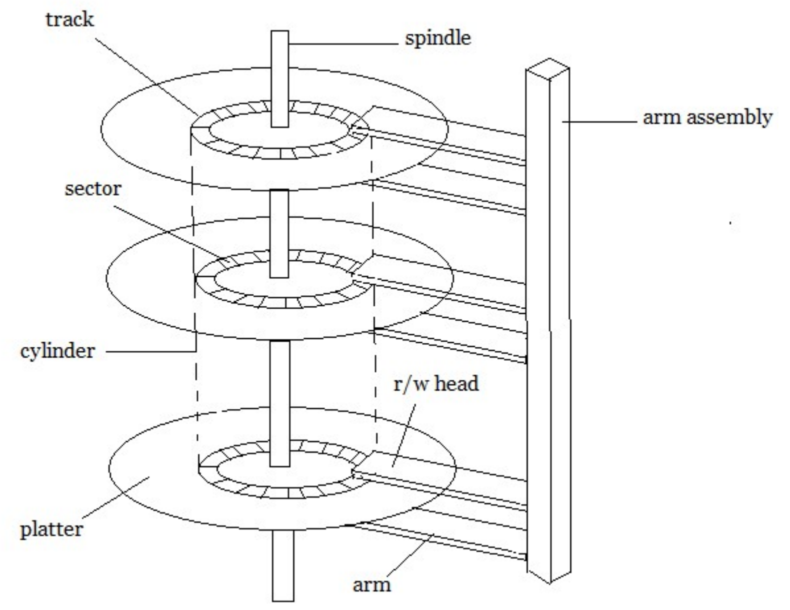
Magnetic Disk Structure

In modern computers, most of the secondary storage is in the form of magnetic disks. Hence, knowing the structure of a magnetic disk is necessary to understand how the data in the disk is accessed by the computer.



Structure of a magnetic disk

<u>Disk Scheduling Algorithms</u>

Disk scheduling is done by operating systems to schedule I/O requests arriving for disk. Disk scheduling is also known as I/O scheduling.

There are many Disk Scheduling Algorithms but before discussing them let's have a quick look at some of the important terms:

Seek Time: Seek time is the time taken to locate the disk arm to a specified track where the data is to be read or write. So the disk scheduling algorithm that gives minimum average seek time is better.

Rotational Latency: Rotational Latency is the time taken by the desired sector of disk to rotate into a position so that it can access the read/write heads. So the disk scheduling algorithm that gives minimum rotational latency is better.

Transfer Time: Transfer time is the time to transfer the data. It depends on the rotating speed of the disk and number of bytes to be transferred.

Disk Access Time: Disk Access Time is:

Disk Access Time = Seek Time + Rotational Latency + Transfer Time

Disk Response Time: Response Time is the average of time spent by a request waiting to perform its I/O operation. Average Response time is the response time of the all requests. Variance Response Time is measure of how individual request are serviced with respect to average response time. So the disk scheduling algorithm that gives minimum variance response time is better.

Disk Scheduling Algorithms

First Come First Serve (FCFS): FCFS is the simplest of all the Disk Scheduling Algorithms. In FCFS, the requests are addressed in the order they arrive in the disk queue.

Shortest Seek Time First (SSTF): In SSTF, requests having shortest seek time are executed first. So, the seek time of every request is calculated in advance in queue and then they are scheduled according to their calculated seek time. As a result, the request near the disk arm will get executed first. SSTF is certainly an improvement over FCFS as it decreases the average response time and increases the throughput of system.

SCAN: In SCAN algorithm the disk arm moves into a particular direction and services the requests coming in its path and after reaching the end of disk, it reverses its direction and again services the request arriving in its path. So, this algorithm works like an elevator and hence also known as elevator algorithm. As a result, the requests at the midrange are serviced more and those arriving behind the disk arm will have to wait.

C-SCAN: In SCAN algorithm, the disk arm again scans the path that has been scanned, after reversing its direction. So, it may be possible that too many requests are waiting at the other end or there may be zero or few requests pending at the scanned area.

These situations are avoided in C-SAN algorithm in which the disk arm instead of reversing its direction goes to the other end of the disk and starts servicing the requests from there. So, the disk arm moves in a circular fashion and this algorithm is also similar to SCAN algorithm and hence it is known as C-SCAN (Circular SCAN).

LOOK: It is similar to the SCAN disk scheduling algorithm except the difference that the disk arm in spite of going to the end of the disk goes only to the last request to be serviced in front of the head and then reverses its direction from there only. Thus it prevents the extra delay which occurred due to unnecessary traversal to the end of the disk.

C-LOOK: As LOOK is similar to SCAN algorithm, in similar way, C-LOOK is similar to C-SCAN disk scheduling algorithm. In C-LOOK, the disk arm in spite of going to the end goes only to the last request to be serviced in front of the head and then from there goes to the other end's last request. Thus, it also prevents the extra delay which occurred due to unnecessary traversal to the end of the disk.

<u>Disk Management</u>

Disk Management is a utility built into different operating systems which can be used to create, delete, format partitions, assign drive letters, and much more. Disk Management can also be used to view partitions and their formatted file systems on the hard drive.

<u>Swap-Space Management</u>

Swap-space management refers to the process where OS breaks the physical RAM that is structured with RAM into some pitches of memory that is called pages. This is the method in which a single page of memory can be copied to the preconfigured space on a hard disk. This is called swap space. This process helps to free up that page of memory.

<u>Protection and security</u>

Protection is a mechanism to control access to resources. Protection mechanisms provide controlled access by limiting the type of access that various users can make.

Security is a measure of confidence that the integrity of a (computer) system relies on. A security system prevents unauthorized access to a system.

<u>Goals of protection</u>

• Prevent accidental and maliciously destructive behavior.

• Ensure fair and reliable resource usage.

• Provide a mechanism for the enforcement of the policies governing resources use.

<u>Domain of Protection</u>

A Computer System is a collection of processes and (both HW & SW) objects, each of which has a unique name, and can be accessed through a well-defined set of operations.

Protection domain is an abstract notion. A process operates within a protection domain. The domain specifies the resources that the process may access.

Domain Structure

- Domain is a set of access rights.
- Each domain defines a set of objects and the types of operations that may be invoked on each object.
- An access right is the ability to execute an operation on an object.
- A domain is defined as a set of < object, { access right set } > pairs

Access Matrix

- Access Matrix is an abstract, general representation of the protection domains model.
- Generally, the users determine the content of the column for objects they create.
- A process may switch from a domain to another domain while it executes.
- Domains can be viewed as objects.

Implementations of Access Matrix

Global Table

- The simplest approach is one big global table with < domain, object, rights > entries.
- Unfortunately this table is very large (even if sparse) and so cannot be kept in memory (without invoking virtual memory techniques.)

Access Lists for Objects

- Each column of the table can be kept as a list of the access rights for that particular object, discarding blank entries.
- For efficiency a separate list of default access rights can also be kept, and checked first.

Capability Lists for Domains

- In a similar fashion, each row of the table can be kept as a list of the capabilities of that domain.
- Capability lists are associated with each domain, but not directly accessible by the domain or any user process.

A Lock-Key Mechanism

- Each resource has a list of unique bit patterns, termed locks.
- Each domain has its own list of unique bit patterns, termed keys.

- Access is granted if one of the domain's keys fits one of the resource's locks.

Revocation of Access Rights

In a dynamic system we may need to revoke access rights to objects.

Several issues are important:

- Revocation is immediate or delayed.
- Revocation is general (affects all users) or selective (affects only certain users).
- Revocation is total (affects all access rights) or partial (affects subset of the rights).
- Revocation is temporary or permanent.

With an access list scheme revocation is easy, immediate, and can be selective, general, partial, total, temporary, or permanent, as desired.

With capabilities lists the problem is more complicated, because access rights are distributed throughout the system. A few schemes that have been developed include:

Reacquisition - Capabilities are periodically revoked from each domain, which must then re-acquire them.

Back-pointers - A list of pointers is maintained from each object to each capability which is held for that object.

Indirection - Capabilities point to an entry in a global table rather than to the object. Access rights can be revoked by changing or invalidating the table entry, which may affect multiple processes, which must then re-acquire access rights to continue.

Keys - A unique bit pattern is associated with each capability when created, which can be neither inspected nor modified by the process.

- A master key is associated with each object.
- When a capability is created, its key is set to the object's master key.
- As long as the capability's key matches the object's key, then the capabilities remain valid.
- The object master key can be changed with the set-key command, thereby invalidating all current capabilities.
- More flexibility can be added to this scheme by implementing a list of keys for each object, possibly in a global table.