

# What is an Operating System?

An operating system (OS) is a resource manager. It takes the form of a set of software routines that allow users and application programs to access system resources (e.g. the CPU, memory, disks, modems, printers network cards etc.) in a **safe**, **efficient** and **abstract** way.

For example, an OS ensures **safe** access to a printer by allowing only one application program to send data directly to the printer at any one time. An OS encourages **efficient** use of the CPU by suspending programs that are waiting for I/O operations to complete to make way for programs that can use the CPU more productively. An OS also provides convenient **abstractions** (such as files rather than disk locations) which isolate application programmers and users from the details of the underlying hardware.

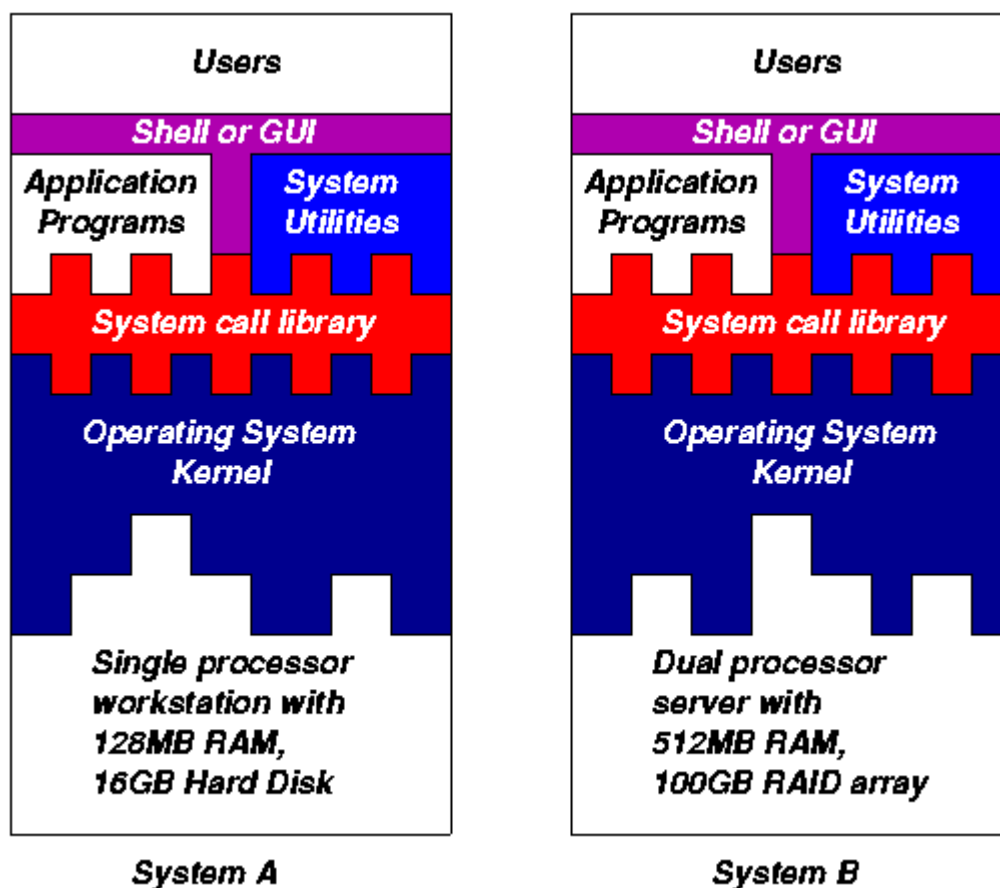


Fig. 1.1: General operating system architecture

Fig. 1.1 presents the architecture of a typical operating system and shows how an OS succeeds in presenting users and application programs with a uniform interface without regard to the details of the underlying hardware. We see that:

- The operating system **kernel** is in direct control of the underlying hardware. The kernel provides low-level device, memory and processor management functions (e.g. dealing with interrupts from hardware devices, sharing the processor among multiple programs, allocating memory for programs etc.)
- Basic hardware-independent kernel services are exposed to higher-level programs through a library of **system calls** (e.g. services to create a file, begin execution of a program, or open a logical network connection to another computer).
- **Application programs** (e.g. word processors, spreadsheets) and **system utility programs** (simple but useful application programs that come with the operating system, e.g. programs which find text inside a group of files) make use of system calls. Applications and system utilities are launched using a **shell** (a textual command line interface) or a **graphical user interface** that provides direct user interaction.

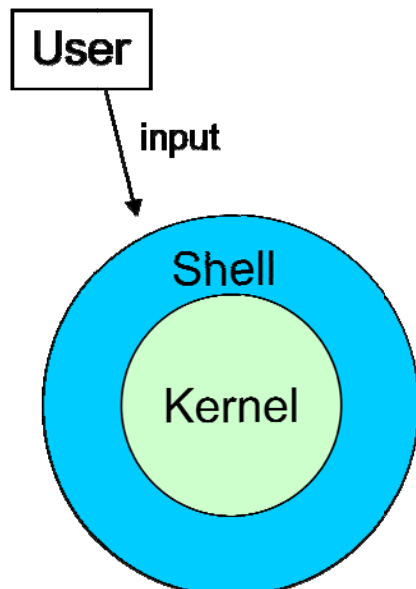
Operating systems (and different flavours of the same operating system) can be distinguished from one another by the system calls, system utilities and user interface they provide, as well as by the resource scheduling policies implemented by the kernel.

- numbers and punctuation.

## Overview of unix System

### Kernel & Shell

- Unix/Linux is operating system (OS).
- Unix system is described as kernel & shell.
- Kernel is a main program of Unix system. it controls hard wares, CPU, memory, hard disk, network card etc.
- Shell is an interface between user and kernel. Shell interprets your input as commands and pass them to kernel.



## Multi-user & Multi-process

- Many people can use one machine at the same time.
- 

## The UNIX Filesystem

The UNIX operating system is built around the concept of a filesystem which is used to store all of the information that constitutes the long-term state of the system. This state includes the operating system kernel itself, the executable files for the commands supported by the operating system, configuration information, temporary workfiles, user data, and various special files that are used to give controlled access to system hardware and operating system functions.

Every item stored in a UNIX filesystem belongs to one of four types:

### 1. **Ordinary files**

Ordinary files can contain text, data, or program information. Files cannot contain other files or directories. Unlike other operating systems, UNIX filenames are not broken into a name part and an extension part (although extensions are still frequently used as a means to classify files). Instead they can contain any keyboard character except for '/' and be up to 256 characters long (note however that characters such as \*, ?, # and & have special meaning in most shells and should not therefore be used in filenames). Putting spaces in filenames also makes them difficult to manipulate - rather use the underscore '\_'.

### 2. **Directories**

Directories are containers or folders that hold files, and other directories.

### 3. **Devices**

To provide applications with easy access to hardware devices, UNIX allows them to be used in much the same way as ordinary files. There are two types of devices in UNIX - **block-oriented** devices which transfer data in blocks (e.g. hard disks) and **character-oriented** devices that transfer data on a byte-by-byte basis (e.g. modems and dumb terminals).

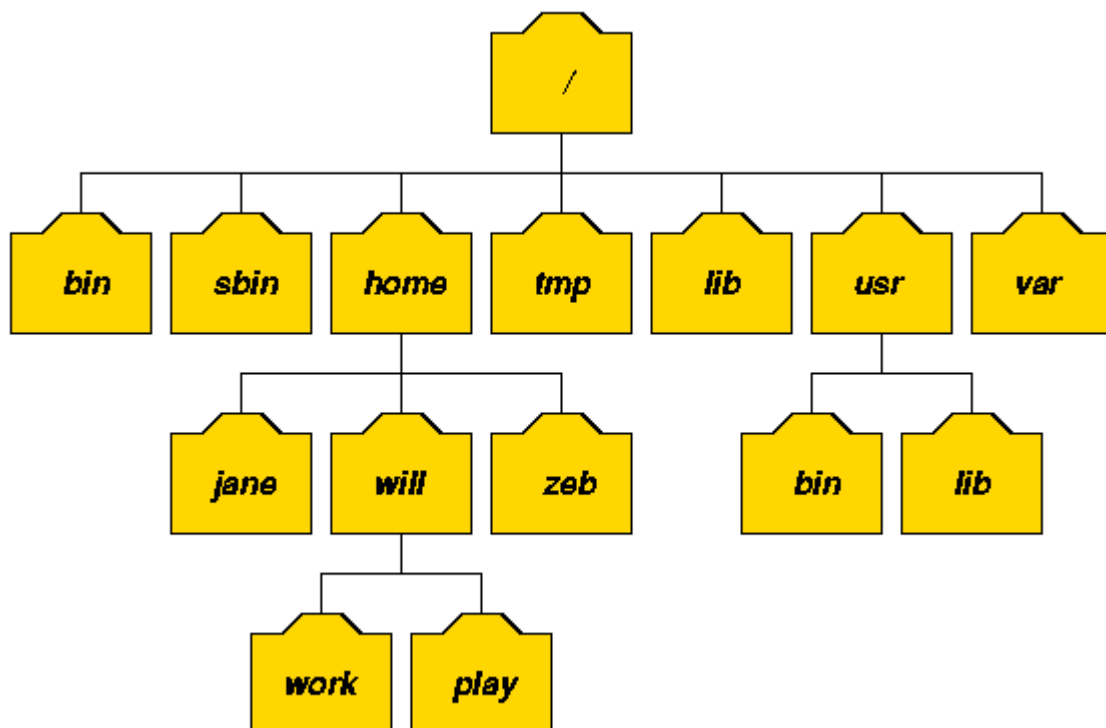
### 4 **Links**

A link is a pointer to another file. There are two types of links - a **hard link** to a file is indistinguishable from the file itself. A **soft link** (or symbolic link) provides an indirect pointer or shortcut to a file. A soft link is implemented as a directory file entry containing a pathname

- numbers and punctuation.

# Directory structure

- Files are put in a directory.
- All directories are in a hierarchical structure (tree structure).
- User can put and remove any directories on the tree.
- Top directory is “/”, which is called slash or root.
- Users have the own directory. (home directory)



To specify a location in the directory hierarchy, we must specify a path through the tree. The path to a location can be defined by an absolute path from the root /, or as a relative path from the current working directory. To specify a path, each directory along the route from the source to the destination must be included in the path, with each directory in the sequence being separated by a slash. To help with the specification of relative paths, UNIX provides the shorthand "." for the current directory and ".." for the parent directory. For example, the absolute path to the directory "play" is `/home/will/play`, while the relative path to this directory from "zeb" is `../will/play`.

- numbers and punctuation.

## Important Directories

- /bin This contains files that are essential for correct operation of the system. These are available for use by all users.
- /home This is where user home directories are stored.
- /var This directory is used to store files which change frequently, and must be available to be written to.
- /etc Various system configuration files are stored here.
- /dev This contains various devices as files, e.g. hard disk, CD-ROM drive, etc.
- /sbin Binaries which are only expected to be used by the super user.
- /tmp Temporary files.

### Normal user and Super user

- In Unix system, there is one special user for administrator, which can do anything.
- This special user is called root or superuser.

### Case Sensitivity

- Unix is case-sensitive.
- MYFILE.doc, Myfile.doc, mYfiLe.Doc are different.

### Online Manual

- Unix has well-written online manuals.

## Logging into (and out of) UNIX Systems


### **Text-based (TTY) terminals:**

When you connect to a UNIX computer remotely (using telnet) or when you log in locally using a text-only terminal, you will see the prompt:

```
login:
```

At this prompt, type in your username and press the enter/return/↵ key. Remember that UNIX is case sensitive (i.e. Will, WILL and will are all different logins). You should then be prompted for your password:

```
login: will
password:
```

Type your password in at the prompt and press the enter/return/  key. Note that your password will not be displayed on the screen as you type it in.

If you mistype your username or password you will get an appropriate message from the computer and you will be presented with the `login:` prompt again. Otherwise you should be presented with a shell prompt which looks something like this:

```
$
```

To log out of a text-based UNIX shell, type "exit" at the shell prompt (or if that doesn't work try "logout"; if that doesn't work press ctrl-d).

## Changing your password

One of the things you should do when you log in for the first time is to change your password.

The UNIX command to change your password is `passwd`:


```
$ passwd 
```

The system will prompt you for your old password, then for your new password. To eliminate any possible typing errors you have made in your new password, it will ask you to reconfirm your new password.

Remember the following points when choosing your password:

- Avoid characters which might not appear on all keyboards, e.g. '£'.
- Make it at least 7 or 8 characters long and try to use a mix of letters, numbers and punctuation.

## General format of UNIX commands

- A UNIX command line consists of the name of a UNIX command followed by its "arguments" (options and the target filenames and/or expressions). The general syntax for a UNIX command is
- ```
$ command -options targets 
```
- 
- numbers and punctuation.

## How to run commands

- Between command name, options and arguments, space is necessary.
- Options always start with “-”
- Example:

`cd ..`

`ls -l .bashrc`

`mv fileA fileB`