System Protection

Goal of Protection –

Protection can improve reliability by differentiating unauthorised users from authorised users.

The role of protection in a computer system is to provide a mechanism that enforces policies when using resources.

Mechanism – determines how something will be done.

Policies – decides what will be done.

Principles of protection – is the "Principle of least privilege"

It dictates that programs, users, and even systems be given just enough privileges to perform their task.

It helps to produce a more secure environment.

Ex – Read only mode is required then it will never give R & W both modes to the user.

Domain of protection –

A computer system is a collection of processes and objects.

objects are -

H/W - CPU, Printer etc

S/W - files, programs etc

(1) A process should be allowed to access only those objects for which it has authorization.

and

(2) A process should be able to access only those resources that it currently requires to complete its task.

This is also known as "need-to-know" principle.

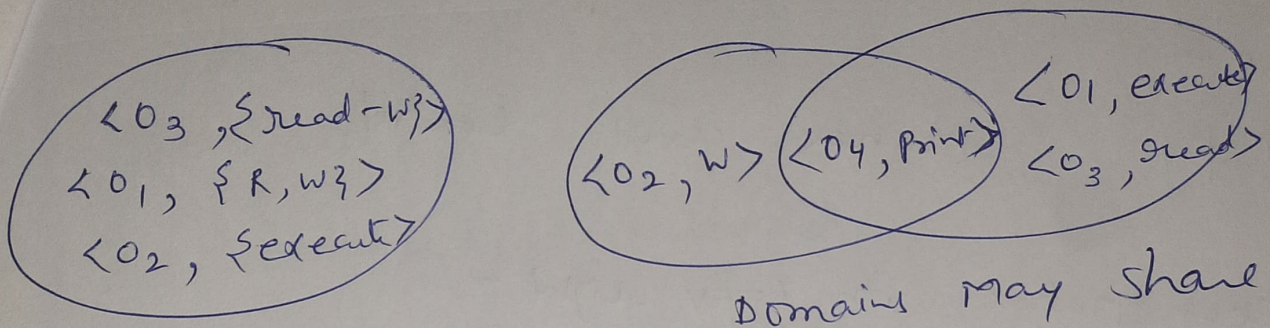
Domain structure - A process operates within a protection domain, which specifies the resources that a process may access.

Domain - (Set of objects + type of operations)

A domain is a collection of access rights that defines ability to execute an operation on an object.

Access Right - < Object-Name, right-set >

D1 D2 D3



Domains may share their access rights

Access Matrix – Protection model can be represented by a matrix called as Access matrix.

rows – Domains
Columns – objects
each entry – access right

Access (i,j) → Process in D_i can invoke an operation on O_j

Obj / Dom	f_1	f_2	f_3	Printer	D1
D1	R		R		
D2				P	Switch
D3		R	E		
D4	R W		R W		

Access (1,1) = R
Process in D_1 can Read (operation) file f_1.

A process in D2 can switch to Domain D1

To allow controlled changes in the ac matrix entries, three operation required are -

Copy, owner and control.

ACCESS matrix with copy right -

Obj / Domain	F1	F2	F3
D1	Execute		Write*
D2	E	Read*	E
D3	E		

⇒

	F1	F2	F3
D1	E		Write*
D2	E	Read*	E
D3	E	Read	

Ability to copy an access right from one domain to another (in the same column) is copy right represented by *.

Propagation of copy right is limited ⇒ means now process Running in Domain D3 can not copy access right Read to some other row as * is not there,

(Copy), (transfer) & (limited copy)

↓ Copied as read*

↓ Copied as read* but read* is removed from previous Domain

↓ Copied as Read

OWNER – It allows addition of new rights and removal of rights from access matrix.

If access (i, j) includes the owner right then a process executing in Domain Di can add and remove any right in any entry of object column j.

D \ O	F1	F2	F3
D1	OWNER E		W
D2		Read* OWNER	R* OWNER
D3	E		

⇒

D \ O	F1	F2	F3
D1	OWNER E		W
D2		OWNER R* W*	R* W OWNER
D3		Write	

D is owner of an object, so can change any right

CONTROL – A mechanism is needed to change the entries in a row.

The control right is applicable only to domain objects.

	F1	F2	F3	D1	D2
D1	R			S	control
D2		W			switch

D1 can change row of D2.

Implementation of Access matrix -

(1) global Table -

<domain, object, right-set>, search all these entries, if this triple found operation is allowed to continue.

(i) large table

(2) Access list of objects -

Each column in the access matrix can be implemented as an access list for one object.

operation M on an object Oj is attempted we search the access list of for entry <Di, M>

(3) Capability list for Domains -