File Systems

COMP755 Advanced Operating Systems

OS Purpose

- An operating system maps a logical concept onto the physical hardware.
- The OS makes disk drives (of various makes and models) with cylinders, heads and sectors look like files with names, security and data streams.

Goals of a File System

- Easily identify files
- Provide secure access
- Allow sharing of data
- Allocate disk space
- Provide rapid access
- Protection against loss of data
- Be hardware transparent

Types of Files

- To the OS, bits are bits, the file type doesn't matter.
- The OS provides low level functionality upon which higher level file structures can be built.
- Basic OS I/O functions are:
  - read a block, write a block, open, close and delete

Logical File Structure

- Most modern systems use a directory tree structure
- A file can be uniquely specified by it's position in the tree, **`/here/there/file.type`**
- Both files and directories have access rights.

- To the OS, directories are just files containing the names of other files.
- Disk space is allocated in units, often called clusters.
- I/O must be done on a sector or cluster basis.

Implementation Requirements

- Locate the data on the disk
- Allocate free space
- Maintain information about the files (access rights, creation dates, backup flags, etc.)

Contiguous Allocation

- Use First fit, Best fit, Next fit, etc.
- Older OS required the administrator to position data on the disk.
- Difficult to expand
- Efficient for reading and writing large data sets.
- Allows for efficient placement to minimize head movement.

Linked Allocation

- A file consists of a linked list of blocks.
- Easy expansion of files.
- Simple to allocate free space.
- Effective for sequential access, but not too good for random access.
- Fragile linked system can be corrupted.
- Used by FAT systems of DOS and Windows

Indexed Allocation

- An index block contains a table of the blocks containing a file.
- Files may require multiple index blocks
- Large files may use multi-level indexing
- Used by Unix and NTFS

Unix File System

- Unix has a tree structured directory system.
- Directories are just files that contain a list of filenames and other directories that are logically contained in the directory.

inodes

- Information about a file or directory is kept in an inode.
- Information in an inode includes:
  - access rights
  - owner
  - file size
  - time stamps
  - use count
  - location of the file on the disk

Locating a file

- To read a file, you have to first read the inode for that file.
- The access rights in the inode are checked to ensure the user is allowed to use the file.
- The location of the file on the disk is in the inode. The OS uses this to read the file data from the disk.

File Allocation Addresses

- The addresses of the first ten blocks of a file are kept in the inode.
- If the file is larger than ten blocks, the inode contains the address of an indirect block.
- The indirect block contains the addresses of the next 256 blocks of the file.

Indirect blocks

- If the file is larger than 10 + 256 blocks, the inode contains the address of a double indirect block.
- The double indirect block contains the address of up to 256 indirect blocks.
- For very large files there is a triple indirect block.

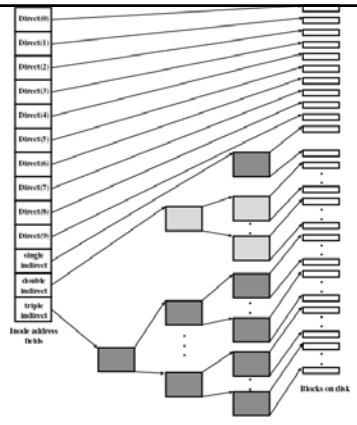

Figure 12.13 UNIX Block Addressing Scheme

Links

- Sometimes it is useful to have a file or subdirectory included in two different directories.
- A link in one directory can point to the file in another directory.
- Microsoft implements this as “shortcuts”.
- Deleting the file under a link can cause consistency problems.