Working of an Operating System

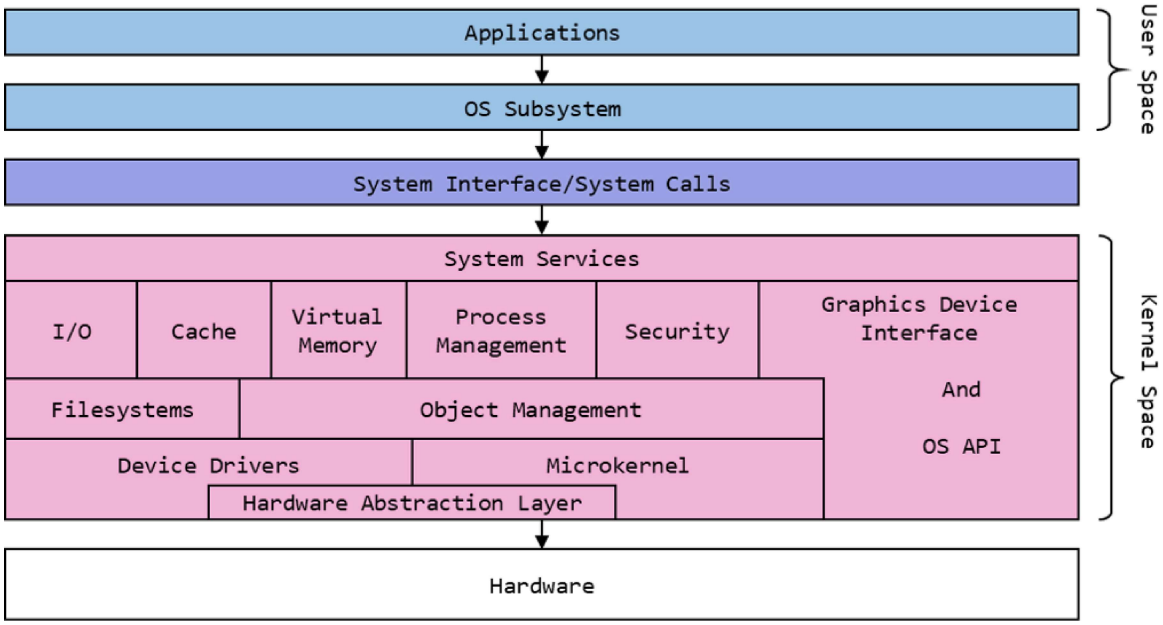
What is an Operating System

- An Operating System is the most important software of a computer, making our tasks simpler by abstracting the hardware from the user.
- Many operating systems have API (Application Programming Interface) to allow developers to easily make applications, and access hardware directly if required.

Functions of an OS

1. **Memory Management:** The OS performs many memory management functions to make sure memory is used effectively and the OS runs smoothly. e.g. Paging, Segmentation, Virtual RAM
2. **Process Management:** The OS manages resources given to each process, to allow all processes to run efficiently.
3. **File Management:** The OS manages the filesystem and files stored on various devices, allowing easy read and write of files.
4. **Device Management:** The OS manages devices connected to the computer, and loads the appropriate drivers for each device.
5. **Security and Access Control:** The OS uses user authentication, and user-based permissions to prevent unauthorized access, and also has anti-virus software to protect against malware.

Parts of an OS



- The specific parts of the OS are outlined in the above diagram.
- The pink shaded region, labelled as **Kernel Space**, refers to the OS's kernel, which forms the base layer of the operating system, allowing it to interface with the hardware of the computer.
- The blue shaded region, labelled as **User Space**, refers to the applications the user is running and the **subsystems of the OS**, which perform different functions to allow the applications to interface with the hardware.
- This communication between the subsystems and the hardware is done using **System Calls.**

Boot process of an OS

Step 1: CPU Initialization

When the CPU receives power, it begins executing the firmware (BIOS/UEFI). The BIOS/UEFI runs Power-On Self Test (POST) to check hardware integrity.

Step 2: Loading and Executing Boot Loader

The firmware locates the bootloader on the storage drive, loads it into memory and executes it. The bootloader then loads the operating system kernel into memory.

Step 3: Kernel Initialization and Startup Services

The kernel initializes system components and hardware drivers, then starts the initial process. The *init* process runs startup scripts and services, allowing the computer to access network, GPU, and other devices/functions.

Step 4: Login Manager and Startup Applications

The login manager displays the login screen to the user. When the user logs in, the OS starts the desktop environment and startup applications.

The User Interface

- The user interface is one of the most important parts of a consumer OS.
- **Graphical User Interface** (GUI) allows the user to easily interact with the computer with a keyboard and a mouse, and perform various tasks, with the output displayed on a screen.
- To have windows displayed on a screen, the OS requires a system software called the *window manager*, and for it to work, the OS requires a windowing server, to communicate with the graphics processor.

Window Manager

- The window manager receives information of what has to be rendered from the application, and renders the window in the appropriate location on the screen.
- It also manages the title bar of the window, moving of windows, closing of windows, minimizing, full-screen, automatic tiling, and virtual desktops.
- The window manager also has a *compositor* to add effects like blur and animations when opening and closing windows.

Desktop Environment

- Apart from the application windows, the user requires interfaces such as the Start Menu, Taskbar, Notifications, Calendar, etc. to work efficiently, the window manager and these elements make up the desktop environment of the Operating System.
- Desktop Environments define the workflow for a user using an operating system, for example, in Windows, to launch an application, a user must open the Start Menu, while on MacOS, they can use the Launchpad, or Spotlight, to find their application and open it.

CLI - Shell

- Though most users do not need to use the terminal on a daily basis, programmers often require it to *configure* their system, *install* applications, or *manage* dependencies for their applications.

- When a user opens the terminal on a computer, they are greeted with a window displaying a prompt.
- This prompt is provided by the **shell** of the operating system, which interprets the commands entered by the user and executes them.
- Most shells have a *scripting language* to run multiple commands programmatically, to automate repeated tasks.
- The shell is a useful tools for developers as it allows to interact with the kernel and perform many functions faster than with a GUI.
- Windows uses *PowerShell* and *cmd*, while Linux and MacOS (Unix) use *bash* or *zsh*, and there are others that can be installed if required.

Major Operating Systems - Windows, MacOS and Linux

Most operating systems running on Non-Embedded Systems can be classified into three types - Windows-*(based?)*, Unix-based, Linux-based.

- Windows, since Windows NT has used the **NT** (New Technology) kernel developed by Microsoft.
- MacOS is an operating system based on the **Unix** kernel.
- Linux refers to the family of different operating systems using the **Linux** Kernel, created by Linus Torvalds. e.g. Ubuntu, Fedora, Arch
- MacOS and Windows are widely used in *consumer computers*, as they are easy to use, reliable, and have great software support.
- Many programmers choose *Linux* based Operating Systems for speed and efficiency, and to customize as per their requirements.
- Linux is also used in IOT, server computing, and super computers, as they have great customizability, allowing it to be adapted to any use case.
- Users can also install a **virtual machine** software like Hyper-V, VirtualBox, or VMware to test out operating systems without making changes to their actual computer.