

Chapter 14: Protection and Security

Policy and Mechanism

- Protection mechanisms are used to authenticate access to resources
  - File protection
  - Memory protection
- A security policy reflects an organization's strategy to authorize access to the computer's resources
  - Managers have access to personnel files
  - OS processes have access to the page table
- Authentication mechanisms are the basis of most protection mechanisms. Two types:
  - External Authentication
  - Internal Authentication

External Authentication

- User/process authentication
  - Is this user/process who it claims to be?
    - ❖ Passwords
    - ❖ More sophisticated mechanisms
- Authentication in networks
  - Is this computer who it claims to be?
    - ❖ File downloading
    - ❖ Obtaining network services
    - ❖ The Java promise

Internal Authentication



- **Sharing parameters:** A process changing the parameter values of another process without access authorization is a violation.
- **Confinement:** Contain all rights to resources so that they do not propagate outside some chosen set of processes.
- **Allocating rights:** A process may provide another process with specific rights to use its resources.
- **Trojan horse:** If the server program takes advantage of the client process's rights to access resources on its own behalf, it is called a Trojan horse.

A Model for Resource Protection

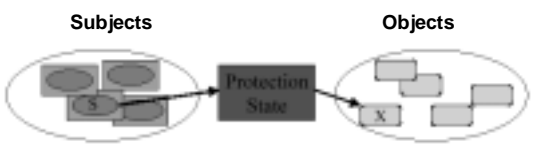
- Active parts (e.g., processes or threads) are called subjects and act on behalf of users.
- Passive parts (i.e., resources) are called objects.
- The particular set of rights a process has at any given time is referred to as its protection domain.
- A subject is a process executing in a specific protection domain.
- A protection system is composed of a set of objects, a set of subjects, and a set of rules specifying the protection policy.
- Want mechanism to implement different security policies for subjects to access objects
  - Many different policies must be possible
  - Policy may change over time

A Protection System



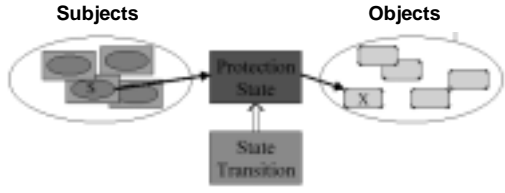
- S desires  $\alpha$  access to X

A Protection System



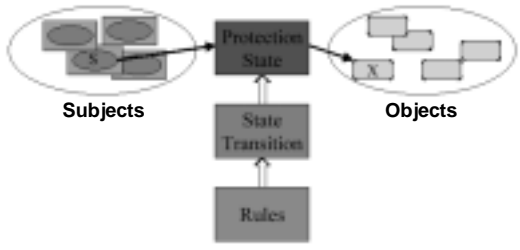
- S desires  $\alpha$  access to X
- Protection state reflects current ability to access X

A Protection System



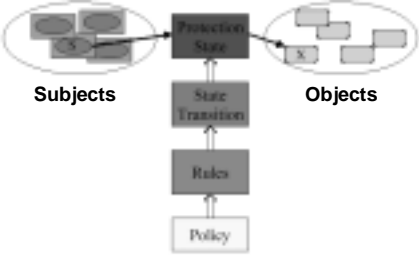
- S desires  $\alpha$  access to X
- Protection state reflects current ability to access X
- Authorities can change

A Protection System



- S desires  $\alpha$  access to X
- Protection state reflects current ability to access X
- Authorities can change
- What are rules for changing authority?

A Protection System



- S desires  $\alpha$  access to X
- Protection state reflects current ability to access X
- Authorities can change
- What are rules for changing authority?
- How are the rules chosen?

Protection System Example



- S desires  $\alpha$  access to X

Protection System Example



		X
S		$\alpha$

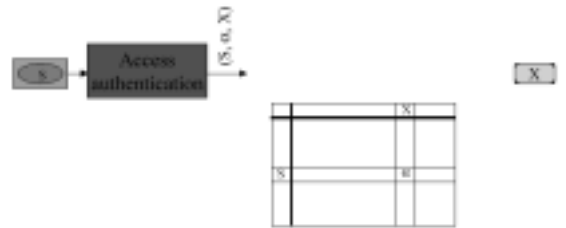
Access matrix

- S desires  $\alpha$  access to X
- Captures the protection state

### Protection State Example

	S <sub>1</sub>	S <sub>2</sub>	S <sub>3</sub>	F <sub>1</sub>	F <sub>2</sub>	D <sub>1</sub>	D <sub>2</sub>
S <sub>1</sub>	control	block wakeup owner	control owner	read* write*		seek	owner
S <sub>2</sub>		control	stop	owner	update	owner	seek*
S <sub>3</sub>			control	delete	execute owner		

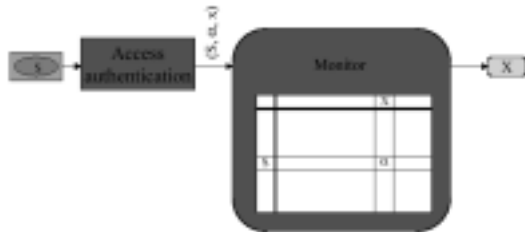
### Protection System Example



Access matrix

- S desires  $\alpha$  access to X
- Captures the protection state
- Generates an unforgeable ID

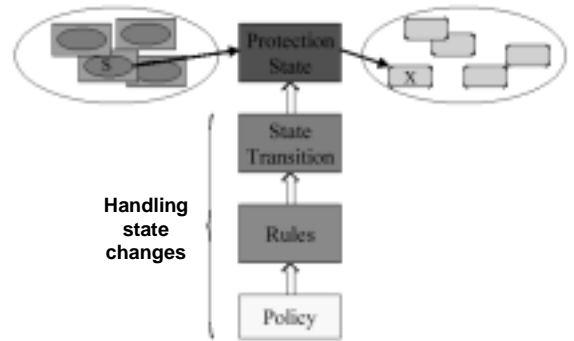
### Protection System Example



Access matrix

- S desires  $\alpha$  access to X
- Captures the protection state
- Generates an unforgeable ID
- Checks the access against the protection state

### A Protection System



### Policy Rules Example

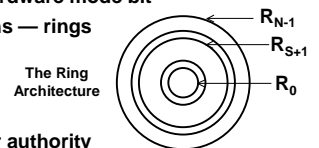
	S <sub>1</sub>	S <sub>2</sub>	S <sub>3</sub>	F <sub>1</sub>	F <sub>2</sub>	D <sub>1</sub>	D <sub>2</sub>
S <sub>1</sub>	control	block wakeup owner	control owner	read* write*		seek	owner
S <sub>2</sub>		control	stop	owner	update	owner	seek*
S <sub>3</sub>			control	delete	execute owner		

### Rules for a Particular Policy

Rule	Command by S <sub>2</sub>	Authorization	Effect
1	transfer( $\alpha$ ) to (S, X)	$\alpha \in A[S_2, X]$	$A[S, X] = A[S, X] \cup \{\alpha\}$
2	grant( $\alpha$ ) to (S, X)	owner $A[S_2, X]$	$A[S, X] = A[S, X] \cup \{\alpha\}$
3	delete $\alpha$ from (S, X)	control $A[S_2, S]$ or owner $A[S_2, X]$	$A[S, X] = A[S, X] - \{\alpha\}$

### Protection Domains

- Lampson model uses processes and domains — how is a domain implemented?
  - Supervisor/user hardware mode bit
  - Software extensions — rings



- Inner rings have higher authority
  - Ring 0 corresponds to supervisor mode
  - Rings 1 to S have decreasing protection, and are used to implement the OS
  - Rings S+1 to N-1 have decreasing protection, and are used to implement applications

### Protection Domains

- Ring crossing is a domain change
- Inner ring crossing  $\Rightarrow$  rights amplification
  - Specific gates for crossing
  - Protected by an authentication mechanism
- Outer ring crossing uses less-protected objects
  - No authentication
  - Need a return path
  - Used in Multics and Intel 80386 (& above) hardware

### Implementing Access Matrix

- Usually a sparse matrix
  - Too expensive to implement as a table
  - Implement as a list of table entries
- Column oriented list is called an access control list (ACL)
  - List kept at the object
  - UNIX file protection bits are one example
- Row oriented list is called a capability list
  - List kept with the subject (i.e., process)
  - Kerberos ticket is a capability
  - Mach mailboxes protected with capabilities

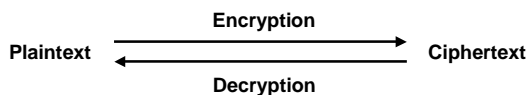
### More on Capabilities

- Provides an address to object from a very large address space
- Possession of a capability represents authorization for access
- Implied properties:
  - Capabilities must be very difficult to guess
  - Capabilities must be unique and not reused
  - Capabilities must be distinguishable from randomly generated bit patterns

### Cryptography

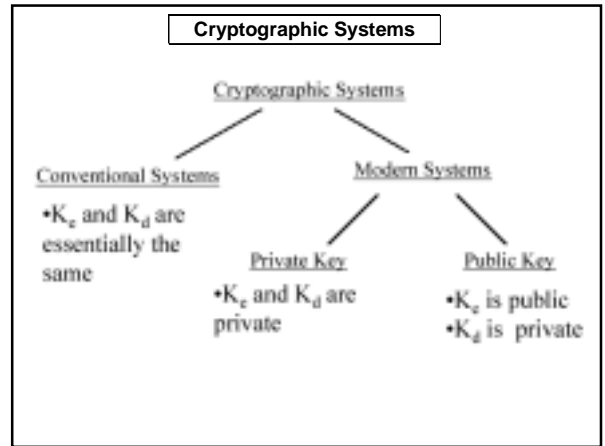
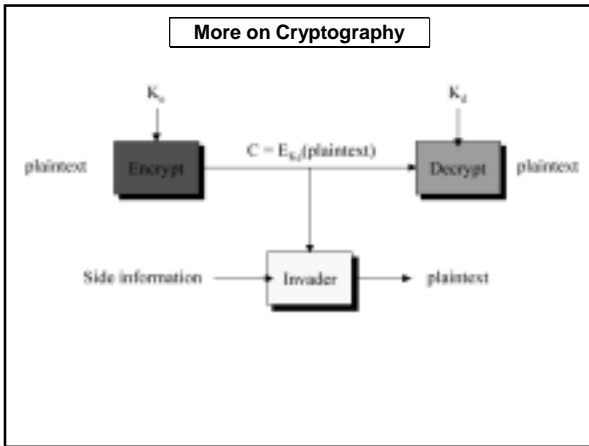
- Information can be encoded using a key when it is written (or transferred) — encryption
- It is then decoded using a key when it is read (or received) — decryption
- Very widely used for secure network transmission

### More on Cryptography

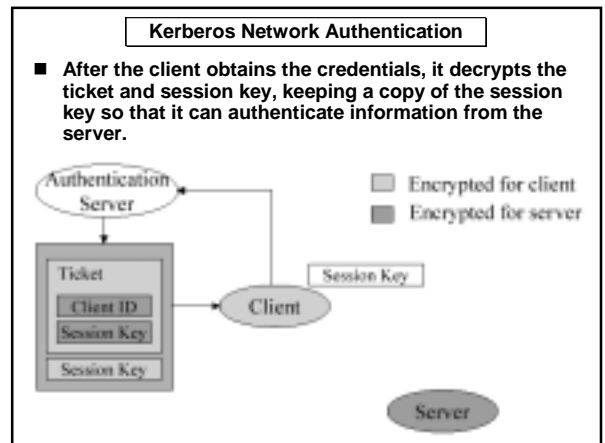
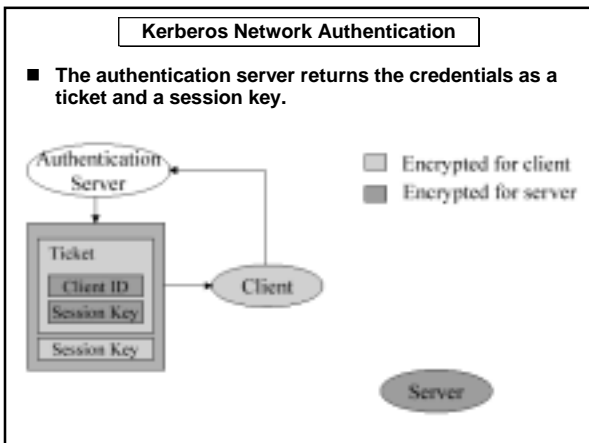
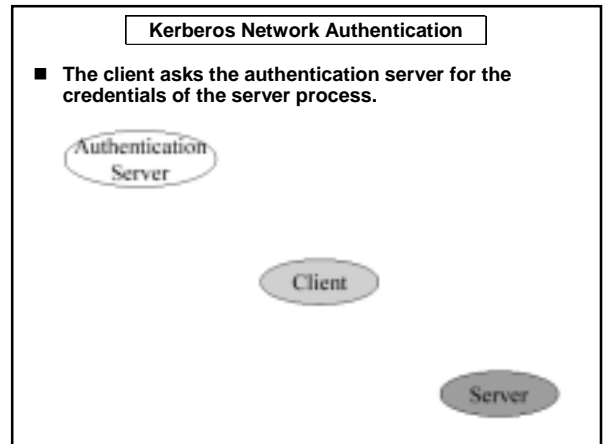


### More on Cryptography



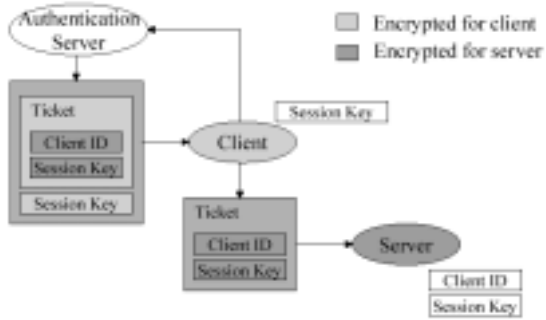


- ### Kerberos Network Authentication
- Kerberos is a set of network protocols that can be used to authenticate access to one computer by a user at a different computer using an unsecure network.
  - In Kerberos, it is assumed that a process on one computer (the client) wishes to employ the services of a process on another computer (the server) using the network for communication.
  - Kerberos assumes that information transmitted over the network could be tampered with during transmission.
  - Kerberos does not assume that the operating systems on the two machines are necessarily secure.



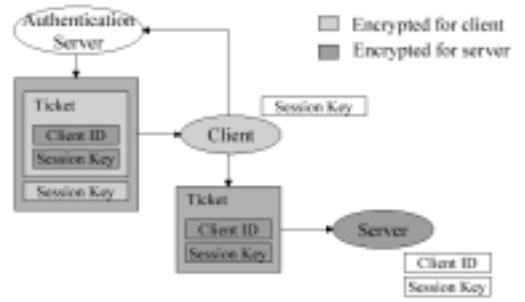
### Kerberos Network Authentication

- The client then sends a copy of the ticket with the encrypted fields intact to the server.



### Kerberos Network Authentication

- The server decrypts the copy of the ticket so that it can obtain a secure copy of the client's identification and of the session key.



END