Protection - System Protection, Goals of Protection, Principles of Protection, Domain of Protection, Access Matrix, Implementation of Access Matrix, Access Control, Revocation of Access Rights, Capability-Based Systems, Language-Based Protection.

1.System protection

we need to provide protection for several reasons. The most obvious is the need to prevent the causing harm, intentional violation of an access restriction by a user .protection can improve reliability by detecting hidden errors at the interface between component subsystems.easy detection of interface errors can prevent contamination of a healthy sub system by a malfunctioning sub system. An un protected resource cannot defend against use by an unauthorized user. A protection oriented system provides means to distinguish between authorized and unauthorized usage. mechanisms are distinct from policies. Mechanisms determine how something will be done. Policies decide what will be done.Policies are likely to change from place to place (or) time to time .protection is no longer the concern solely of the designer of an operating system.

2.Principles of protection

frequently, a guiding principle can be used through out a project, such as the design of an operating system. A key, time tested guiding principle for protection is the principle of least privilege. It dictates that programs, users, systems be given just enough privileges to perform their tasks.an operating system following the principle of least privilege implements its features, programs, system calls so that failure of a component does the minimum damage and allows the minimum damage to be done. Managing users with the principle of least privileges creating a separate account for each user, with just the privileges that the user needs. An operator who needs to mount tapes and backup files on the system has access to just these commands and files needed to accomplish the job. Computers implemented in a computing facility under the principle of least privilege can be limited to running specific services, accessing specific remote hosts and doing so during specific times. These restriction are implemented through enabling or disabling each service and through using access control lists.

3.Domain of protection:- A computer system is a collection of processes and objects .objects means both hardware objects(cpu, printer, disks, tape driver) software objects (files, programs, semaphores).

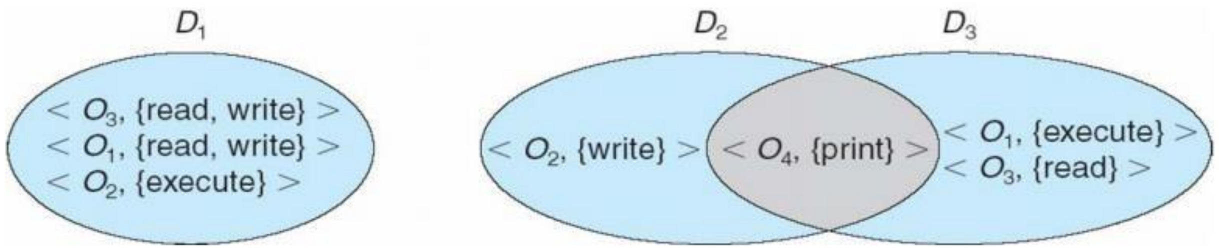
Each objects has a unique name that differentiates it from all other objects in the system, and each can be accessed only through well defined and meaningful operations. Objects are essentially abstract data types. The operations that are possible may depend on the object.
On a cpu, we can only execute memory segments can be read and written, cd-rom, dvd-rom can be only be read. Tape drivers can read, written. Data files can be opened, created ,read, written, executed, deleted. A process should be allowed to access only those resource for which it has authorization. At any time, a process should be able to access only those resources that it currently requires to complete its task. ***Need to know*** principle is useful in limiting the amount of damage a faulty process can cause in the system. When process p invokes procedure A(), the procedure should be allowed to access only its own variables and formal parameters passed to it. It should not be access all the variables of process p.

4.Domain structure

a process operates with in a protection domain, which specifies the resources that the process may access. Each domain defines a set of objects and the types of operations that may be invoked on each object. The ability to execute an operation on an object is an access right. A domain is a collection of access rights, each of which is an ordered pair <object name, right- set>.

Example: if domain D has the access right <file F,{read,write}>, then a process executing in domain d can be both read and write file F.

Domains do not need to be disjoint; they may share access rights.



Systems with 3 protection domains

We have 3 domains D1,D2,D3. The access right <O4,{print}>

is shared by D2 and D3, implying that a process executing in either of these two domains can print object O4. A process must be executing in domain D1 to read and write object O1, while only processes in domain D3 may execute object O1.

5.Access Matrix

our model of protection can be viewed abstractly as a matrix, called an access matrix. The rows of the access matrix represents domains, columns represent objects. Each entry in the matrix consists of a set of access rights. The entry access(i,j) defines the set of operations that a process executing in domain Di can invoke on object Oj.

Object/domain	F1	F2	F3	Printer
D1	read		read	
D2				print
D3		read	execute	
D4	read		read	

ACCESS MATRIX

`there are 4 domains and 4 objects- 3 files(F1,F2,F2) And one printer. A process executing in domain D1 can read files F1 and F3. A process executing in domain D4 has the same privileges as one executing in domain D1, but in addition, it can also write onto files F1 and F3. Note that the printer can be accessed only by a process executing in domain D2.

Object/doma	F1	F2	F3	printer	D1	D2	D3	D4
D1	rea		read			switch		
D2				Print			Switch	switch
D3		read	Execute					
D4	Rea d		Rea d		switch			

Access matrix with domains as objects

Object/domain	F1	F2	F3
D1	Execute		Write*
D2	Execute	Read*	execute
D3	Execute		

Object/domain	F1	F2	F3
D1	Execute		Write*
D2	Execute	Read*	execute
D3	Execute	read	

3.Capability lists for domain :-

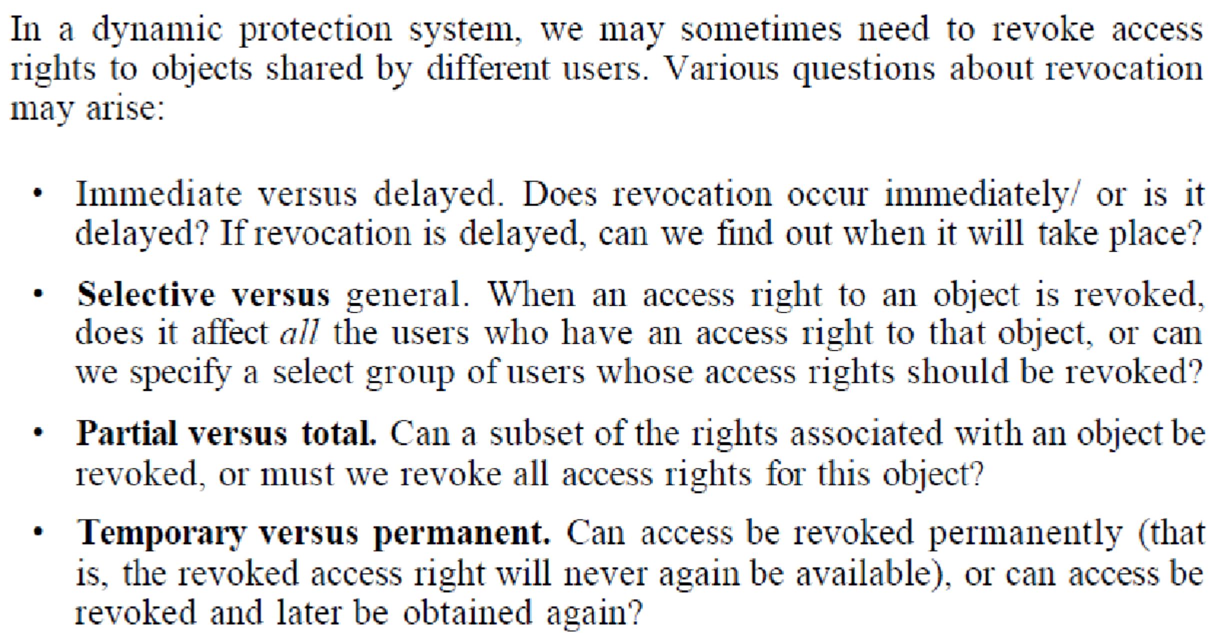
Rather than associating the column of the access matrix with the objects as access lists, we can associate each row with its domain. A capability list for a domain is a list of objects together with the opertions allowed on those objects. An object is often represented by its physical name (or) address, called a capability. To excute opertion M an object Oj, the process executes the opertion M , specifying the capability for object Oj as a parameter.

The capability list is associated with a domain, but it is never directly accessible to a process executing in that domain. the capability list is itself a protected object, maintained by the operating system and accessed by the user only indirectly. capability based protection relies on the fact that the capabilities are never allowed to migrate into any address space directly accessible by a user process (Where they colud be modified). If all the capabilities are secure,the object they protect Is also secure aganist unauthorised access.

4.A Lock Key mechanism :-

It is a compromise between access list and capability lists.Each object has a list of unique bit patterns,called locks.each domain has a list of unique bit patterns,called **keys**.A process executing in a domain can access an object only if that domain has a key that matches one of the locks of the object.

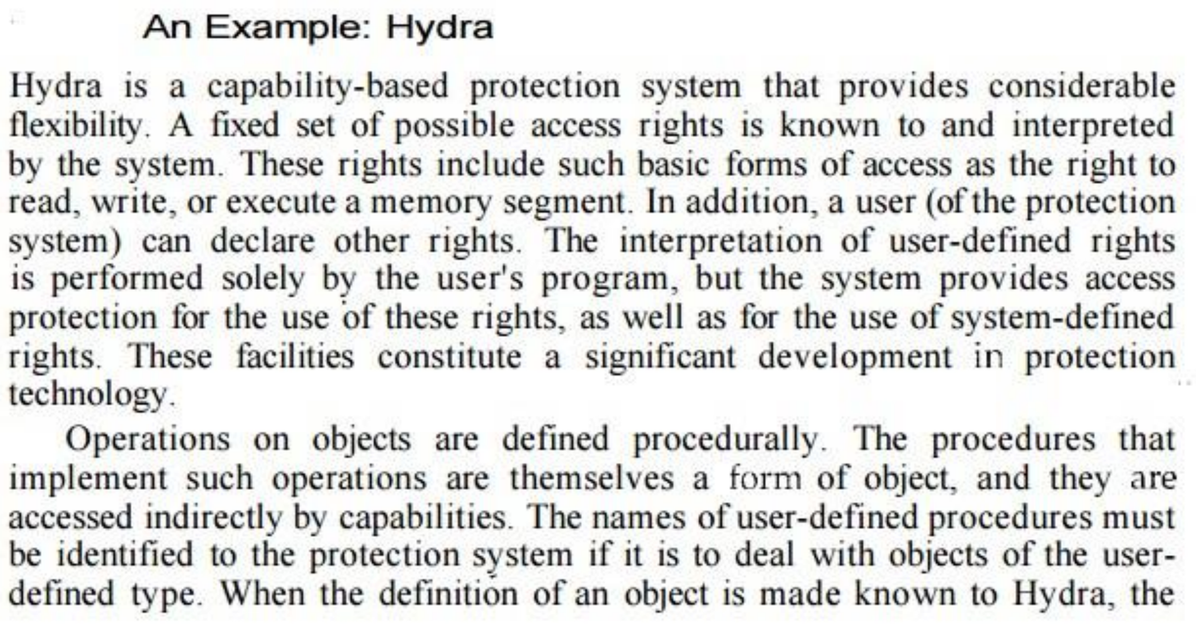
Revocation of access rights



In a dynamic protection system, we may sometimes need to revoke access rights to objects shared by different users. Various questions about revocation may arise:

- Immediate versus delayed. Does revocation occur immediately/ or is it delayed? If revocation is delayed, can we find out when it will take place?
- **Selective versus** general. When an access right to an object is revoked, does it affect *all* the users who have an access right to that object, or can we specify a select group of users whose access rights should be revoked?
- **Partial versus total.** Can a subset of the rights associated with an object be revoked, or must we revoke all access rights for this object?
- **Temporary versus permanent.** Can access be revoked permanently (that is, the revoked access right will never again be available), or can access be revoked and later be obtained again?

Capability - based systems



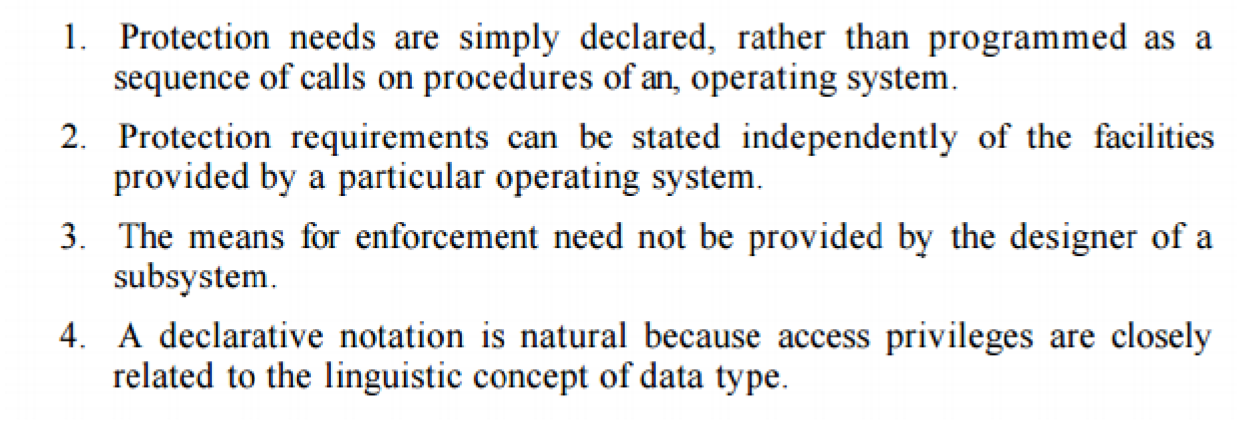
An Example: Hydra

Hydra is a capability-based protection system that provides considerable flexibility. A fixed set of possible access rights is known to and interpreted by the system. These rights include such basic forms of access as the right to read, write, or execute a memory segment. In addition, a user (of the protection system) can declare other rights. The interpretation of user-defined rights is performed solely by the user's program, but the system provides access protection for the use of these rights, as well as for the use of system-defined rights. These facilities constitute a significant development in protection technology.

Operations on objects are defined procedurally. The procedures that implement such operations are themselves a form of object, and they are accessed indirectly by capabilities. The names of user-defined procedures must be identified to the protection system if it is to deal with objects of the user-defined type. When the definition of an object is made known to Hydra, the

Language based protection

Kernal acts as a security agent to inspect and validate each attempt to access a protected resource. Operating systems provide higher level user interfaces and goals of protection drawn heavily on ideas that originated in programming languages and especially on the concepts of abstract data types and objects.



1. Protection needs are simply declared, rather than programmed as a sequence of calls on procedures of an, operating system.
2. Protection requirements can be stated independently of the facilities provided by a particular operating system.
3. The means for enforcement need not be provided by the designer of a subsystem.
4. A declarative notation is natural because access privileges are closely related to the linguistic concept of data type.