File System

File

A file is a named collection of related information that is recorded on secondary storage such as magnetic disks, magnetic tapes and optical disks. In general, a file is a sequence of bits, bytes, lines or records whose meaning is defined by the files creator and user.

File Structure

A File Structure should be according to a required format that the operating system can understand.

* A file has a certain defined structure according to its type.
* A text file is a sequence of characters organized into lines.
* A source file is a sequence of procedures and functions.
* An object file is a sequence of bytes organized into blocks that are understandable by the machine.
* When operating system defines different file structures, it also contains the code to support these file structure. Unix, MS-DOS support minimum number of file structure.

File Type

File type refers to the ability of the operating system to distinguish different types of file such as text files source files and binary files etc. Many operating systems support many types of files. Operating system like MS-DOS and UNIX have the following types of files −

Ordinary files

* These are the files that contain user information.
* These may have text, databases or executable program.
* The user can apply various operations on such files like add, modify, delete or even remove the entire file.

Directory files

* These files contain list of file names and other information related to these files.

Special files

* These files are also known as device files.
* These files represent physical device like disks, terminals, printers, networks, tape drive etc.

These files are of two types −

* **Character special files** − data is handled character by character as in case of terminals or printers.
* **Block special files** − data is handled in blocks as in the case of disks and tapes.

File Access Mechanisms

File access mechanism refers to the manner in which the records of a file may be accessed. There are several ways to access files −

* Sequential access
* Direct/Random access
* Indexed sequential access

Sequential access

* A sequential access is that in which the records are accessed in some sequence, i.e., the information in the file is processed in order, one record after the other. This access method is the most primitive one. Example: Compilers usually access files in this fashion.

Direct/Random access

* Random access file organization provides, accessing the records directly.
* Each record has its own address on the file with by the help of which it can be directly accessed for reading or writing.
* The records need not be in any sequence within the file and they need not be in adjacent locations on the storage medium.

Indexed sequential access

* This mechanism is built up on base of sequential access.
* An index is created for each file which contains pointers to various blocks.
* Index is searched sequentially and its pointer is used to access the file directly.

Space Allocation

Files are allocated disk spaces by operating system. Operating systems deploy following three main ways to allocate disk space to files.

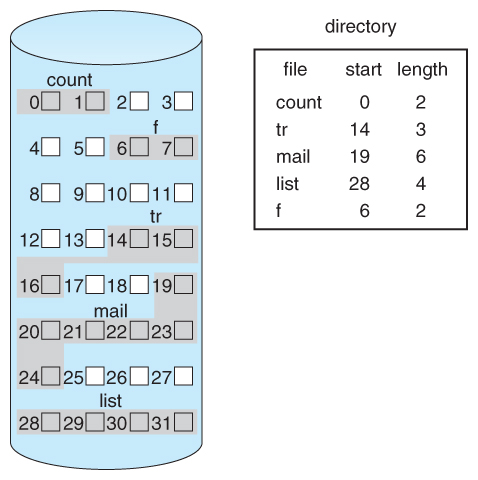
* Contiguous Allocation
* Linked Allocation
* Indexed Allocation

### Allocation Methods

* There are three major methods of storing files on disks: contiguous, linked, and indexed.

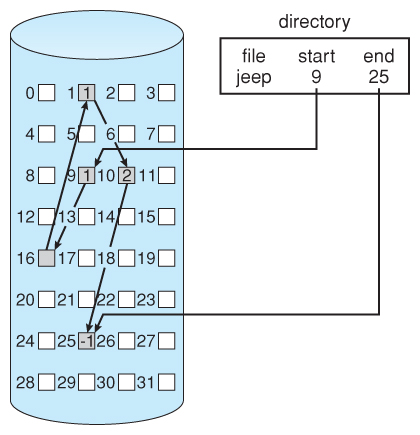
**Contiguous Allocation**

* ***Contiguous Allocation***requires that all blocks of a file be kept together contiguously.
* Performance is very fast, because reading successive blocks of the same file generally requires no movement of the disk heads, or at most one small step to the next adjacent cylinder.
* Storage allocation involves the same issues discussed earlier for the allocation of contiguous blocks of memory ( first fit, best fit, fragmentation problems, etc. ) The distinction is that the high time penalty required for moving the disk heads from spot to spot may now justify the benefits of keeping files contiguously when possible.
* ( Even file systems that do not by default store files contiguously can benefit from certain utilities that compact the disk and make all files contiguous in the process. )
* Problems can arise when files grow, or if the exact size of a file is unknown at creation time:
  + Over-estimation of the file's final size increases external fragmentation and wastes disk space.
  + Under-estimation may require that a file be moved or a process aborted if the file grows beyond its originally allocated space.
  + If a file grows slowly over a long time period and the total final space must be allocated initially, then a lot of space becomes unusable before the file fills the space.
* A variation is to allocate file space in large contiguous chunks, called ***extents.***When a file outgrows its original extent, then an additional one is allocated. ( For example an extent may be the size of a complete track or even cylinder, aligned on an appropriate track or cylinder boundary. ) The high-performance files system Veritas uses extents to optimize performance.

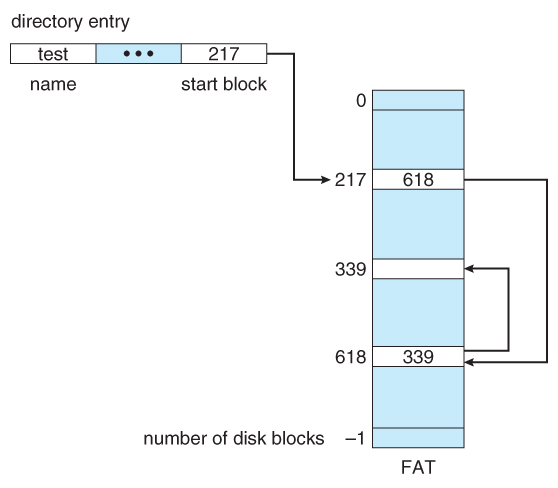
  
**Figure 12.5 - Contiguous allocation of disk space.**

**Linked Allocation**

* Disk files can be stored as linked lists, with the expense of the storage space consumed by each link. ( E.g. a block may be 508 bytes instead of 512. )
* Linked allocation involves no external fragmentation, does not require pre-known file sizes, and allows files to grow dynamically at any time.
* Unfortunately linked allocation is only efficient for sequential access files, as random access requires starting at the beginning of the list for each new location access.
* Allocating ***clusters*** of blocks reduces the space wasted by pointers, at the cost of internal fragmentation.
* Another big problem with linked allocation is reliability if a pointer is lost or damaged. Doubly linked lists provide some protection, at the cost of additional overhead and wasted space.

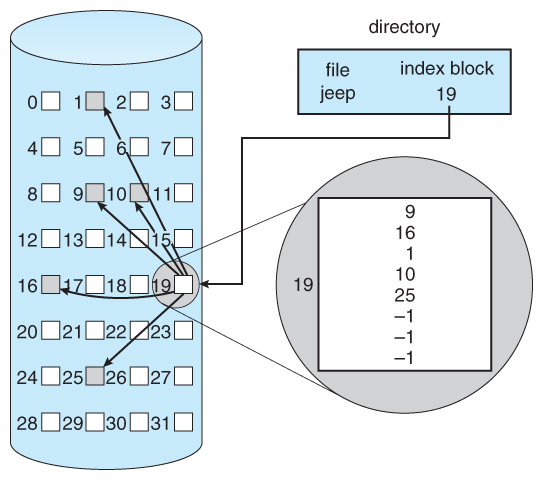
  
**Figure 12.6 - Linked allocation of disk space.**

* The ***File Allocation Table, FAT,***used by DOS is a variation of linked allocation, where all the links are stored in a separate table at the beginning of the disk. The benefit of this approach is that the FAT table can be cached in memory, greatly improving random access speeds.

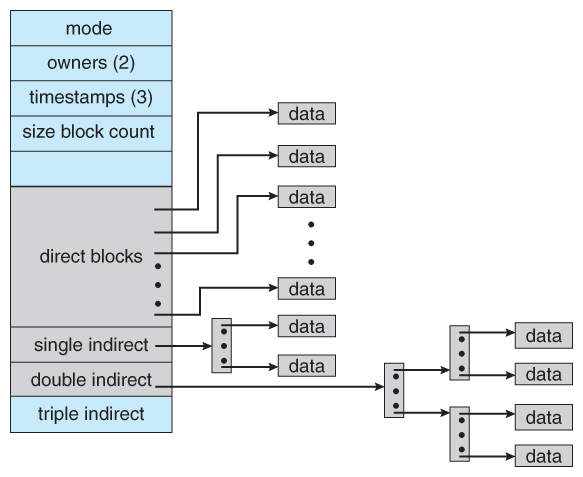
  
**Figure 12.7 File-allocation table**

Indexed Allocation

* ***Indexed Allocation***combines all of the indexes for accessing each file into a common block ( for that file ), as opposed to spreading them all over the disk or storing them in a FAT table.

  
**Figure 12.8 - Indexed allocation of disk space.**

* Some disk space is wasted ( relative to linked lists or FAT tables ) because an entire index block must be allocated for each file, regardless of how many data blocks the file contains. This leads to questions of how big the index block should be, and how it should be implemented. There are several approaches:
  + **Linked Scheme -**An index block is one disk block, which can be read and written in a single disk operation. The first index block contains some header information, the first N block addresses, and if necessary a pointer to additional linked index blocks.
  + **Multi-Level Index -**The first index block contains a set of pointers to secondary index blocks, which in turn contain pointers to the actual data blocks.
  + **Combined Scheme -**This is the scheme used in UNIX inodes, in which the first 12 or so data block pointers are stored directly in the inode, and then singly, doubly, and triply indirect pointers provide access to more data blocks as needed. ( See below. ) The advantage of this scheme is that for small files ( which many are ), the data blocks are readily accessible ( up to 48K with 4K block sizes ); files up to about 4144K ( using 4K blocks ) are accessible with only a single indirect block ( which can be cached ), and huge files are still accessible using a relatively small number of disk accesses ( larger in theory than can be addressed by a 32-bit address, which is why some systems have moved to 64-bit file pointers. )



Contiguous Allocation

* Each file occupies a contiguous address space on disk.
* Assigned disk address is in linear order.
* Easy to implement.
* External fragmentation is a major issue with this type of allocation technique.

Linked Allocation

* Each file carries a list of links to disk blocks.
* Directory contains link / pointer to first block of a file.
* No external fragmentation
* Effectively used in sequential access file.
* Inefficient in case of direct access file.

Indexed Allocation

* Provides solutions to problems of contiguous and linked allocation.
* A index block is created having all pointers to files.
* Each file has its own index block which stores the addresses of disk space occupied by the file.
* Directory contains the addresses of index blocks of files.

Security

Security refers to providing a protection system to computer system resources such as CPU, memory, disk, software programs and most importantly data/information stored in the computer system. If a computer program is run by an unauthorized user, then he/she may cause severe damage to computer or data stored in it. So a computer system must be protected against unauthorized access, malicious access to system memory, viruses, worms etc. We're going to discuss following topics in this chapter.

* Authentication
* One Time passwords
* Program Threats
* System Threats
* Computer Security Classifications

Authentication

Authentication refers to identifying each user of the system and associating the executing programs with those users. It is the responsibility of the Operating System to create a protection system which ensures that a user who is running a particular program is authentic. Operating Systems generally identifies/authenticates users using following three ways −

* **Username / Password** − User need to enter a registered username and password with Operating system to login into the system.
* **User card/key** − User need to punch card in card slot, or enter key generated by key generator in option provided by operating system to login into the system.
* **User attribute - fingerprint/ eye retina pattern/ signature** − User need to pass his/her attribute via designated input device used by operating system to login into the system.

One Time passwords

One-time passwords provide additional security along with normal authentication. In One-Time Password system, a unique password is required every time user tries to login into the system. Once a one-time password is used, then it cannot be used again. One-time password are implemented in various ways.

* **Random numbers** − Users are provided cards having numbers printed along with corresponding alphabets. System asks for numbers corresponding to few alphabets randomly chosen.
* **Secret key** − User are provided a hardware device which can create a secret id mapped with user id. System asks for such secret id which is to be generated every time prior to login.
* **Network password** − Some commercial applications send one-time passwords to user on registered mobile/ email which is required to be entered prior to login.

Program Threats

Operating system's processes and kernel do the designated task as instructed. If a user program made these process do malicious tasks, then it is known as **Program Threats**. One of the common example of program threat is a program installed in a computer which can store and send user credentials via network to some hacker. Following is the list of some well-known program threats.

* **Trojan Horse** − Such program traps user login credentials and stores them to send to malicious user who can later on login to computer and can access system resources.
* **Trap Door** − If a program which is designed to work as required, have a security hole in its code and perform illegal action without knowledge of user then it is called to have a trap door.
* **Logic Bomb** − Logic bomb is a situation when a program misbehaves only when certain conditions met otherwise it works as a genuine program. It is harder to detect.
* **Virus** − Virus as name suggest can replicate themselves on computer system. They are highly dangerous and can modify/delete user files, crash systems. A virus is generatlly a small code embedded in a program. As user accesses the program, the virus starts getting embedded in other files/ programs and can make system unusable for user

System Threats

System threats refers to misuse of system services and network connections to put user in trouble. System threats can be used to launch program threats on a complete network called as program attack. System threats creates such an environment that operating system resources/ user files are misused. Following is the list of some well-known system threats.

* **Worm** − Worm is a process which can choked down a system performance by using system resources to extreme levels. A Worm process generates its multiple copies where each copy uses system resources, prevents all other processes to get required resources. Worms processes can even shut down an entire network.
* **Port Scanning** − Port scanning is a mechanism or means by which a hacker can detects system vulnerabilities to make an attack on the system.
* **Denial of Service** − Denial of service attacks normally prevents user to make legitimate use of the system. For example, a user may not be able to use internet if denial of service attacks browser's content settings.

### 