Security in Distributed Systems

Agenda

❑ Introduction

❑ Cryptography

❑ Secure Channels

❑ Access Control

❑ Security Management

Introduction

- Security is one of the most important principles , since security need to be pervasive through the system.

- Security policies decide the security goals of a computer system and these goals are achieved through various security mechanism.

Goals of computer security

- Secrecy : information within the system must be accessible only to authorized users.
- Privacy : information given to the users must be used only for the purpose for which was given.
- Authenticity :the user must be able to verify that the data obtained is from expected sender only.
- Integrity :information must be protected from unauthorized access.

Potential threat and Attacks on computer security

- **Threat:** is a possible danger that might exploit a vulnerability to breach security and thus cause possible harm.

- **Attack:** is any attempt to destroy, expose, alter, disable, steal or gain unauthorized access to or make unauthorized use of an asset.

Type of threats

- **Interception.** Unauthorized user gaining access to a service or data. E.g. eavesdropping, illegal copying.
- **Interruption**. Services or data becoming unavailable, unusable, destroyed. E.g. intentional file corruption, denial of service attacks.
- **Modification**. Unauthorized changing of data or service so that it no longer adheres to its original specification.
- **Fabrication.** Additional data or activity is generated that would normally not exist. E.g., adding entry to password file or database, breaking into a system by replaying previously sent messages.

Security Policy and Mechanisms

- Security policy describes what actions the entities in a system are allowed to take and which ones are prohibited
- Security mechanisms implement security policies.
- The following techniques are used:
  - **Encryption:** provides a means to implement confidentiality , since it transforms the data into some think which attacker cannot understand.
  - **Authentication**: to verify weather the user, client , server ,etc.

    Are authentic. User are authenticated by password.
  - **Authorization:** to check as a weather the client is authorized to perform specific task.
  - **Auditing:** tools are used to trace which clients accessed what information and when they did so.

Type of attack

- **Passive attacks**
- Browsing
- Inferencing
- Masquerading

- **Active attacks**
- Virus
- Worm
- Logic bomb
- Integrity attack
- Authenticity attack
- Delay attack
- Replay attack
- Denial attack

Intruder : person/program vying for unauthorized access to data

Passive attack

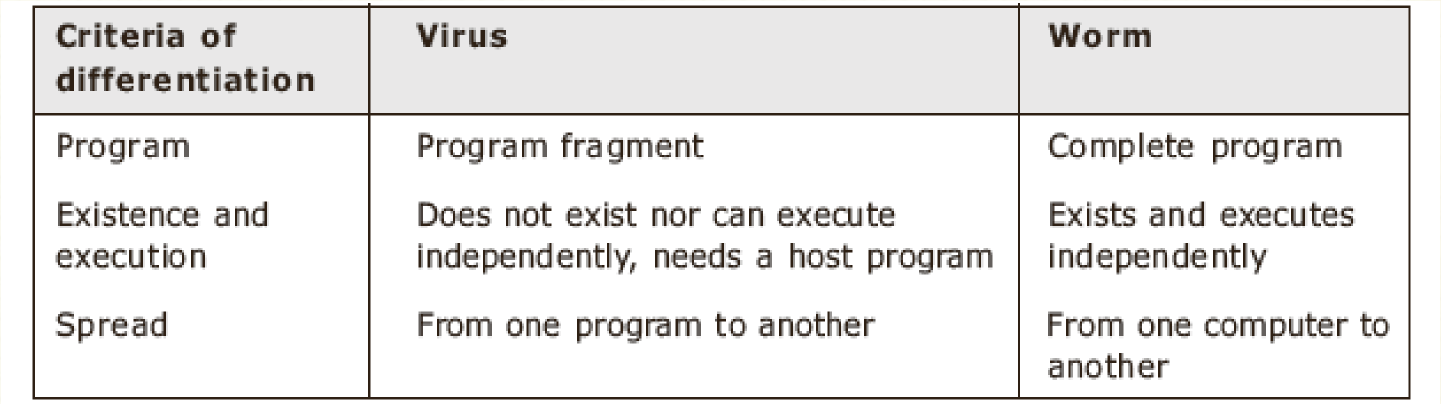
- ❖ Intruder access unauthorized information from a computer system but not cause harm to the system.
- Browsing: Intruders here attempt to read stored files, traverse message packet on the network, access other process memory, etc.
- Inferencing: The intruder records and analyzes past activities and access methods and uses this information to draw inferences
- Masquerading: an intruder Masquerades as an authorized user or a program to gain access to unauthorized data or resource.

Active attack

- **Virus:** is a small computer program that needs to be executed by either running it or having it loaded from boot sector of a disk.
- an intruder writes a useful program and attaches the virus to it, such as when the program executed the virus is also executed.

- **Worm:** A worm is a small piece of software the uses computer network and security holes to replicate it self.

Virus vs worm



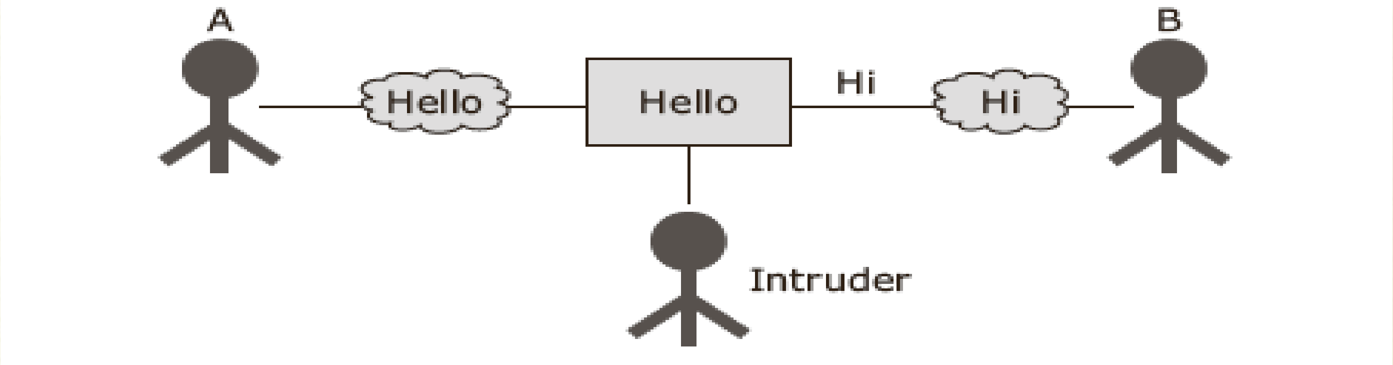
Criteria of differentiation	Virus	Worm
Program	Program fragment	Complete program
Existence and execution	Does not exist nor can execute independently, needs a host program	Exists and executes independently
Spread	From one program to another	From one computer to another

Active attack

- **Logic bomb:** is a piece of code intentionally inserted into a software system that will set off a malicious function when specified conditions are met. When ‘exploded’ may be designed to display a message ,delete or corrupt data.

Active attack

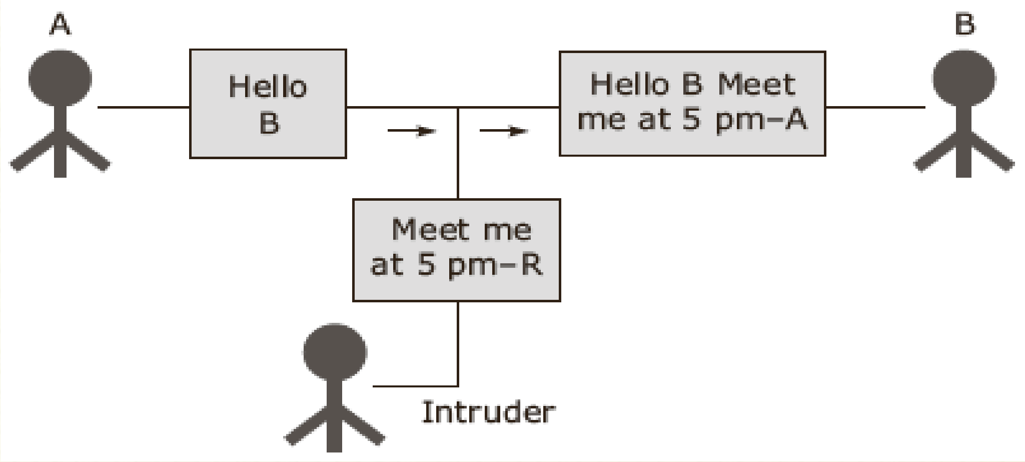
- **Integrity attack :** an intruder can change the message while it is traveling in the communication channel and the receiver may interpret it as original message.



Integrity attack

Active attack

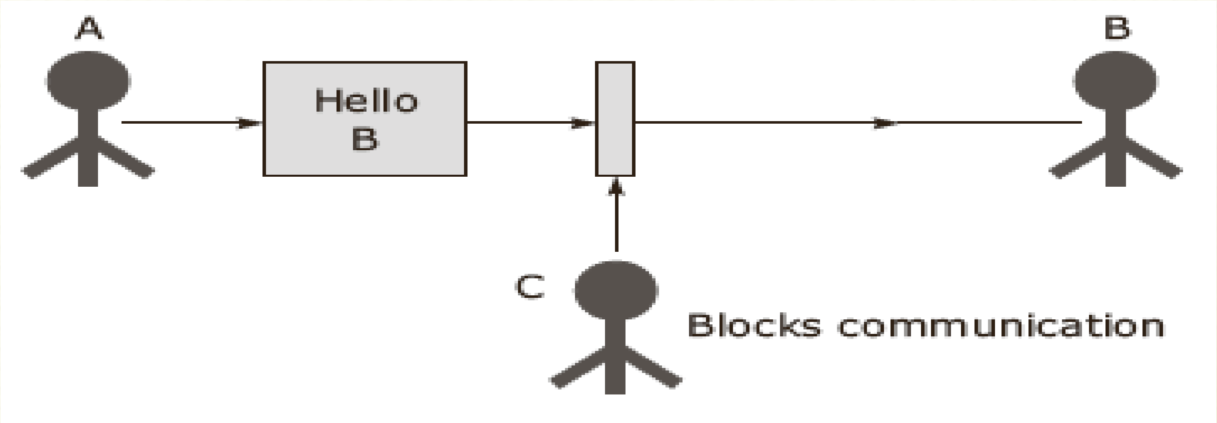
- **Authenticity attack :**an intruder can illegally connect to computer network , impersonate and insert bogus message with valid address in the system . These will then be delivered as genuine message



Authenticity attack

Active attack

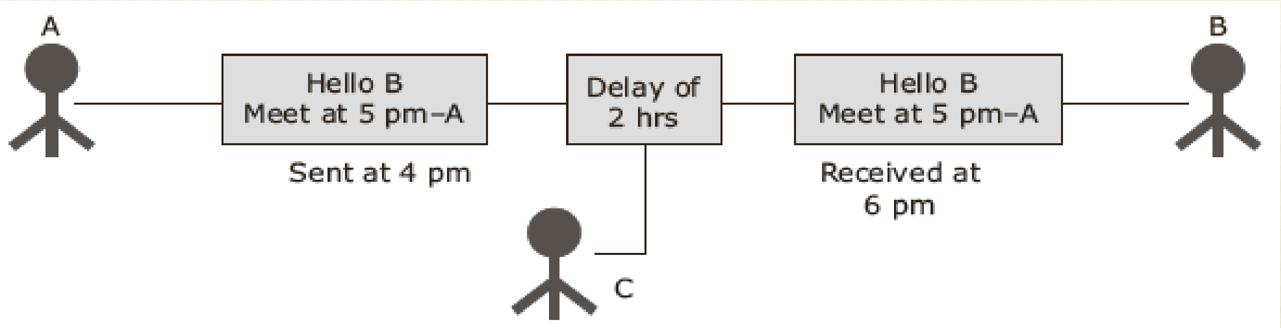
- **Denial attack:** an intruder might partly or completely block communication path between two processes.



Denial attack

Active attack

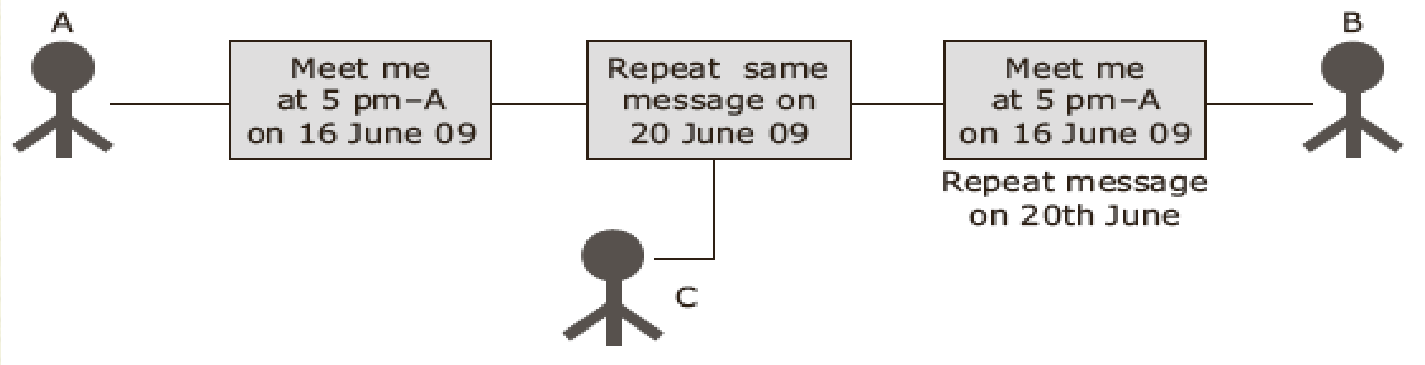
- **Delay attack:** an intruder can delay the message delivery that can make it useless to receive if it is received late.



Delay attack

Active attack

- **Replay attack:** an intruder retransmit an old message that is accepted as new message by the receiver.


A
Meet me at 5 pm-A on 16 June 09
Repeat same message on 20 June 09
Meet me at 5 pm-A on 16 June 09
B
Repeat message on 20th June
C

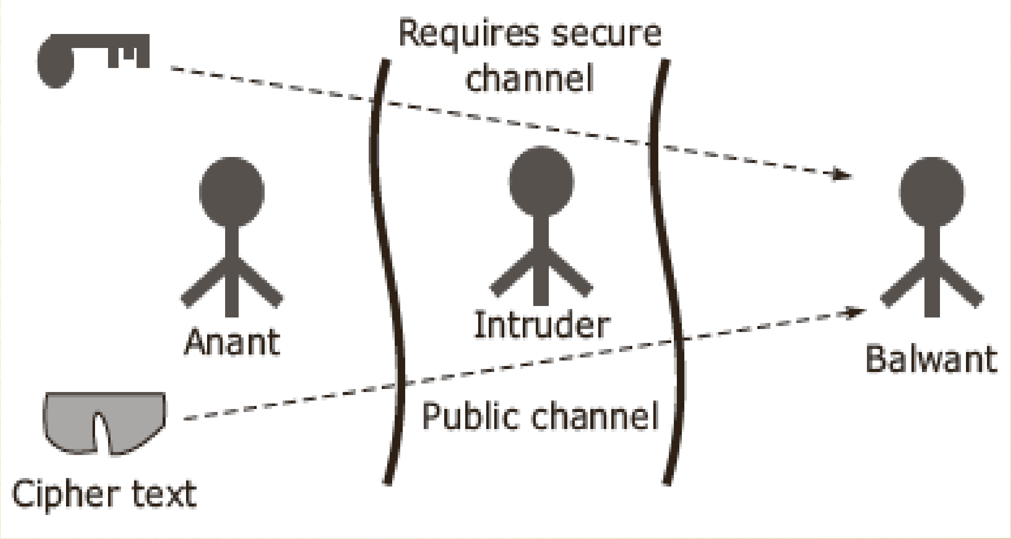
Replay attack

Confinement problems

- Prevention of such leakage of information is called confinement problem.

- The following types of channels can be used by a program to leak information:

1. Legitimate channel
2. Storage channel
3. Covert channel



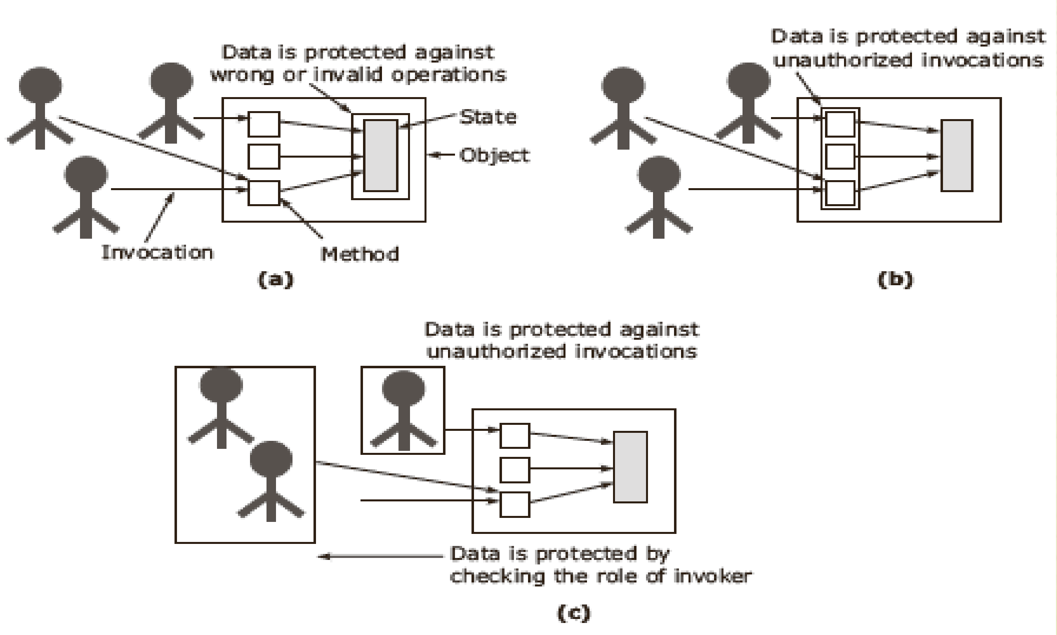
Channels

Design issues

- The major design issues in building secure distributed system are:
- Focus of control.
- Layering of security mechanism.

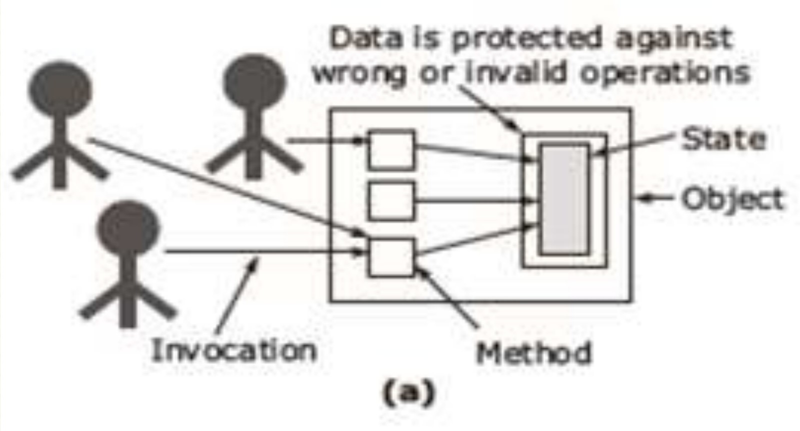
Focus of control.

- The are three approaches that can be followed to protect a distributed application:-
- Protection against invalid operations on secure data
- Protection against unauthorized invocations
- Protection against unauthorized users


Data is protected against
wrong or invalid operations
State
Object
Invocation
Method
(a)
Data is protected against
unauthorized invocations
(b)
Data is protected against
unauthorized invocations
Data is protected by
checking the role of invoker
(c)

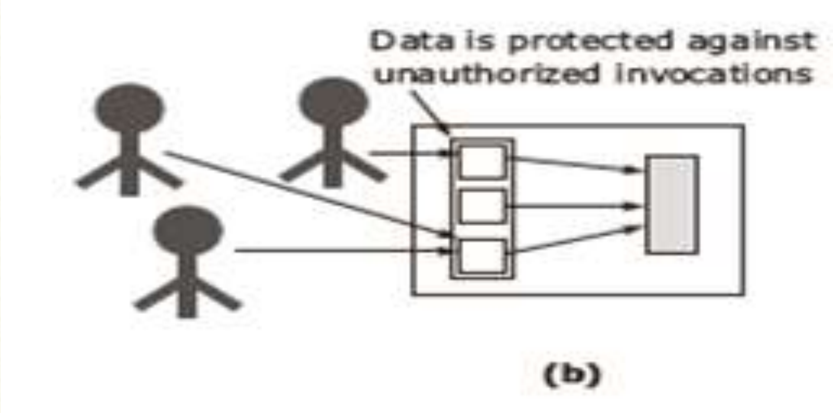
Protection against invalid operations on secure data

Protecting the data that is associated with the application, i.e. insure data integrity .



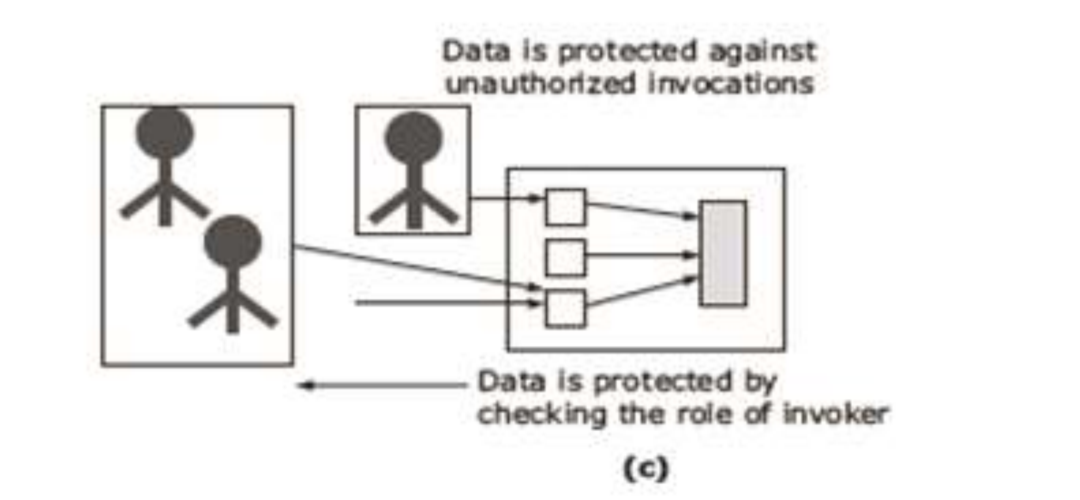
Protection against unauthorized invocations

Specifying which operations can be invoked and by whom and when the data resources are accessed.



Protection against unauthorized users

- Specifying which users should be allowed to access the application irrespective of the operation to be performed.
- Roles are defined for the users and once that role is verified , access to the resource is either granted or denied.


Data is protected against unauthorized invocations
Data is protected by checking the role of invoker
(c)

Layering of security mechanism

- One of important aspect of designing secure system is to decide which level the security mechanism should be placed.
- Security mechanism is normally placed in middleware in a distributed system.

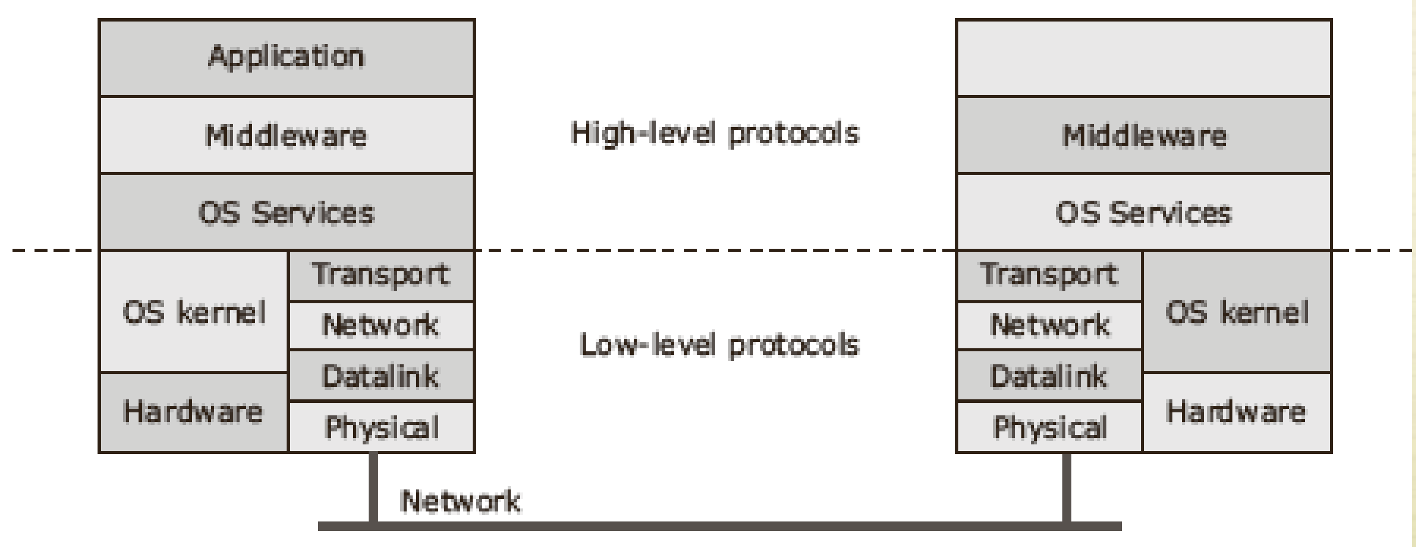
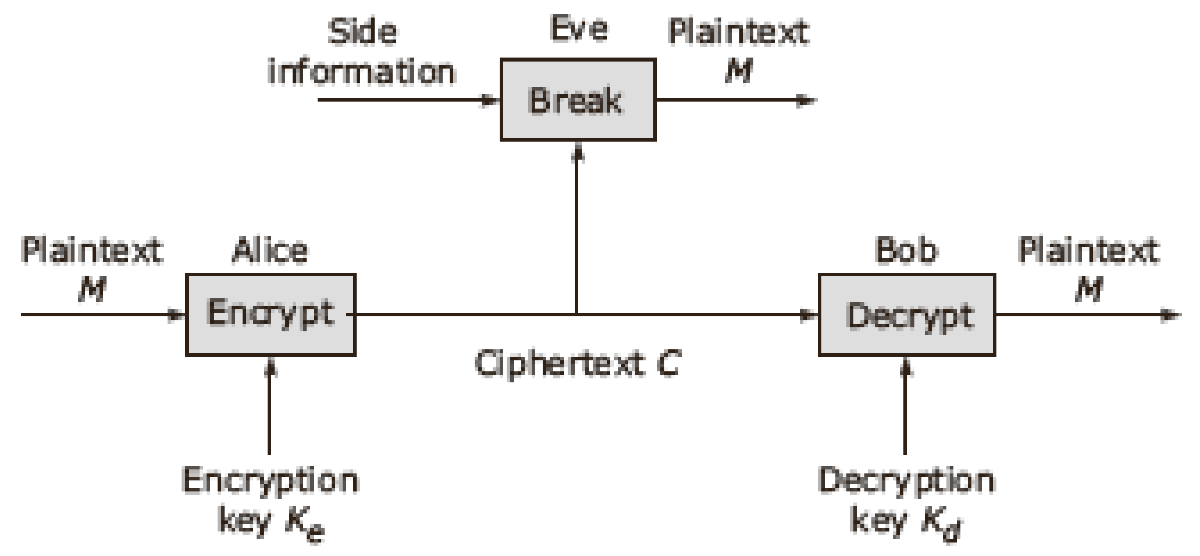


Figure 10-5 Layered organization of a distributed system

Cryptography

- **Cryptography:** is defined as a means of protecting private information against unauthorized access in case where physical security is difficult to achieve.
- Two basic operation : encryption and decryption.



cryptography

- The encryption algorithm has following form:

$$C = E(P, K_e)$$

Where P =plaintext to be encrypted

K_e =encryption key

C =resulting ciphertext

- The decryption algorithm has following form:

$$P = D(C, K_d)$$

Where C = ciphertext to be decrypted

K_d = decryption key

P =resulting plaintext

cryptography

- There are two board classes of cryptosystem based on weather the encryption and decryption keys are the same namely symmetric and asymmetric systems.

- Symmetric cryptosystem: uses the same key for both encryption decryption.

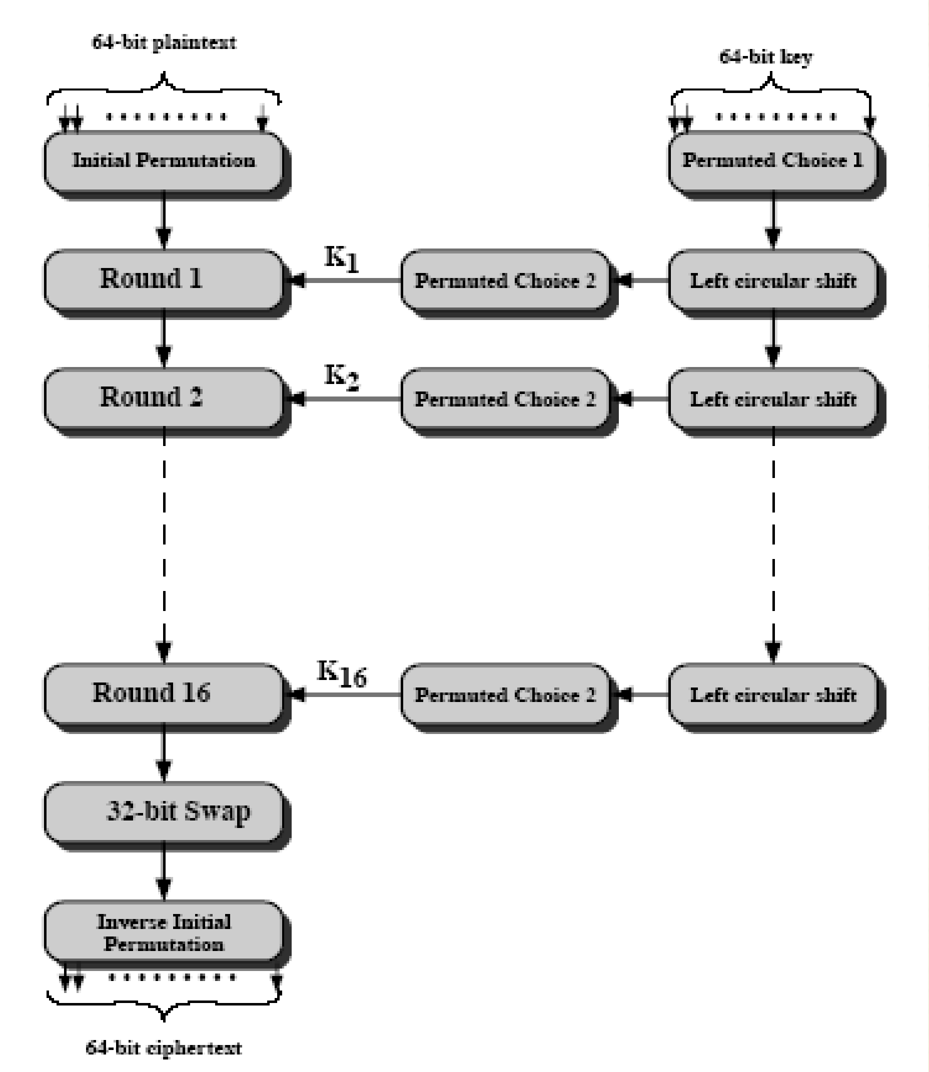
- Asymmetric cryptosystem: the key for both encryption decryption are different but they form unique.

Symmetric Cryptosystem Algorithm: DES

- Symmetric (secret key)

- *E(K, M)*= $\{M\}_K$ *D(K, E(K,M))=M*

- Message *M*, key *K*, published encryption functions *E, D*
- Same key for *E* and *D*
- M must be hard (infeasible) to compute if K is not known.
- Usual form of attack is brute-force: try all possible key values for a known pair M, $\{M\}_K$. Resisted by making K sufficiently large~ 128 bits


64-bit plaintext
64-bit key
Initial Permutation
Permuted Choice 1
Round 1
K1
Permuted Choice 2
Left circular shift
Round 2
K2
Permuted Choice 2
Left circular shift
Round 16
K16
Permuted Choice 2
Left circular shift
32-bit Swap
Inverse Initial Permutation
64-bit ciphertext

DES Flow Chart

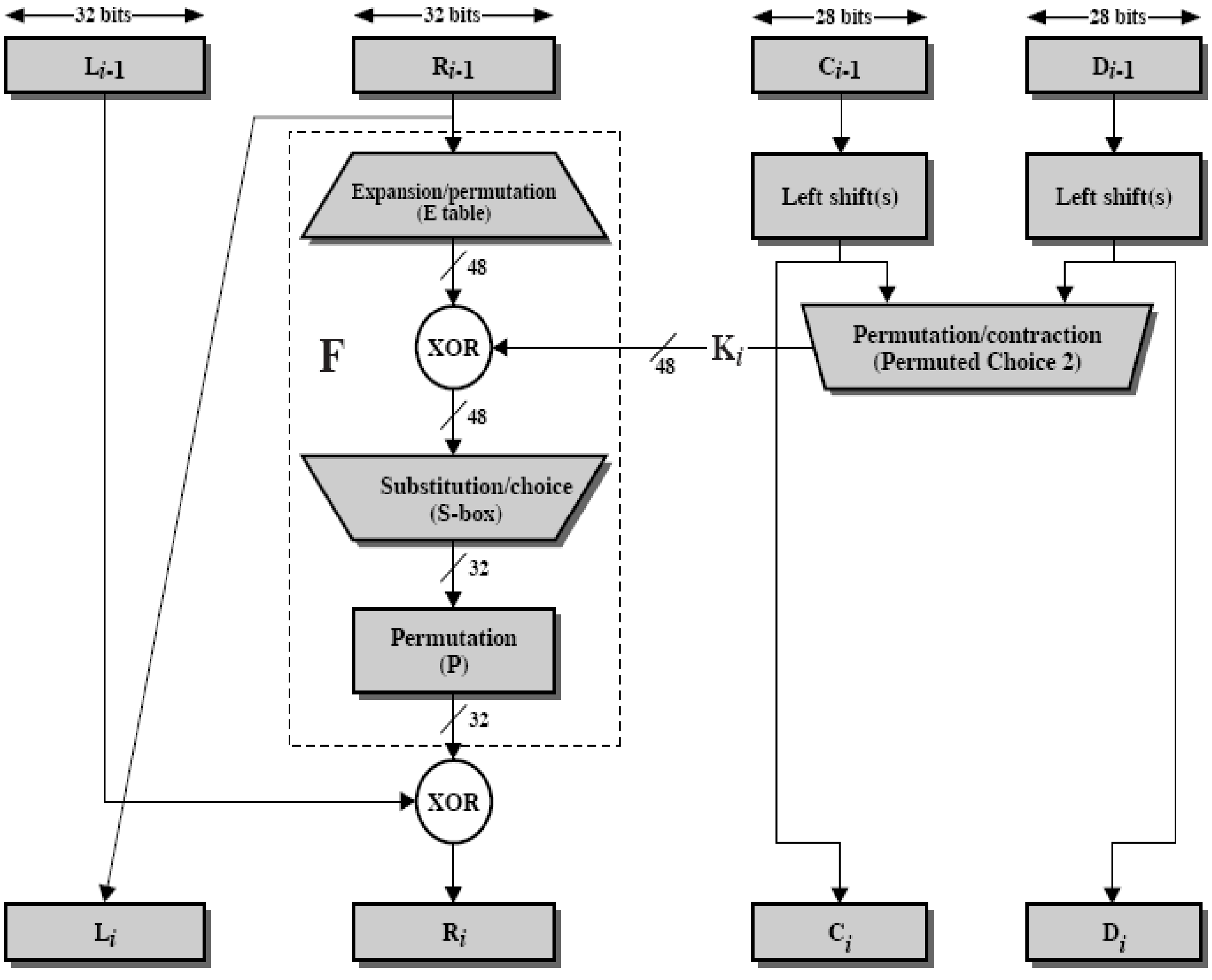


Figure 3.8 Single Round of DES Algorithm

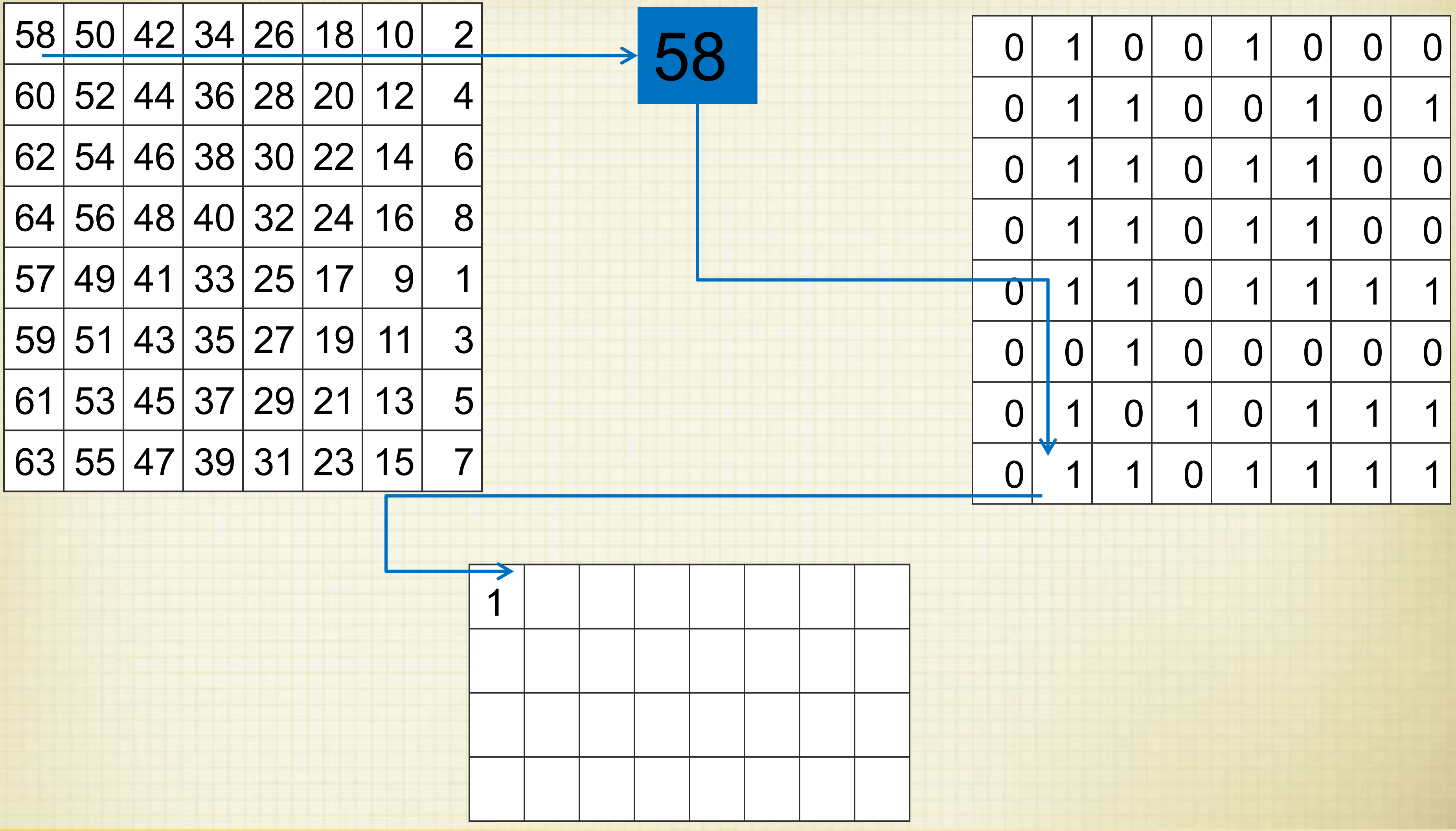
Data Encryption Standard Algorithm

- **Initial permutation:**

each bit of a block is subject to initial permutation, permutation table shows, when reading the table from left to right then from top to bottom, that the 58th bit of the 64-bit block is in first position, the 50th in second position and so forth.

Once the initial permutation is completed, the 64-bit block is divided into two 32-bit blocks, respectively denoted **L** and **R** (for left and right). The initial status of these two blocks is denoted $\mathbf{L}_0$ and $\mathbf{R}_0$:

Data Encryption Standard Algorithm



58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

58

0	1	0	0	1	0	0	0
0	1	1	0	0	1	0	1
0	1	1	0	1	1	0	0
0	1	1	0	1	1	0	0
0	1	1	0	1	1	1	1
0	0	1	0	0	0	0	0
0	1	0	1	0	1	1	1
0	1	1	0	1	1	1	1

1							

Data Encryption Standard Algorithm

- **Expansion function**

The 32 bits of the $\mathbf{R}_0$ block are expanded to 48 bits thanks to a table called an *expansion table* (denoted **E**), in which the 48 bits are mixed together and 16 of them are

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

expansion table

1	0	1	0	1	0	1	0
1	0	0	0	1	1	1	0
0	1	0	1	1	0	0	1
1	0	0	1	0	0	1	0

0	1				

Data Encryption Standard Algorithm

- **Exclusive OR with the key:**
- The resulting 48-bit table is called **R'**$_0$ or **E**[**R**$_0$]. The DES algorithm then *exclusive ORs* the first key **K**$_1$ with **E**[**R**$_0$]. The result of this *exclusive OR* is a 48-bit table we will call **R**$_0$ out of convenience (it is not the starting **R**$_0$!).

0	1	0	0	1	0
0	1	1	0	0	1
0	1	1	0	1	1
0	1	1	0	1	1
0	1	1	0	1	1
0	0	1	0	0	0
0	1	0	1	0	1
0	1	1	0	1	1

⊕

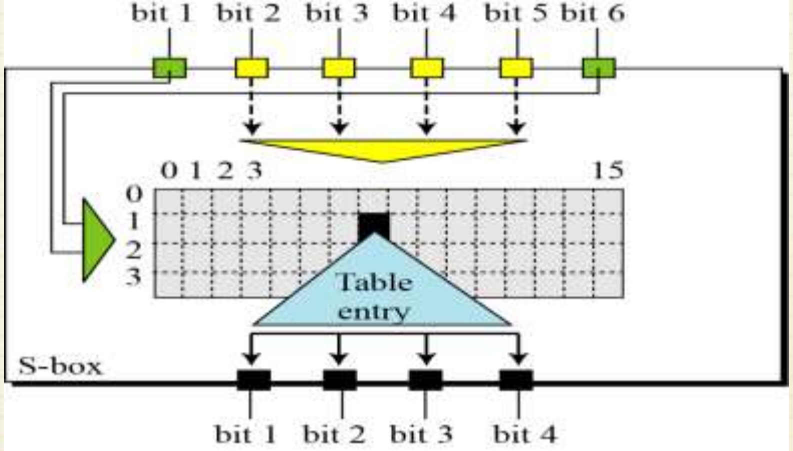
0	0	1	0	0	0
1	0	0	1	0	1
1	0	1	1	0	0
1	0	1	1	0	0
1	0	1	1	1	1
1	0	0	0	0	0
0	1	0	1	1	1
1	0	1	1	1	1

=

0	1	1	0	1	0
1	1	1	1	0	0

Data Encryption Standard Algorithm

- **Substitution function**
- $\mathbf{R}_0$ is then divided into 8 6-bit blocks, denoted $\mathbf{R}_{0i}$. Each of these blocks is processed by **selection functions** (sometimes called *substitution boxes*), generally denoted $\mathbf{S}_i$.
  The first and last bits of each $\mathbf{R}_{0i}$ determine (in binary value) the line of the selection function; the other bits (respectively 2, 3, 4 and 5) determine the column.



Data Encryption Standard Algorithm

- **Permutation:**

The obtained 32-bit block from the earlier step (**Substitution function)** is then subject to a permutation P here is the table:

P							
16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

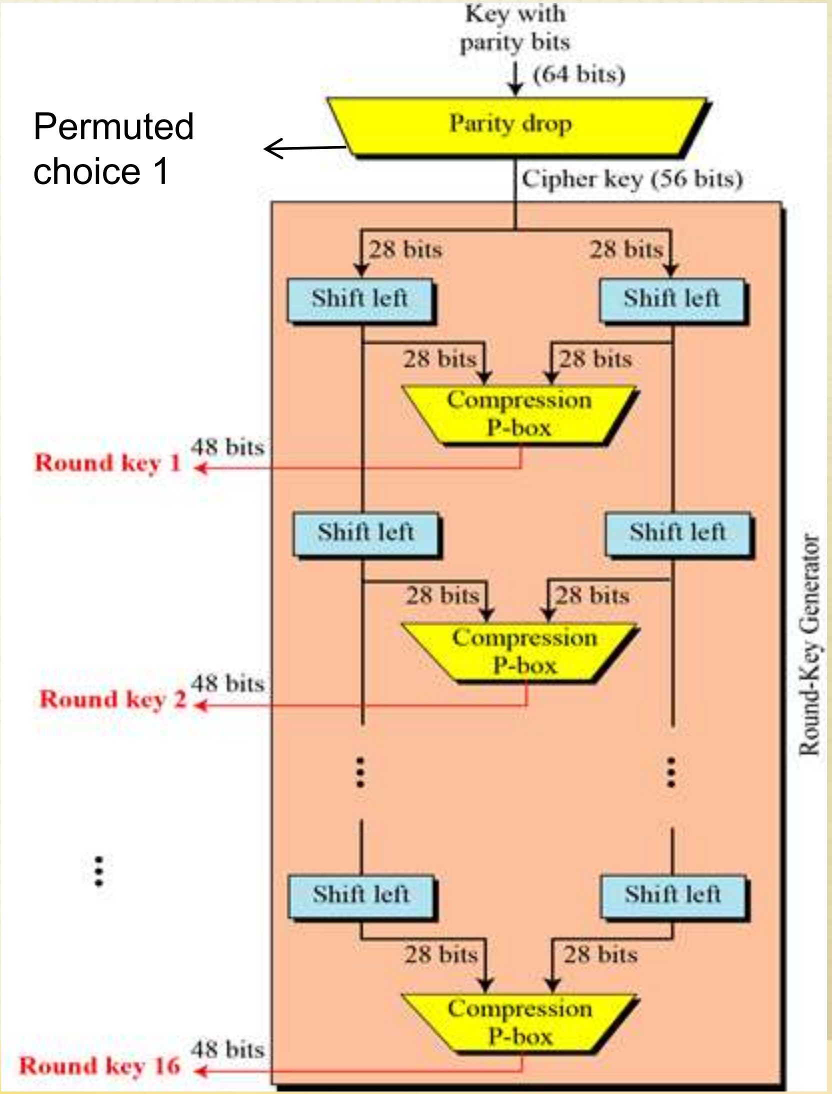
- **Exclusive OR**

All of these results output from **P** are subject to an *Exclusive OR* with the starting$\mathbf{L}_0$ (as shown on the first diagram) to give R1, whereas the initial $\mathbf{R}_0$ gives $\mathbf{L}_1$**.**

Data Encryption Standard Algorithm

- **Key generation** :

is the process of generating keys for cryptography. A key is used to encrypt and decrypt whatever data is being encrypted/decrypted.



Des Decryptions

- ❖ The decryption process with DES is essentially the same as the encryption process and is as follows:
- Use the cipher text as the input to the DES algorithm but use the keys K_i In reverse order. That is, use K_{16} on the first iteration, K_{15} on the second until K_1 which is used on the 16th and last iteration.

Needham-Schroeder protocol

- Can refer to one of the two communication protocol intended for use over insecure network, there are two types of it:

- Needham-Schroeder symmetric key protocol.
- Needham-Schroeder public key protocol.

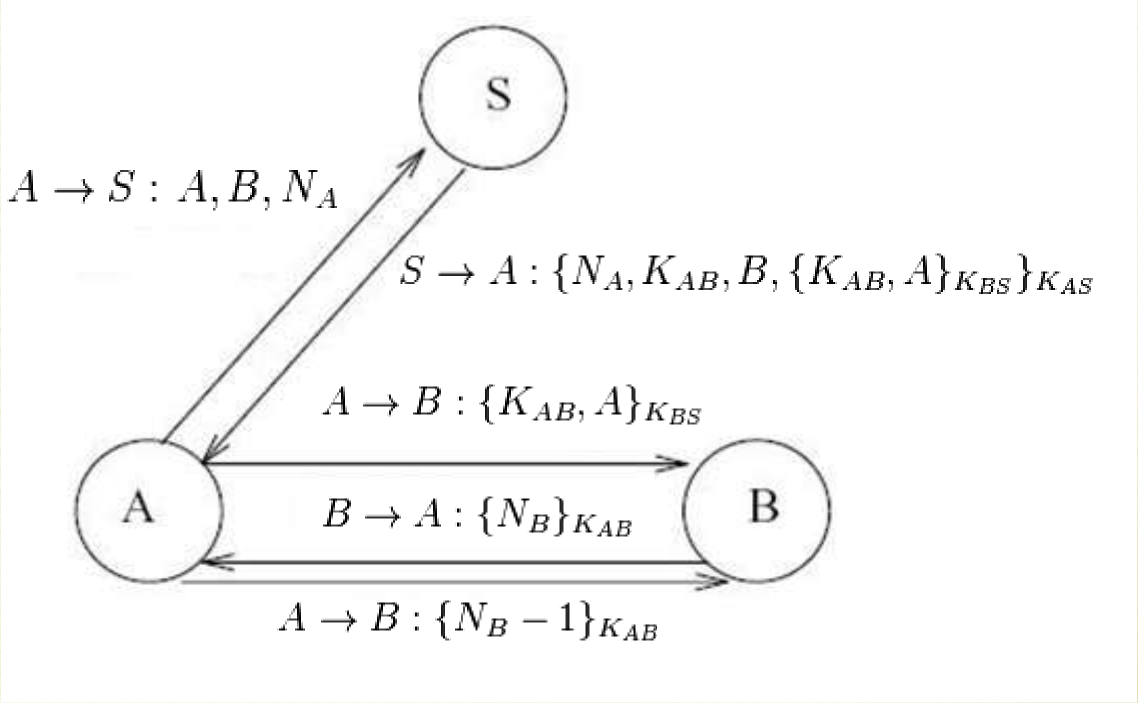
Needham-Schroeder symmetric key protocol:

- It is based on symmetric encryption algorithm.
- This protocol aims to establish session key between two parties on a network .
- Alice (A) initiates the communication to Bob (B). S is a server trusted by both parties. In the communication:

- ➢ A and B are identities of Alice and Bob respectively
- ➢ K_{AS} is a symmetric key known only to A and S
- ➢ K_{BS} is a symmetric key known only to B and S
- ➢ N_A and N_B are nonces generated by A and B respectively
- ➢ K_{AB} is a symmetric, generated key, which will be the session key of the session between A and B

Needham-Schroeder symmetric key protocol

- The protocol can be specified as follow in security notation:


S
$A \rightarrow S : A, B, N_A$
$S \rightarrow A : \{N_A, K_{AB}, B, \{K_{AB}, A\}_{K_{BS}}\}_{K_{AS}}$
$A \rightarrow B : \{K_{AB}, A\}_{K_{BS}}$
A
$B \rightarrow A : \{N_B\}_{K_{AB}}$
B
$A \rightarrow B : \{N_B - 1\}_{K_{AB}}$

Needham-Schroeder public key protocol.

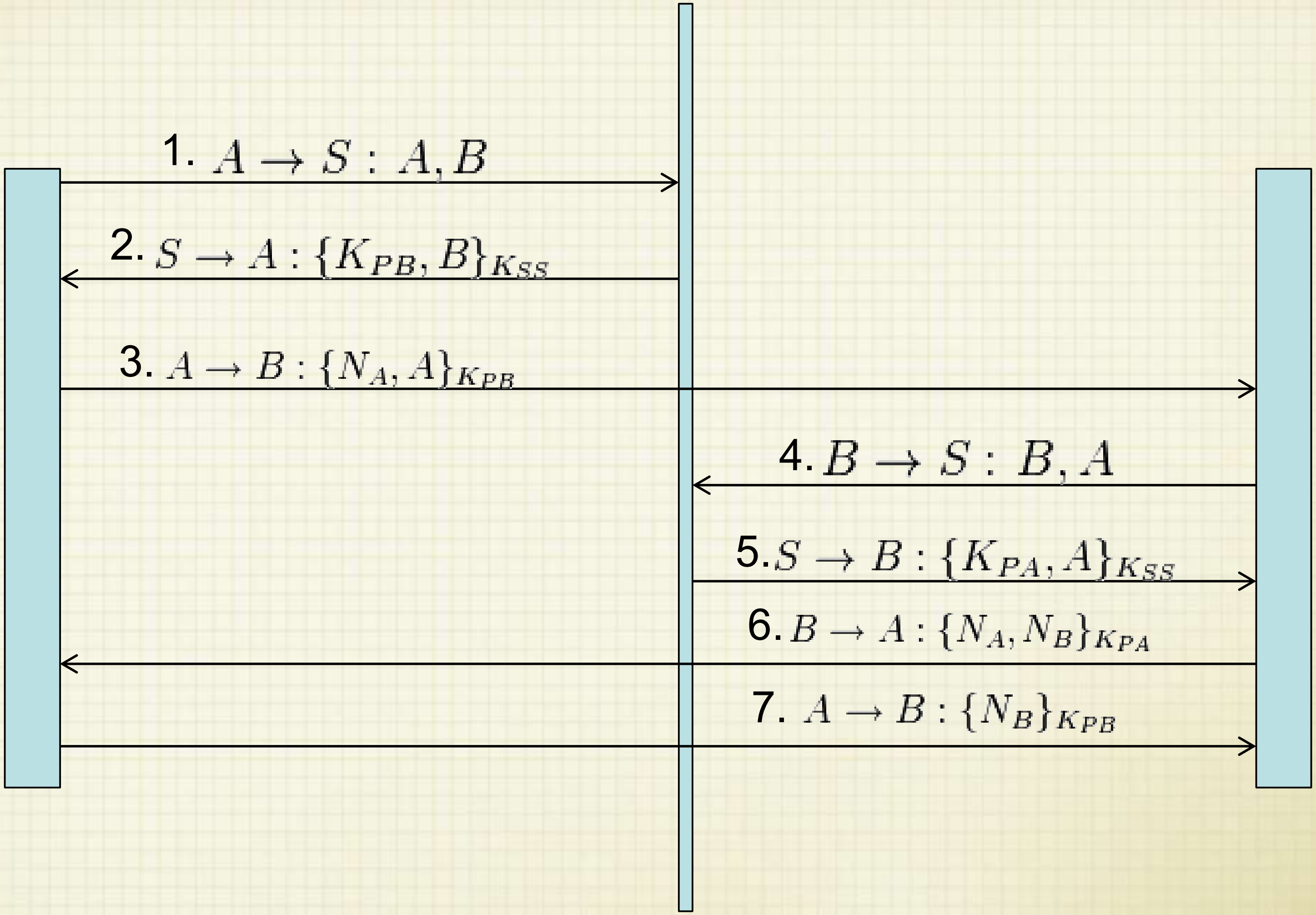
- It based on public-key cryptography. This protocol is intended to provide mutual authentication between two parties communicating on a network.
- Here, Alice (A) and Bob (B) use a trusted server (S) to distribute public keys on request. These keys are:

- K_{PA} and K_{SA}, respectively public and private halves of an encryption key-pair belonging to A (S stands for "secret key" here)
- K_{PB} and K_{SB}, similar belonging to B
- K_{PS} and K_{SS}, similar belonging to S. (Note this has the property that K_{SS} is used to *encrypt* and K_{PS} to *decrypt*).

Needham-Schroeder public key protocol.

- The protocol runs as follows:
- A requests B's public keys from S
- S responds with public key K_{PB} alongside B's identity, signed by the server for authentication purposes.
- B requests A's public keys.
- Server responds.
- A chooses a random N_A and sends it to B.
- B chooses a random N_B, and sends it to A along with N_A to prove ability to decrypt with K_{SB}.
- A confirms N_B to B, to prove ability to decrypt with K_{SA}

Needham-Schroeder public key protocol



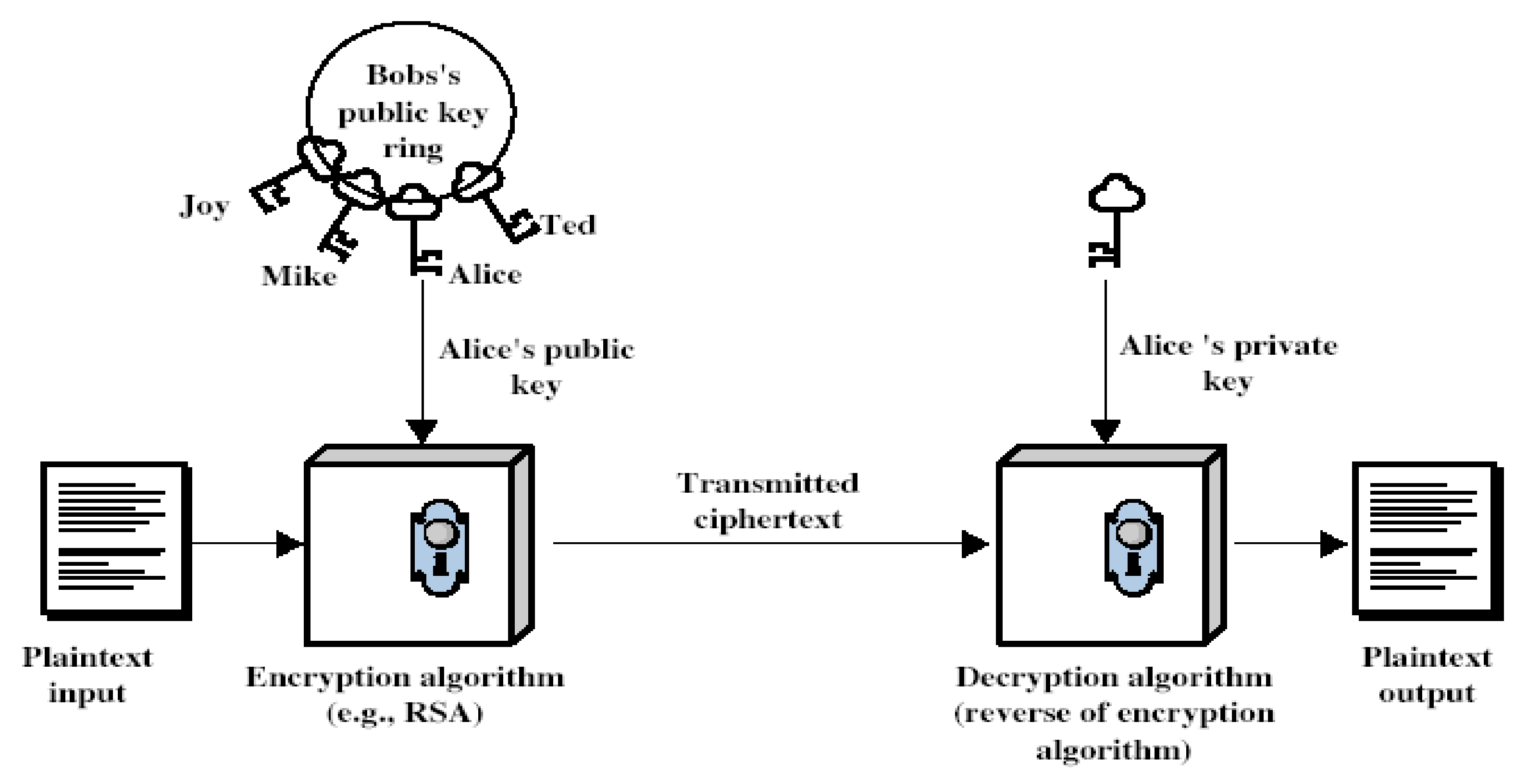
1. $A \rightarrow S : A, B$
2. $S \rightarrow A : \{K_{PB}, B\}_{K_{SS}}$
3. $A \rightarrow B : \{N_A, A\}_{K_{PB}}$
4. $B \rightarrow S : B, A$
5. $S \rightarrow B : \{K_{PA}, A\}_{K_{SS}}$
6. $B \rightarrow A : \{N_A, N_B\}_{K_{PA}}$
7. $A \rightarrow B : \{N_B\}_{K_{PB}}$

Asymmetric Cryptosystem

- There are two separate key for encryption and decryption , one of them is kept private and the other one public.

RSA protocol

- Is an algorithm for a public-key systems



RSA Algorithm

- each user generates a public/private key pair by:
- selecting two large primes at random - p, q
- computing their system modulus $N = p.q$
  - $note\ ø(N) = (p-1)(q-1)$

- selecting at random the encryption key e
  - where $1 < e < ø(N), \gcd\big(e, ø(N)\big) = 1$

- solve following equation to find decryption key d
  - $e.d = 1\ mod\ ø(N) and\ 0 \leq d \leq N$

- publish their public encryption key: KU={e,N}
- keep secret private decryption key: KR={d,p,q}

Hash function MD5

- The MD5 message-digest algorithm is a widely used cryptographic hash function producing a 128-bit (16-byte) hash value.
- MD5 has been utilized in a wide variety of cryptographic applications, and is also commonly used to verify data integrity.

Hash function MD5 algorithm

- The input message is broken up into chunks of 512-bit blocks (sixteen 32-bit words).
- the message is padded so that its length is divisible by 512.
- The padding works as follows: first a single bit, 1, is appended to the end of the message. This is followed by as many zeros as are required to bring the length of the message up to 64 bits fewer than a multiple of 512.
- The remaining bits are filled up with a 64-bit integer representing the length of the original message, in bits.
- The main MD5 algorithm operates on a 128-bit state, divided into four 32-bit words, denoted *A*, *B*, *C*, and *D*. These are initialized to certain fixed constants.

MD5 pseudo-code

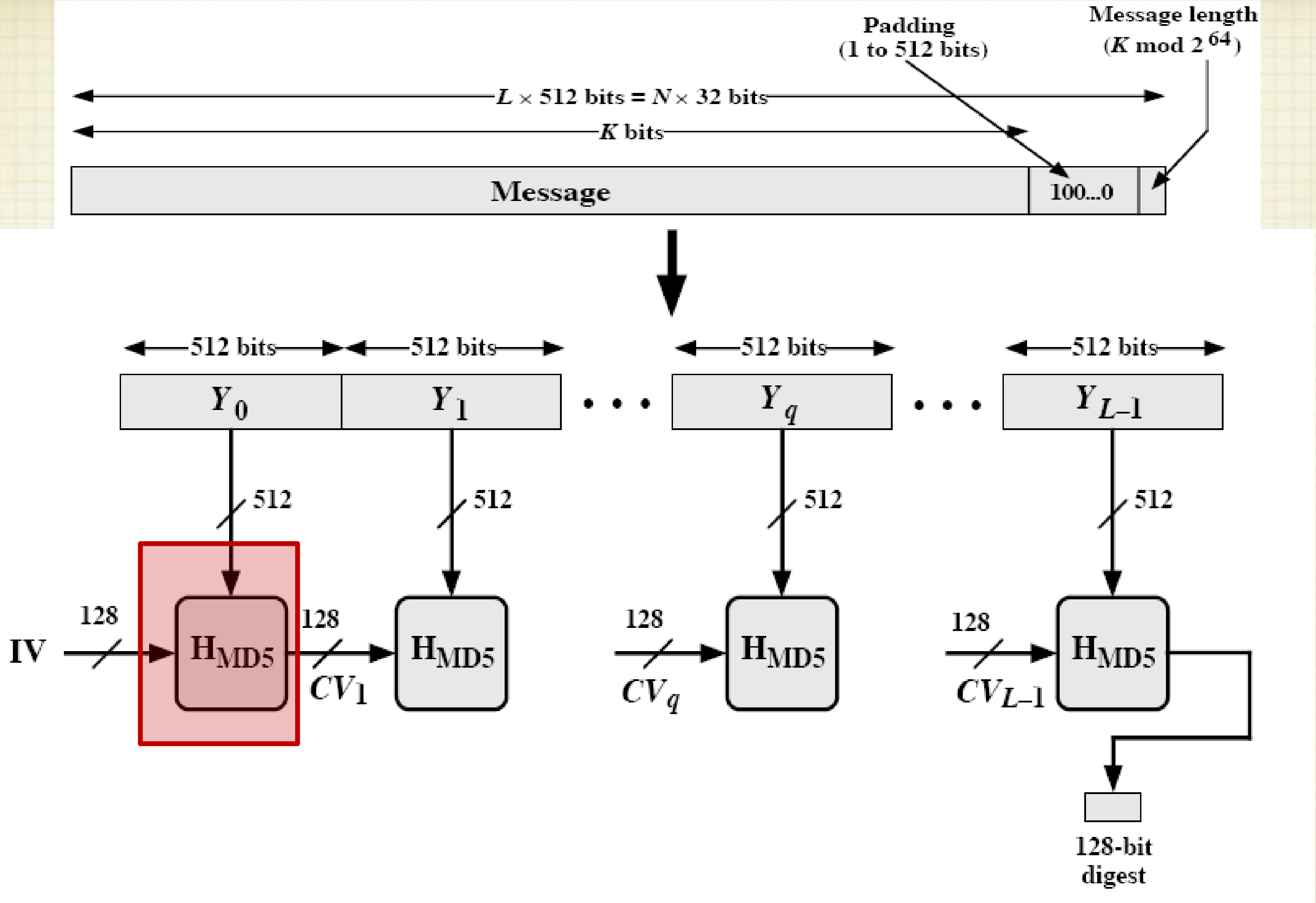
- The main MD5 algorithm operates on a 128-bit state, divided into four 32-bit words, denoted *A*, *B*, *C*, and *D*. These are initialized to certain fixed constants.

A = 0x67452301 B = 0xEFCDAB89
C = 0x98BADCFE D = 0x10325476.

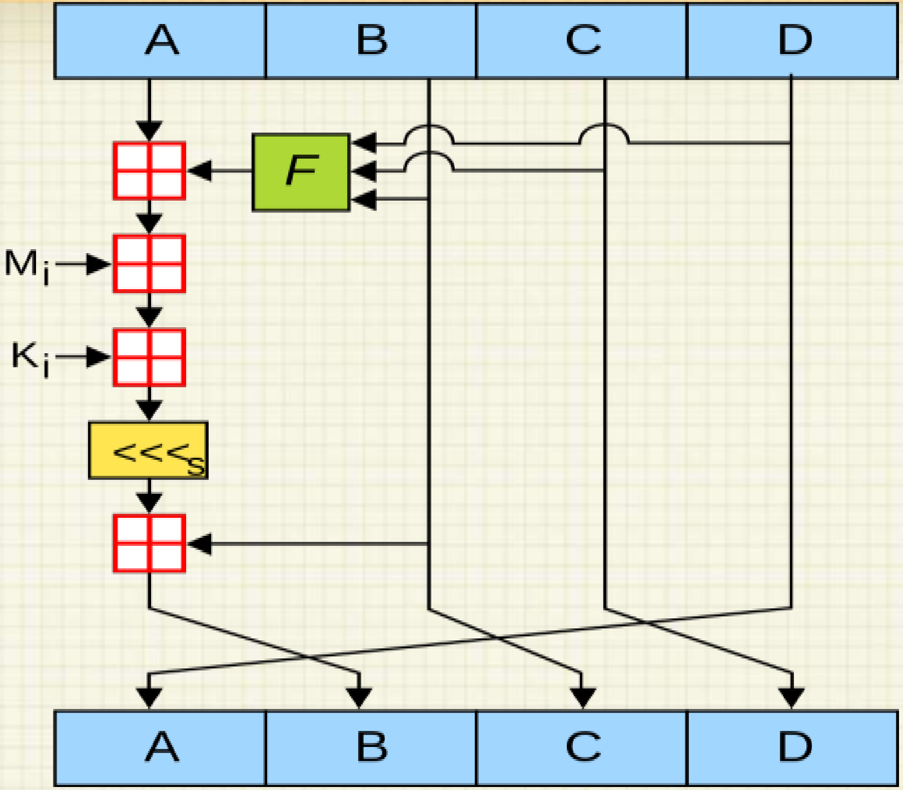
- There are four possible functions *F*; a different one is used in each round:

F(X,Y,Z) = (X & Y) | (~(X) & Z)
G(X,Y,Z) = (X & Z) | (Y & ~(Z))
H(X,Y,Z) = X ^ Y ^ Z
I(X,Y,Z) = Y ^ (X | ~(Z))

Where &, |, ^, and ~ are the bit-wise AND, OR, XOR, and NOT operators


Padding
(1 to 512 bits)
Message length
(K mod 2^64)
L × 512 bits = N × 32 bits
K bits
Message
100...0
512 bits
512 bits
512 bits
512 bits
Y0
Y1
Yq
YL–1
512
512
512
512
IV
128
HMD5
128
CV1
HMD5
128
CVq
HMD5
128
CVL–1
HMD5
128-bit
digest

One MD5 operation



MD5 consists of 64 of these operations, grouped in four rounds of 16 operations. F is a nonlinear function; one function is used in each round. M_i denotes a 32-bit block of the message input, and K_i denotes a 32-bit constant, different for each operation. s denotes a left bit rotation by s places; s varies for each operation. denotes addition modulo 2^{32}.

Secure channels

- A secure channel protects senders and receivers from against :

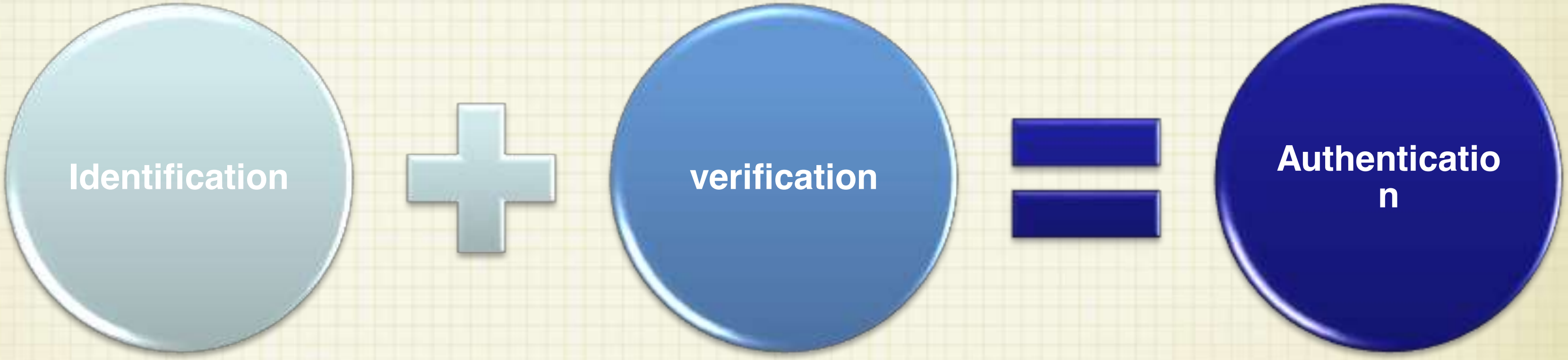
1. fabrication
2. Modification.
3. Interception.

- To have a secure communication we should have:

1. an authentication of communicating parties.
2. Ensure data integrity and confidentiality.

Authentication

- Deals with verifying the identity of users before allowing them to access a resource.
- This mechanism prevent unauthorized user from accessing the system resource.



Authentication

- **Identification:** is the process of claiming a certain identity by a user
- **Verification:** is the process of verifying the user’s claimed identity.
- Authentication in distributed system can be categorized into following :

1. User login authentication
2. One-way authentication of communicating entities
3. Two-way authentication of communicating entities

✓

User login authentication

- Deals with verifying the identity of the user by the system while logging in.
- Correct user identification during login becomes essential since all access control decisions and accounting functions depend on this identity.
- For ensuring security ,a password-based authentication system must have the following mechanisms:

1. Maintain secrecy of passwords
2. Make passwords difficult to guess
3. Limit damage due to a compromised password

User login authentication Cont.

1-Maintain secrecy of passwords :

- User must keep the password secret from external world.
- Hiding the character displayed when the user logs in to the terminal.
- Password table is protected and accessible only to the authentication program and the entries in password table is encrypted.

User login authentication Cont.

2-Make passwords difficult to guess:

To keep passwords secret in distributed environment:

A.Use long passwords

B.Include special character in password

C.Avoid passwords used earlier

3-Limit damage due to a compromised password:

➢The users must change the password periodically and this change should not make password guessing easy.

One way authenticated of communicated entities

- An entity *A* wants to communicate with entity *B,B* may want to verify the identity of *A* before initiating communication.
- This authentication protocol directly uses cryptosystem-based design principle which categorized into:

1. Protocols based on symmetric cryptosystem
2. Protocols based on asymmetric cryptosystem

One way authenticated of communicated entities

1- Protocols based on symmetric cryptosystem:

- Knowledge of shared key allows encryption and decryption of messages.
- It also called challenge-response protocol.
- **Example:** user A wants to communicate with user B:

- A encrypts its ID using K obtains the cipher-text , and sends this message to user B.
- User B decrypts using K and compares the results with the ID of the message .
- Incase they match user A is accepted ; else rejected.

One way authenticated of communicated entities

2- Protocols based on asymmetric cryptosystem:

- The public key of each user is published while the secret key of each user is known only to the user.
- **Example:** user A wants to communicate with user B:
  - A sends its ID in plaintext form to B.
  - B receive message and sends a random number to A in plaintext form.
  - User A receives this message , encrypts the random key using its secret key obtains cipher-text and send it to B
  - User B receives the message ,decrypts the cipher-text using public key of user A.
  - Compares the result with the original random number if they are equal then user A is accepted; else rejected.

Two-way authentication of communicating entities

- Two-way protocol ensure that both the sender and the receiver processes identify each other before establishing a secure logical communication channel between them.
- Mutual authentication can be also carried out by performing one-way authenticating twice.

Two-way authentication of communicating entities

- If two communicating user A and B want authenticate each other ,A can first Authenticate B by perform one-way Authenticate and then B Authenticate A by repeating same process.
- K_{A+} = public key of A K_{A-} = Private key of A
- K_{B+} = public key of B K_{B-} = Private key of B
- K_{S+} = public key of AS K_{S+} = Private key of AS
- A = ID of *A* B = ID of *B*
- Q_{A} = Code for request by A Q_{B} = Code for request by B
- *K*= A session key generated by the authentication server for user *A* and *B.*

Two-way authentication of communicating entities

- The authentication protocol consists of the following steps:
- User A sends a request message m_1 to AS indicating that it wants to establish a secure logical communication with user B. m_1 contains Q_A ,ID_A and ID_B.
- AS extracts the public key of $\mathrm{B}(K_B)$ form database that corresponds to user id of the message.by using the secret key K_{s-},the AS encrypts to generate $C_1 = E((B, K_{B+}), K_{s-})$ sends C_1 to A in message m_2.
- *A* decrypts C_1 *by using AS(*K_{s+}*)*. It then generate a R_A,

  $C_2 = E((A, R_A)K_{B+})$ and send C_2 to B in message m_3.
- User B decrypts C_2 by using K_{B-}.it then sends m_4 to AS requesting the *A*'s public key and a session key *K* .the message is encrypted with public key of AS(K_{s+}), the message contains

  $C_3 = E((Q_B, A, B, R_A), K_{s+})$ send it to AS.

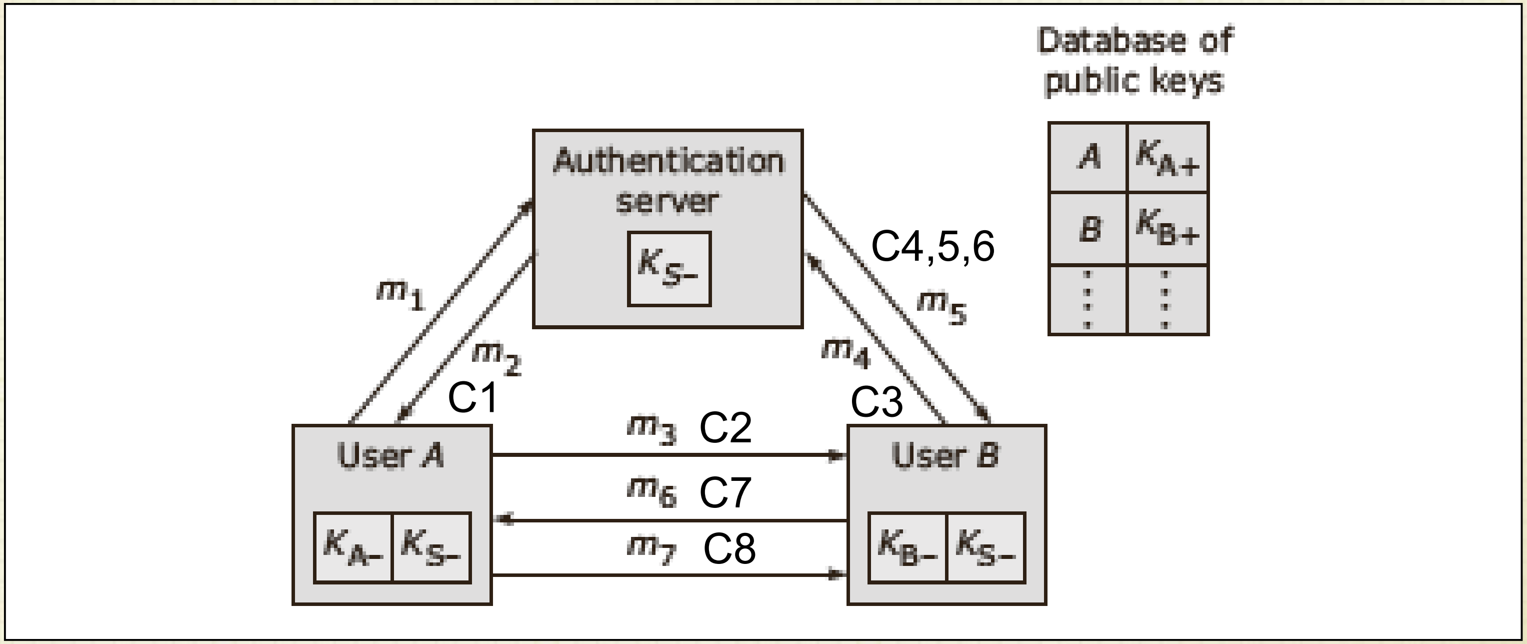
Two-way authentication of communicating entities

- The AS decrypts C_3 with the its secret key K_{s-}. It generates a new session key *K* for *A and B.* Next it generates three cipher-text:

  C_4=*E((*A, K_{A+}), K_{s-}*)* C_5=*E(*(B,K, R_A), K_{s-}*)* C_6=*E(*(C_5, K_{B+}))

- User B decrypts C_4 and C_6 with K_{s+} and K_{B-}. It then generates random number R_B and creates cipher text: C_7=*E((*C_5, R_B)K_{A+}*)* , then *sends* C_7 A in m_6*.*
- User A first decrypts $C_7$by using $K_{A-}$and then decrypts C_5 by using AS(K_{s+}) now both A and B have the session key C_8=*E(*R_B,K) then send it to *B.*
- User B decrypts C_8 by using the K and compares the results with the original value R_B **if they are equal it proved that the logical communication is established.**

Two way authentication based on asymmetric cryptosystem



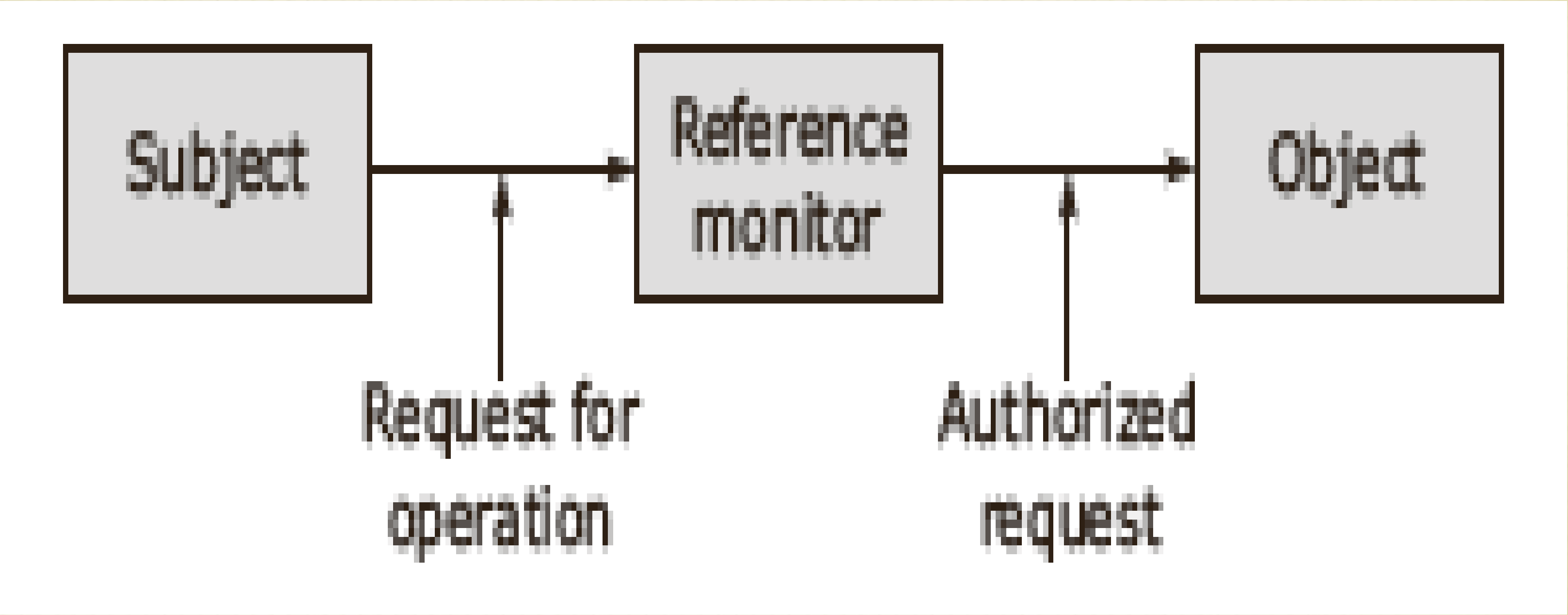
Access Control

General Issues in Access Control.

Firewalls.

Secure Mobile Code.

General Issues in Access Control



General model to control access to object

General Issues in Access Control

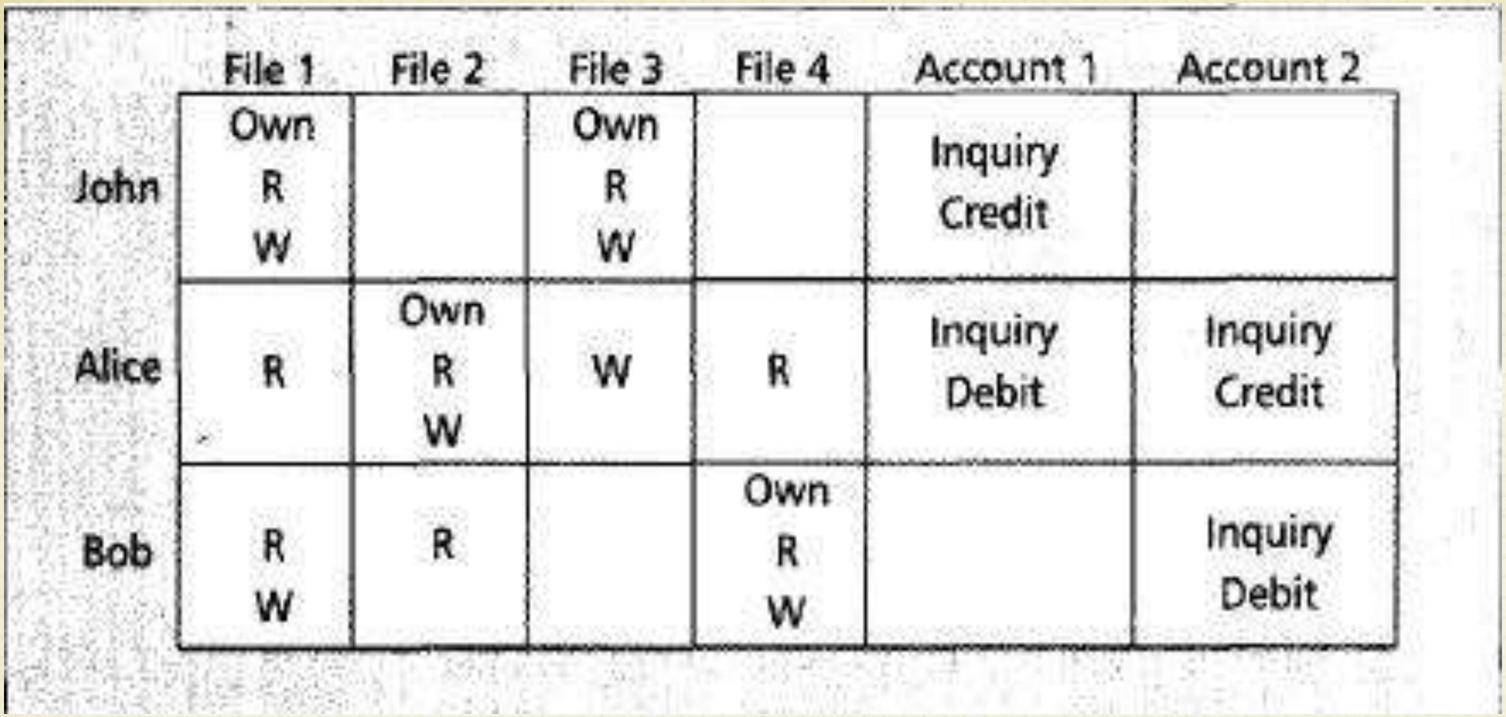
- **Subjects**
  - can best be thought of as being processes acting on behalf of users, but can also be objects that need the services of other objects in order to carry out their work.
- **Objects**
  - can be hardware objects (*e.g.* CPU, memory segment and printers) or software objects (*e.g.* file and program).
- **Reference monitor**
  - records which subject may do what, and decides whether a subject is allowed to have a specific operation carried out.

General Issues in Access Control(Cont.)

Access Control Matrix

- A common approach to modeling the access rights of subjects with respect to objects.
- Each subject is represented by a row in this matrix, each object is represented by a column.
- Disadvantages :
  - Considering that a system may easily need to support thousands of users and millions of objects that require protection Many entries in the matrix will be empty : a single subject will generally have access to relatively few objects.

General Issues in Access Control(Cont.)



	File 1	File 2	File 3	File 4	Account 1	Account 2
John	Own R W		Own R W		Inquiry Credit	
Alice	R	Own R W	W	R	Inquiry Debit	Inquiry Credit
Bob	R W	R		Own R W		Inquiry Debit

Access control

Own, R, and W refers to owner, read and write respectively.

Example:

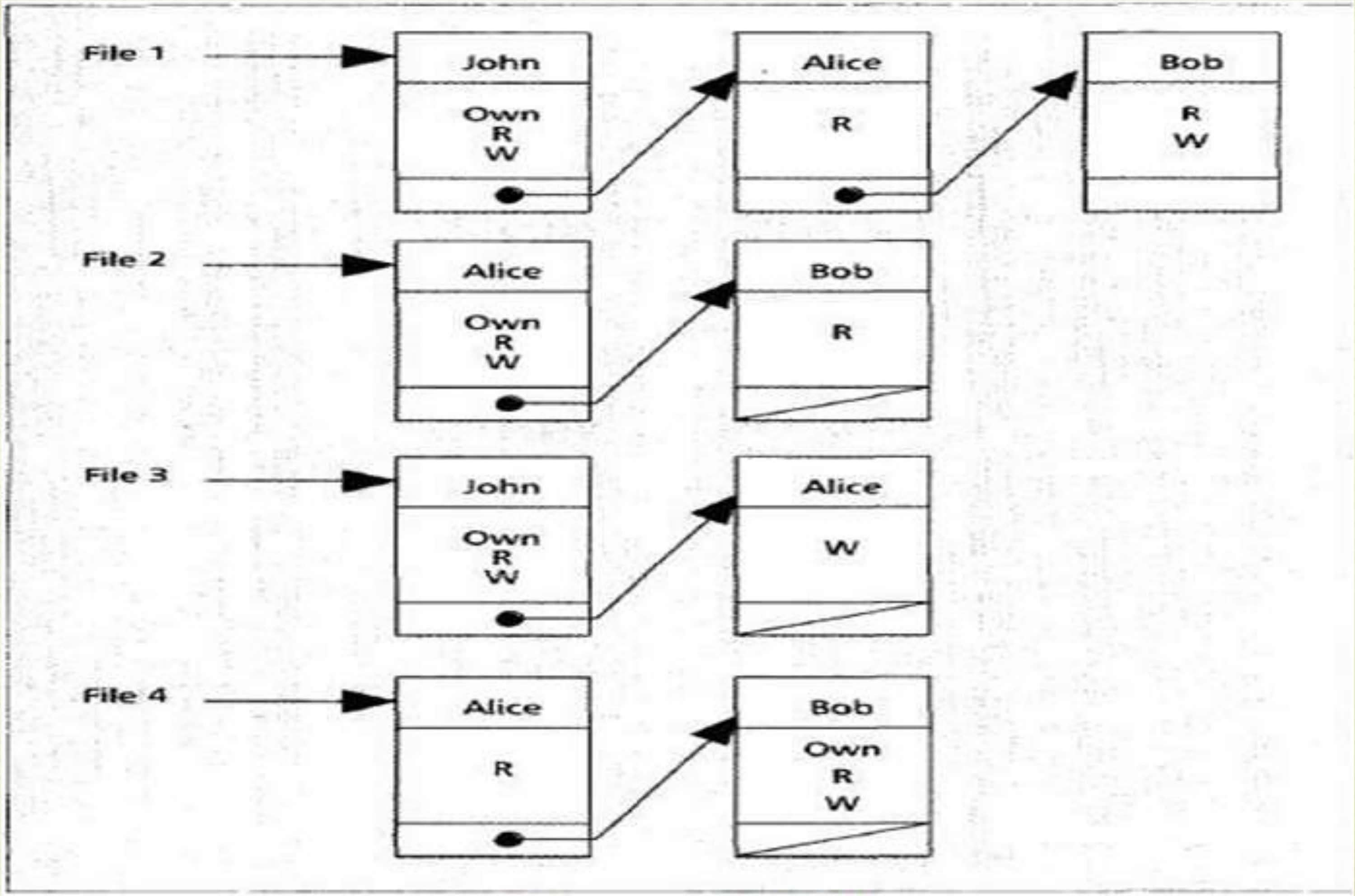
Alice is the owner of the file2, and she can read and write that file.

General Issues in Access Control(Cont.)

Access Control Lists

- Each object is associated with a an ACL.
- ACL has an entry of each subject if it has some kind of access to that object and that empty entries are left out.
- This approach corresponds to storing the access matrix by column (column-wise)
- No empty entries.
- Easy to revoke all access to an object

General Issues in Access Control(Cont.)



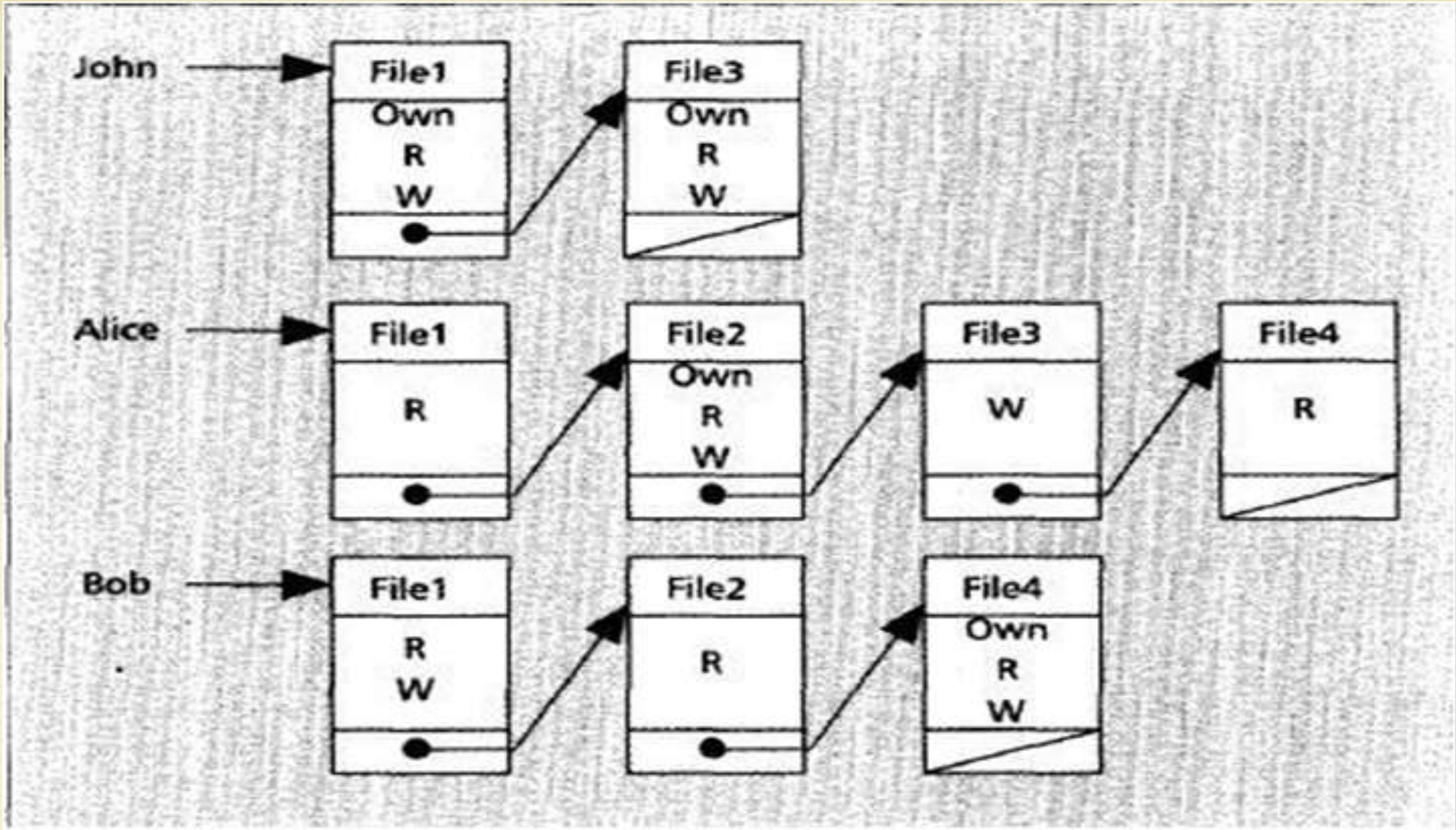
Access Control Lists

General Issues in Access Control(Cont.)

❑ Capabilities

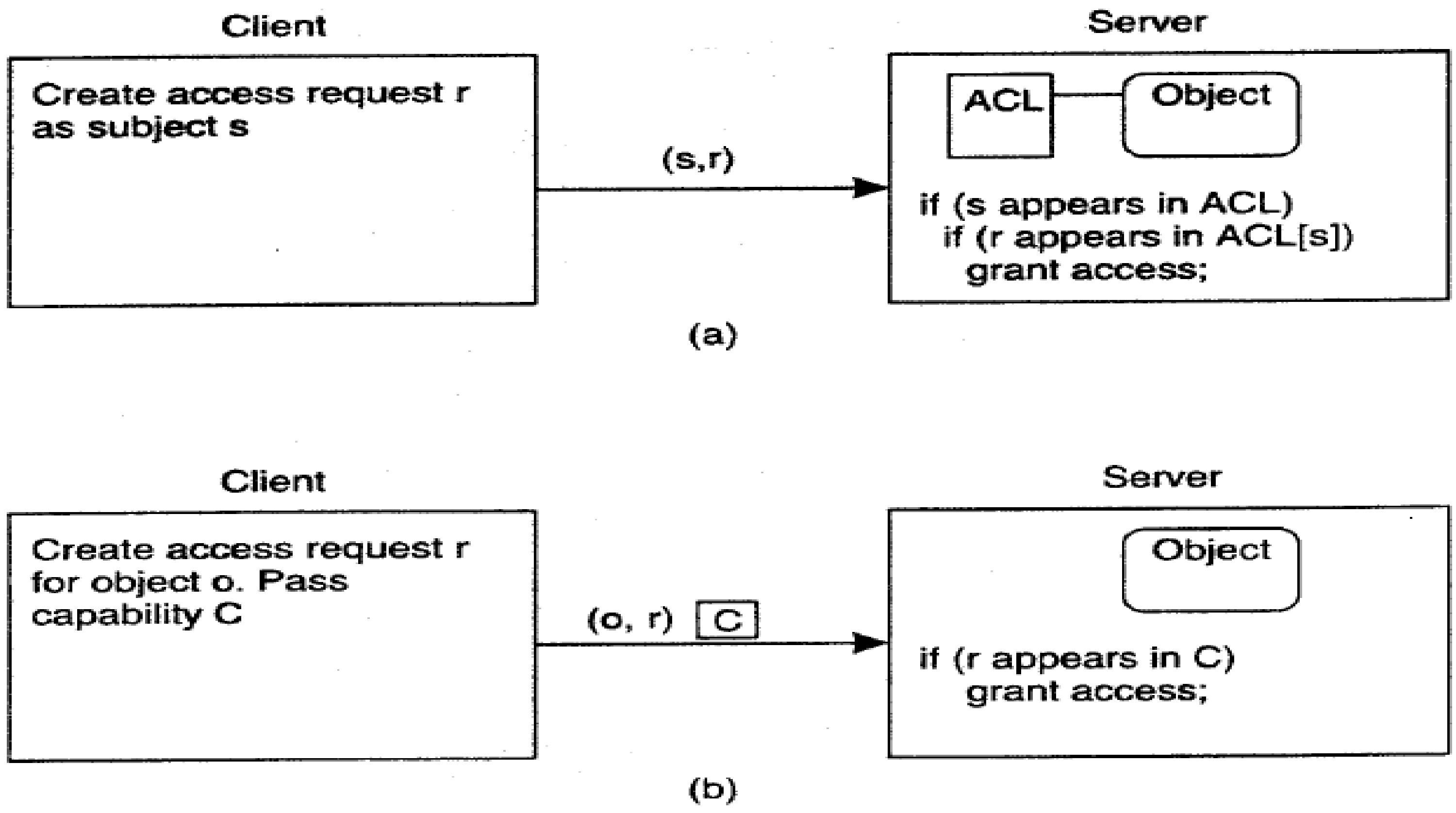
- ➢ Each subject is associated with a list (call the capability list).
- ➢ A capability list of a subject is a list of objects for which subject has some kind of access.
- ➢ This approach corresponds to storing the access matrix by row (row-wise).
- ➢ Easy to find all accesses that a subject is authorized to perform.
- ➢ Easy to revoke all accesses to a subject

General Issues in Access Control(Cont.)



Capabilities lists

General Issues in Access Control(Cont.)

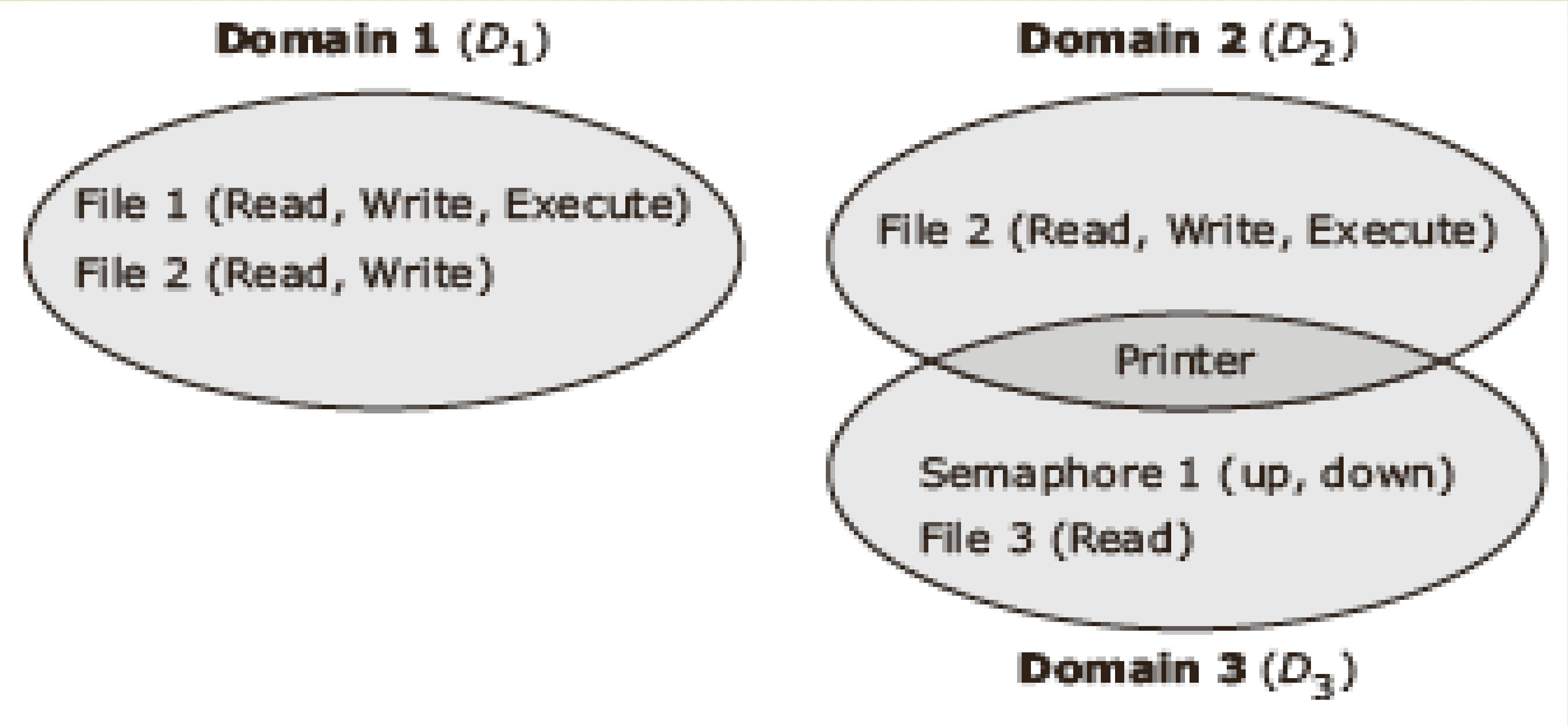


Comparison between ACLs and capabilities for protecting objects.
(a) Using an ACL.. (b) Using capabilities.

General Issues in Access Control(Cont.)

- ❑ Protection domains
  - ➢ In order to reduce the list length, use groups (Protection domains) instead of individual subject identifiers.
  - ➢ Domain is an abstract definition of a set of access rights.
  - ➢ Requests for carrying out an operation are always issued within a domain. Therefore, whenever a subject requests an operation the reference monitor looks up the domain associated with that request.

General Issues in Access Control(Cont.)

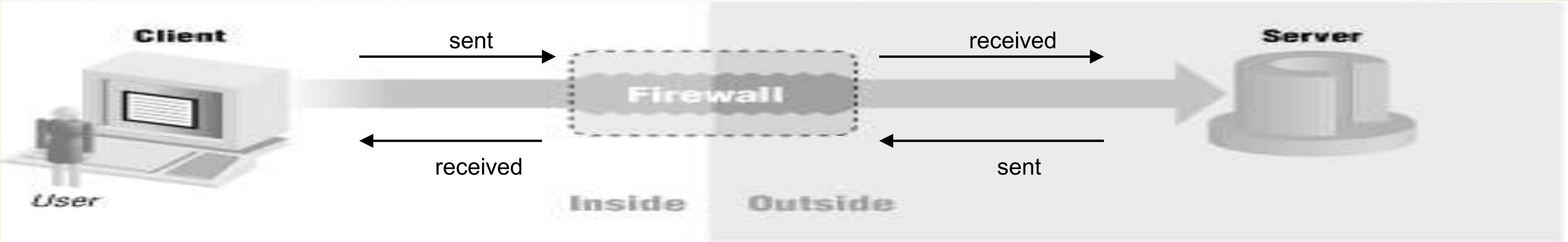


A system with three protection domains

A process executing in *D1* can read and write File2.

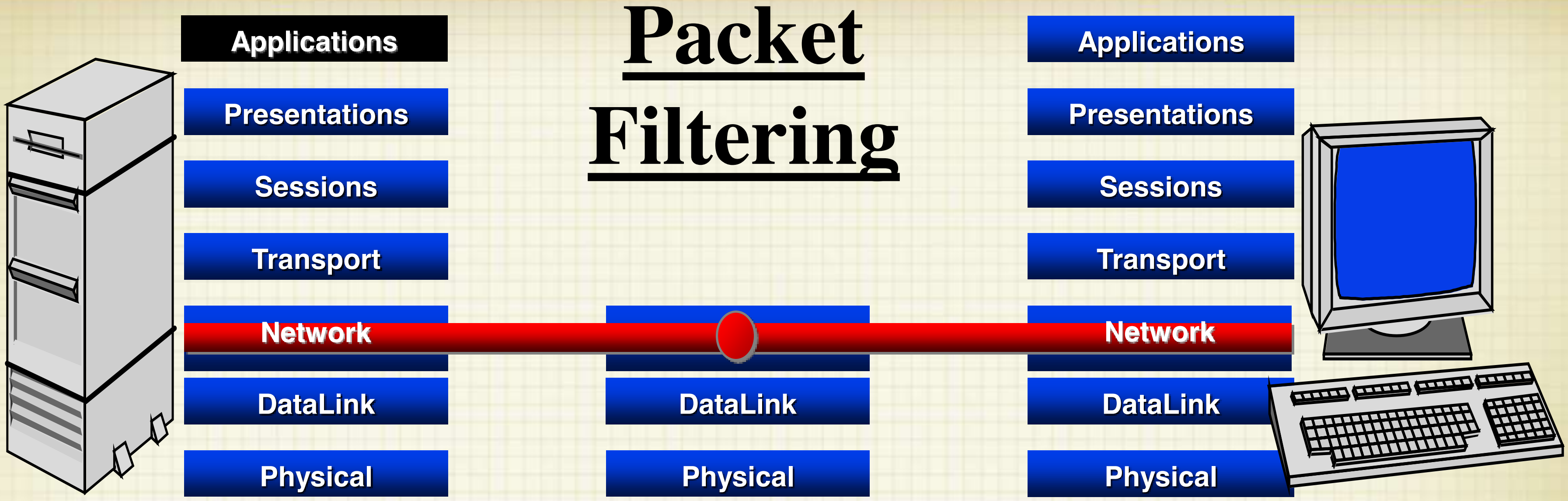
Firewalls

- It is a special kind of reference monitor.
- Disconnects any part of a distributed system from the outside world.
- Provides secure connectivity and prevent unauthorized programs from accessing the system.
- May be a hardware, software, or a combination of both.
- There are a different types of firewalls (*e.g.* packet-filtering and application-level).



✓

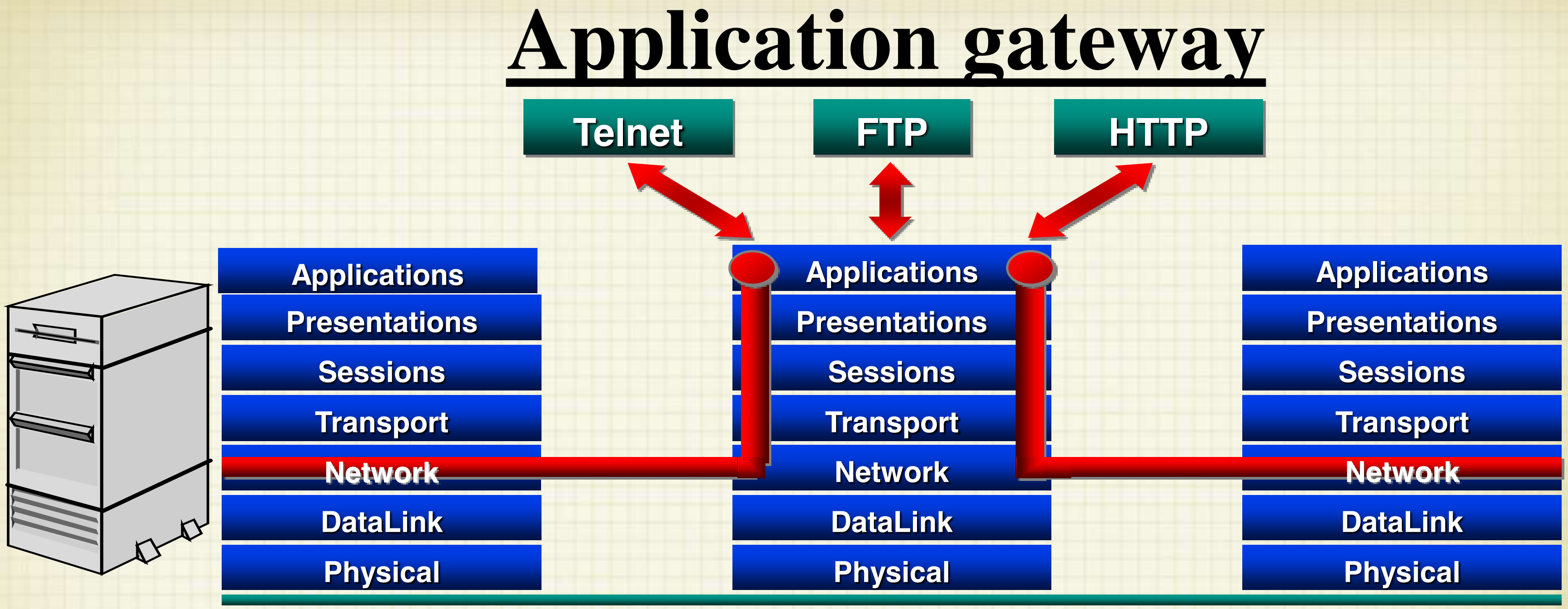
Firewalls (Cont.)



Packet-filtering gateway

- This type of firewall looks at each packet entering or leaving the network and accepts or rejects it based on the source and destination address as contained in the packet's header.
- Effective but difficult to configure.

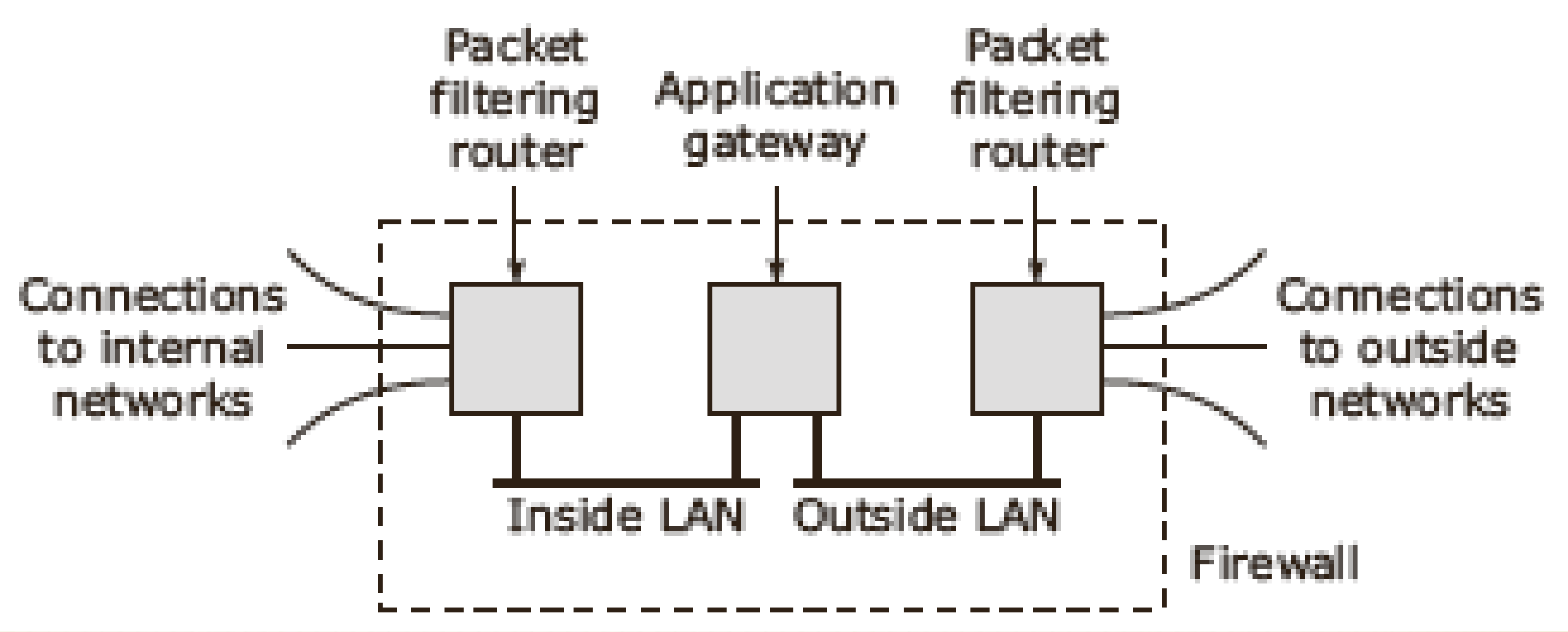
Firewalls (Cont.)



Application-level gateway

- In contrast to a packet-filtering gateway, which inspects only the header of network packets, this type of firewall actually inspects the content of an incoming or outgoing message.
- Very effective, but can impose a performance degradation.

Firewalls (Cont.)

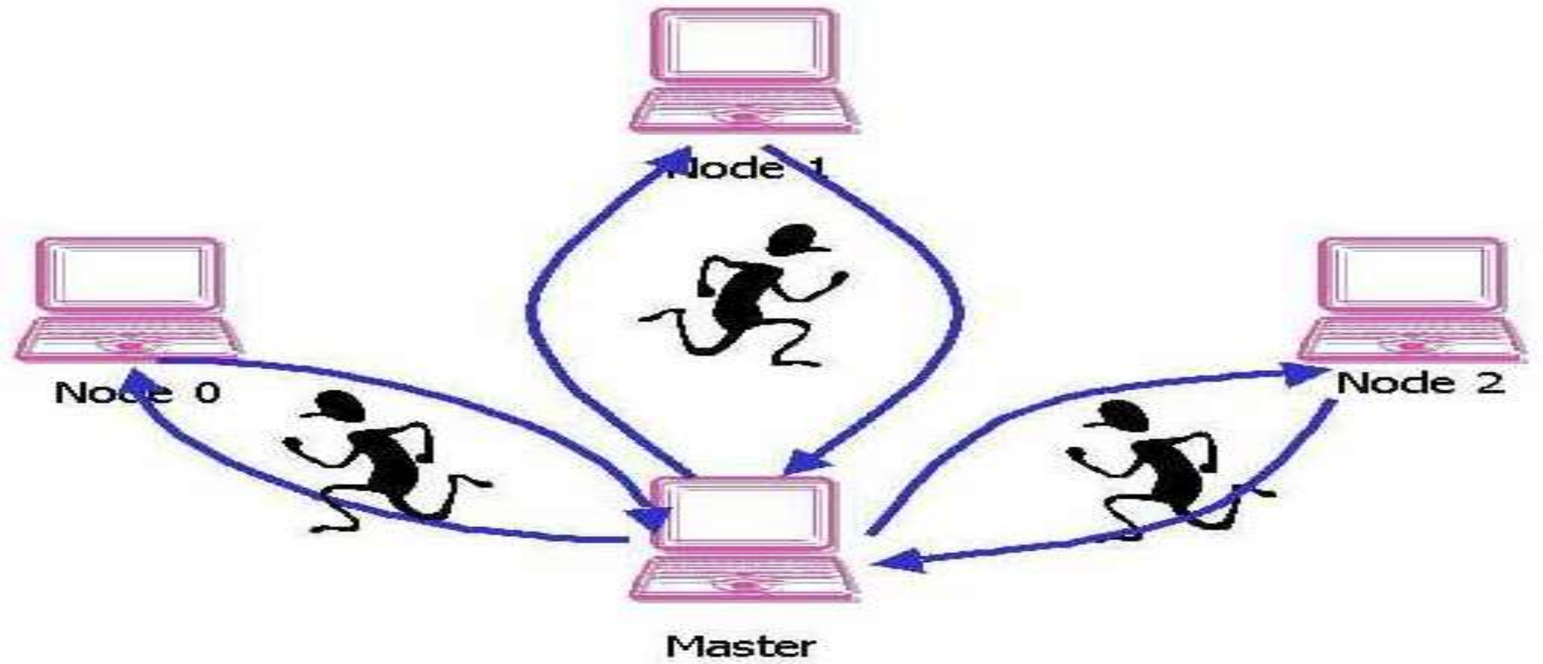


Firewall

A firewall model where Packet-filtering and Application-level gateway are combined

Secure Mobile Code

- Mobile code (agent)
  - Software code that has the ability to travel (*able to transport itself*) from one place(*host*) to another to do the work assigned to it (*e.g. Viruses*).



Secure Mobile Code (Cont.)

Mobile Code (Agent) Applications

- □ Data collection from many place
  - Implement a network backup tool.
- □ Searching and filtering
  - Visit many sites, search through the information available at each site to match a search criterion.
- □ Monitoring
  - *E.g*. in a stock market host, wait for a certain stock to hit a certain price, notify its user or even buy some of the stocks.
- □ Parallel processing
  - Distribute processes easily over many computers in the system.

Secure Mobile Code (Cont.)

Protecting an agent

- Fully protecting an agent against all kinds of attacks is impossible.
- An alternative is to organize agents in such a way that modifications can at least be detected.
- Three mechanisms to detect modifications in an agent
  I. Read-only state.
  II. Append-only logs.
  III. Selective revealing of state.

Secure Mobile Code (Cont.)

- Read-only state

  The owner encrypts a message digest(MD) with its private key and easily check if the MD has been tampered with by verifying the state against the signed MD of the original state.

- Append-only logs

  Have the restrictions that the data can only be appended to the log, therefore is no way that data can be removed or modified without the owner being able to detect this.

- Selective revealing of state

  Providing an array of data items, where each entry for a specific server. Each entry is encrypted with the specific server's public key. then the agent's owner signs the entire array to ensure integrity.

Secure Mobile Code (Cont.)

- Protecting the target (Host)
  - Two approaches to protect the host
    - Sandbox.
    - Playground.

Secure Mobile Code (Cont.)

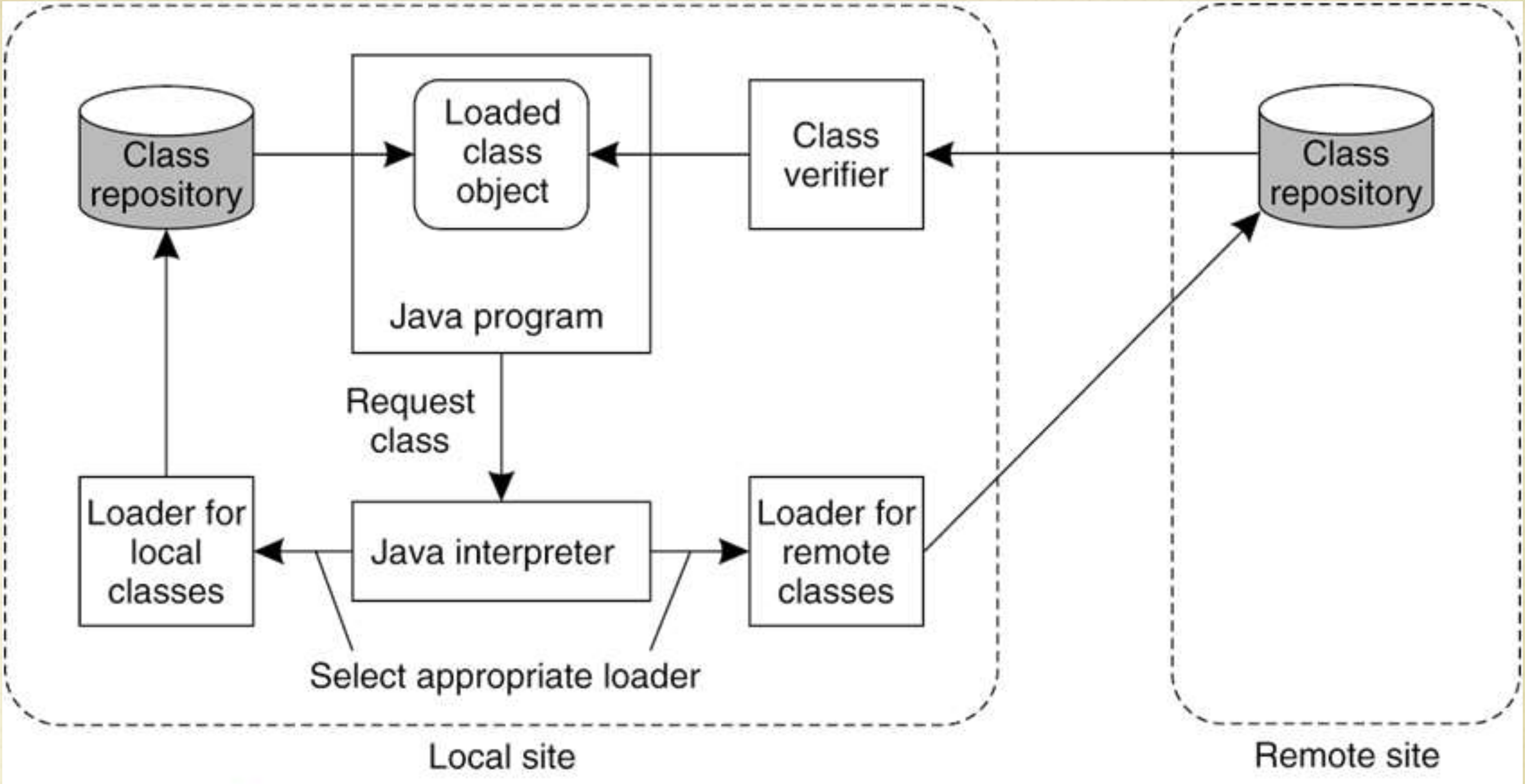
- **Sandbox**
  - It is a technique by which a downloaded program is executed in such a way that each of its instructions can be fully controlled.
  - Any attempt by the foreign code to execute a forbidden instructions halts the execution.
  - IF the foreign code tries to access a forbidden registers or memory areas then also the execution is halted
  - Implementing a sandbox using java program (Java sandbox model)

Secure Mobile Code (Cont.)

Java sandbox model

- It is an implementation of a sandbox using java program.
- It has three main players
  - ✓ Class loader
    - Is responsible for fetching a specified class from a server and installing it in the client's address space so that the JVM can create objects from it.
  - ✓ Byte code verifier
    - Checks that the class contains no illegal instructions or instructions that could somehow corrupt the stack or memory.
  - ✓ Security manager
    - Perform various checks at runtime. The security manager plays the role of a reference monitor.

Secure Mobile Code (Cont.)



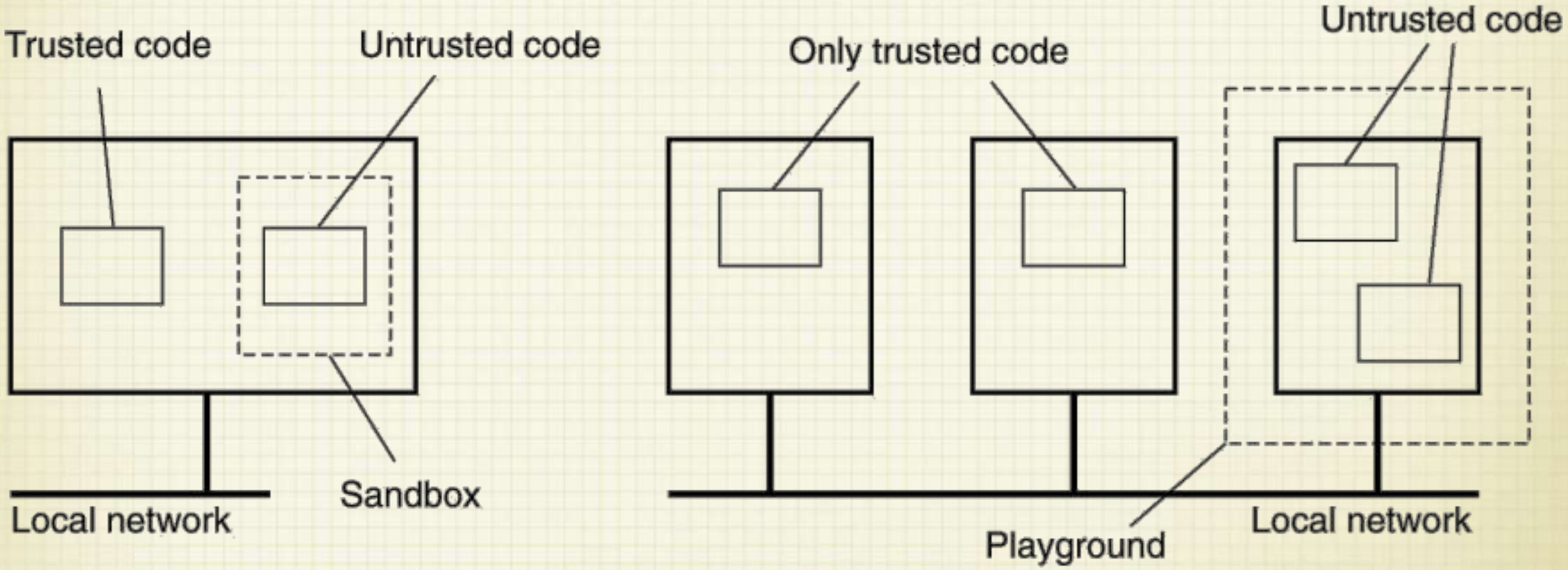
Java sandbox model

Secure Mobile Code (Cont.)

- **Playground**
  - Is a separate, specified machine exclusively reserved for running mobile code.
  - Offers more flexibility to the mobile code.
  - Local resources such as files or network connections to external servers are available to programs executing in the playground.
  - Local resources to other machines are physically disconnected from the playground and cannot be accessed by downloaded code
  - Users on these other machines can access the playground in a traditional way, for example, by means of RPC (Remote procedure call)

Secure Mobile Code (Cont.)

- Sandbox VS. Playground



Security Management

Key Management

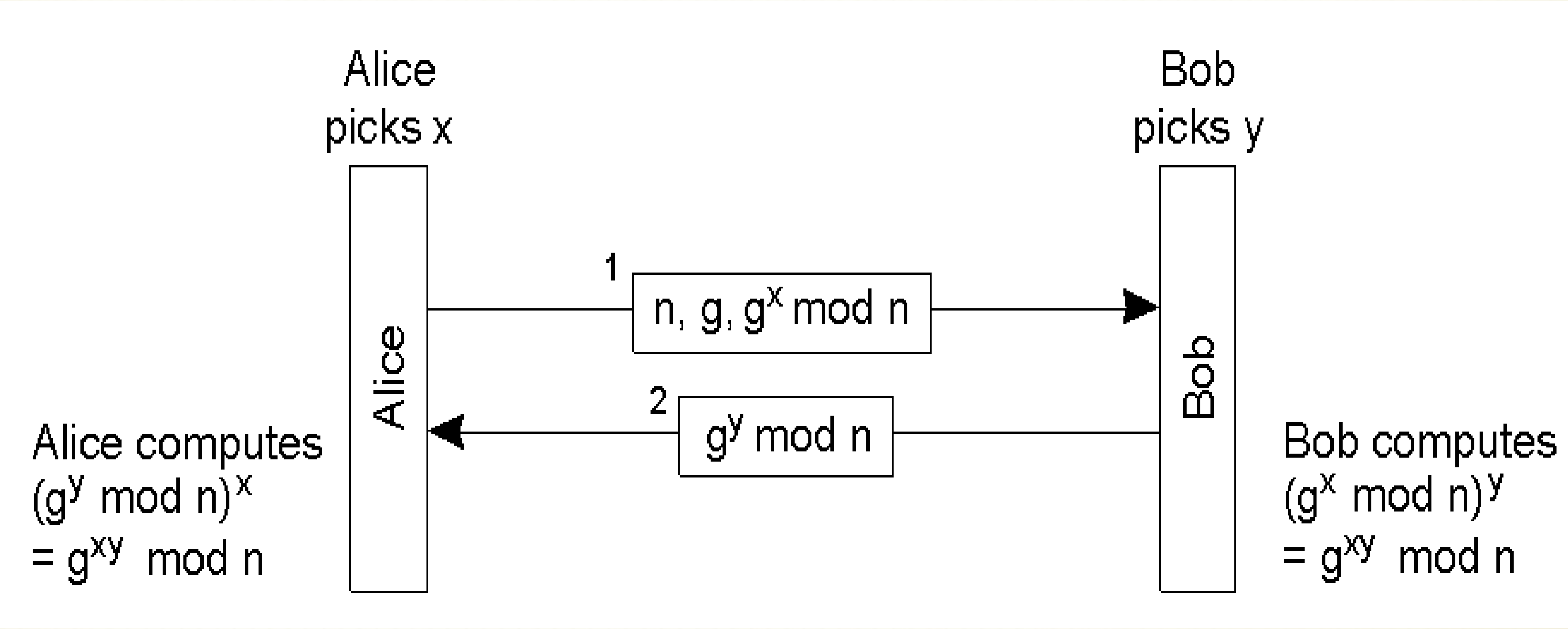
Issues in Key Distribution

Secure Group Management

Authorization Management

Key Management

- Diffie-Hellman key exchange
  - It is a simple and popular scheme for sending a shared secret key across an insecure channel.



The principle of Diffie-Hellman key exchange.

Key Management (Cont.)

- Diffie-Hellman key exchange works as follows :

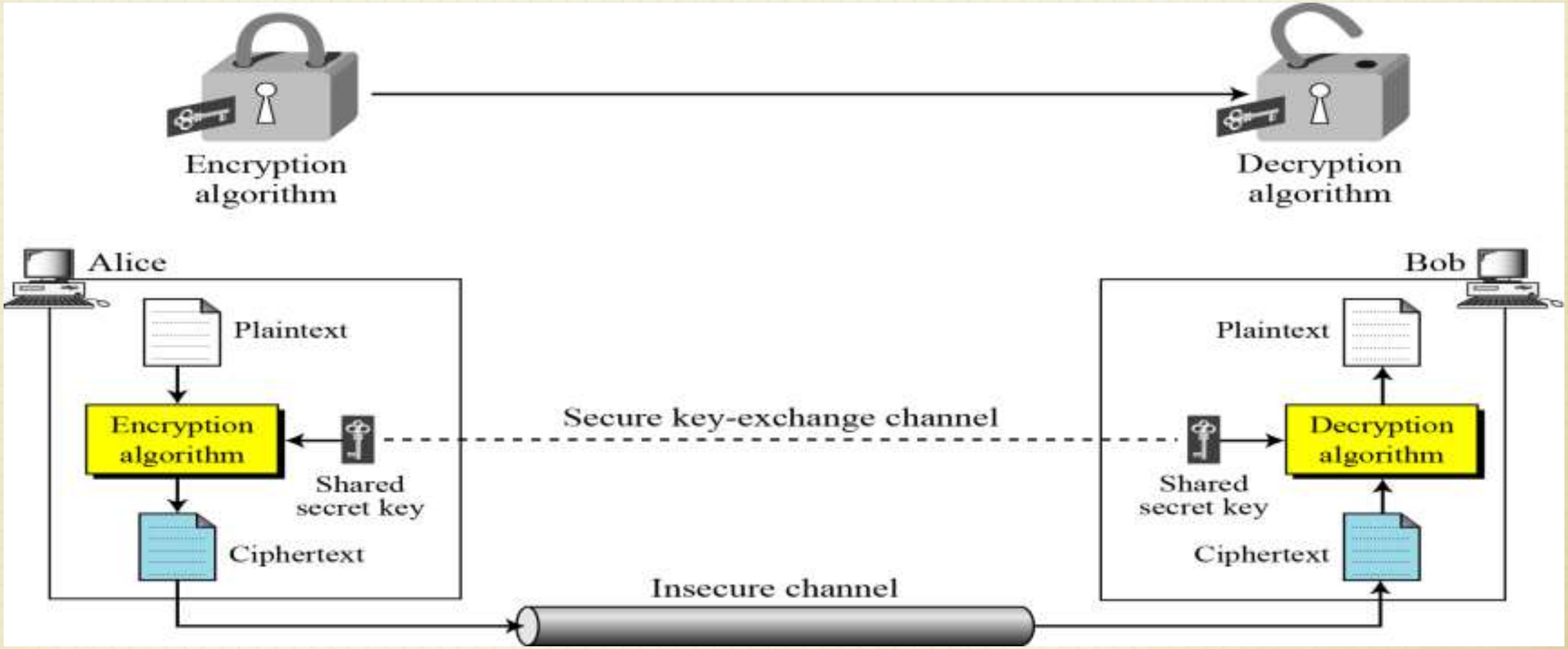
Suppose Alice and Bob want to establish a secret key.

1. They agree on two numbers *n* and *g* (where *n* is a prime and *g* is primitive root and can be public)
2. Alice picks a large secret random number *x* and Bob picks secret random number *y* .
3. Alice sends *g^x mod n* to Bob with *n* and *g.*
4. Bob calculates *(g^x mod n) ^y = g^xy mod n.*
5. Bob knows *g, n,* and *y* so he sends *g^y mod n* to Alice .
6. Alice computes *(g^y mod n) ^x = g^xy mod n.*

Thus both Alice and Bob have a shared secret key *g^xy mod n* that is known only to them and has been sent across a non-secure channel.

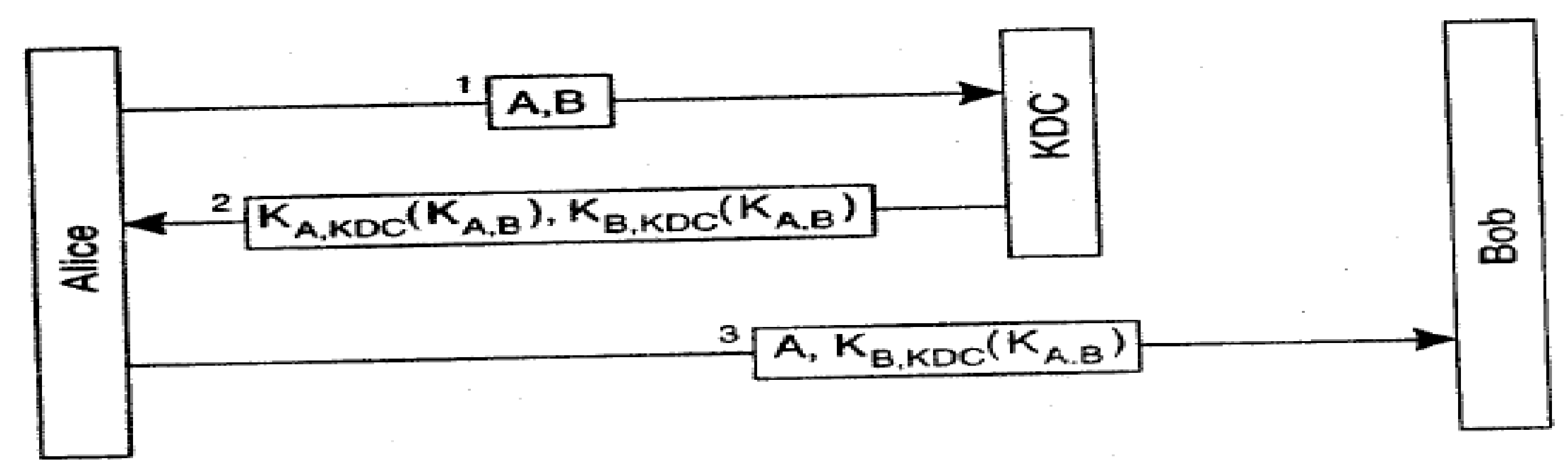
Issues in Key Distribution

- Key distribution in symmetric cryptosystem
  - Centralized approach
  - Fully distributed approach
  - Partially distributed approach



Issues in Key Distribution (Cont.)

- **Centralized approach**
  - In this approach, a single KDC (key distribution center) maintains a table of secret keys for each user.
  - Drawbacks
    - Performance bottleneck due to a single KDC.
    - Poor reliability.
    - KDC node may get crashed.
    - KDC may get overloaded in a system having too many users.



Key distribution in Centralized symmetric cryptosystem

Issues in Key Distribution (Cont.)

- **Fully distributed approach**
  - The KDC resides at each node in the distributed system and the secret keys are distributed well in advance.
  - Each KDC has a table of secret keys with private keys of all KDCs.
  - To establish a secure logical communication channel, user *A* makes a request to the local KDC (plaintext form).
  - This approach is more reliable than the centralized one.

Issues in Key Distribution (Cont.)

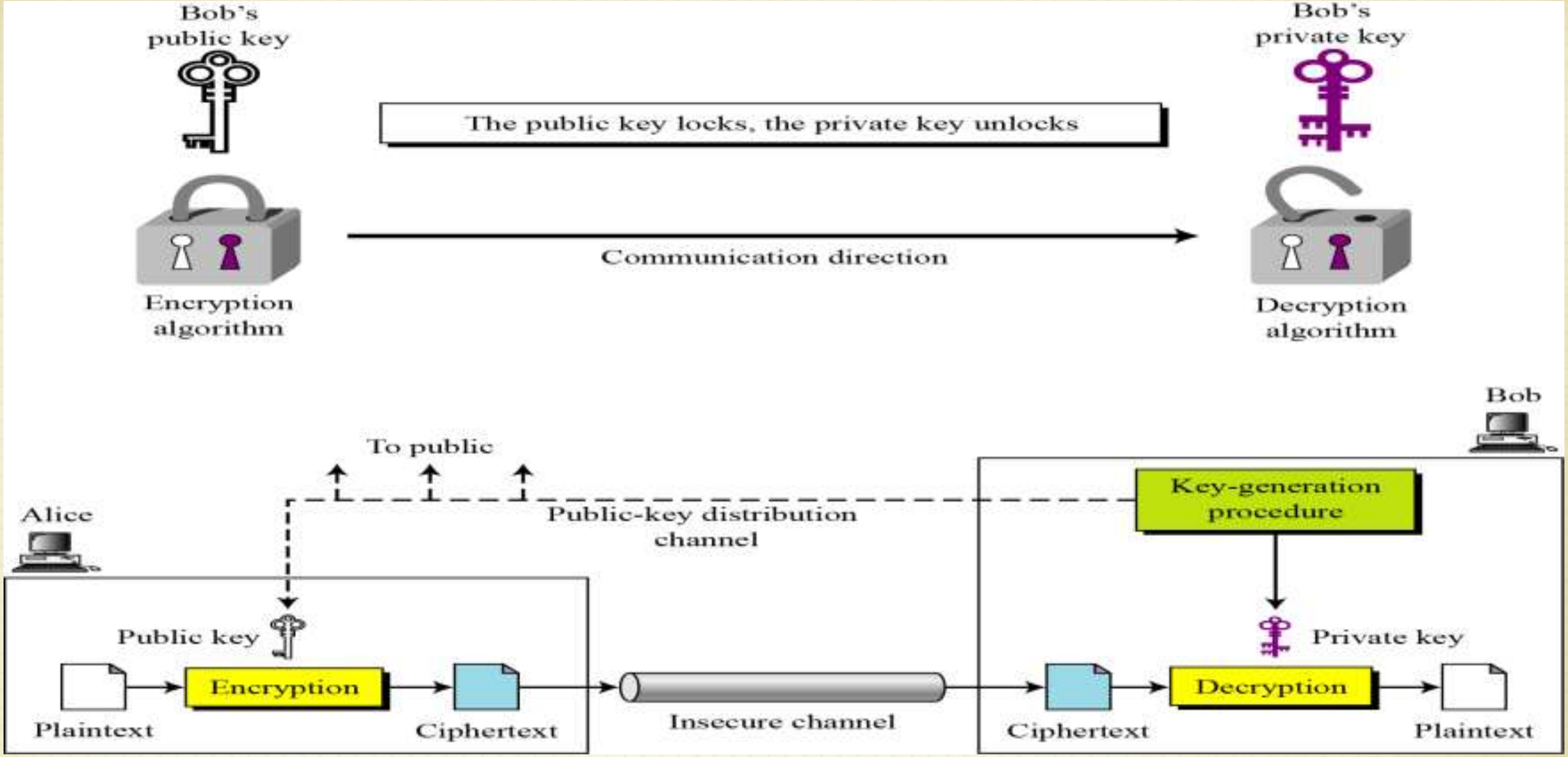
- **Partially distributed approach**
  - The nodes are partitioned into regions and each region has a KDC.
  - The prior distribution of secret keys allows each KDC to communicate securely with each user of its own region and with KDCs of other regions.
  - The failure of the KDC of a particular region affects the key distribution activities only of that region.
  - The reliability of this approach lies between the reliabilities of the centralized and fully distributed approaches.

Issues in Key Distribution (Cont.)

Key distribution in asymmetric cryptosystem

- Only public keys are distributed which in anyway need not be kept secret and can be transmitted over an insecure channel.
- The key distribution procedure involves an authentication procedure to prevent an intruder for generating a pair of keys and sending a public key to establish a secure communication channel.
- Public key manager (PKM) maintains a directory of public keys of all users in the system.

Issues in Key Distribution (Cont.)



The general idea behind asymmetric-key cryptography

Issues in Key Distribution (Cont.)

- **Public-key certificates**
  - Consist of public keys and the identities to which the keys are associated.
  - A certification authority signs the <public key, identifier> pair and places it on the certificate.
  - A private key of the certification authority is used to sign the certificate.

Issues in Key Distribution (Cont.)

- **Lifetime of certificates**
  - The certificates can be revoked in various ways using
    - Using certification revocation list (CRL)
      - The CRL is periodically published by the certification authority. when a client checks a certificate it first checks the CRL to see if it is revoked or not.
    - Restricting the lifetime of the certificate
      - Restrict the lifetime of the certificate by using the concept of leases.
    - Reduce the lifetime of the certificate to zero
      - Even if the lifetime of the certificate is reduced to zero, the client will always have to contact the certification authority for checking the validity of the public key.

Secure Group Management

- Key Distribution Centers (KDCs) or Certification Authorities (CAs) must be trusted.
- It is necessary that security services (*e.g.* KDCs and CAs) offer high availability.
- If a process asks to join a group *G*, the integrity of the group must not be compromised.

Authorization Management

Capabilities

- Are the tickets that the users can carry to access a named object.
- Comprised from two parts
  - An object identifier, that contains a pointer to the object and acts as unique name for it.
  - Rights information, which is a set of bits that determine the operation allowed on the object.
- Basically used for
  - To allow its holder to access the object.
  - To uniquely identifying an object.
- Working and properties of a security system based on Capabilities
  - Access validation.
  - Granting and passing rights.
  - Protecting capabilities against unauthorized access.
  - Rights amplification.
  - Rights revocation.

Authorization Management(Cont.)

- **Access validation**
  - In capability-based, there is no need to search a list to verify whether access is allowed. It is only required to verify that the capability supplied by a process is valid.
  - There is no checking of user identity.

- **Granting and passing rights**
  - The capability-based security system uses one or more object managers for each object type.
  - All requests to create an object or to perform some operation on the object sent to the object manager of that object type.
  - When a new object is created , the corresponding object manager generates a capability with all access rights for the object and generated capability is returned to the owner for use.

Authorization Management(Cont.)

- **Protecting capabilities against unauthorized access**
  - A capability must uniquely identify an object in the entire system. The capability must not be reused after the object is deleted.
  - Capabilities must be protected from user tampering.
  - It should be difficult to guess the capability.
- **Rights amplification**
  - Each object type has a set of kernel rights such as get, put, and add to manipulate the data part of an object, and load, store, delete, copy, and create to manipulate the capability part of the list object.

Authorization Management(Cont.)

- **Rights revocation**
  - It is difficult to determine which subjects have what rights for object. This because capabilities for an object may be stored in several capability lists that are distributed throughout the system.
  - Methods can be used for implementing revocation for capabilities
    - ✓ Back pointers
      - Maintain a list of pointers with an object, pointing to all capabilities associated with the object.
    - ✓ Indirection
      - Each capability points to an indirect object via a table entry instead of the object itself and the table entry points to the real object.
    - ✓ Use of key
      - In addition to the object identifier and rights information field, each capability has a key field.
      - Each object has a master key that can be changed with the *set_key* operation.
      - Revocation involves replacement of the master key with new value by using the *set_key* operation.

Questions