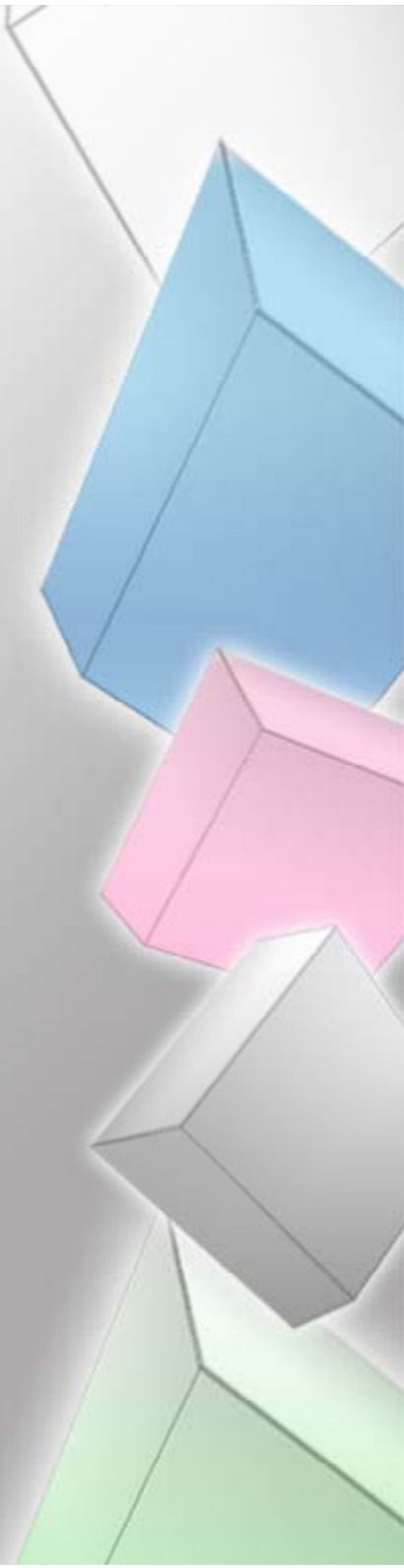
Advanced Topics in Operating Systems

MSc in Computer Science
UNYT-UoG

Dr. Marenglen Biba
8-9-10 January 2010



Lesson 9

01: Introduction
02: Architectures
03: Processes
04: Communication
05: Naming
06: Synchronization
07: Consistency & Replication
08: Fault Tolerance
09: Security
10: Distributed Object-Based Systems
11: Distributed File Systems
12: Distributed Web-Based Systems
13: Distributed Coordination-Based Systems

Security

- Introduction to security
  - Cryptography
- Secure channels
- Access Control
- Security Management

Security Threats

- Types of security threats to consider:

- Interception
- Interruption
- Modification
- Fabrication

Properties

- Confidentiality
  - The property of a computer system to disclose information only to authorized parties
- Integrity
  - Alterations to a system can be made only in an authorized way.
- Security Policy
  - What actions can entities in a CS take
- Security Mechanisms
  - How to enforce a security policy

Security Mechanisms

1. Encryption
   - Transform data an attacker cannot understand
2. Authentication
   - Verify claimed identity of an entity
3. Authorization
   - Once authenticated, is the entity authorized to do something?
4. Auditing
   - Tools to trace who is accessing what

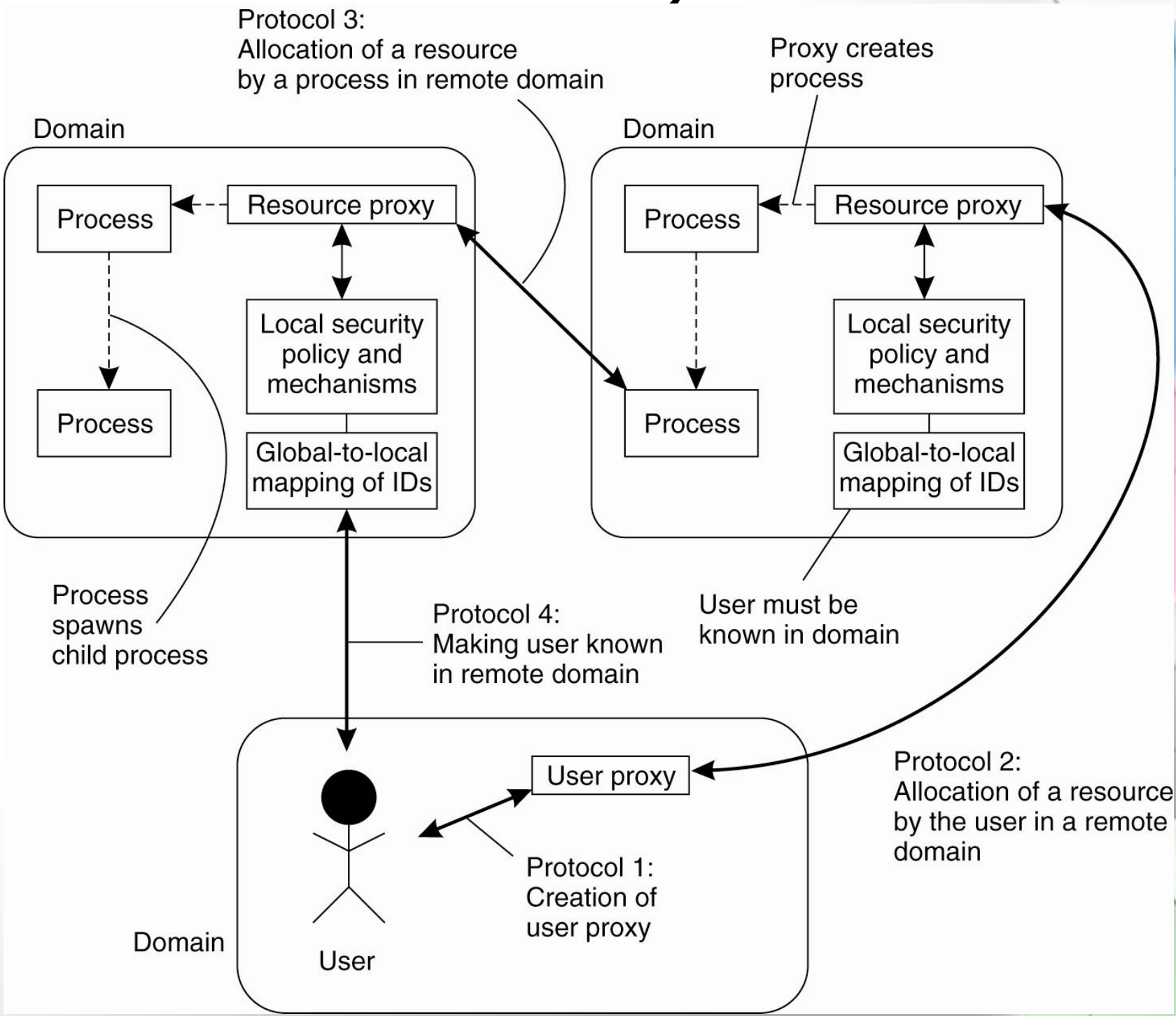
Example: The Globus Security Architecture

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.

The Globus Security Architecture



Security Design Issues

1. Focus of Control
2. Layering of Security Mechanisms
3. Simplicity

Focus of Control: protection of data

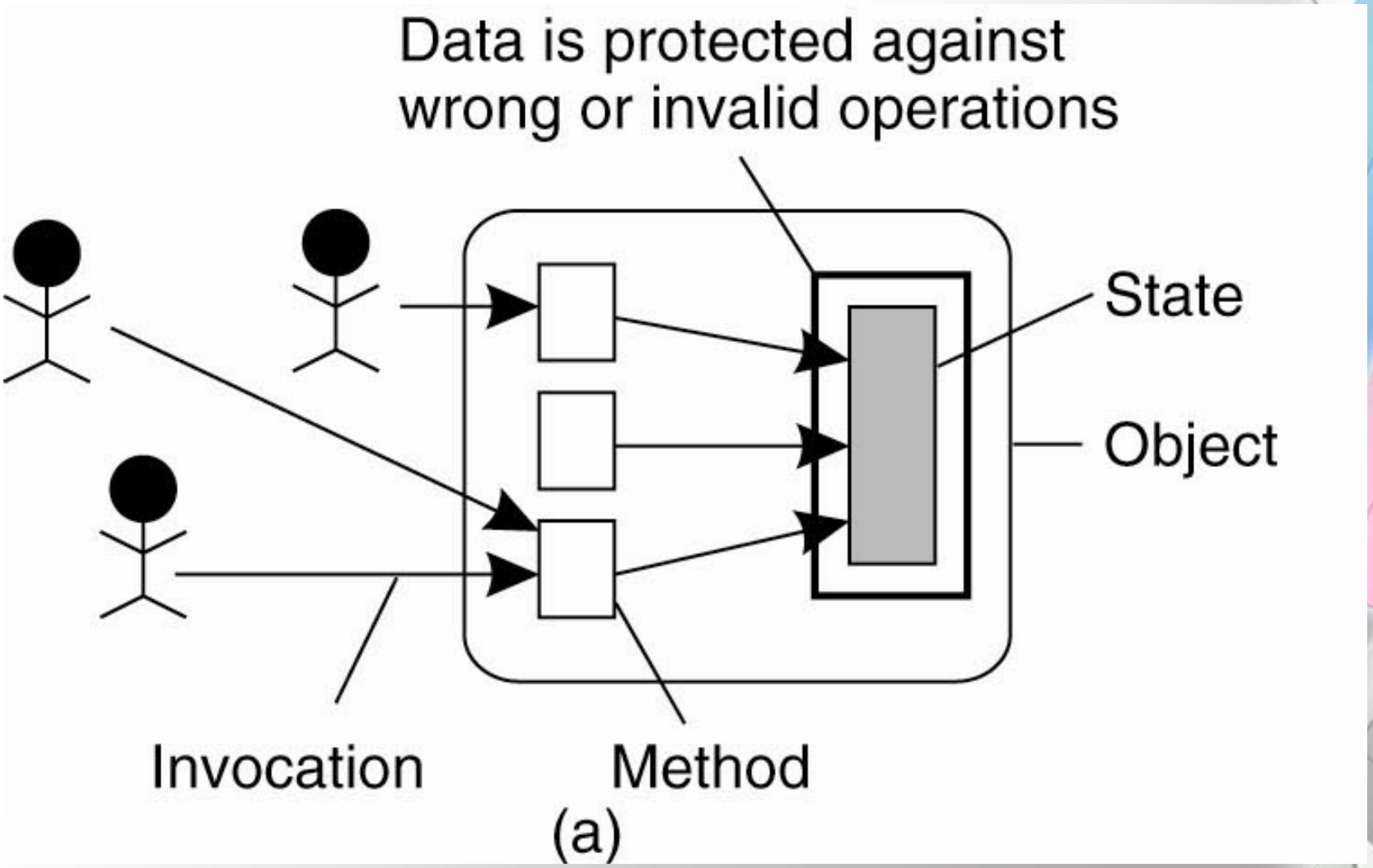


Figure 9-2. Three approaches for protection against security threats. (a) Protection against invalid operations

Focus of Control: invocations

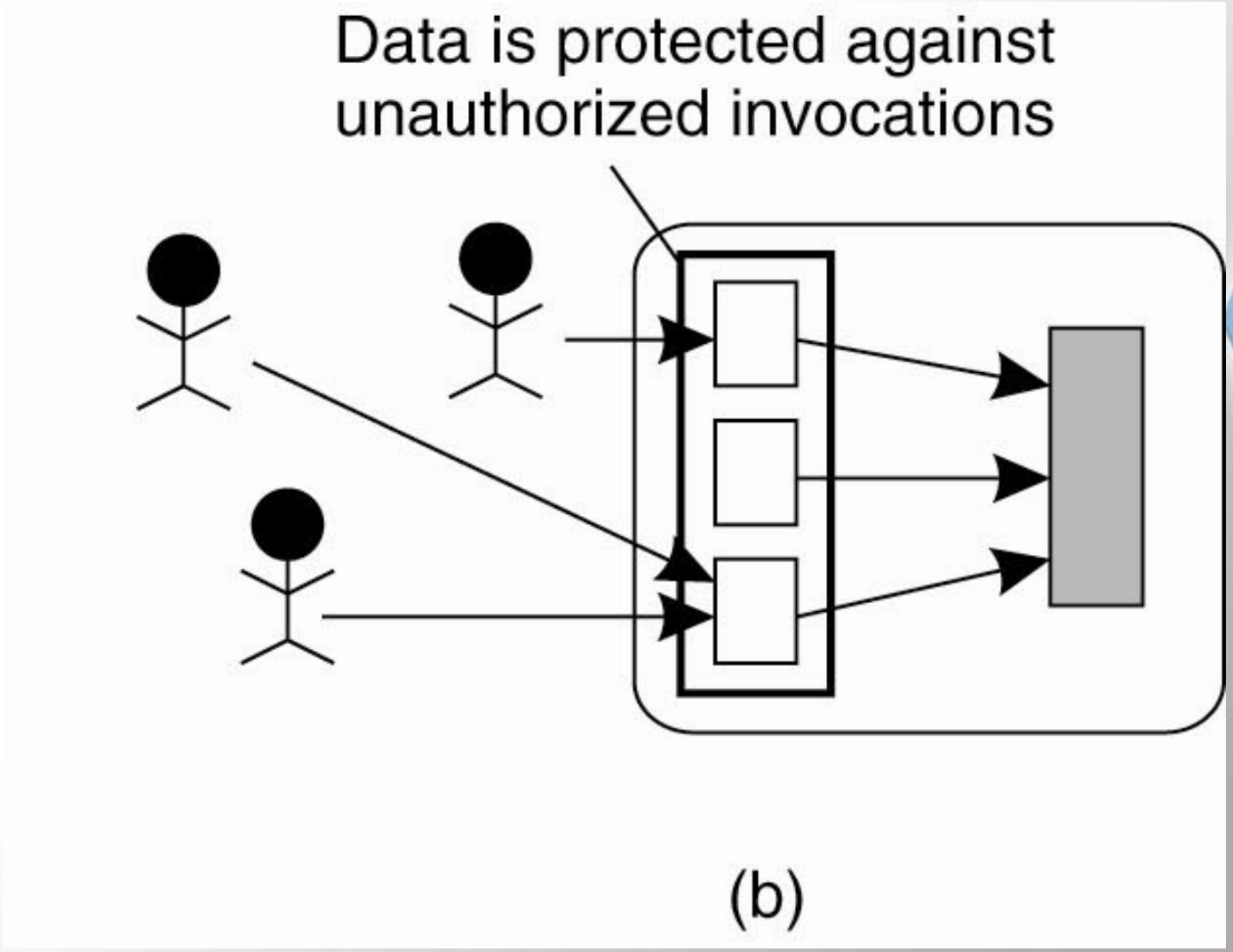


Figure 9-2. Three approaches for protection against security threats. (b) Protection against unauthorized invocations.

Focus of Control: users

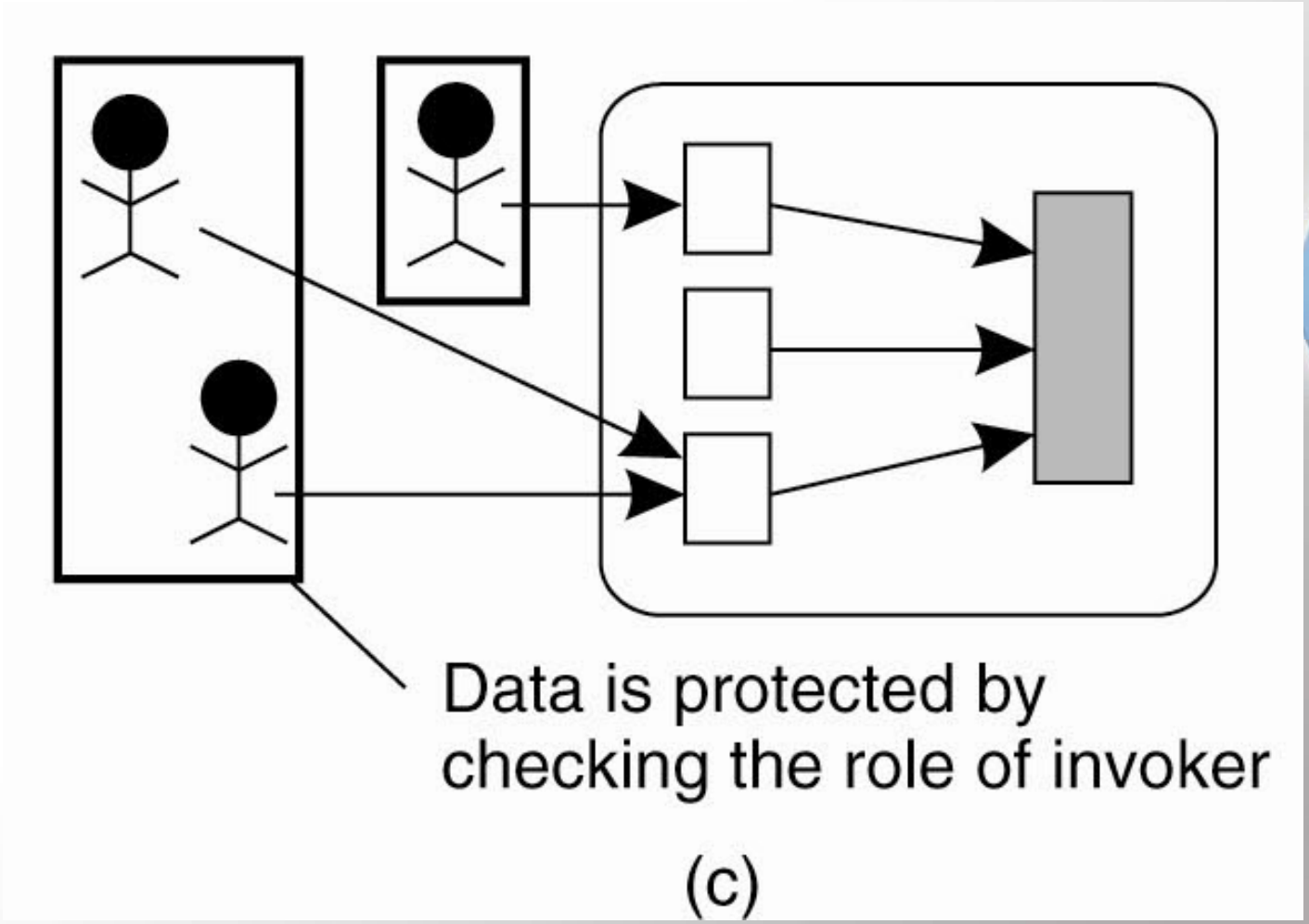


Figure 9-2. Three approaches for protection against security threats.
(c) Protection against unauthorized users.

Layering of Security Mechanisms

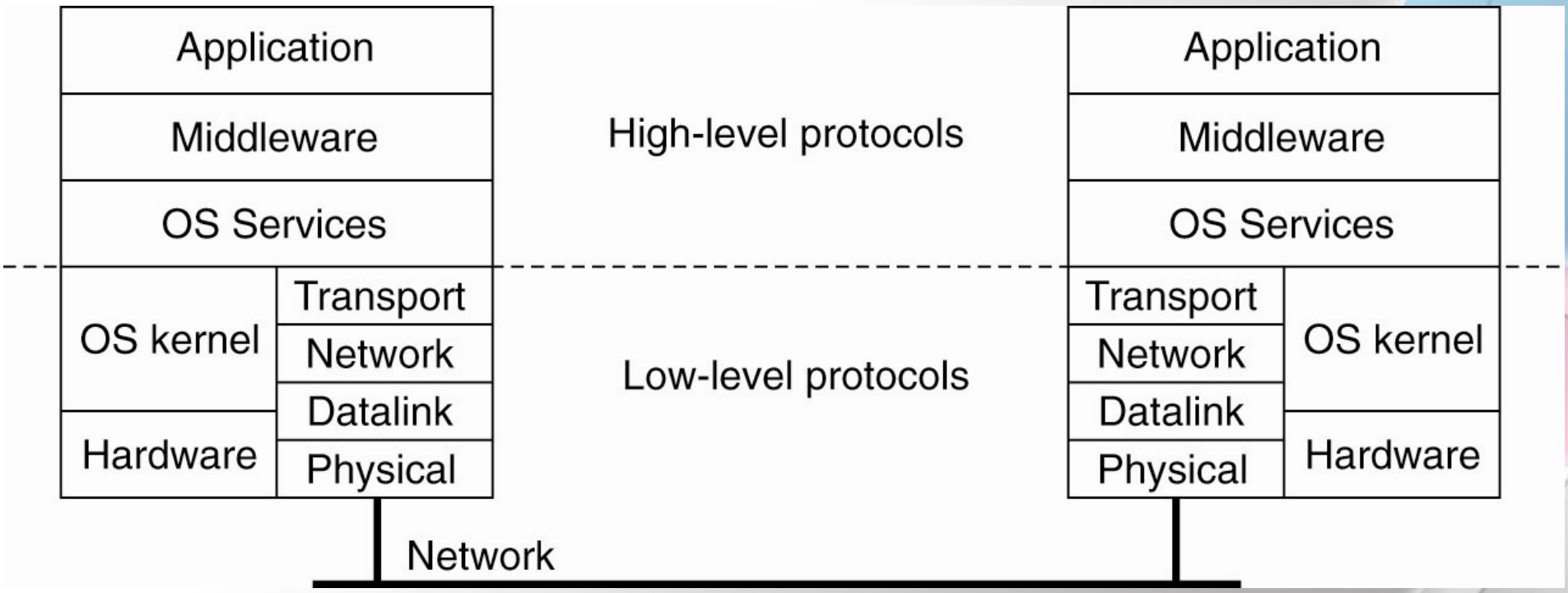
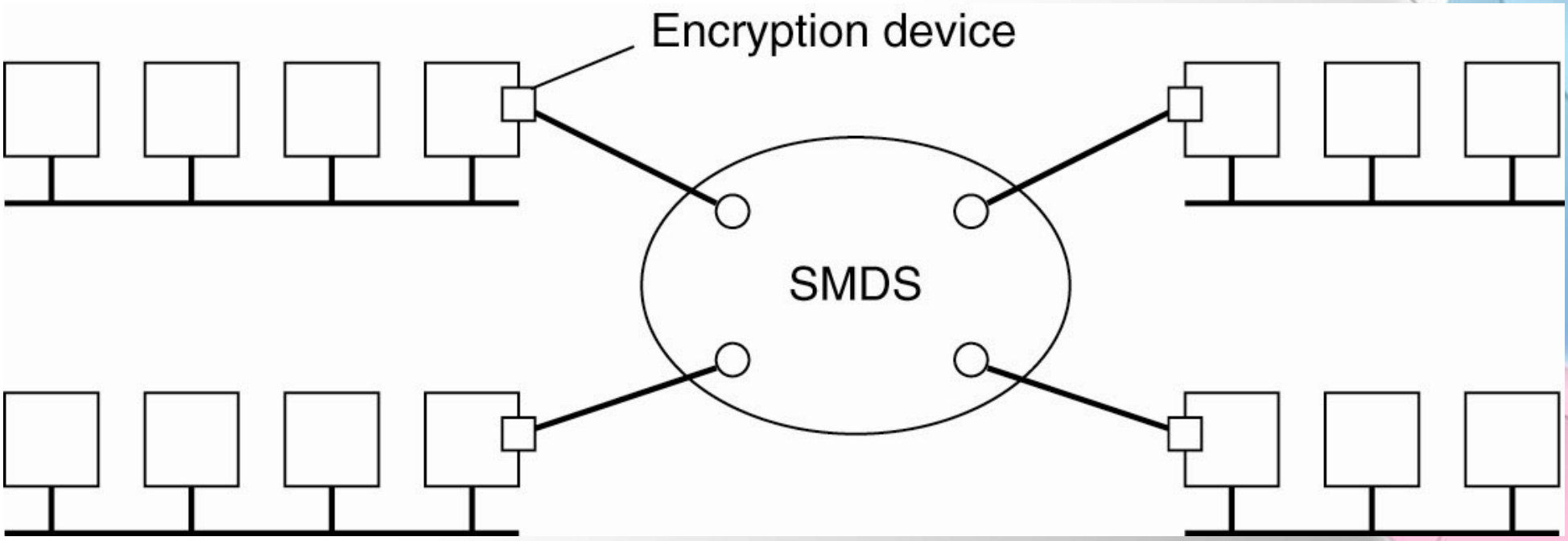


Figure 9-3. The logical organization of a distributed system into several layers.

Layering of Security Mechanisms



Different sites connected through Switched Multi-megabit Data Service (SMDS). An SMDS network can be thought of as a link-level backbone connecting various local-area networks.

Security can be provided by placing encryption devices at each SMDS router,

SSL

- Another approach is that of using a transport-level security service such as SSL.
- SSL stands for Secure Sockets Layer and can be used to securely send messages across a TCP connection.
- All transport-level messages will be encrypted - and at the SMDS level as well, but that is of no concern to us.
  - In this case, we will have to put our trust into SSL.
  - In other words, we believe that SSL is secure.

- In distributed systems, security mechanisms are often placed in the middleware layer.

Distribution of Security Mechanisms

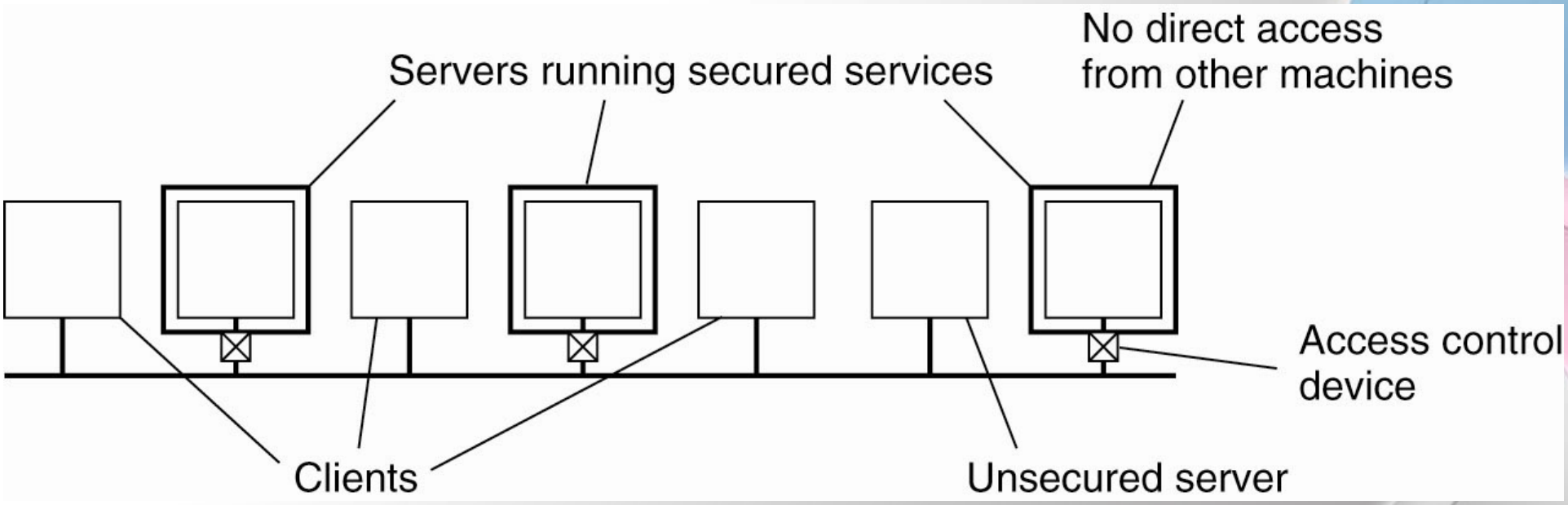
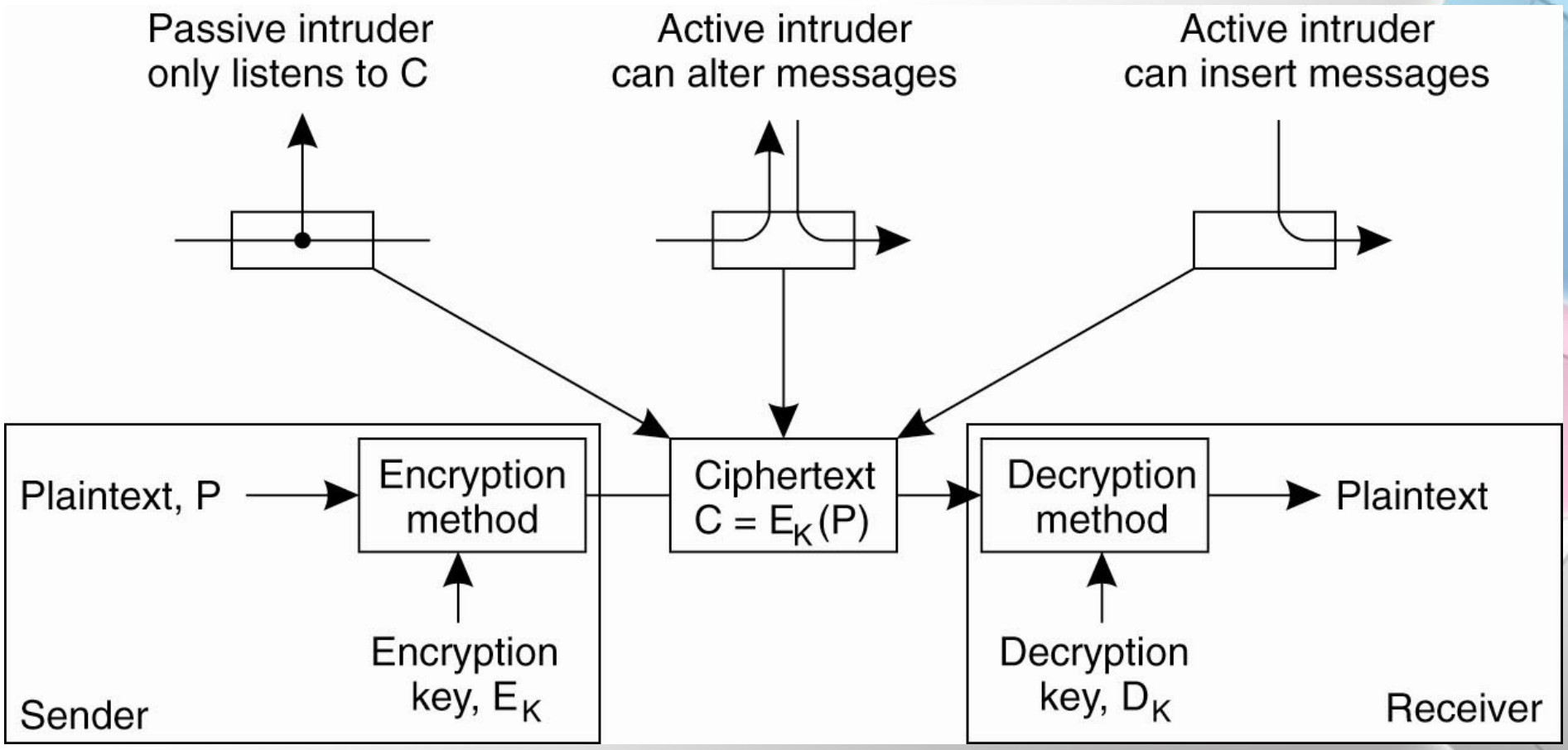


Figure 9-5. The principle of Reduced Interfaces for Secure System Components (RISSC) as applied to secure distributed systems.

Simplicity

- Unfortunately, simple mechanisms are not always sufficient for implementing security policies.
- Link-level encryption is a simple and easy-to-understand mechanism to protect against interception of intersite message traffic.
  - However, much more is needed if we want to be sure that only the right person will receive the messages.
- In that case, user-level authentication services are needed, and we may need to be aware of how such services work in order to put our trust in it.
- User-level authentication may therefore require at least a notion of cryptographic keys and awareness of mechanisms such as certificates, despite the fact that many security services are highly automated and hidden from users.

Cryptography (1)



- Figure 9-6. Intruders and eavesdroppers in communication.

Symmetric and Asymmetric cryptosystems

- There is a fundamental distinction between different cryptographic systems, based on whether or not the encryption and decryption key are the same.
- In a **symmetric cryptosystem**, the same key is used to encrypt and decrypt a message. Symmetric cryptosystems are also referred to as **secret-key** or **shared-key** systems.
- In an **asymmetric cryptosystem**, the keys for encryption and decryption are different, but together form a **unique pair**.
- One of the keys in an asymmetric cryptosystem is kept private; the other is made public. For this reason, asymmetric cryptosystems are also referred to as **public-key** systems

Hash functions: essential properties

- A hash function *H* takes a message *m* of arbitrary length as input and produces a bit string *h* having a fixed length as output, *h = H(m).*
- Properties:
  - One-way function: it is computationally infeasible to find the input *m* that corresponds to a known output *h.*
  - Weak collision resistance: given an input *m* and its associated output *h = H (m),* it is computationally infeasible to find another, different input *m‘ ≠ m,* such that *H (m) = H (m').*
  - Strong collision resistance: given only *H,* it is computationally infeasible to find any two different input values *m* and *m‘* such that *H(m) = H(m').*

Cryptography

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

Figure 9-7. Notation used.

Symmetric Cryptosystems: DES

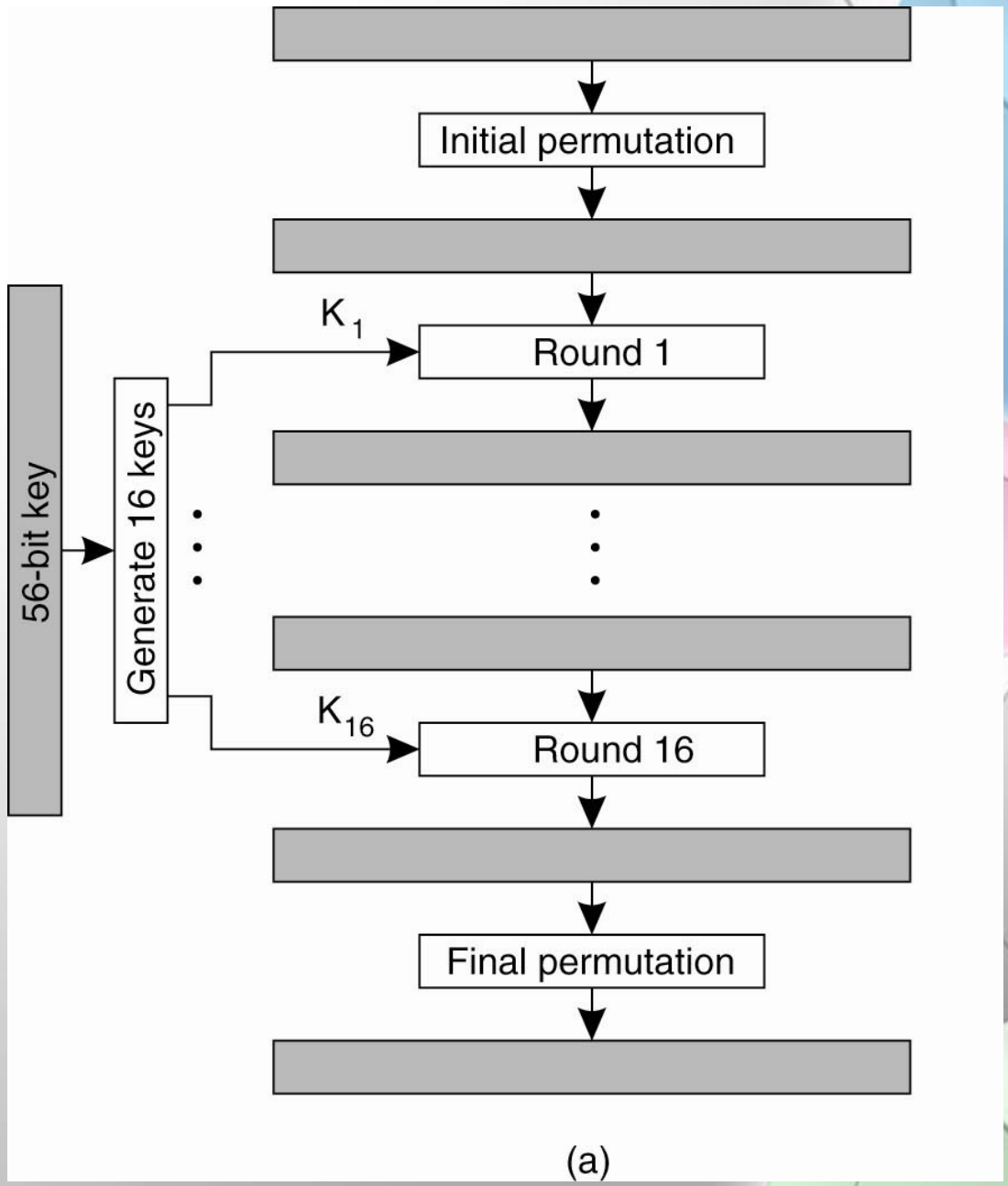


Figure 9-8. (a) The principle of the Data Encryption Standard (DES) algorithm.

Factorization complexity

- If a large, *b*-bit number is the product of two primes that are roughly the same size, then no algorithm has been published that can factor in polynomial time.

Public-Key Cryptosystems: RSA

- Generating the private and public keys requires four steps:
  - Choose two very large prime numbers, *p* and *q*.
  - Compute $n = p \times q$ and $z = (p - 1) \times (q - 1)$.
  - Choose a number *d* that is relatively prime to *z*.
  - Compute the number *e* such that $e \times d = 1 \bmod z$.
- One of the numbers, say d, can be used for decryption, the other for encryption.

- When the numbers are very large, no efficient prime factorization algorithm is publicly known; a 2005 effort by F. Bahr, M. Boehm, J. Franke, T. Kleinjung factored a 193-digit number (RSA-640) utilizing 30 2.2 GHz-Opteron-CPU years over a span of 5 months. The presumed difficulty of this problem is at the heart of certain algorithms in cryptography such as RSA.

Hash Functions: MD5

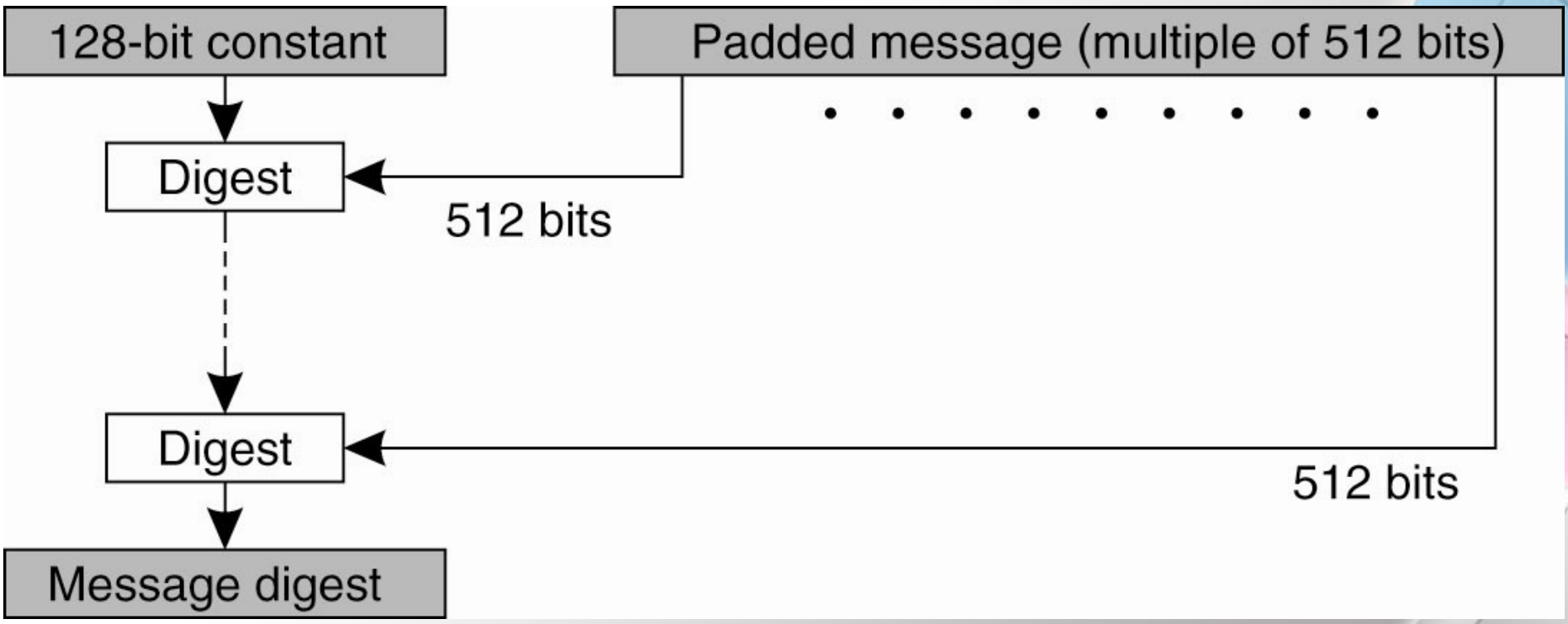


Figure 9-10. The structure of MD5. k phases where k is the number of 512-bit blocks comprising the padded message.

Security

- Introduction to security
  - Cryptography
- Secure channels
- Access Control
- Security Management

Secure Channels

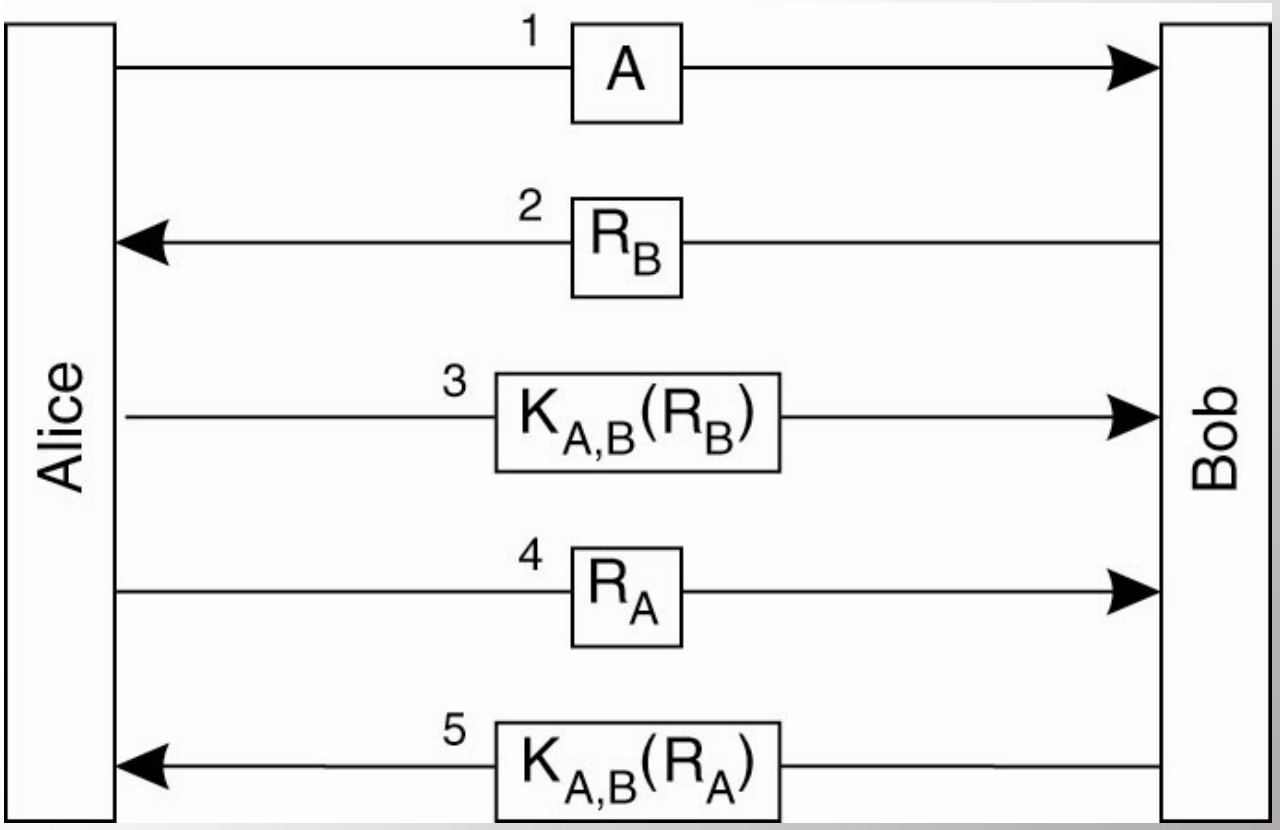
- When considering security in distributed systems, it is once again useful to think in terms of clients and servers.
- In particular, making a distributed system secure needs to solve how to make the communication between clients and servers secure. Secure communication requires authentication of the communicating parties.
- The issue of protecting communication between clients and servers, can be thought of in terms of setting up a **secure channel** between communicating parties.

Authentication and message integrity

Authentication and message integrity cannot do without each other.

Authentication Based on a Shared Secret Key (1)

Figure 9-12. Authentication based on a shared secret key.



One party challenges the other to a response R that can be correct only if the other knows the shared secret key. Such solutions are also known as challenge-response protocols.

Authentication Based on a Shared Secret Key (2)

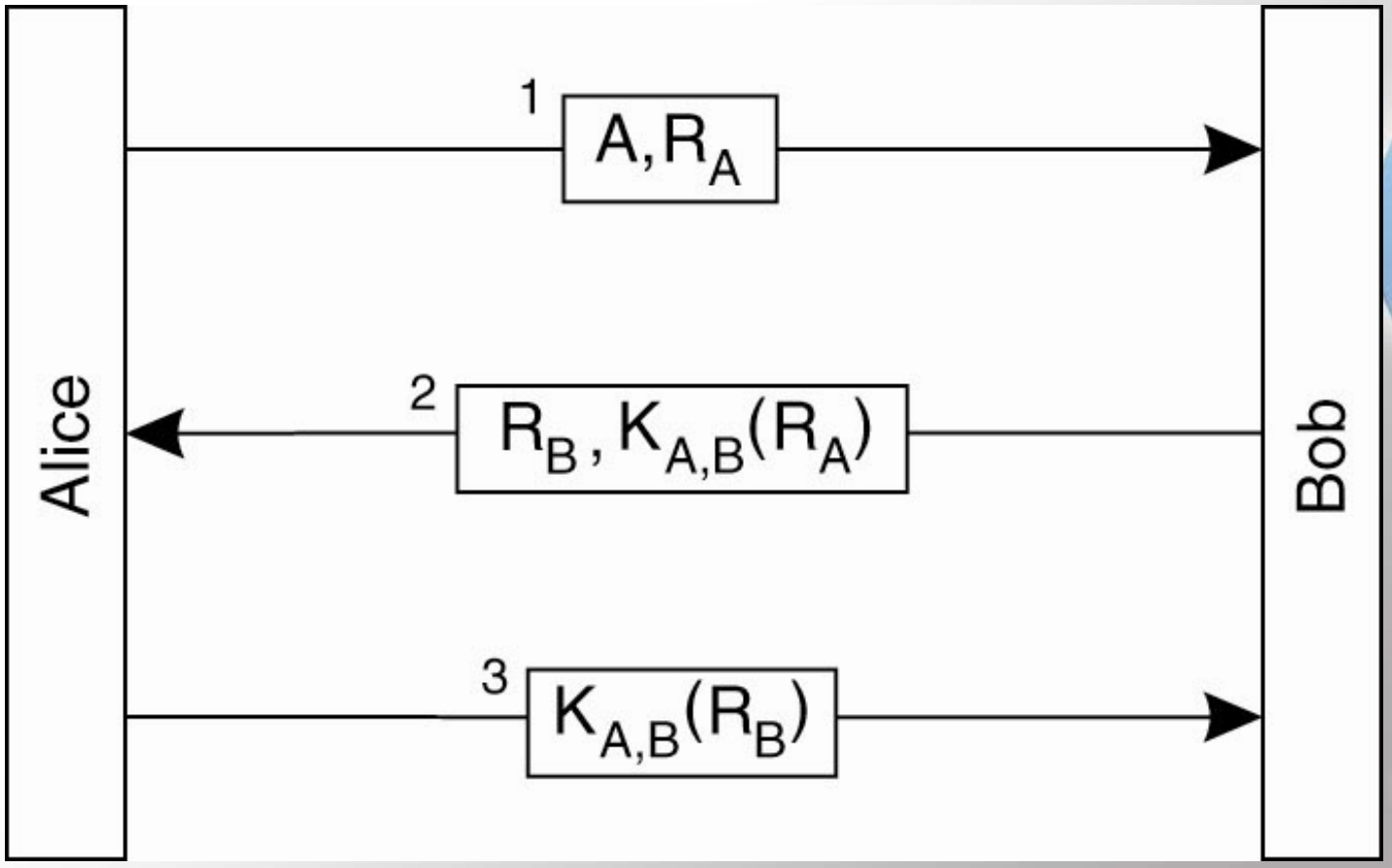


Figure 9-13. Authentication based on a shared secret key, but using three instead of five messages.

Authentication Based on a Shared Secret Key (3)

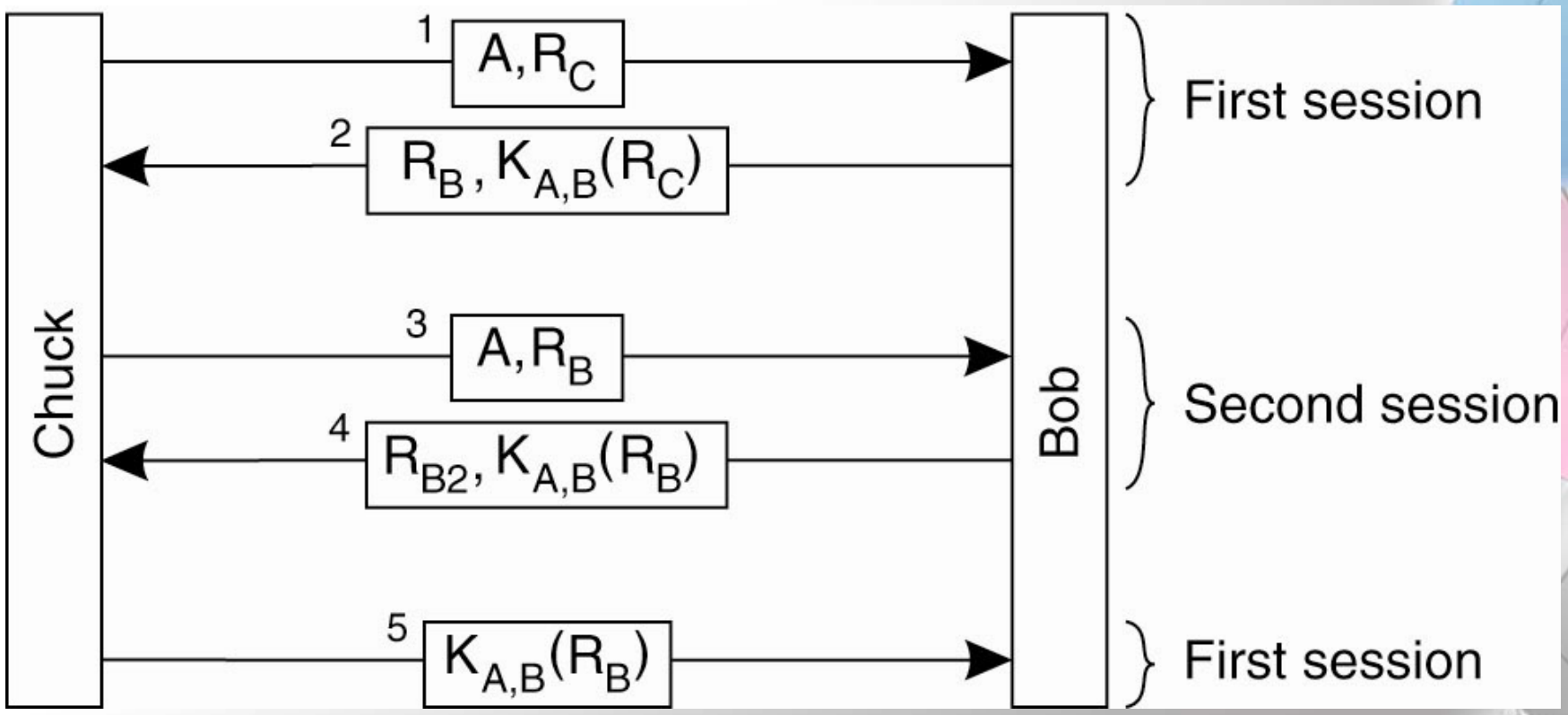


Figure 9-14. The reflection attack.

Authentication Using a Key Distribution Center (1)

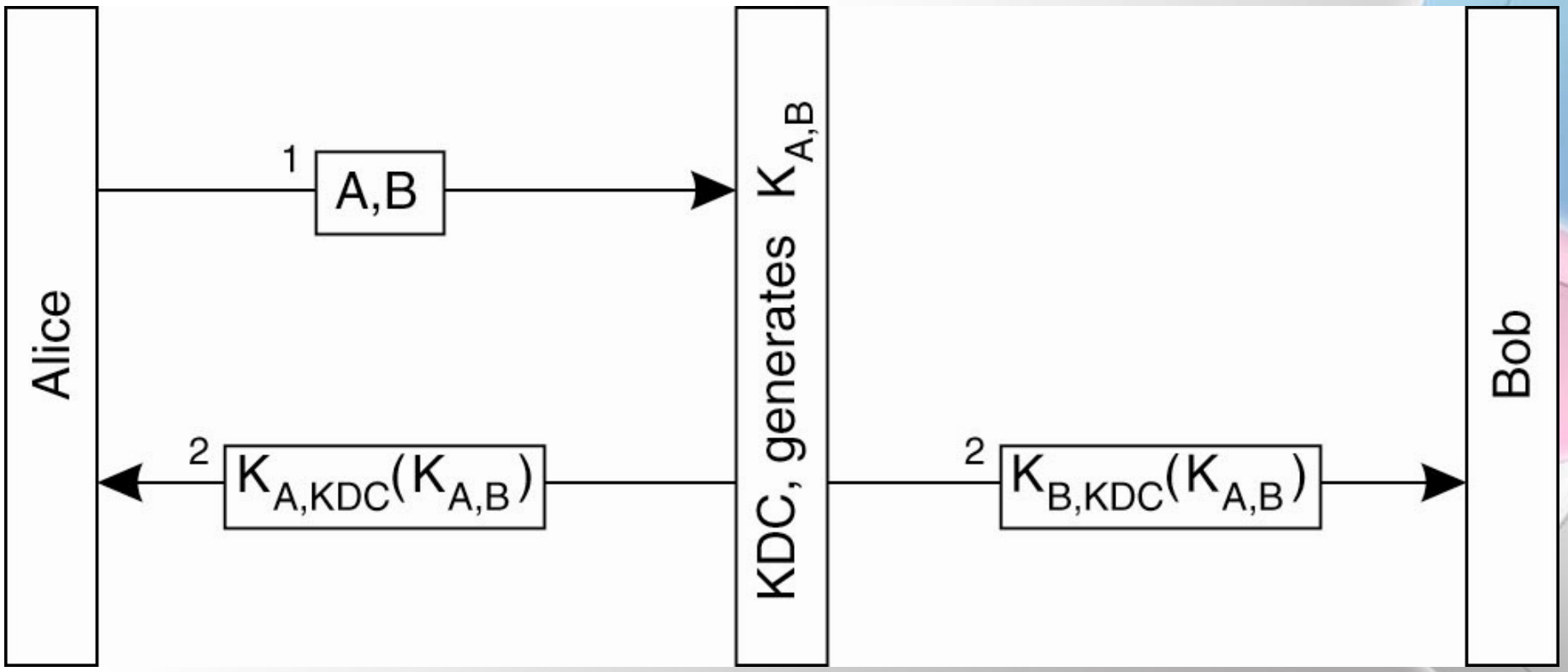


Figure 9-15. The principle of using a KDC.

Authentication Using a Key Distribution Center (2)

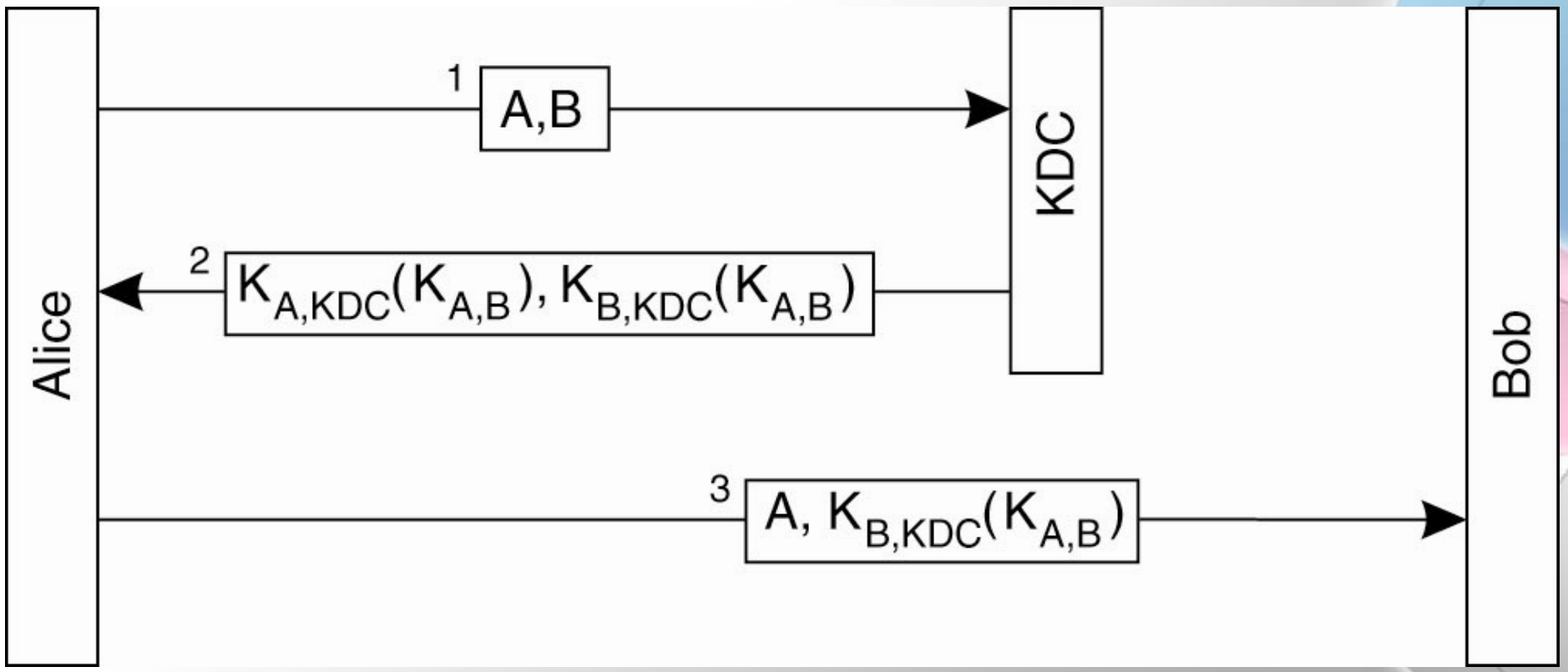


Figure 9-16. Using a ticket [$K_{B,KDC}(K_{A,B})$] and letting Alice set up a connection to Bob.

Authentication Using a Key Distribution Center (3)

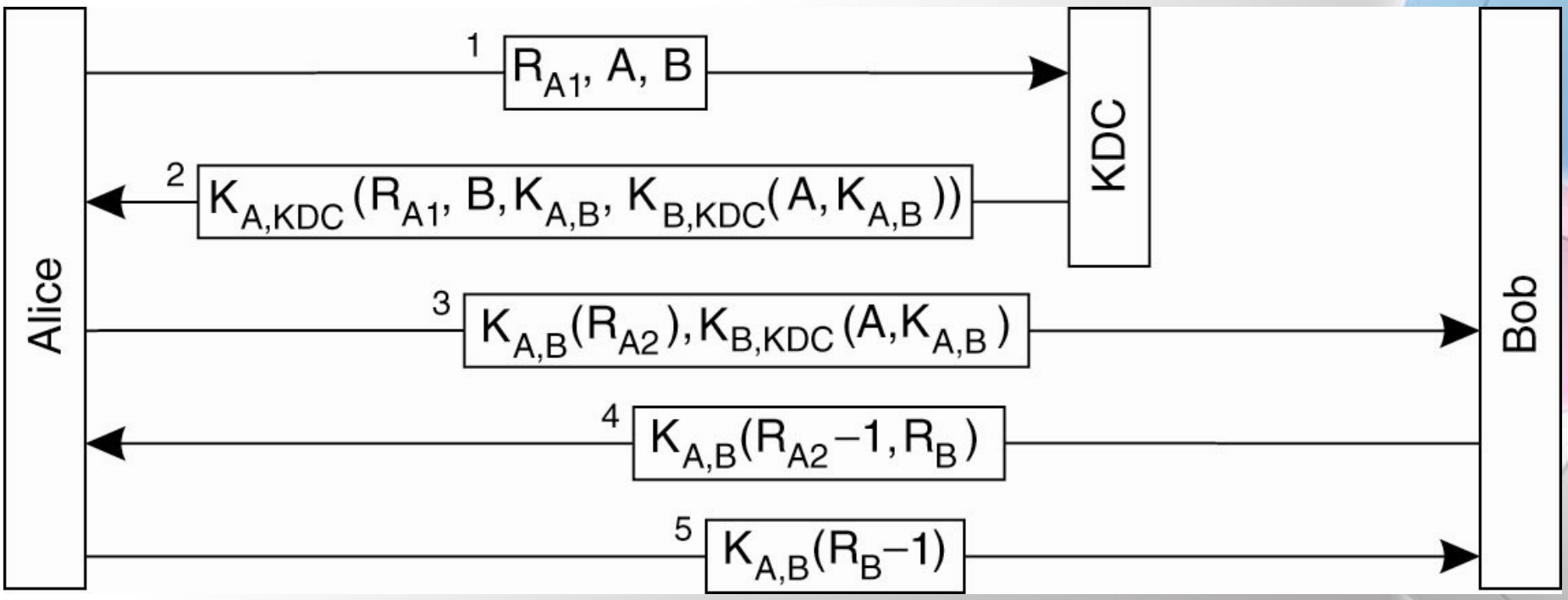


Figure 9-17. The Needham-Schroeder authentication protocol.

Authentication Using a Key Distribution Center (4)

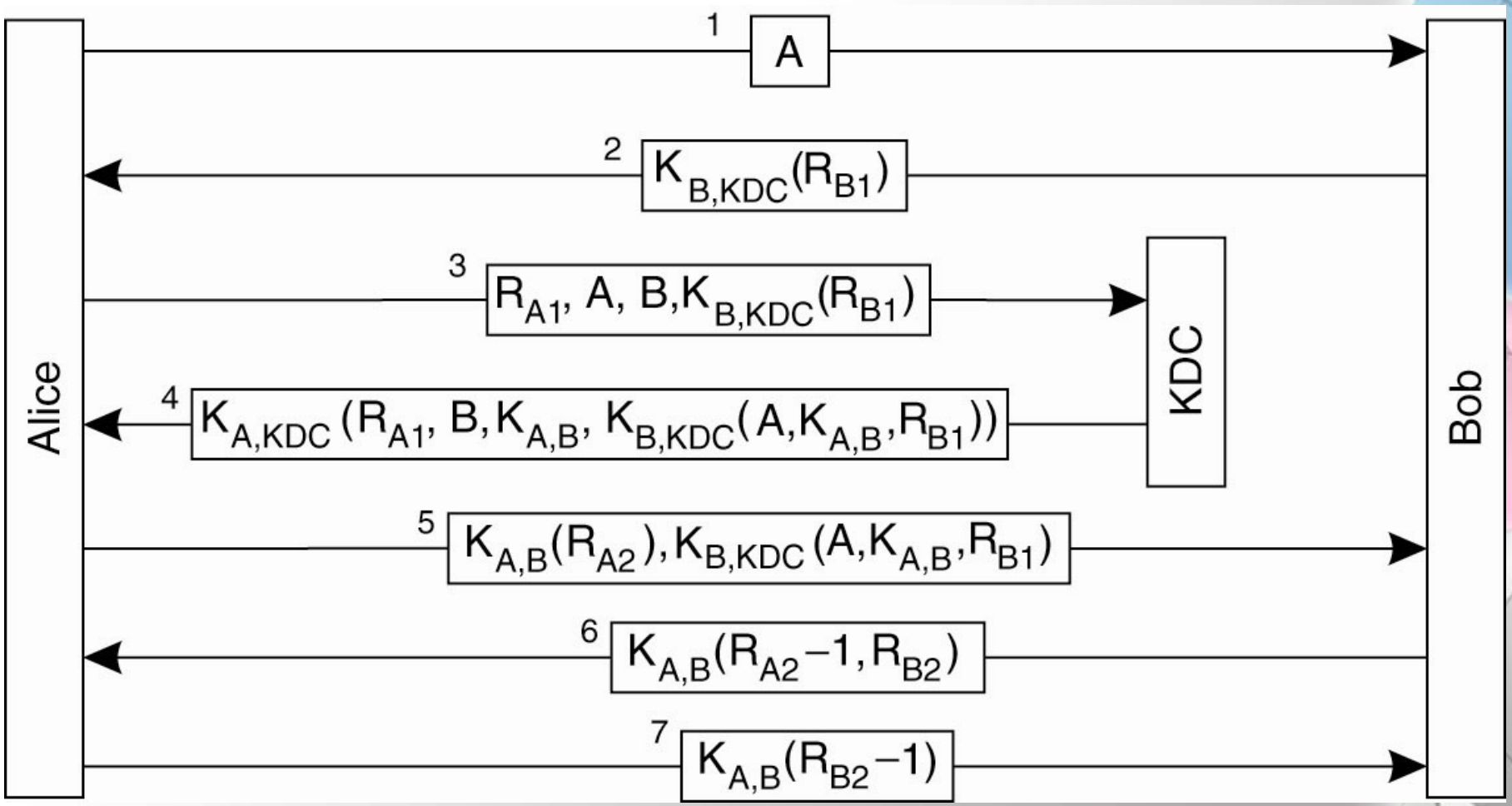


Figure 9-18. Protection against malicious reuse of a previously generated session key in the Needham-Schroeder protocol.

Public-key cryptosystems

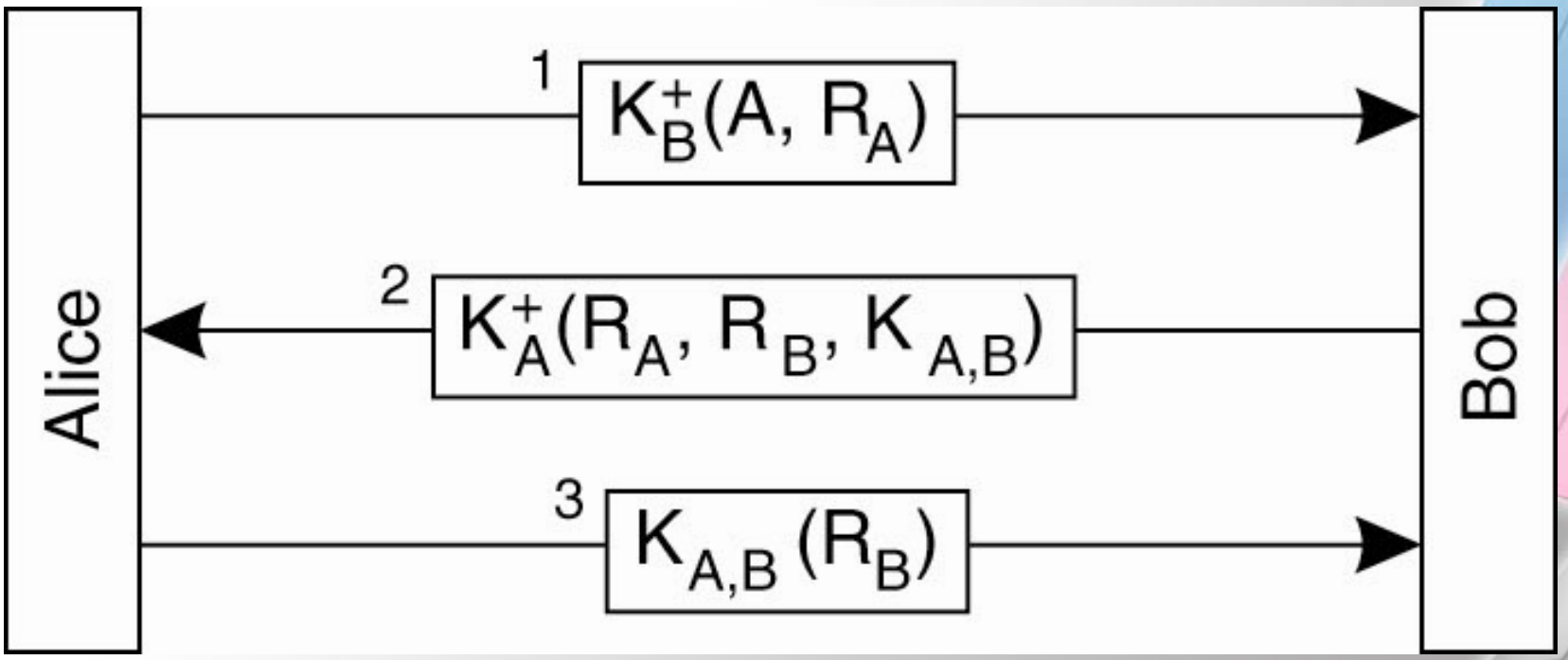


Figure 9-19. Mutual authentication in a public-key cryptosystem.

Message Integrity and Confidentiality

- Besides authentication, a secure channel should also provide guarantees for message integrity and confidentiality.
  - Message integrity means that messages are protected against clandestine modification;
  - Confidentiality ensures that messages cannot be intercepted and read by eavesdroppers.
- Confidentiality is easily established by simply encrypting a message before sending it.
  - Encryption can take place either through a secret key shared with the receiver or alternatively by using the receiver's public key.
- However, protecting a message against modifications is somewhat more complicated.

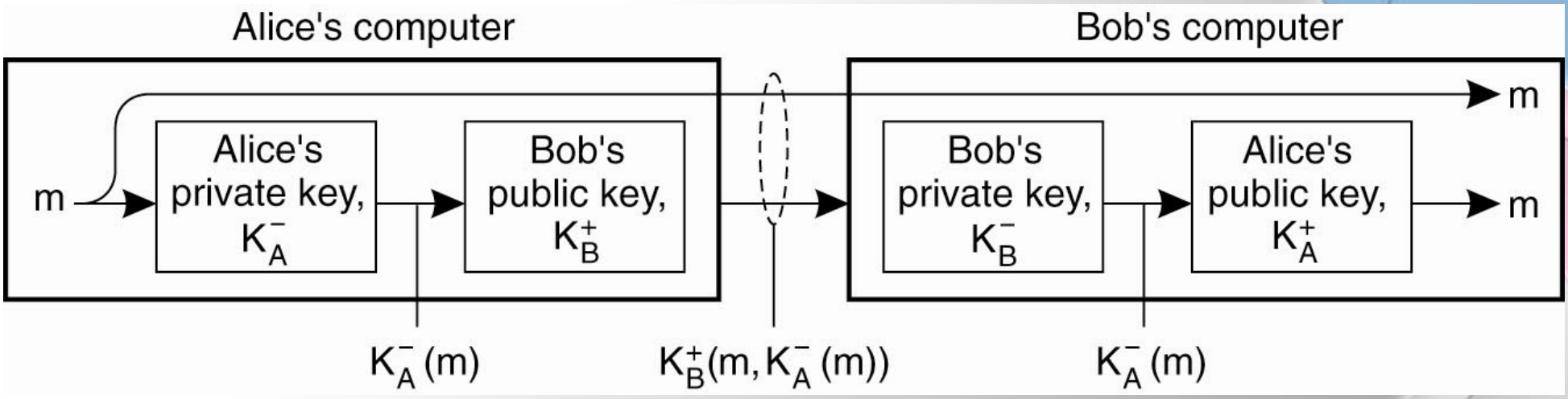
Message changes: why bother?

- Consider the situation in which Bob has just sold Alice something for $500. The whole deal was done through e-mail. In the end, Alice sends Bob a message confirming that she will buy the record for $500. In addition to authentication, there are at least two issues that need to be taken care of regarding the integrity of the message.

1. Alice needs to be assured that Bob will not maliciously change the $500 mentioned in her message into something higher, and claim she promised more than $500.
2. Bob needs to be assured that Alice cannot deny ever having sent the message, for example, because she had second thoughts.

Digital Signatures (1)

Figure 9-20. Digital signing a message using public-key cryptography.



Alice digitally signs the message in such a way that her signature is uniquely tied to its content.

Digital Signatures (2)

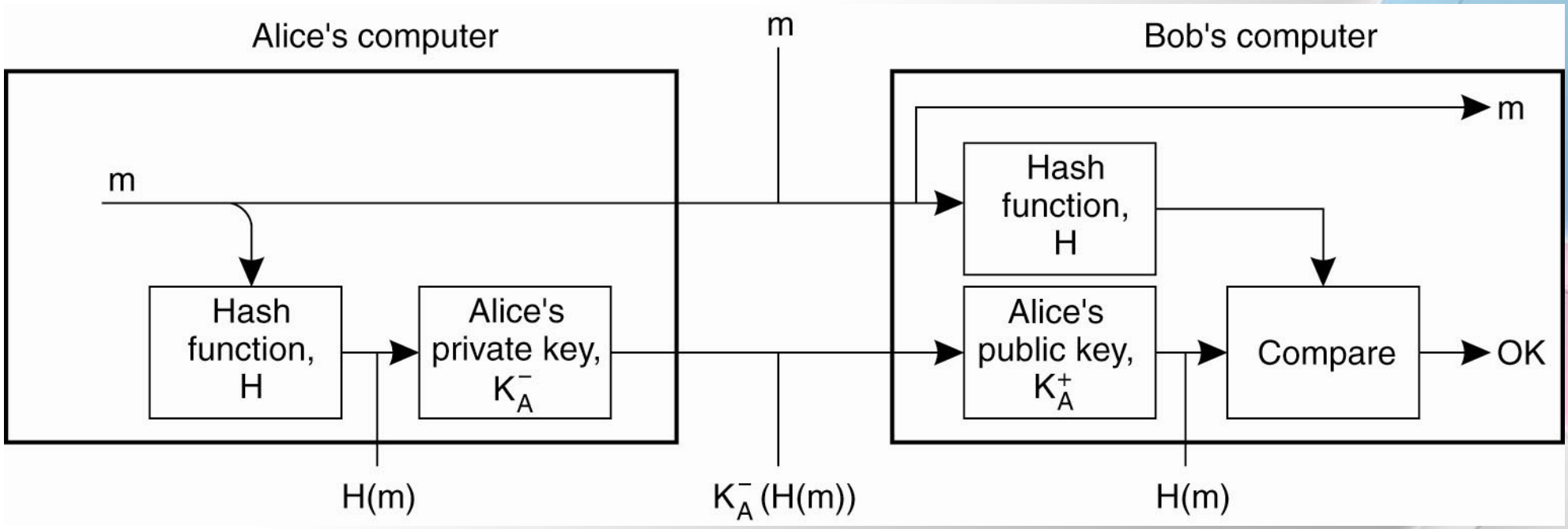


Figure 9-21. Digitally signing a message using a message digest.

Session key

- During the establishment of a secure channel, after the authentication phase has completed, the communicating parties generally use a unique shared session key for confidentiality.
  - The session key is safely discarded when the channel is no longer used.
- Benefits to using session keys
  - Much safer when authentication keys are used as little as possible.
  - Ensure protection against replay attacks

Secure Group Communication

- In distributed systems, it is often necessary to enable secure communication between more than just two parties.
- A typical example is that of a replicated server for which all communication between the replicas should be protected against modification, fabrication, and interception, just as in the case of two-party secure channels.

Confidential Group Communication

Consider the problem of protecting communication between a group of *N* users against eavesdropping.

1. Let all group members share the same secret key, which is used to encrypt and decrypt all messages transmitted between group members.
   - Single secret key: more vulnerable
2. Use a separate shared secret key between each pair of group members
   - need to keep $N(N-1)/2$ keys, which may be a difficult problem by itself.
3. Using a public-key cryptosystem can improve matters. In that case, each member has its own *(public key, private key)* pair, in which the public key can be used by all members for sending confidential messages. In this case, a total of *N* key pairs are needed.

Secure Replicated Servers

- Simple solution
  - Collect the responses from all servers and authenticate each one of them. If a majority exists among the responses from the noncorrupted (i.e., authenticated) servers, the client can trust the response to be correct as well. Unfortunately, this approach reveals the replication of the servers, thus violating replication transparency
- Secret Sharing
  - When multiple users (or processes) share a secret, none of them knows the entire secret. Instead, the secret can be revealed only if they all get together.

Secure Replicated Servers

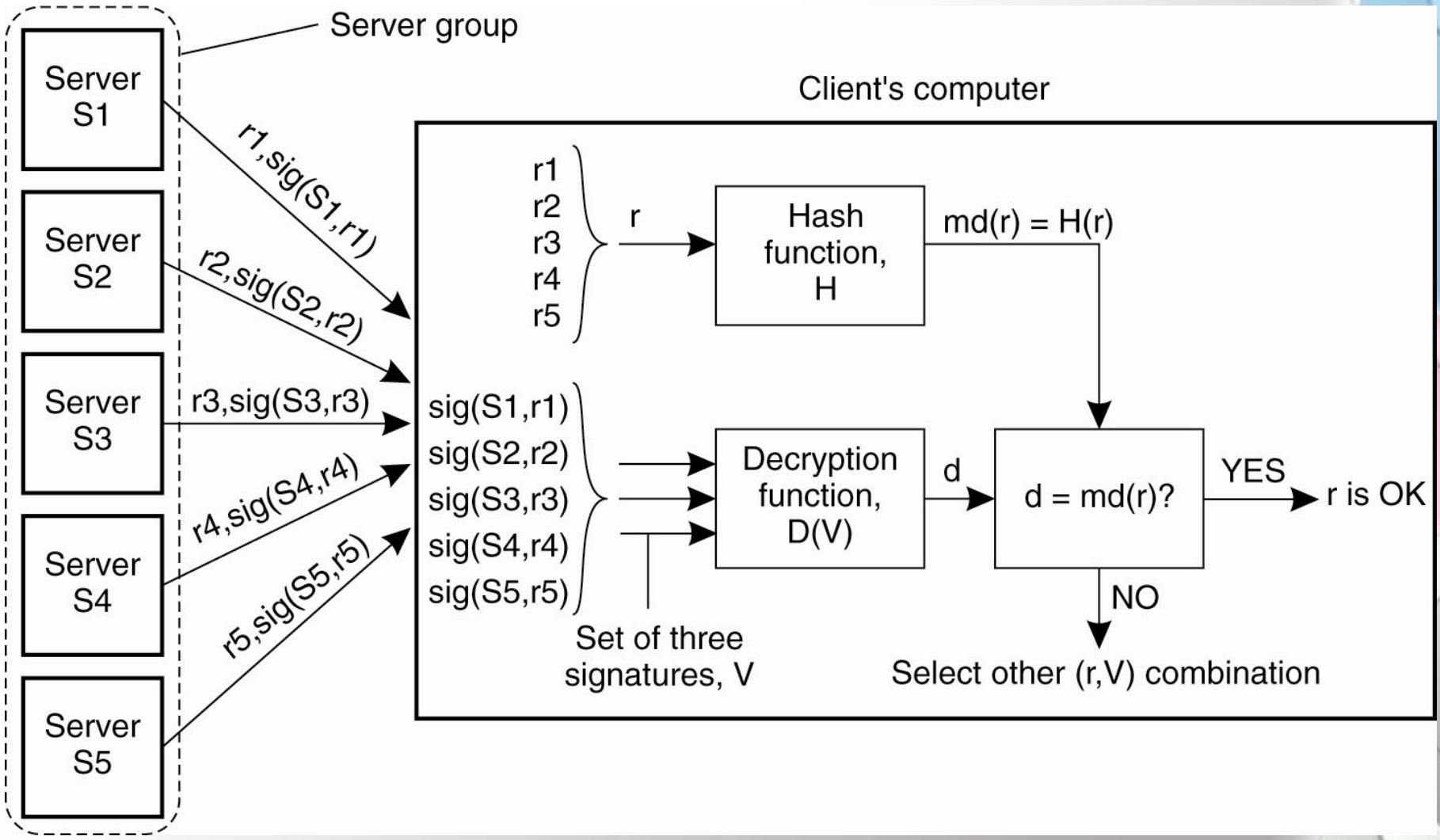


Figure 9-22. Sharing a secret signature in a group of replicated servers.

Example: Kerberos (1)

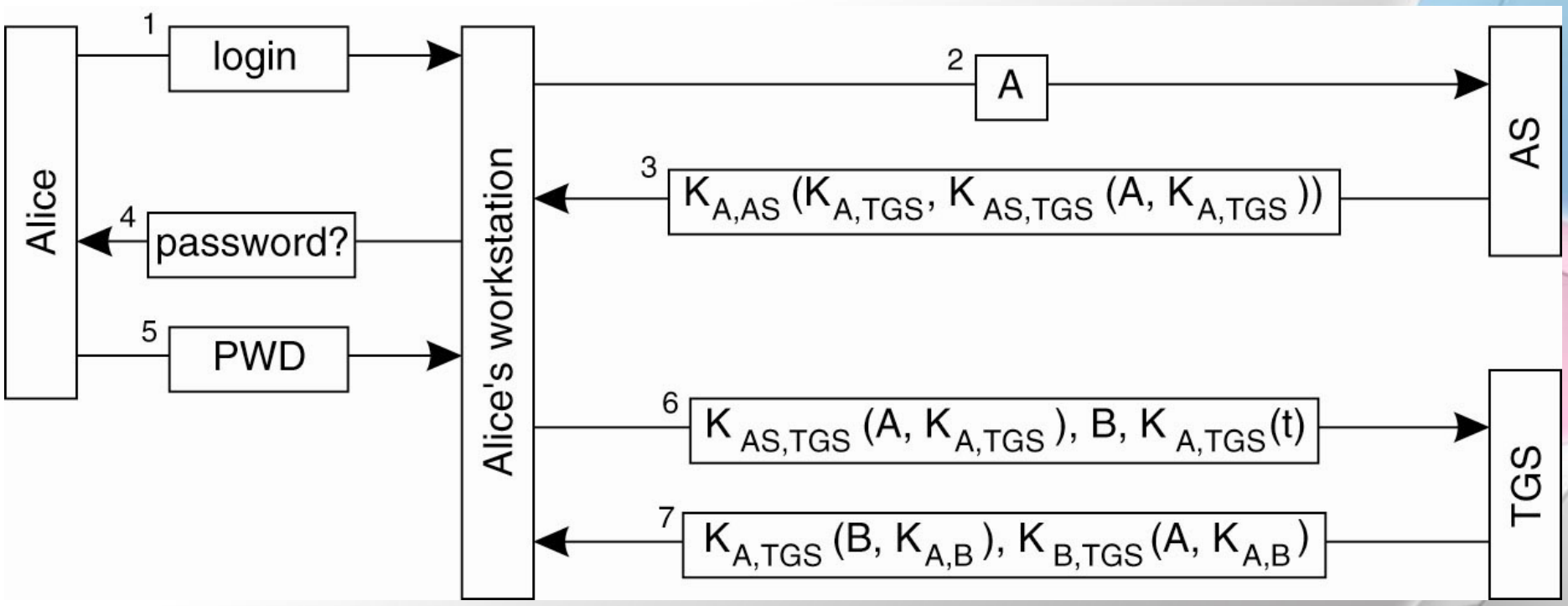


Figure 9-23. Authentication in Kerberos.

Example: Kerberos (2)

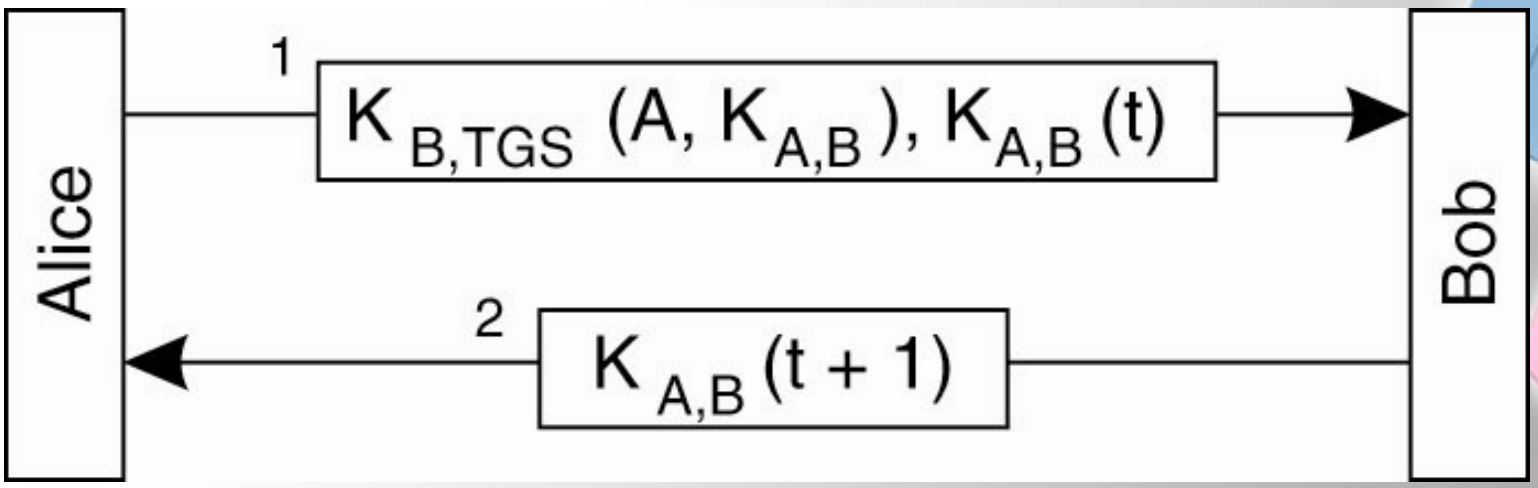


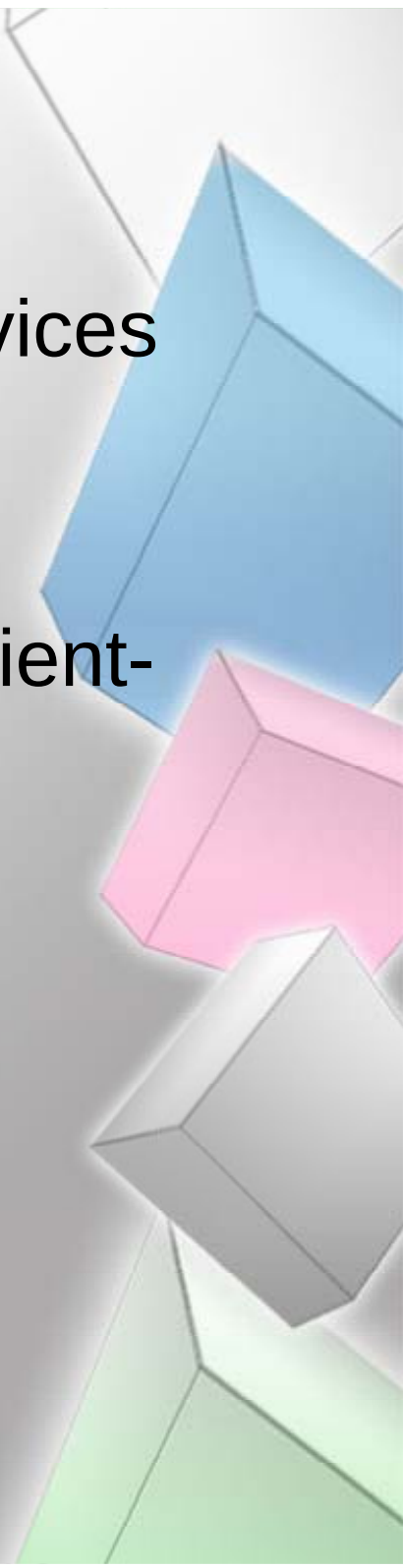
Figure 9-24. Setting up a secure channel in Kerberos.

Project Proposal 1

- Kerberos
  - Setup and Configuration of Kerberos: KDC
  - Create a very simple KDC database
  - Simple test of the system: authenticate yourself and
- Requirements
  - Basic knowledge of Linux
  - If you have only Windows, and do not want a new partition, you can use virtualization software, ex. VMWare.
- You will not be left alone! ☺

GSSAPI

- GSSAPI stands for Generic Security Services Application Programming Interface.
- The GSSAPI is a generic API for doing client-server authentication.
- JAVA GSS-API



Security

- Introduction to security
  - Cryptography
- Secure channels
- Access Control
- Security Management

General Issues in Access Control

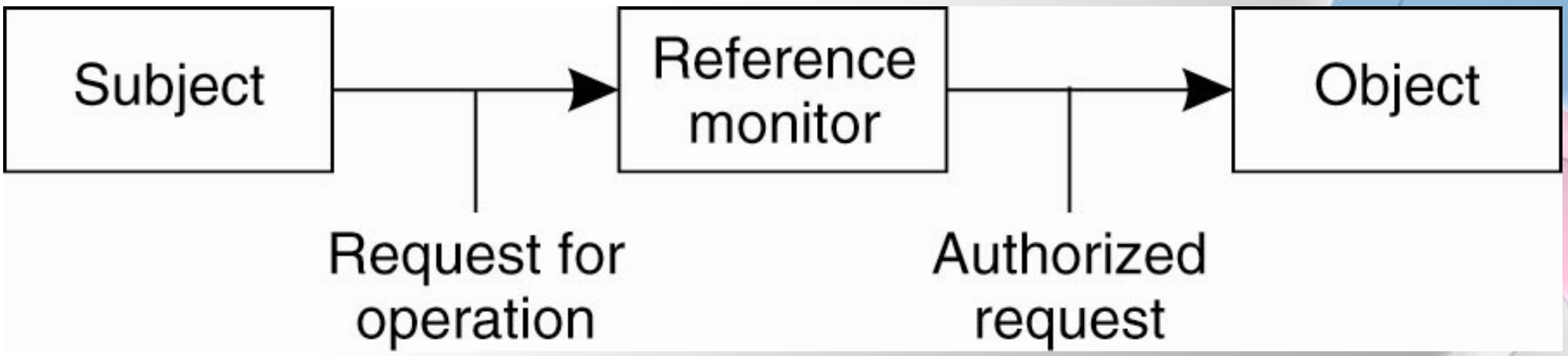
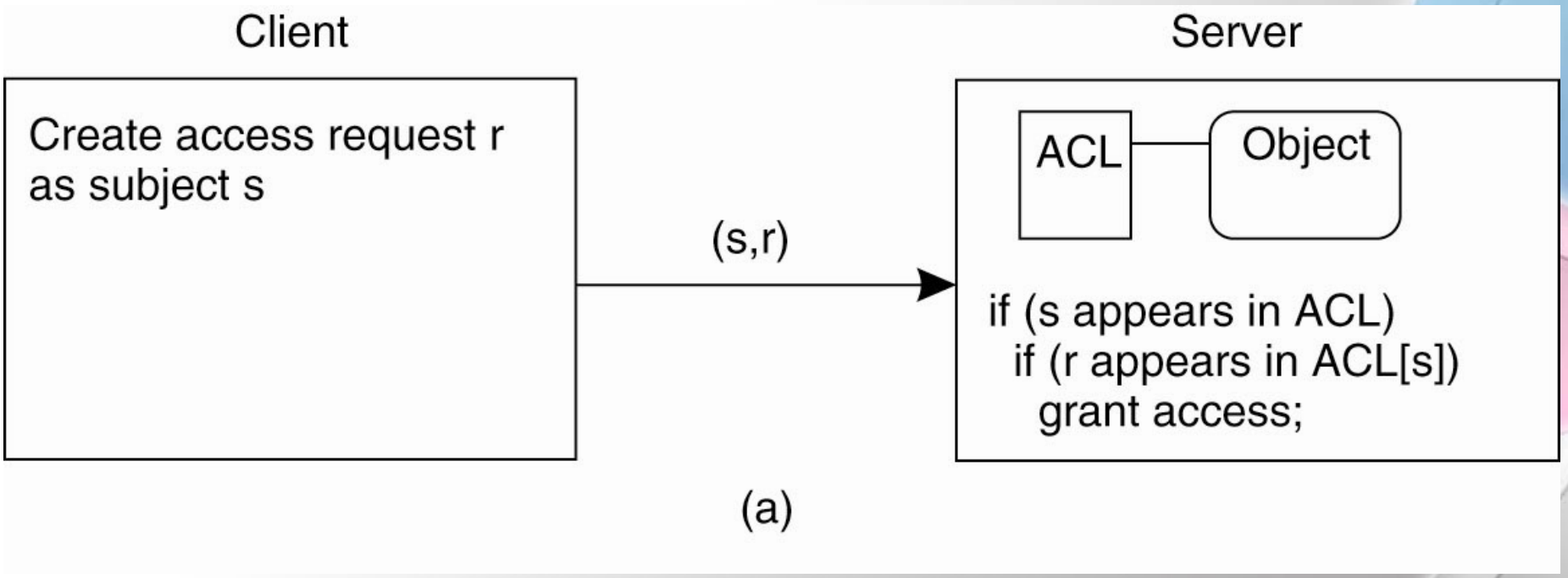


Figure 9-25. General model of controlling access to objects.

Access Control Matrix (1)



Access Control Matrix: maintain a unique matrix for subjects and objects.

ACL (Access Control List), distributed matrix, where each object has it's own ACL.

Access Control Matrix (2)

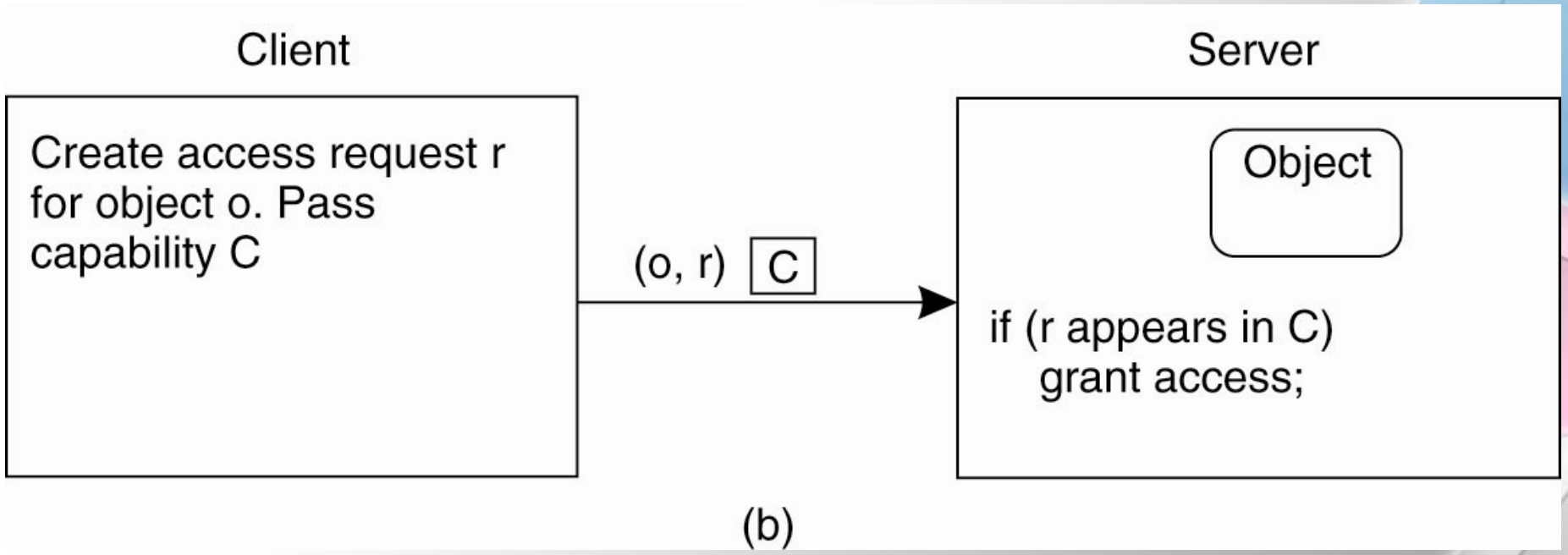


Figure 9-26. Comparison between ACLs and capabilities for protecting objects. Each subject is given a list of capabilities for each object.

Protection Domains

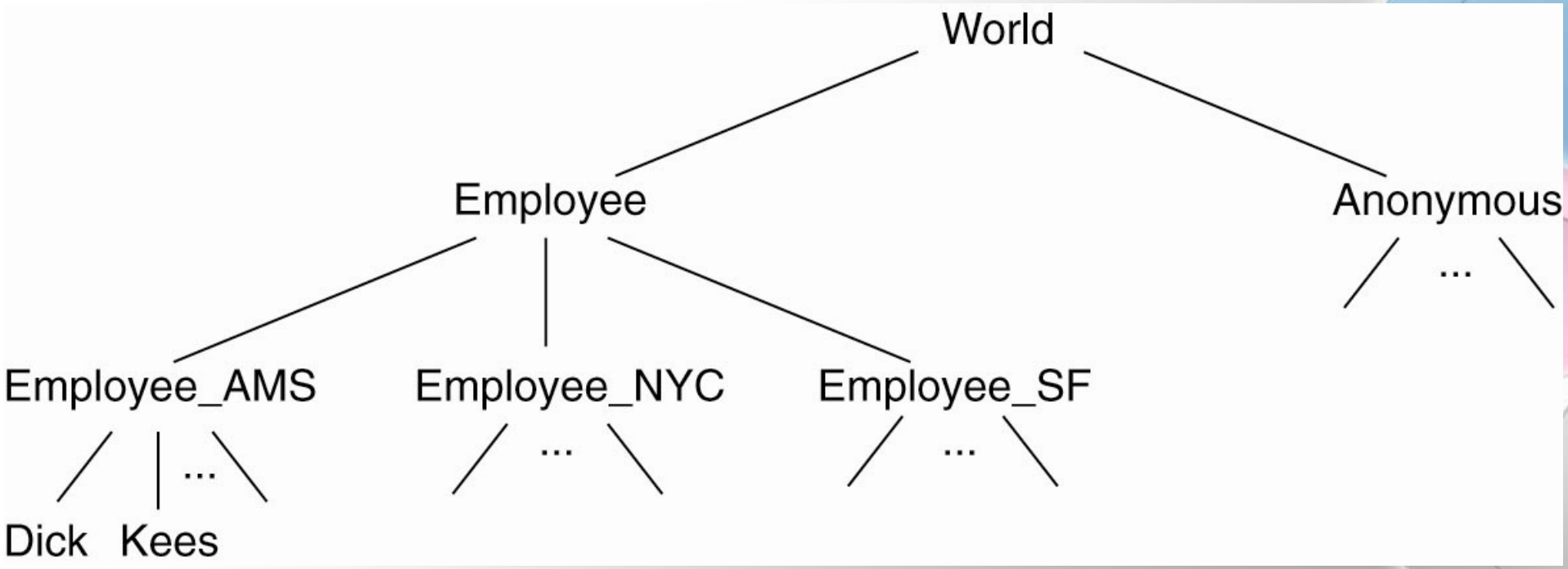


Figure 9-27. The hierarchical organization of protection domains as groups of users.

Firewalls: packets level

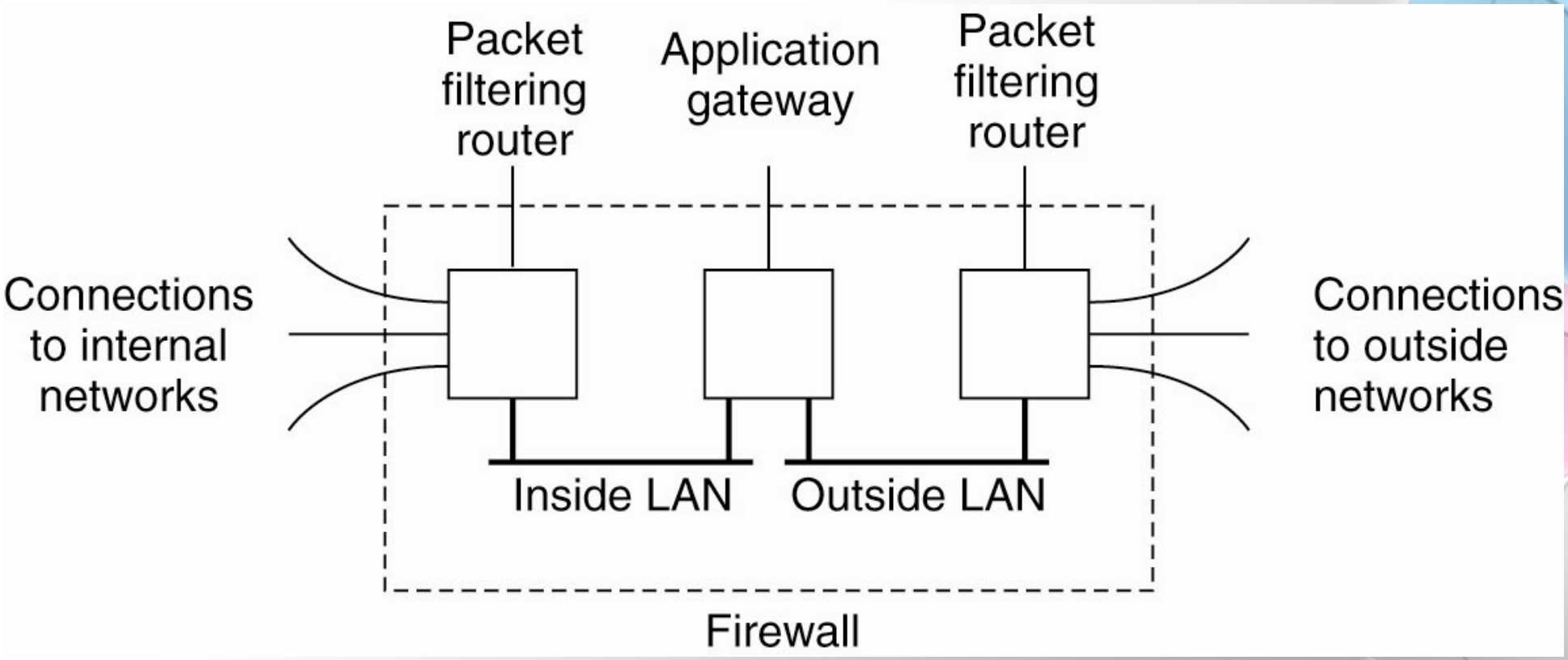


Figure 9-28. A common implementation of a firewall.

Firewall: application level

- Application-level gateway
  - This type of firewall checks the content of incoming and outgoing messages.

Examples

- Mail gateway
  - Filters spam or size-exceeding mails.
- Digital library service
  - Usually commercial services provide only abstracts to external users. If the user wants more, an electronic payment protocol is started.

Protecting the Target (1)

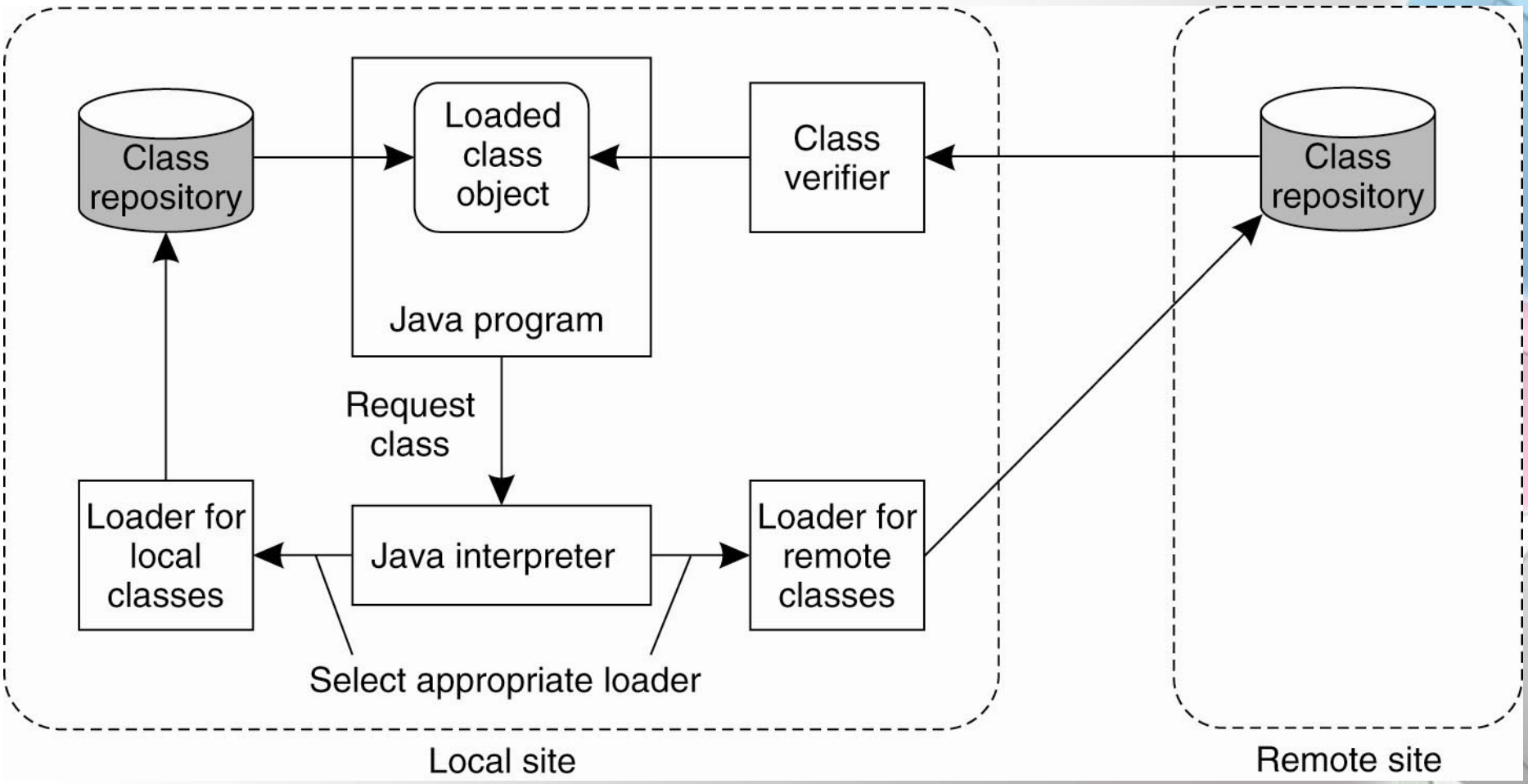
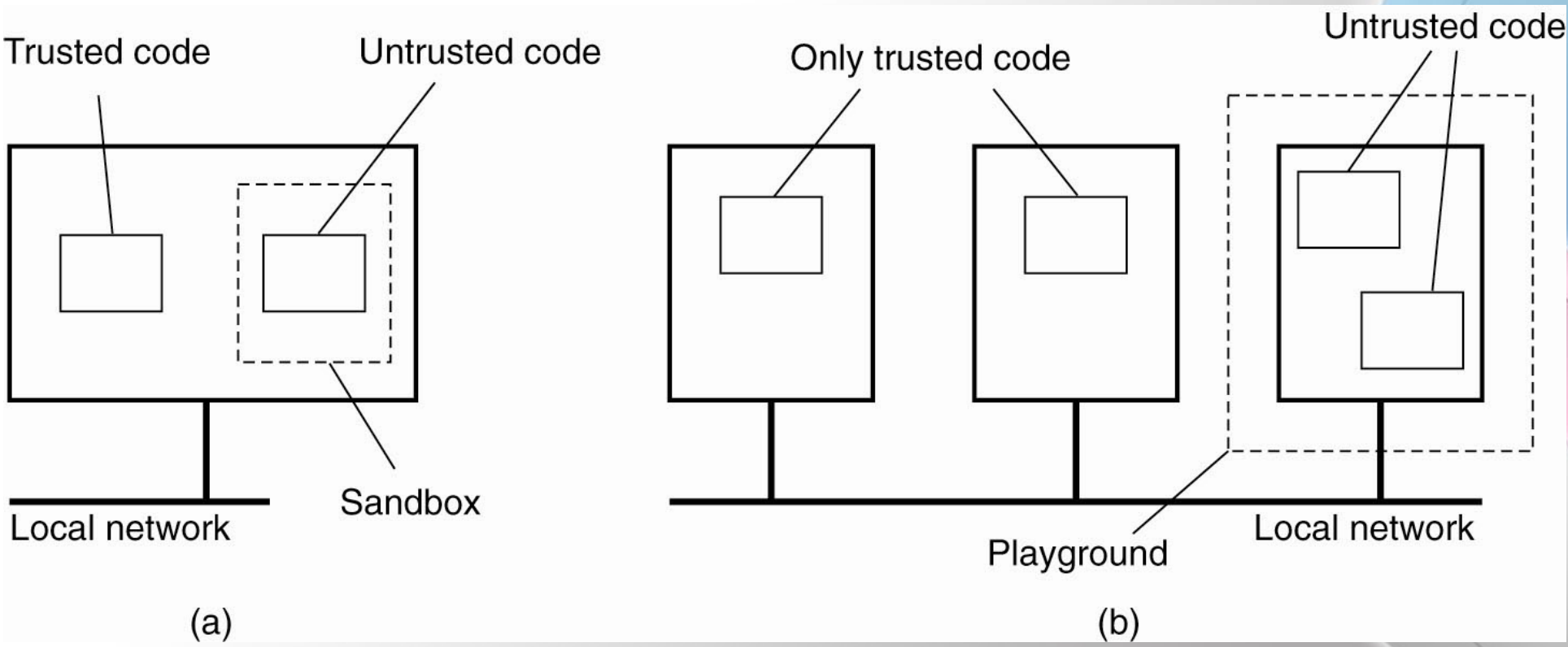


Figure 9-29. The organization of a Java sandbox.

Protecting the Target (2)

Figure 9-30. (a) A sandbox. (b) A playground.



A playground is a separate, designated machine exclusively reserved for running mobile code.

Code-signing

- A next step toward increased flexibility is to require that each downloaded program can be authenticated, and to subsequently enforce a specific security policy based on where the program came from.
- Demanding that programs can be authenticated is relatively easy: mobile code can be signed, just like any other document.
- This code-signing approach is often applied as an alternative to sandboxing as well.
  - In effect, only code from trusted servers is accepted.

Denial of service

- A type of attack that is related to access control is maliciously preventing authorized processes from accessing resources.
  - Defenses against such denial-of-service (DoS) attacks are becoming increasingly important as distributed systems are opened up through the Internet.
- Where DoS attacks that come from one or a few sources can often be handled quite effectively, matters become much more difficult when having to deal with distributed denial of service (DDoS).
- In DDoS attacks, a huge collection of processes jointly attempt to bring down a networked service.
  - In these cases, we often see that the attackers have succeeded in hijacking a large group of machines which unknowingly participate in the attack.

Denial of service

- Two types of attacks: those aimed at bandwidth depletion and those aimed at resource depletion.
- Bandwidth depletion can be accomplished by simply sending many messages to a single machine.
  - The effect is that normal messages will hardly be able to reach the receiver.
- Resource depletion attacks concentrate on letting the receiver use up resources on otherwise useless messages.
  - A well-known resource-depletion attack is TCP SYN-flooding. In this case, the attacker attempts to initiate a huge amount of connections (i.e., send SYN packets as part of the three-way handshake), but will otherwise never respond to acknowledgments from the receiver.

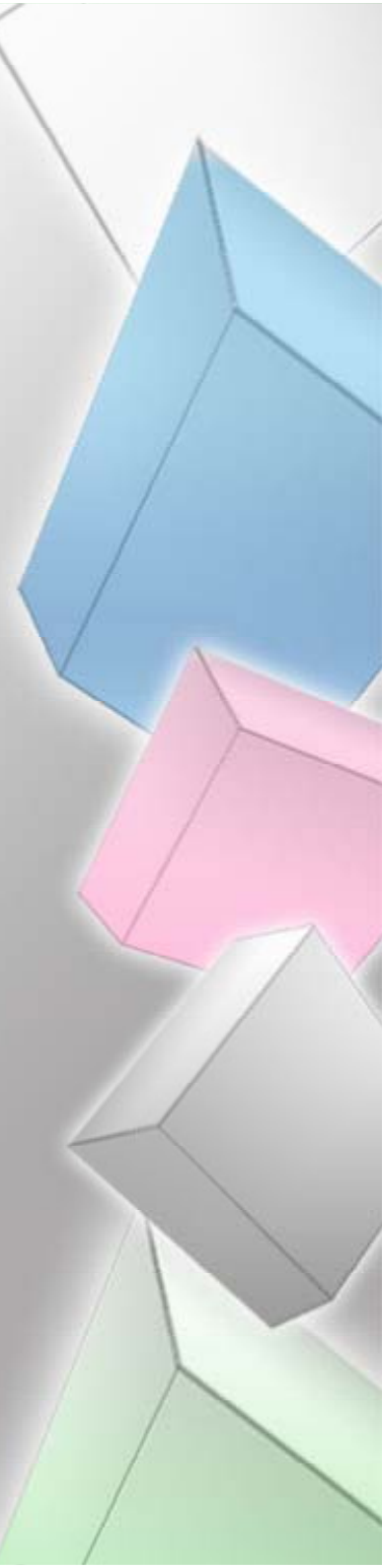
Denial of service

Solutions:

1. Attackers make use of innocent victims by secretly installing software on their machines. In these cases, the only solution is to have machines continuously monitor their state by checking files for pollution.
2. Continuously monitor network traffic, for example, starting at the egress routers where packets leave an organization's network.
3. Concentrate on ingress routers, that is, where traffic flows into an organization's network.
4. Have routers further in the Internet, such as in the networks of ISPs, start dropping packets when they suspect that 'an attack is going on.

Security

- Introduction to security
  - Cryptography
- Secure channels
- Access Control
- Security Management



Key Establishment

The first requirement is that Alice and Bob, both agree on two large numbers *n* and *g. (these are subject to some mathematical properties).*

It is virtually impossible to compute *x* given $g^x \bmod n$.

$g^{xy} \bmod n$ is the shared secret key.

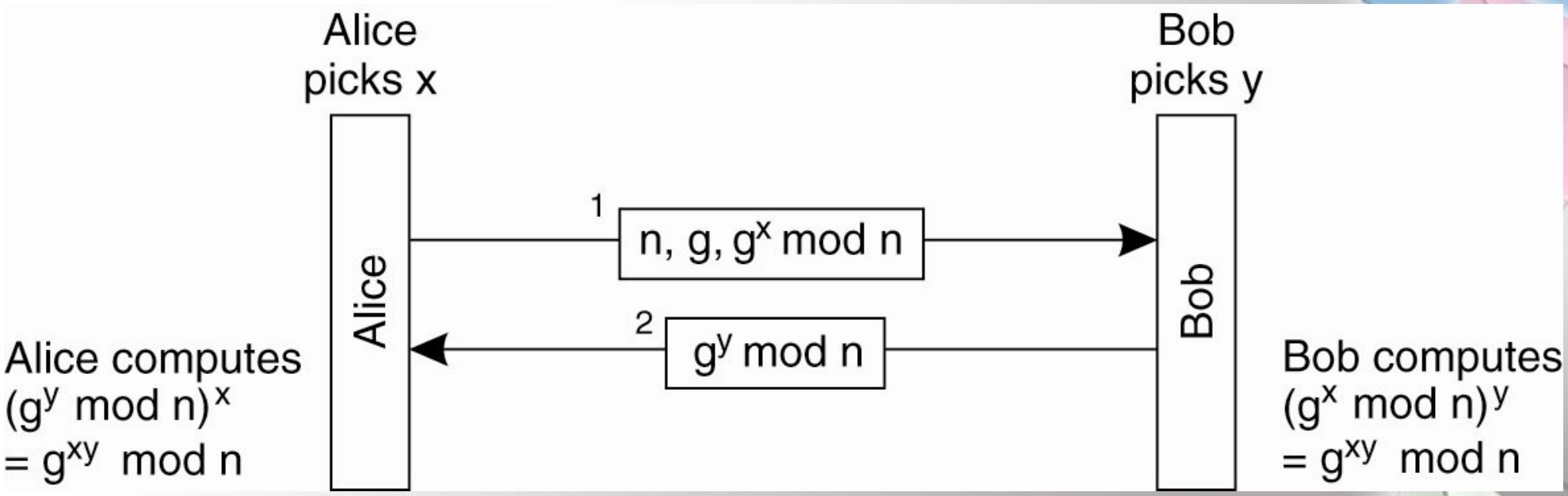


Figure 9-33. The principle of Diffie-Hellman key exchange.

Key Distribution (1)

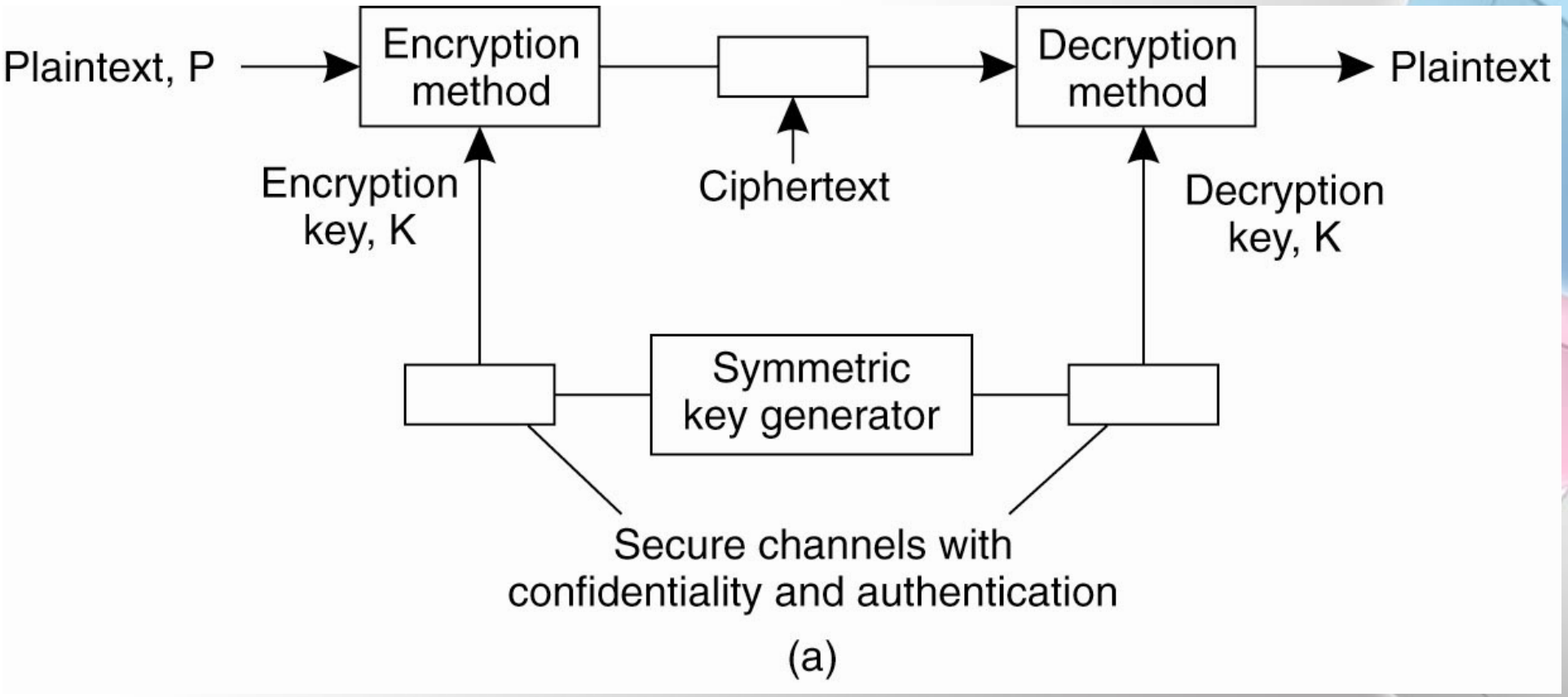


Figure 9-34. (a) Secret-key distribution.

Key Distribution (2)

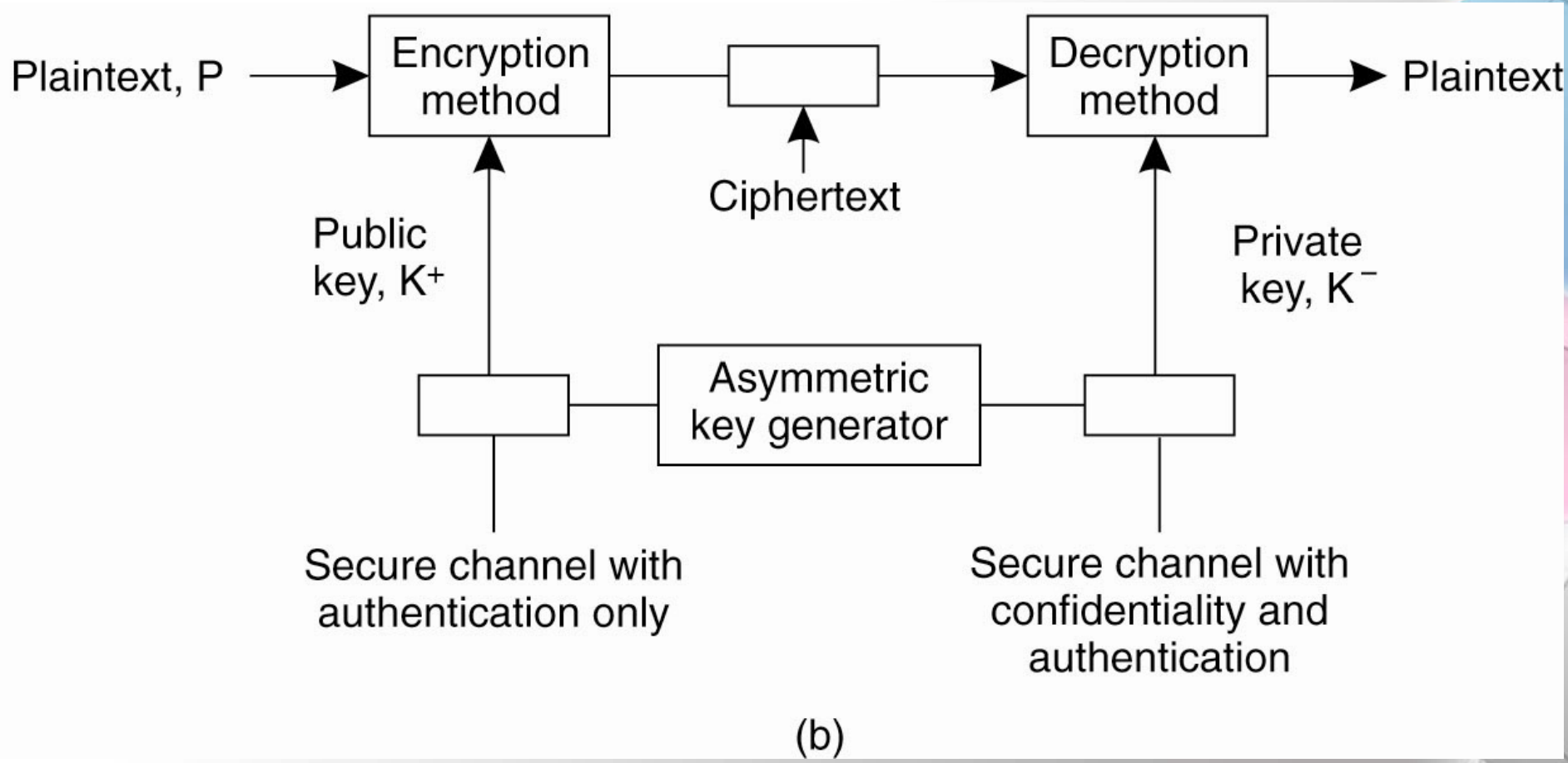


Figure 9-34. (b) Public-key distribution

Public-key certificates

- In practice, public-key distribution takes place by means of public-key certificates.
  - Such a certificate consists of a public key together with a string identifying the entity to which that key is associated.
- The entity could be a user, but also a host or some special device. The public key and identifier have together been signed by a certification authority, and this signature has been placed on the certificate as well
  - The identity of the certification authority is naturally part of the certificate.
- Signing takes place by means of a private key K^-_{CA} that belongs to the certification authority. The corresponding public key K^+_{CA} is assumed to be well known.
  - Public keys of various certification authorities are built into most Web browsers and shipped with the binaries.

Lifetime of Certificates

- An important issue concerning certificates is their longevity. First let us consider the situation in which a certification authority hands out lifelong certificates.
  - Essentially, what the certificate then states is that the public key will always be valid for the entity identified by the certificate.
- Clearly, such a statement is not what we want. If the private key of the identified entity is ever compromised…
- There are several ways to revoke a certificate. One common approach is with a Certificate Revocation List (CRL) published regularly by the certification authority.
  - Whenever a client checks a certificate, it will have to check the CRL to see whether the certificate has been revoked or not.

End of Lesson 9

- Readings
  - Distributed Systems, Chapter 9. Except. 9.4.2 and 9.4.3.

Project Proposal 1

- Kerberos
  - Setup and Configuration of Kerberos
  - Create a very simple KDC database
  - Simple test of the system: authenticate yourself
- Requirements
  - Basic knowledge of Linux
  - If you have only Windows, and do not want a new partition, you can use virtualization software, ex. VMWare.
- You will not be alone! ☺