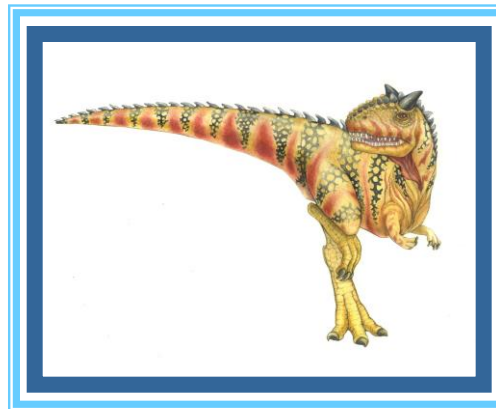
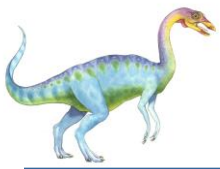


Chapter 14: Protection

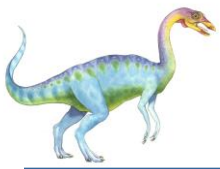




Chapter 14: Protection

- Goals of Protection
- Principles of Protection
- Domain of Protection
- Access Matrix

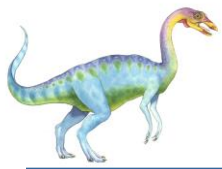




Objectives

- Discuss the goals and principles of protection in a modern computer system
- Explain how protection domains combined with an access matrix are used to specify the resources a process may access





Goals of Protection

- In one protection model, computer consists of a collection of objects, hardware or software
- Each object has a unique name and can be accessed through a well-defined set of operations
- Protection problem - ensure that each object is accessed correctly and only by those processes that are allowed to do so

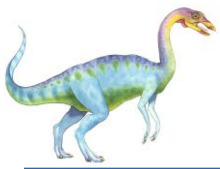




Principles of Protection

- Guiding principle – **principle of least privilege**
 - Programs, users and systems should be given just enough **privileges** to perform their tasks
 - Limits damage if entity has a bug, gets abused
 - Can be static (during life of system, during life of process)
 - Or dynamic (changed by process as needed) – **domain switching, privilege escalation**
 - “Need to know” a similar concept regarding access to data

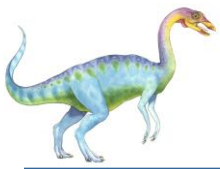




Principles of Protection (Cont.)

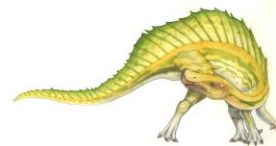
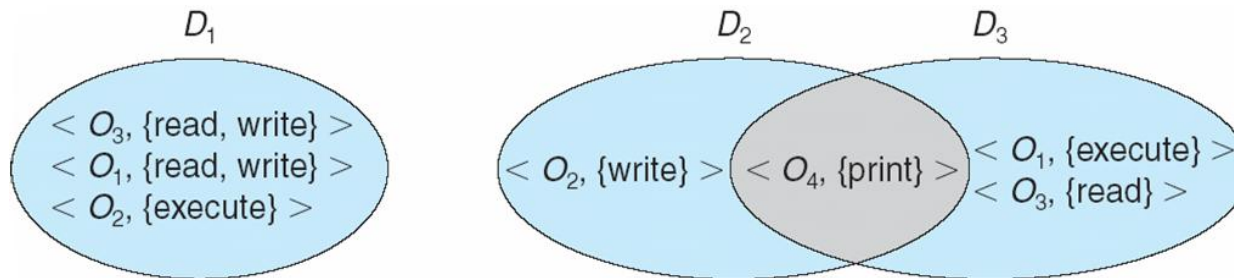
- Must consider “grain” aspect
 - Rough-grained privilege management easier, simpler, but least privilege now done in large chunks
 - ▶ For example, traditional Unix processes either have abilities of the associated user, or of root
 - Fine-grained management more complex, more overhead, but more protective
 - ▶ File ACL lists, RBAC
- Domain can be user, process, procedure





Domain Structure

- Access-right = $\langle \text{object-name}, \text{rights-set} \rangle$
where *rights-set* is a subset of all valid operations that can be performed on the object
- Domain = set of access-rights



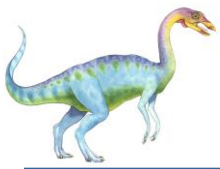


Access Matrix

- View protection as a matrix (**access matrix**)
- Rows represent domains
- Columns represent objects
- **Access** (i, j) is the set of operations that a process executing in Domain_i can invoke on Object_j

object \ domain	F_1	F_2	F_3	printer
D_1	read		read	
D_2				print
D_3		read	execute	
D_4	read write		read write	

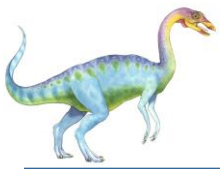




Use of Access Matrix

- If a process in Domain D_i tries to do “op” on object O_j , then “op” must be in the access matrix
- User who creates object can define access column for that object
- Can be expanded to dynamic protection
 - Operations to add, delete access rights
 - Special access rights:
 - ▶ *owner of O_i*
 - ▶ *copy op from O_i to O_j (denoted by “*”)*
 - ▶ *control – D_i can modify D_j access rights*
 - ▶ *transfer – switch from domain D_i to D_j*
 - *Copy and Owner* applicable to an object
 - *Control* applicable to domain object





Use of Access Matrix (Cont.)

- **Access matrix** design separates mechanism from policy
 - Mechanism
 - ▶ Operating system provides access-matrix + rules
 - ▶ If ensures that the matrix is only manipulated by authorized agents and that rules are strictly enforced
 - Policy
 - ▶ User dictates policy
 - ▶ Who can access what object and in what mode
- But doesn't solve the general confinement problem

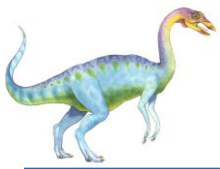




Access Matrix of Figure A with Domains as Objects

domain \ object	F_1	F_2	F_3	laser printer	D_1	D_2	D_3	D_4
D_1	read		read			switch		
D_2				print			switch	switch
D_3		read	execute					
D_4	read write		read write		switch			





Access Matrix with Copy Rights

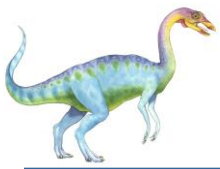
domain \ object	F_1	F_2	F_3
D_1	execute		write*
D_2	execute	read*	execute
D_3	execute		

(a)

domain \ object	F_1	F_2	F_3
D_1	execute		write*
D_2	execute	read*	execute
D_3	execute	read	

(b)





Access Matrix With Owner Rights

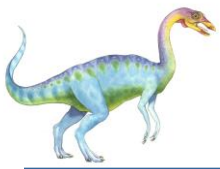
domain \ object	F_1	F_2	F_3
D_1	owner execute		write
D_2		read* owner	read* owner write
D_3	execute		

(a)

domain \ object	F_1	F_2	F_3
D_1	owner execute		write
D_2		owner read* write*	read* owner write
D_3		write	write

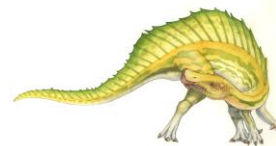
(b)





Modified Access Matrix of Figure B

domain \ object	F_1	F_2	F_3	laser printer	D_1	D_2	D_3	D_4
D_1	read		read			switch		
D_2				print			switch	switch control
D_3		read	execute					
D_4	write		write		switch			



End of Chapter 14

