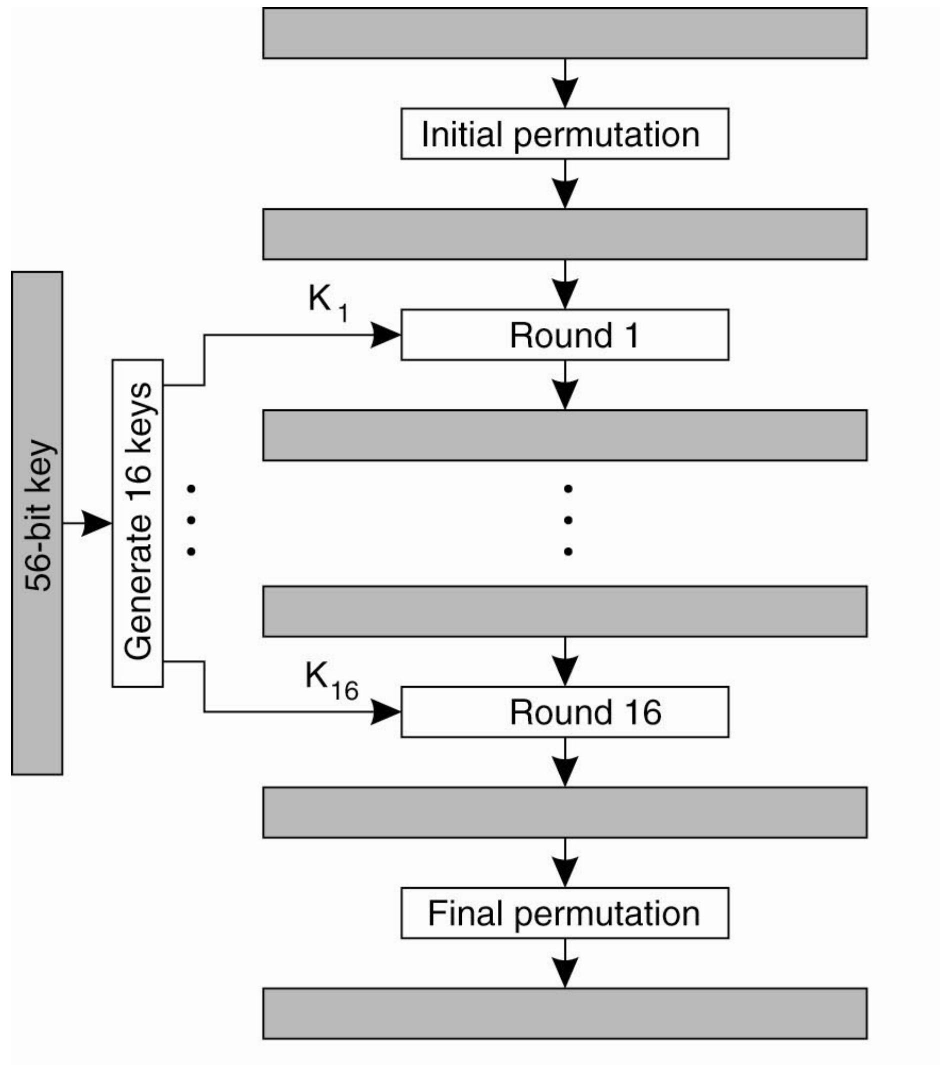
- The principle of RISSC as applied to secure distributed systems.

Cryptography



Symmetric Cryptosystems: DES

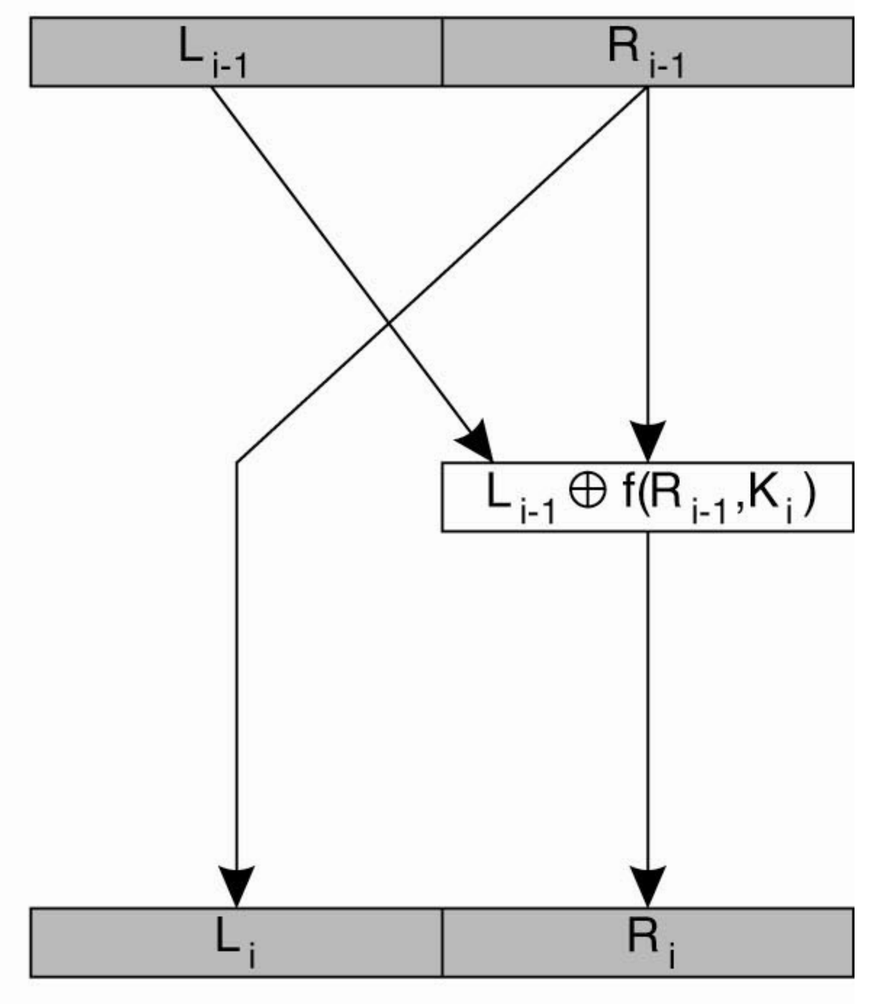
- The principle of DES.



(a)

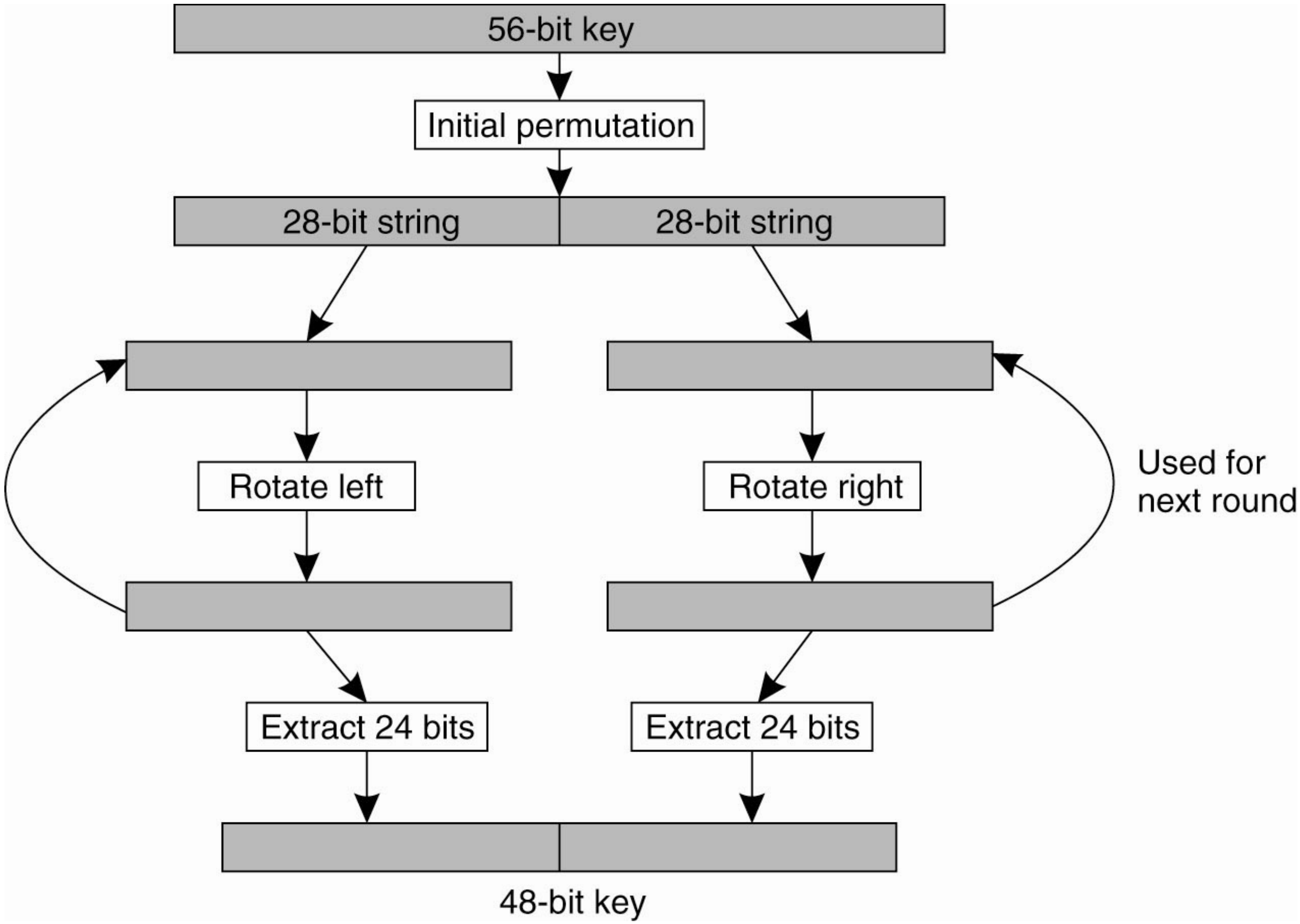
Symmetric Cryptosystems: DES

- one encryption round…



(b)

Symmetric Cryptosystems: DES



- the per-round key generation in DES.

Public-Key Cryptosystems: RSA

- Generating the private and public keys requires four steps:
  1. Choose two very large prime numbers, p and q.
  2. Compute $n = p \times q$ and $z = (p - 1) \times (q - 1)$.
  3. Choose a number d that is relatively prime to z.
  4. Compute the number e such that $e \times d = 1 \bmod z$.

Cryptography

- Notation ...

Notation	Description
$K_{A,B}$	Secret key shared by A and B
K_A^+	Public key of A
K_A^-	Private key of A

Worst case assumptions and design guidelines

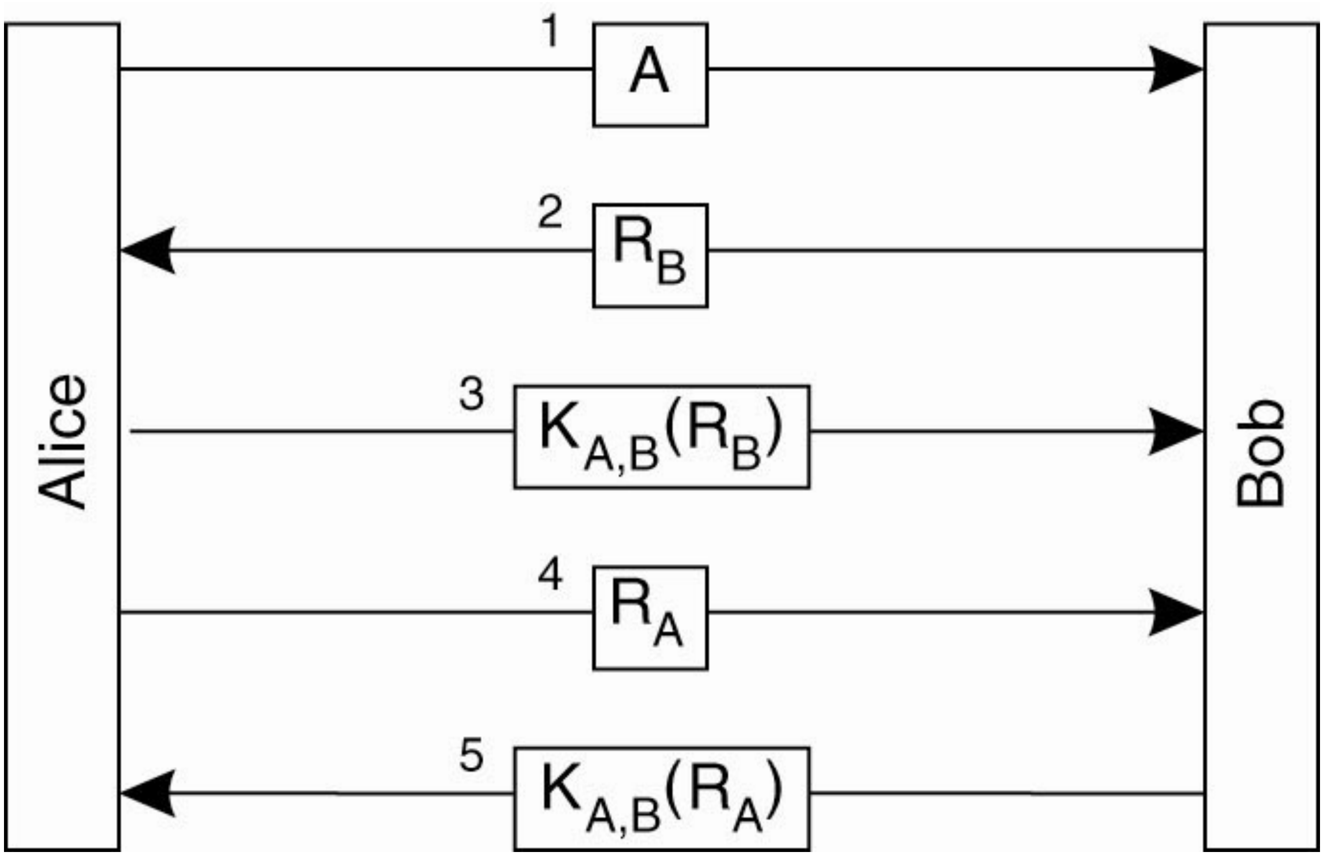
- Interfaces are exposed
- Networks are insecure
- Limit the lifetime and scope of each secret
- Algorithms and program code are available to attackers
- Attackers may have access to large resources
- Minimize the trusted base

Cryptography

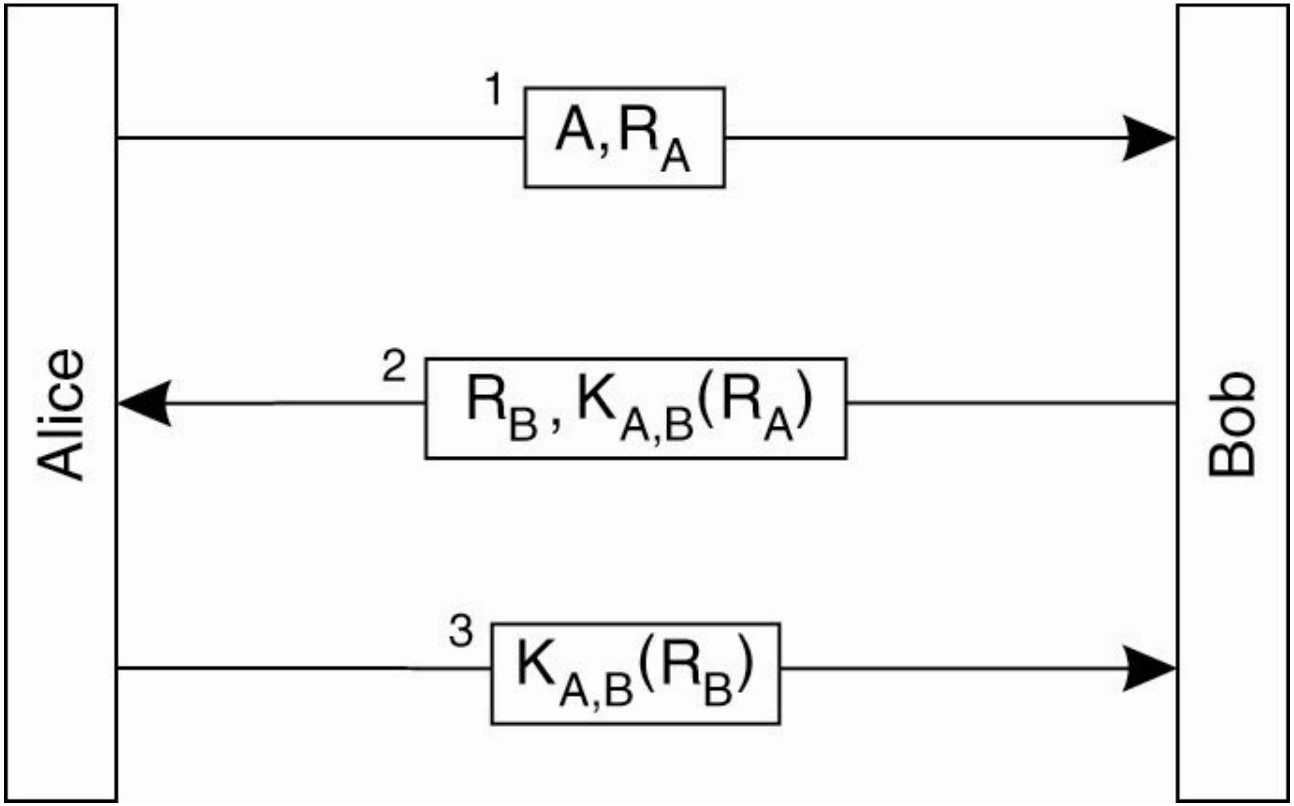
- Notation ...

Notation	Description
$K_{A,B}$	Secret key shared by A and B
K_A^+	Public key of A
K_A^-	Private key of A

Authentication Based on a Shared Secret Key

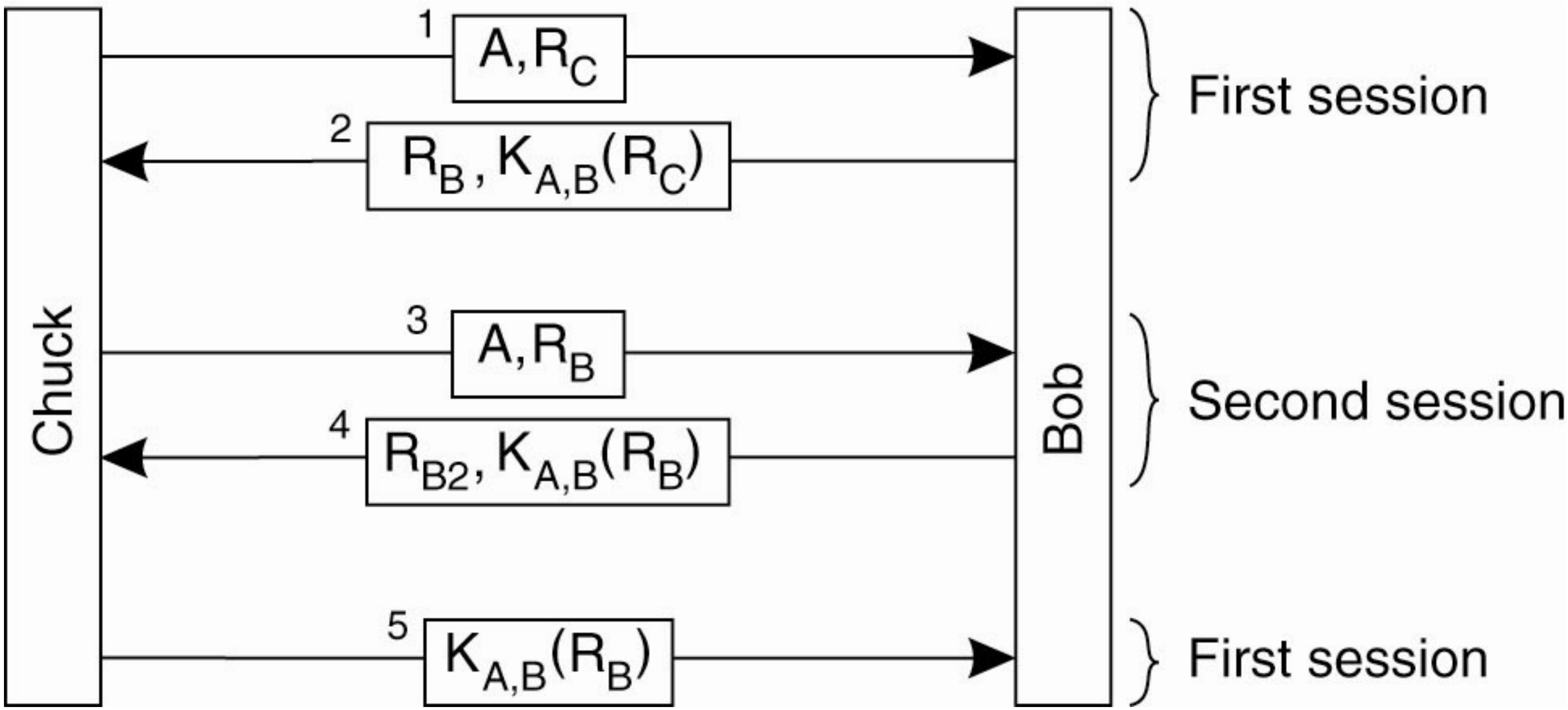


Authentication Based on a Shared Secret Key

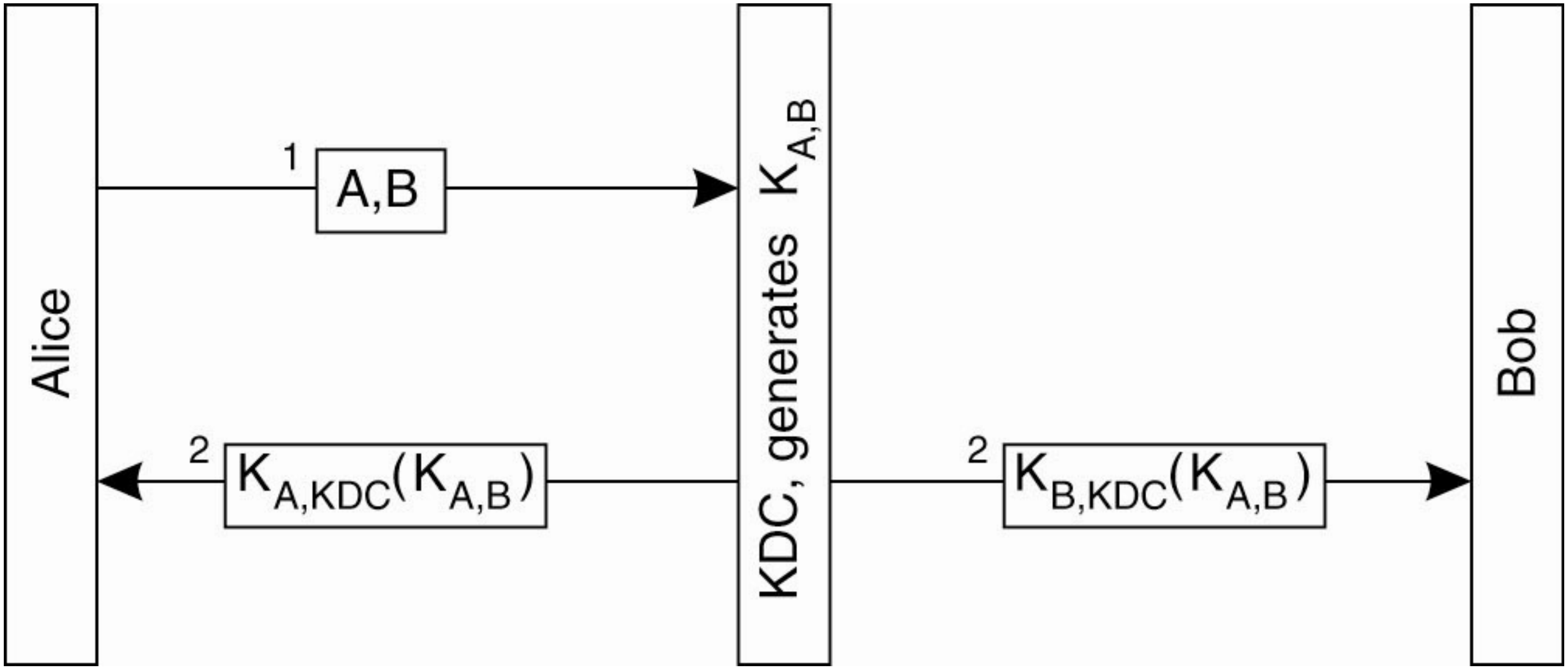


Authentication Based on a Shared Secret Key

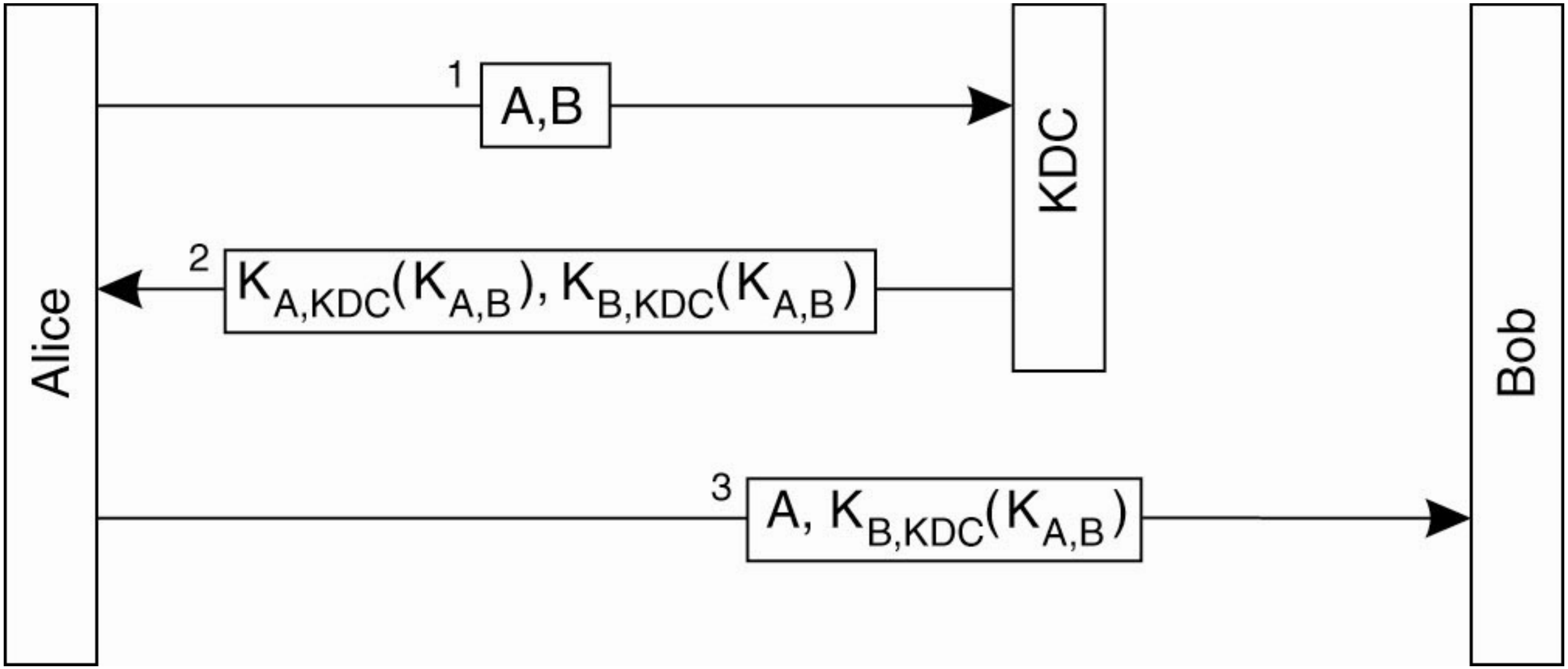
- The reflection attack.



Authentication Using a Key Distribution Center

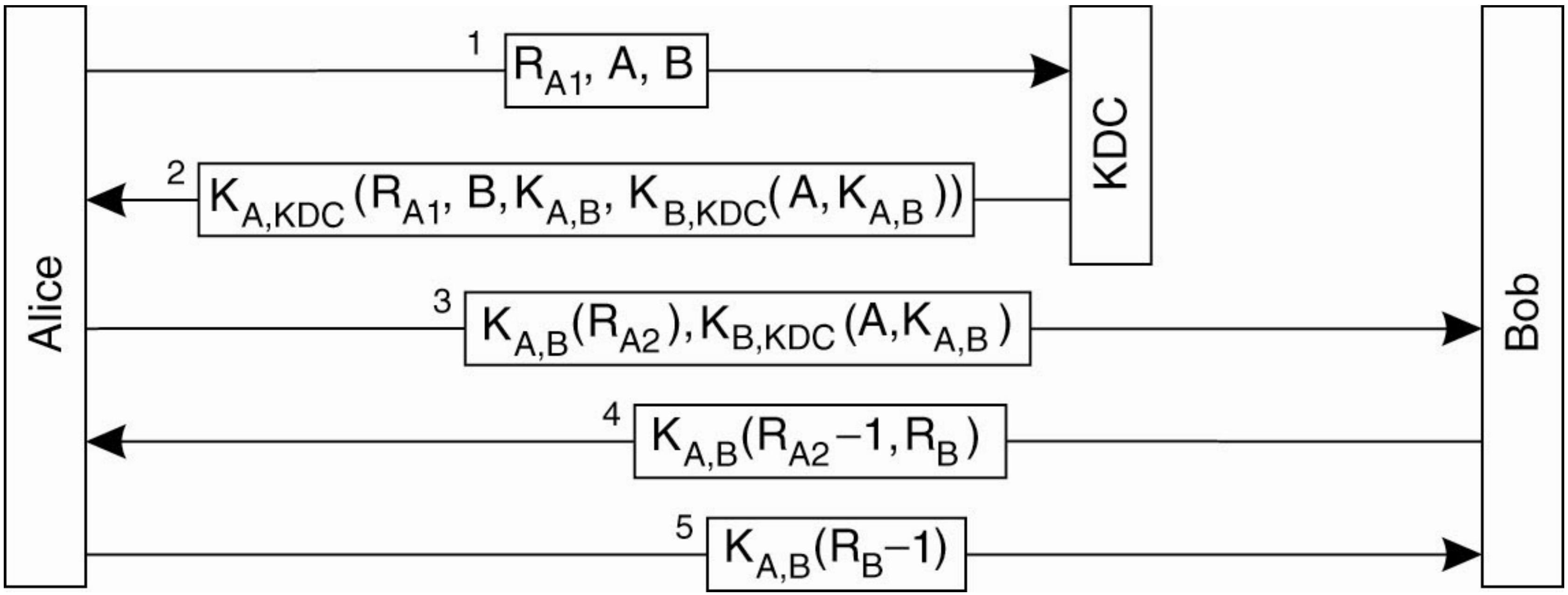


Authentication Using a Key Distribution Center



- …a ticket….

Authentication Using a Key Distribution Center



- The Needham-Schroeder authentication protocol.

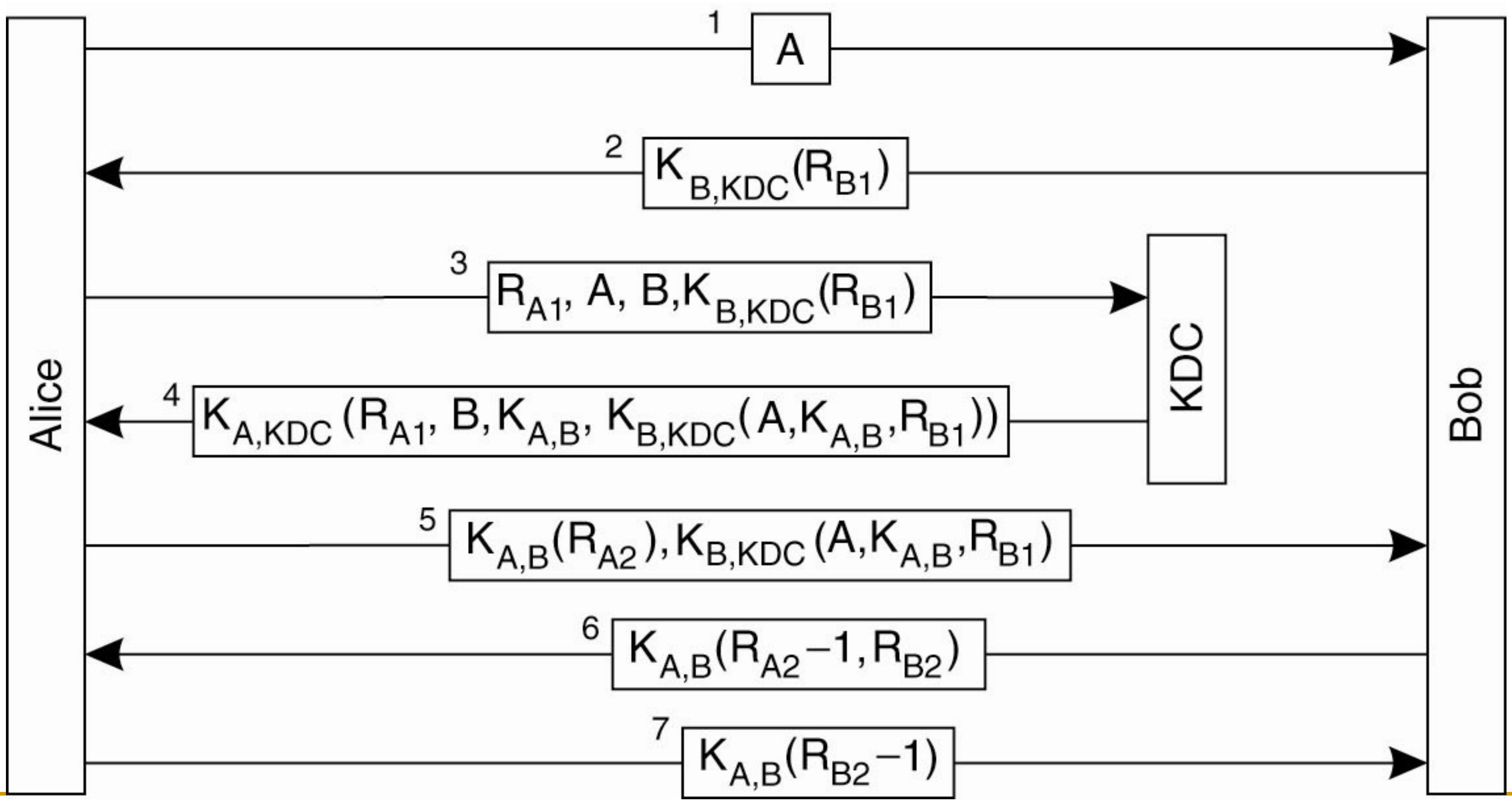
… the Needham - Schroeder protocol

- In early distributed systems (1974-84) it was difficult to protect the servers
  - E.g. against masquerading attacks on a file server
  - because there was no mechanism for authenticating the origins of requests
  - public-key cryptography was not yet available or practical

- Needham and Schroeder developed an authentication and key-distribution protocol for use in a local network
  - An early example of the care required to design a safe security protocol
  - Introduced several design ideas including the use of *nonces*.

Authentication Using a Key Distribution Center

- Protection against malicious reuse of a previously generated session key in the Needham-Schroeder protocol.



The Needham–Schroeder secret-key authentication protocol

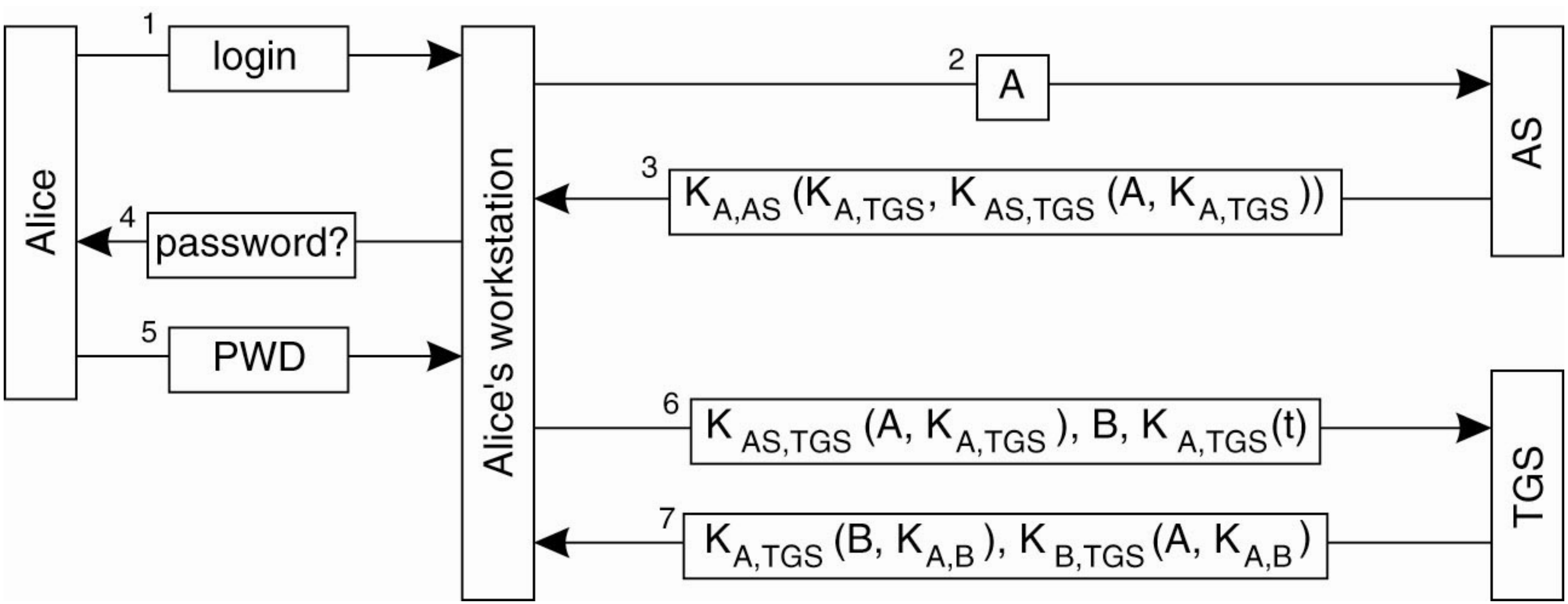
Header	*Message*	*Notes*
1. A->S:	A, B, N_A	A requests S to supply a key for communication with B.
2. S->A:	$\{N_A, B, K_{AB}, \boxed{\{K_{AB}, A\}_{K_B}}\}_{K_A}$ Ticket	S returns a message encrypted in A's secret key, containing a newly generated key K_{AB} and a 'ticket' encrypted in B's secret key. 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.

The Needham–Schroeder secret-key authentication protocol

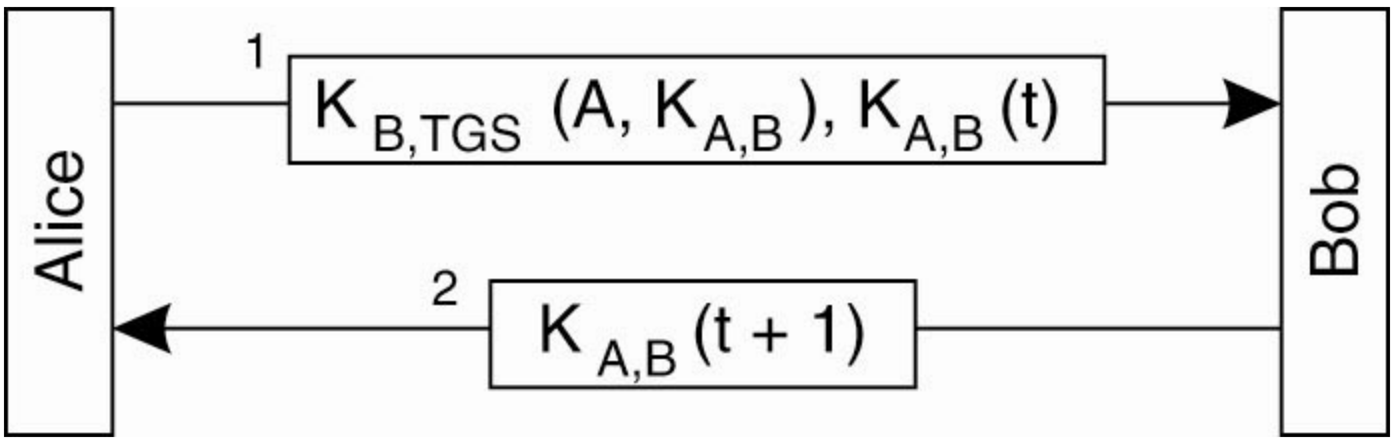
N_A is a nonce. Nonces are integers that are added to messages to demonstrate the freshness of the transaction. They are generated by the sending process when required, for example by incrementing a counter or by reading the (microsecond resolution) system clock.

Weakness: Message 3 might not be fresh - and K_{AB} could have been compromised in the store of A's computer. Kerberos addresses this by adding a timestamp or a nonce to message 3.

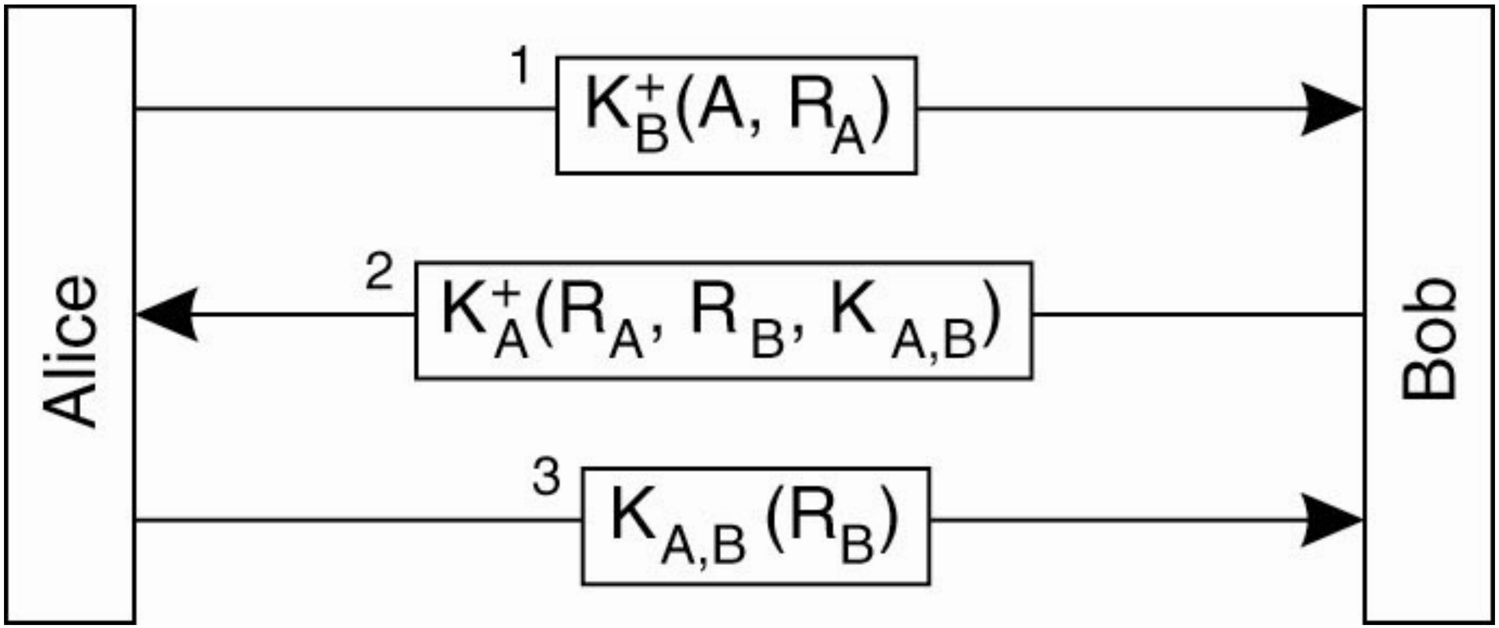
Example: Kerberos (1)



Example: Kerberos (2)

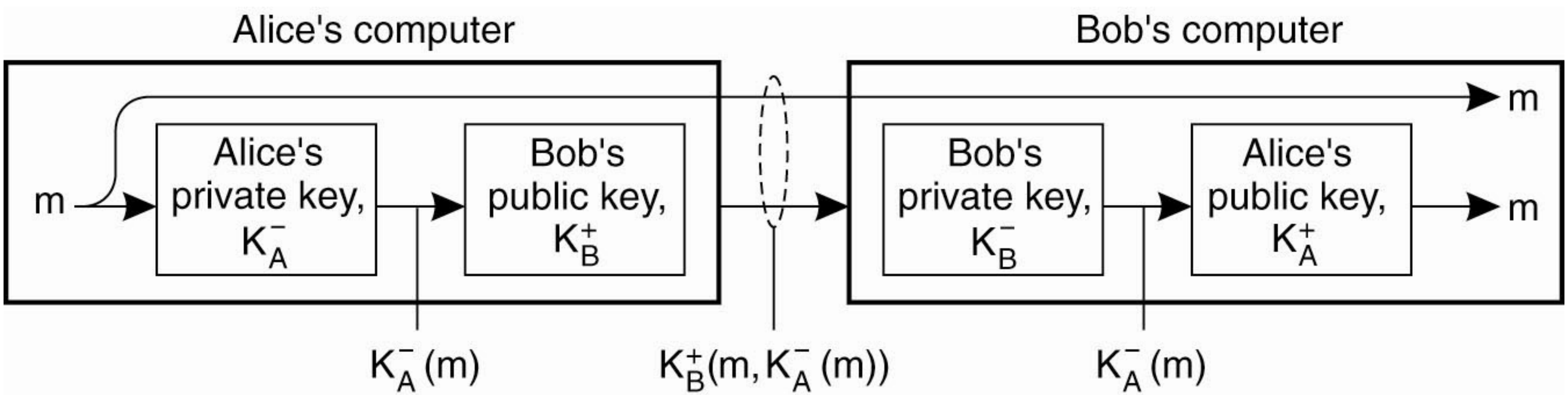


Mutual authentication in a public-key cryptosystem.



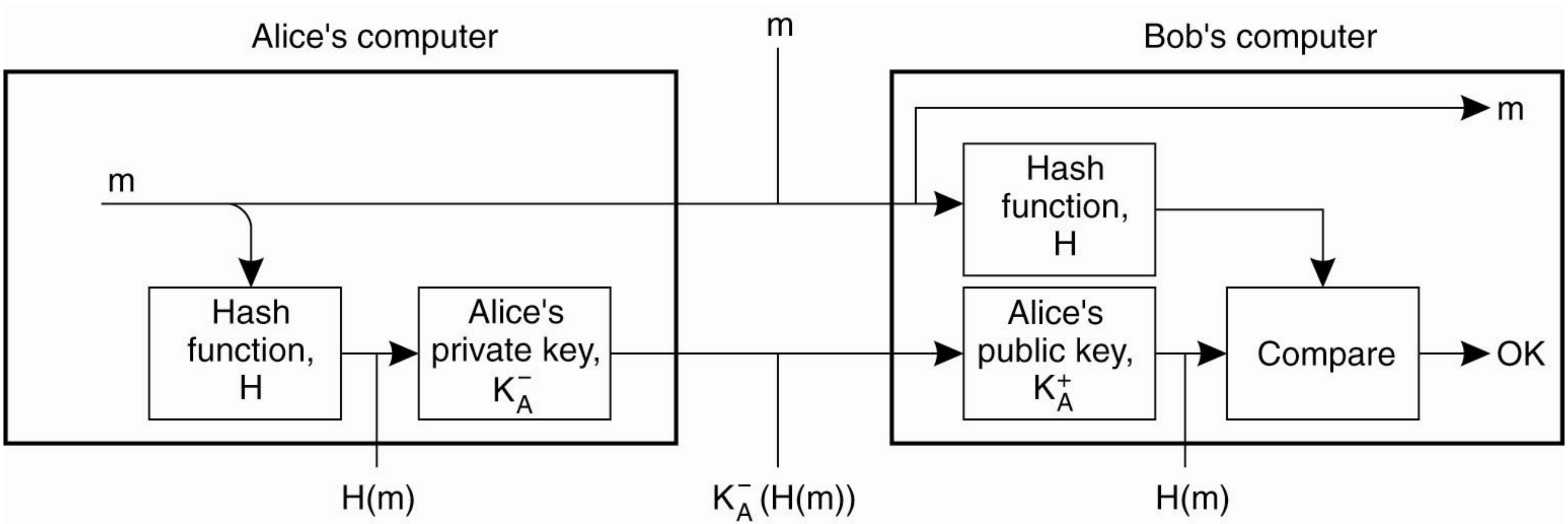
- Mutual authentication in a public-key cryptosystem.

Digital Signatures



- Digital signing a message using public-key cryptography.

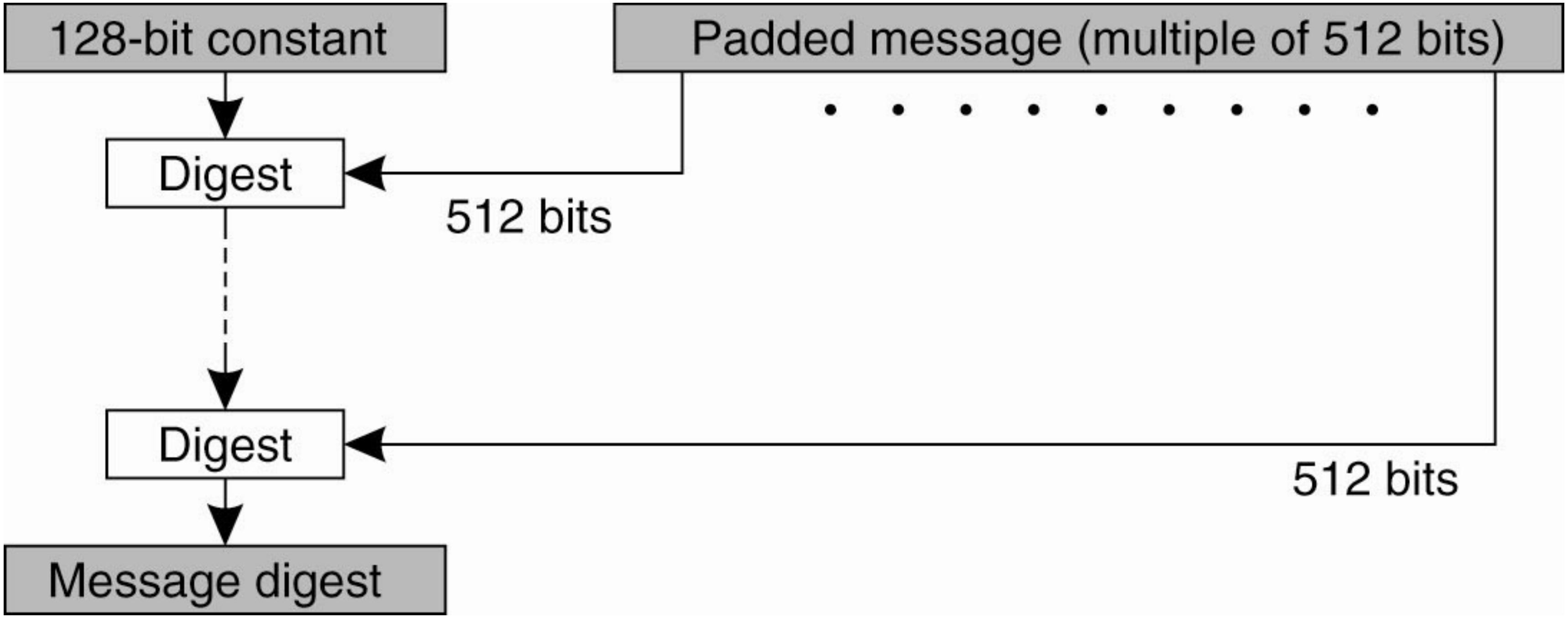
Digital Signatures



- Digitally signing a message using a message digest.

Hash Functions: MD5

- The structure of MD5.



Hash Functions: MD5 (2)

- The 16 iterations during the first round in a phase in MD5.

Iterations 1–8	Iterations 9–16
$p \leftarrow (p + F(q,r,s) + b_0 + C_1) \lll 7$	$p \leftarrow (p + F(q,r,s) + b_8 + C_9) \lll 7$
$s \leftarrow (s + F(p,q,r) + b_1 + C_2) \lll 12$	$s \leftarrow (s + F(p,q,r) + b_9 + C_{10}) \lll 12$
$r \leftarrow (r + F(s,p,q) + b_2 + C_3) \lll 17$	$r \leftarrow (r + F(s,p,q) + b_{10} + C_{11}) \lll 17$
$q \leftarrow (q + F(r,s,p) + b_3 + C_4) \lll 22$	$q \leftarrow (q + F(r,s,p) + b_{11} + C_{12}) \lll 22$
$p \leftarrow (p + F(q,r,s) + b_4 + C_5) \lll 7$	$p \leftarrow (p + F(q,r,s) + b_{12} + C_{13}) \lll 7$
$s \leftarrow (s + F(p,q,r) + b_5 + C_6) \lll 12$	$s \leftarrow (s + F(p,q,r) + b_{13} + C_{14}) \lll 12$
$r \leftarrow (r + F(s,p,q) + b_6 + C_7) \lll 17$	$r \leftarrow (r + F(s,p,q) + b_{14} + C_{15}) \lll 17$
$q \leftarrow (q + F(r,s,p) + b_7 + C_8) \lll 22$	$q \leftarrow (q + F(r,s,p) + b_{15} + C_{16}) \lll 22$

Birthday attack

1. Alice prepares two versions M and M' of a contract for Bob. M is favourable to Bob and M' is not.
2. Alice makes several subtly different versions of both M and M' that are visually indistinguishable from each other by methods such as adding spaces at the ends of lines. She compares the hashes of all the versions of M with all the versions of M'. (She is likely to find a match because of the Birthday Paradox).
3. When she has a pair of documents M and M' that hash to the same value, she gives the favourable document M to Bob for him to sign with a digital signature using his private key. When he returns it, she substitutes the matching unfavourable version M', retaining the signature from M.

- The digest function must be secure against the *birthday attack*

- **Birthday paradox**

  (Statistical result): if there are 23 people in a room, the chances are even that 2 of them will have the same birthday.

- If our hash values are 64 bits long, we require only 2^{32} versions of M and M' on average.
- This is too small for comfort. We need to make our hash values at least 128 bits long to guard against this attack.

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

…notation

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

Scenario 1: Secret communication with a shared secret key

Alice and Bob share a secret key K_{AB}.

1. Alice uses K_{AB} and an agreed encryption function $E(K_{AB}, M)$ to encrypt and send any number of messages $\{M_i\}_{K_{AB}}$ to Bob.
2. Bob reads the encrypted messages using the corresponding decryption function $D(K_{AB}, M)$.

Alice and Bob can go on using K_{AB} as long as it is safe to assume that K_{AB} has not been *compromised*.

Issues:

Key distribution: How can Alice send a shared key K_{AB} to Bob securely?

Freshness of communication: How does Bob know that any $\{M_i\}$ isn't a copy of an earlier encrypted message from Alice that was captured by Mallory and replayed later?

Scenario 2:
Authenticated communication with a server

Bob is a file server; Sara is an authentication service. Sara shares secret key K_A with Alice and secret key K_B with Bob.

1. Alice sends an (unencrypted) message to Sara stating her identity and requesting a *ticket* for access to Bob. →
2. Sara sends a response to Alice. $\{\{\text{Ticket}\}_{K_B}, K_{AB}\}_{K_A}$. It is encrypted in K_A and consists of a ticket (to be sent to Bob with each request for

A ticket is an encrypted item containing the identity of the principal to whom it is issued and a shared key for a communication session.

5. The ticket is actually $\{K_{AB}, \text{Alice}\}_{K_B}$. Bob uses K_B to decrypt it, checks that Alice's name matches and then uses K_{AB} to encrypt responses to Alice.

Scenario 2:
Authenticated communication with a server

- … a simplified version of the Needham and Schroeder (and Kerberos) protocol.
- Timing and replay issues – addressed in N-S and Kerberos.
- Not suitable for e-commerce because authentication service doesn't scale…

Scenario 3: Authenticated communication with public keys

Bob has a public/private key pair <K_{Bpub}, K_{Bpriv}>

1. Alice obtains a certificate that was signed by a trusted authority stating Bob's public key K_{Bpub}
2. Alice creates a new shared key K_{AB} , encrypts it using K_{Bpub} using a public-key algorithm and sends the result to Bob.
3. Bob uses the corresponding private key K_{Bpriv} to decrypt it.

(If they want to be sure that the message hasn't been tampered with, Alice can add an agreed value to it and Bob can check it.)

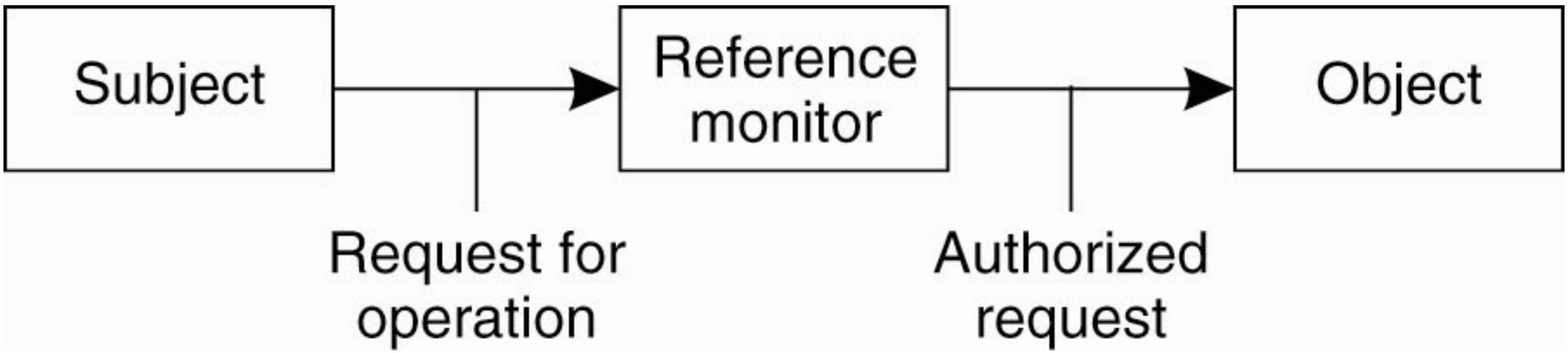
- Mallory might intercept Alice's initial request to a key distribution service for Bob's public-key certificate and send a response containing his own public key. He can then intercept all the subsequent messages.

Scenario 4:
Digital signatures with a secure digest function

Alice wants to publish a document M in such a way that anyone can verify that it is from her.

1. Alice computes a fixed-length digest of the document Digest(M).
2. Alice encrypts the digest in her private key, appends it to M and makes the resulting signed document (M, $\{Digest(M)\}_{K_{Apriv}}$) available to the intended users.
3. Bob obtains the signed document, extracts M and computes Digest(M).
4. Bob uses Alice's public key to decrypt $\{Digest(M)\}_{K_{Apriv}}$ and compares it with his computed digest. If they match, Alice's signature is verified.

General Issues in Access Control



- A general model …

…a matrix-based solution

	Obj 1	Obj 2		Obj j	Obj m
Subj 1	m1	m2			
Subj 2		m1,m2			
Subj i					
Subj n	m4				

- Space problems due to scalability
- A two approaches:
  - Column-wise
  - Row-wise

..ACL…

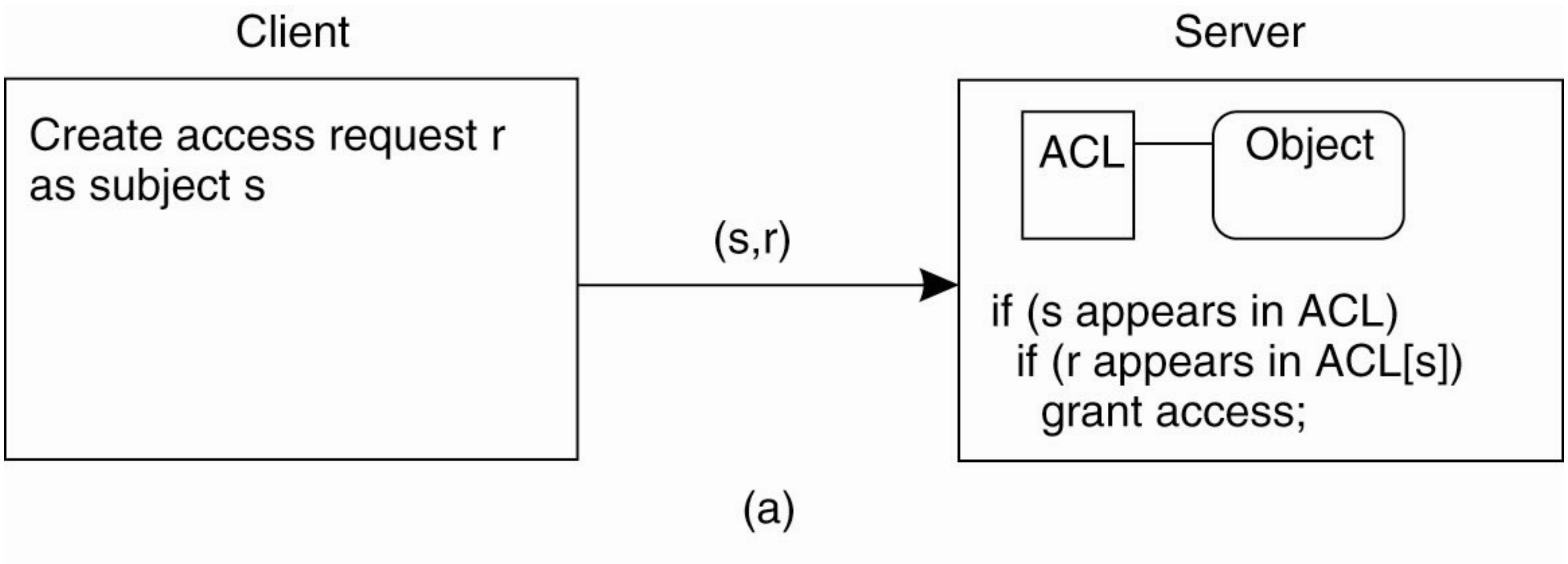
- an **Access Control List** is a key allowing the object to know the subjects that want to access its method.
- Format: <subject id, required operations>
- Problems: eavesdropping, difficulty of cancellation

…a "simple" implementation:

- Like the Unix file access permissions

```
drwxr-xr-x  gfc22  staff       264 Oct 30 16:57
Acrobat User Data
-rw-r--r--  gfc22  unknown       0 Nov  1 09:34 Eudora
Folder
-rw-r--r--  gfc22  staff    163945 Oct 24 00:16
Preview of xx.pdf
drwxr-xr-x  gfc22  staff       264 Oct 31 13:09 iTunes
-rw-r--r--  gfc22  staff       325 Oct 22 22:59 list
of broken apps.rtf
```

Access Control Matrix



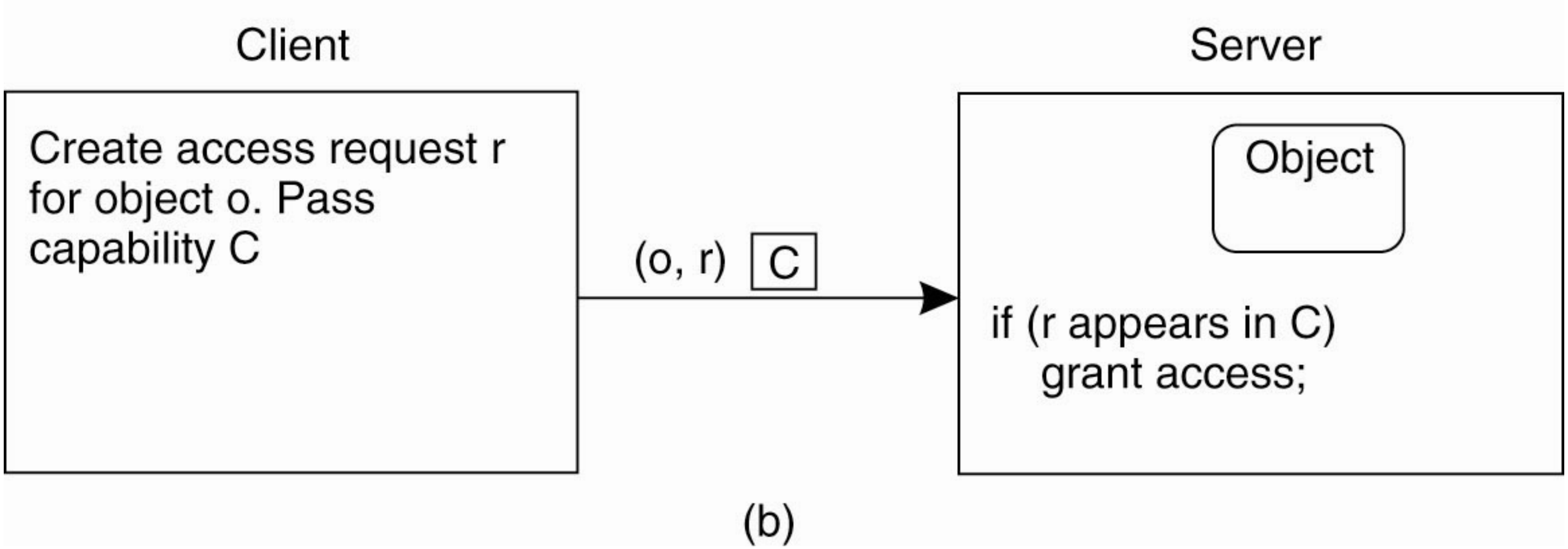
- Using an ACL for protecting objects.

…Capability…

- a **capability** is a key allowing the holder to access one or more of the operations supported by a resource.
- Format: <resource id, permitted operations, authentication code>

- Problems: eavesdropping, difficulty of cancellation

Access Control Matrix



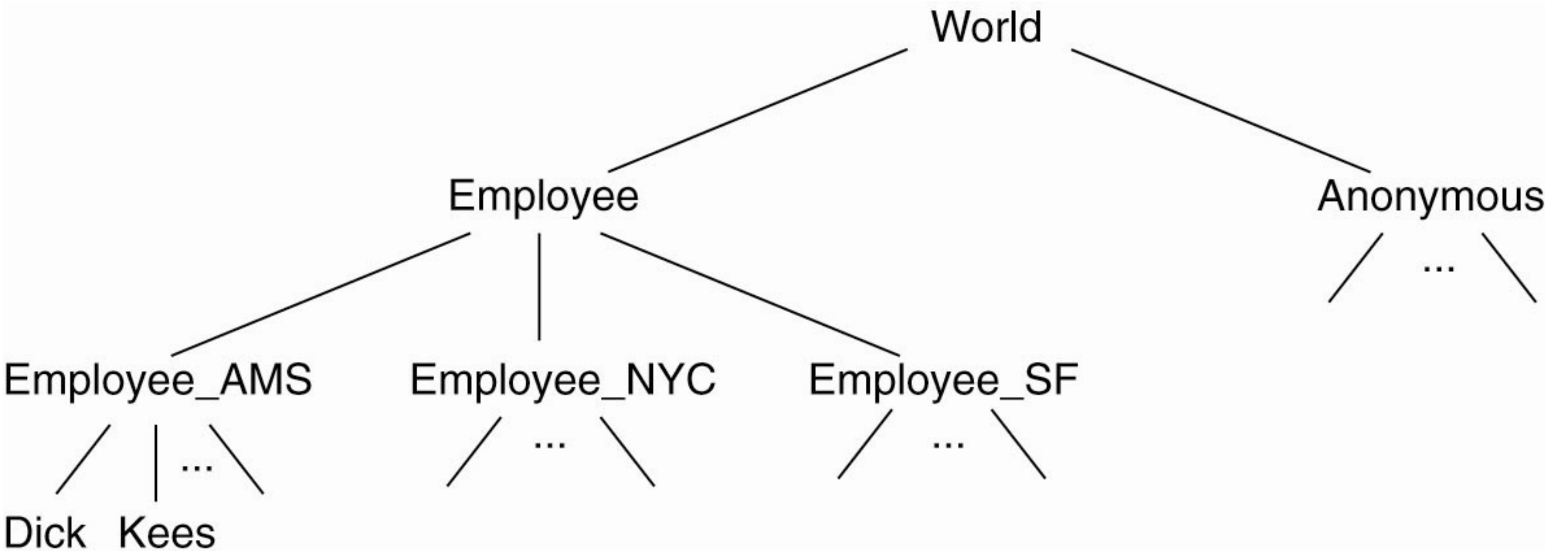
- Using capabilities for protecting objects.

- For more complex object types and user communities, ACLs, as well as capabilities, can become very complex and very difficult to manage

Access control

- Protection domain
  - A set of <resource, rights> pairs

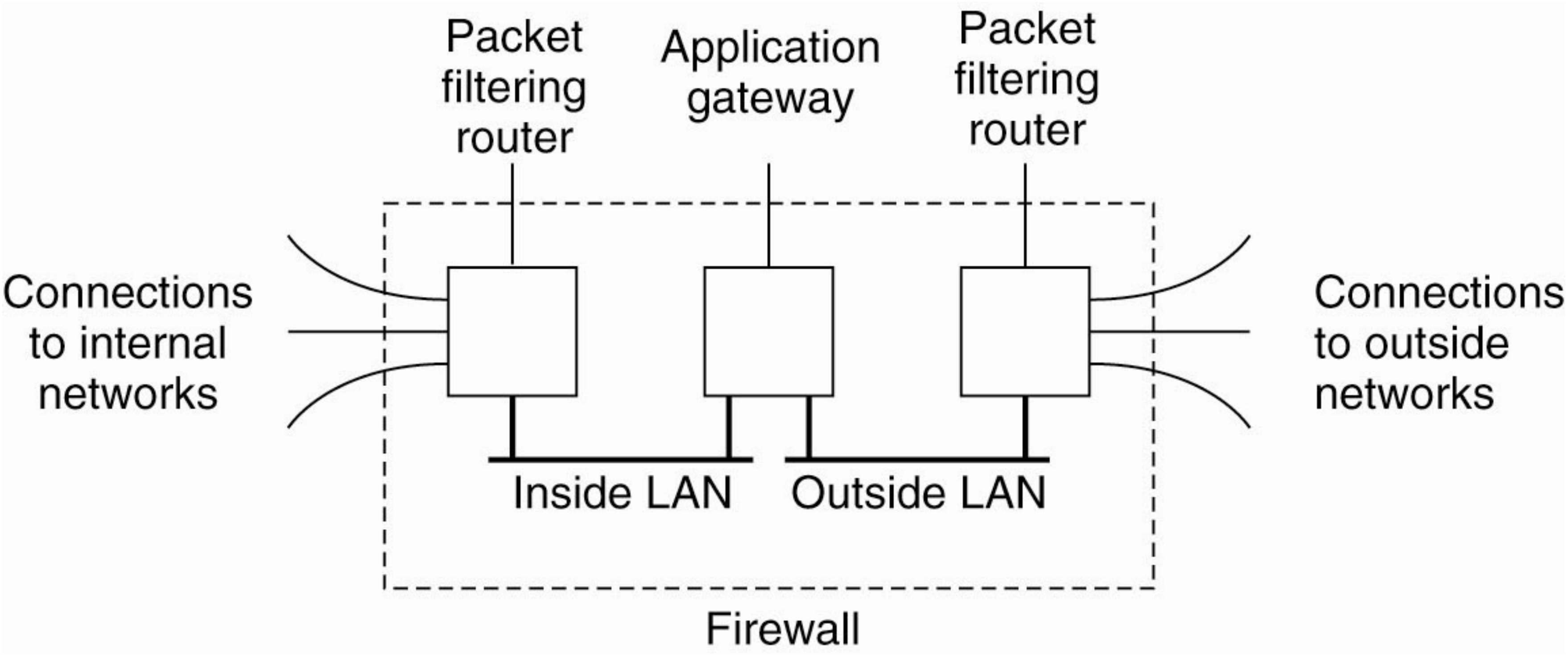
Protection Domains



- The hierarchical organization of protection domains as groups of users.

- …certificates…
- …roles…

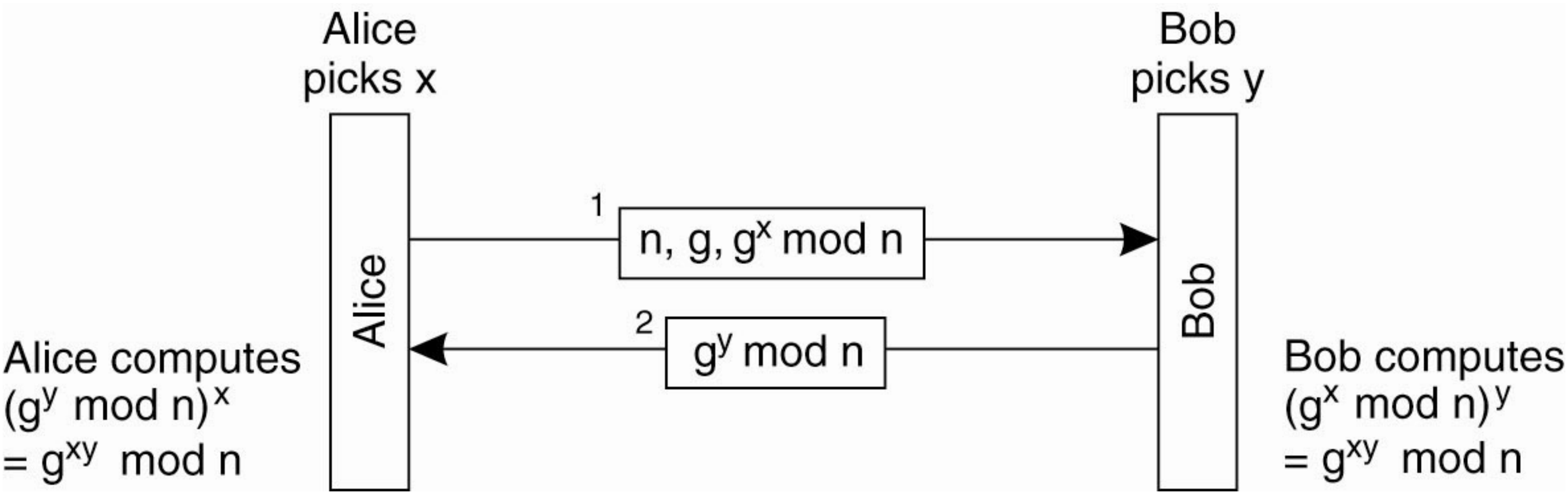
Firewalls



- A common implementation of a firewall.

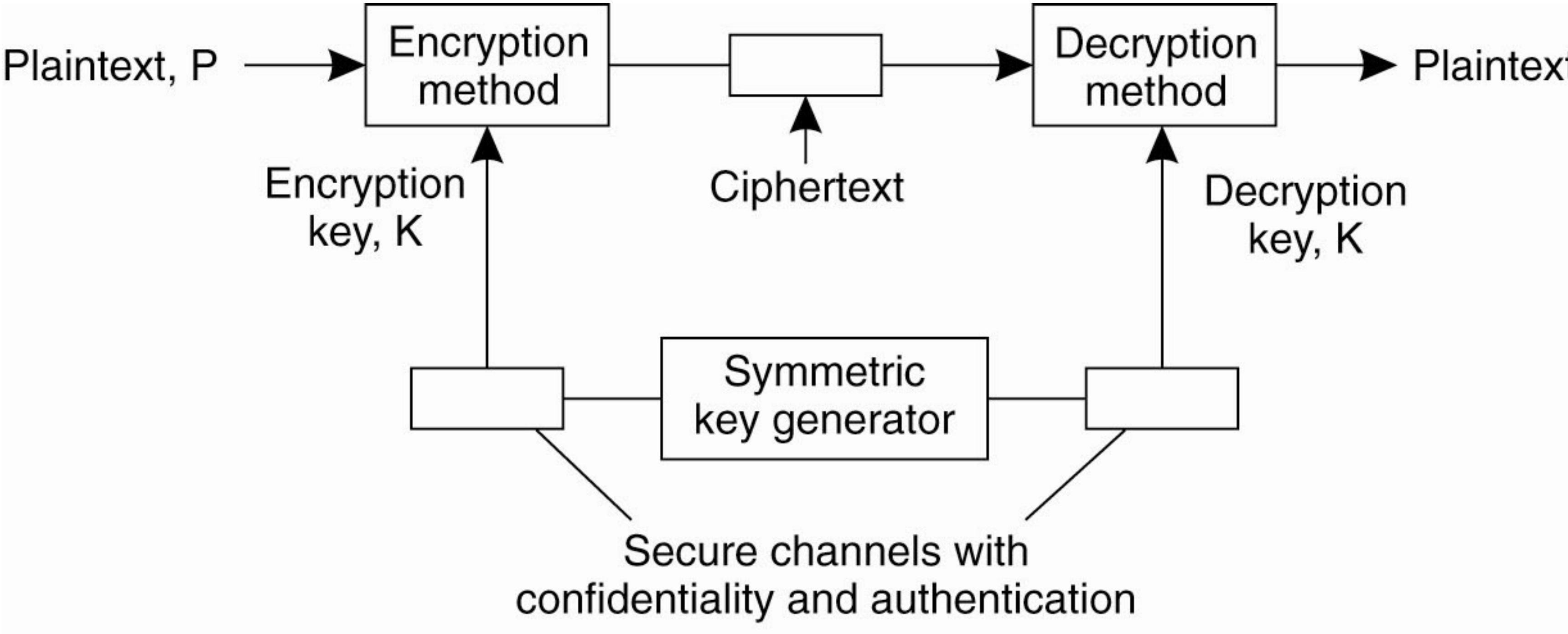
- …packet filtering gateway…
- …application-level gateway…
- …proxy gateway…

Key Establishment



- The principle of Diffie-Hellman key exchange.

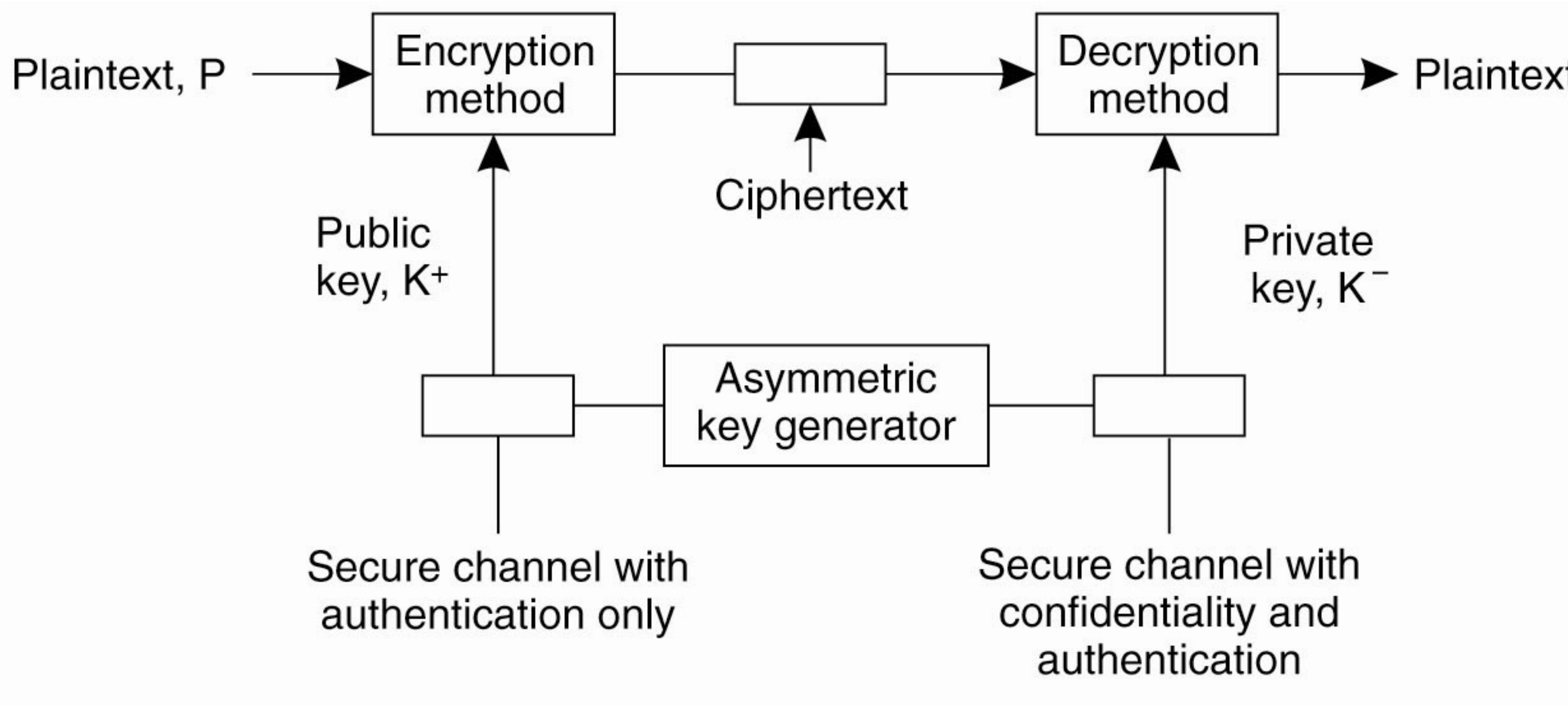
Key Distribution



(a)

- Secret-key distribution

Key Distribution



(b)

- Public-key distribution

- The *speaks for* idea
  - We don't want users to have to give their password every time their PC accesses a server holding protected resources.
- Requests to access resources must be accompanied by *credentials*:

…credentials…

- Evidence for the requesting principal's right to access the resource
- Simplest case: an identity certificate for the principal, signed by the principal.
- Credentials can be used in combination. E.g. to send an authenticated email as a member of University of Padova, I would need to present a certificate of membership of UP and a certificate of my email address.

Delegation: a simple example…

- Consider a server that prints files:
  - wasteful to copy the files, should access users' files *in situ*
  - server must be given restricted and temporary rights to access protected files

Certificates

- Certificate: a statement signed by an appropriate authority.
- Certificates require:
  - An agreed standard format
  - Agreement on the construction of chains of trust.
  - Expiry dates, so that certificates can be revoked.

Certificates

Alice's bank account certificate

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

Public-key certificate for Bob's Bank

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

X509 Certificate format

Subject	Distinguished Name, Public Key
Issuer	Distinguished Name, Signature
Period of validity	Not Before Date, Not After Date
Administrative information	Version, Serial Number
Extended Information	

Certificates as credentials

- Certificates can act as *credentials*
  - Evidence for a principal's right to access a resource
- The two certificates shown in the next slide could act as credentials for Alice to operate on her bank account
  - She would need to add her public key certificate

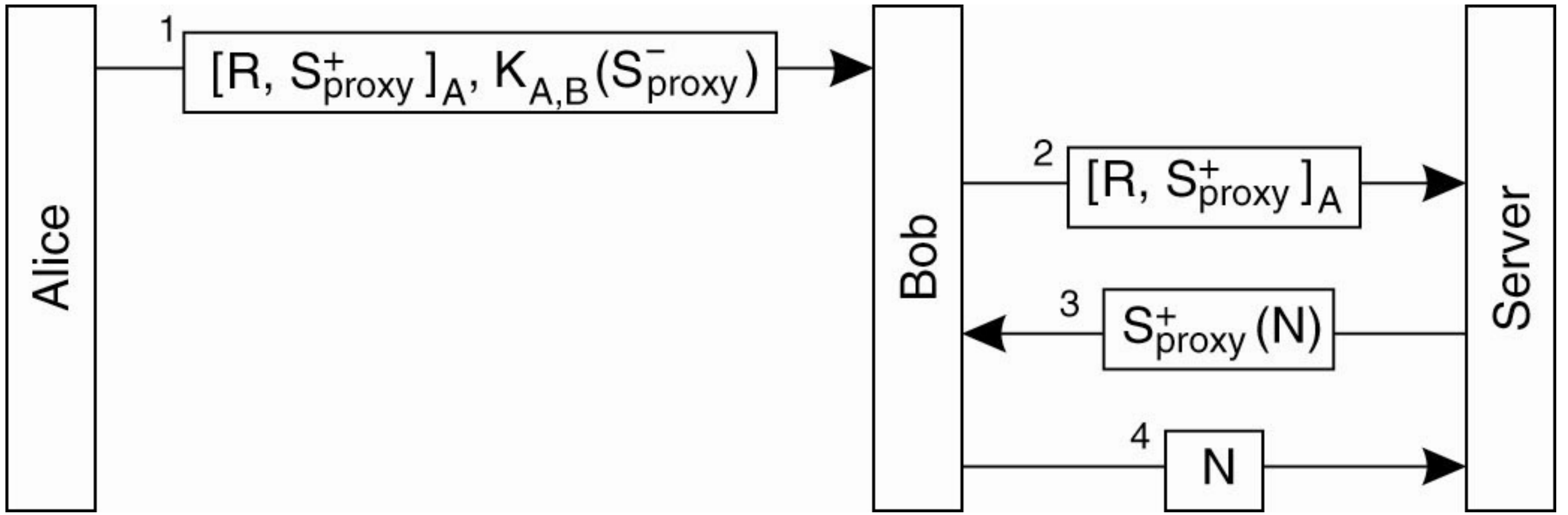
…a delegation certificate…

- a **delegation certificate** is a signed request authorizing another principal to access a named resource in a restricted manner.
- The temporal restriction can be achieved by adding expiry times.
- CORBA Security Service supports delegation certificates

Biometrics…

- Fingerprints, irix, face, voice, gesture etc…
- Multibiometrics
- Systems

Delegation (2)



- Figure 9-39. Using a proxy to delegate and prove ownership of access rights.

Summary

- It is essential to protect resources, communication channels and interfaces of distributed systems and applications against attacks.
- This is achieved by the use of access control mechanisms and secure channels.
- Public-key and secret-key cryptography provide the basis for authentication and for secure communication.

End of lectures