Reading Reference: Textbook 1 Chapter 1, Textbook 2 Introduction

OPERATING SYSTEMS: DEFINITION, ROLES, GOALS

TANZIR AHMED

CSCE 313, Spring 202

Introduction to OS – Today's Theme

- *Answering "What is an OS?" question*
  - Also, what an OS is not?
- *Challenges faced when developing an OS*

What is an OS? – Virtualization View

- Many programs share the computer simultaneously
- How to make this sharing easy?
  - *Or not a nightmare to the app programmers?*
- Let us consider the following example code

```
#include <iostream>
using namespace std;

int main (int ac, char** av){
   for (int i = 0; i < 10; i++){
     cout<<"hello from "<<av[1]<< endl;
     sleep (1);
   }
}
```

CPU Virtualization

When 3 instances are run together, the following happens:

```
tanzir@cse-buggatti:~$ ./a.out Tanzir & ./a.out Tamanna & ./a.out Maya
[1] 39
[2] 40
hello from Tanzir
hello from Tamanna
hello from Maya
hello from Tanzir
hello from Tamanna
hello from Maya
hello from Tanzir
hello from Tamanna
hello from Maya
hello from Tanzir
hello from Tamanna
hello from Maya
hello from Tanzir
hello from Tamanna
hello from Maya
[1]-  Done                    ./a.out Tanzir
[2]+  Done                    ./a.out Tamanna
```

Memory Virtualization

```c
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
int main (){
    int p = getpid();
    sleep (1);
    printf ("Address: %p, Content: %d\n" , &p, p);
}
```

Every program gets its own ID, even the same program running multiple times

Sleeps for a sec

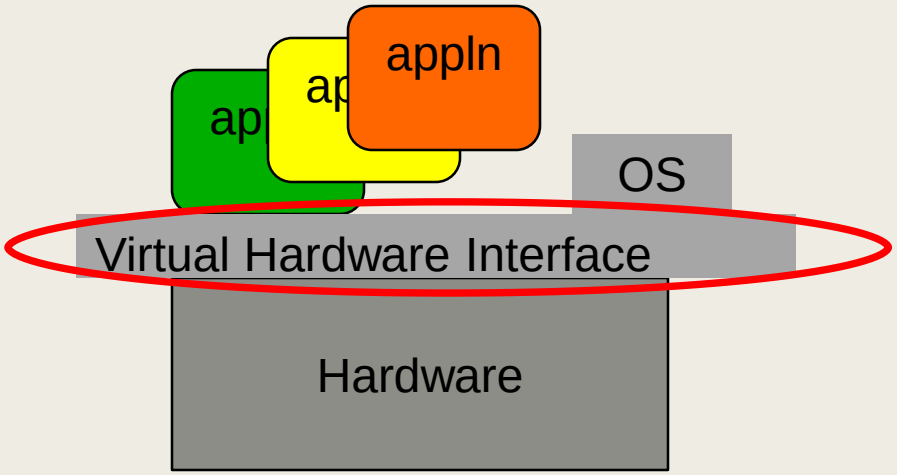
```
osboxes@osboxes:~/Documents$ ./a.out & ./a.out & ./a.out & ./a.out &
[1] 2570
[2] 2571
[3] 2572
[4] 2573
osboxes@osboxes:~/Documents$ Address: 0x7fffffffe084, Content: 2572
Address: 0x7fffffffe084, Content: 2570
Address: 0x7fffffffe084, Content: 2573
Address: 0x7fffffffe084, Content: 2571
```

These are coming from the shell, showing process IDs

Same address, different content

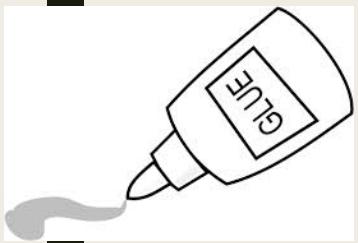
What's the OS's Role?

- In a single word: **Virtualization**
  - *The OS is hiding the physical machine and giving each program a **virtual machine** to play with*
  - *Each program can be agnostic of what else is going on in the System*
    - If apps wanted to know and manage all that, their job would be a lot more difficult



What is an Operating System?

- Referee
  - *Manage sharing of resources, Protection, Isolation*
    - Resource allocation, isolation, communication
- Illusionist
  - *Provide clean, easy-to-use abstractions of physical resources*
    - Infinite memory, dedicated machine
    - Masking limitations, virtualization
- Glue
  - *Common services*
    - Drivers for common devices, File System, Window System, Networking Stack
    - Sharing, Authorization
    - Look and feel



Goals of an Operating System?

- The OS controls and coordinates the use of system resources.

- Primary goal: Provide a convenient environment for a user to access the available resources (CPU, memory, I/O)
  - *Provide appropriate abstractions (files, processes, …)*
  - *"virtual machine"*

- Secondary goal: Efficient operation of the computer system
  - *Make sure that the OS does not kill performance too much*
  - *Efficient Resource Management*
    - **Multiplexing:** Create the illusion of multiple resources from a single resource
    - **Scheduling:** "Who gets the resource when?"

What an operating system is **not**

- *OS is **not** a language or a compiler*
- *OS is **not** a command interpreter / window system*
- *OS is **not** a library of commands*
- *OS is **not** a set of utilities*

OS Challenges

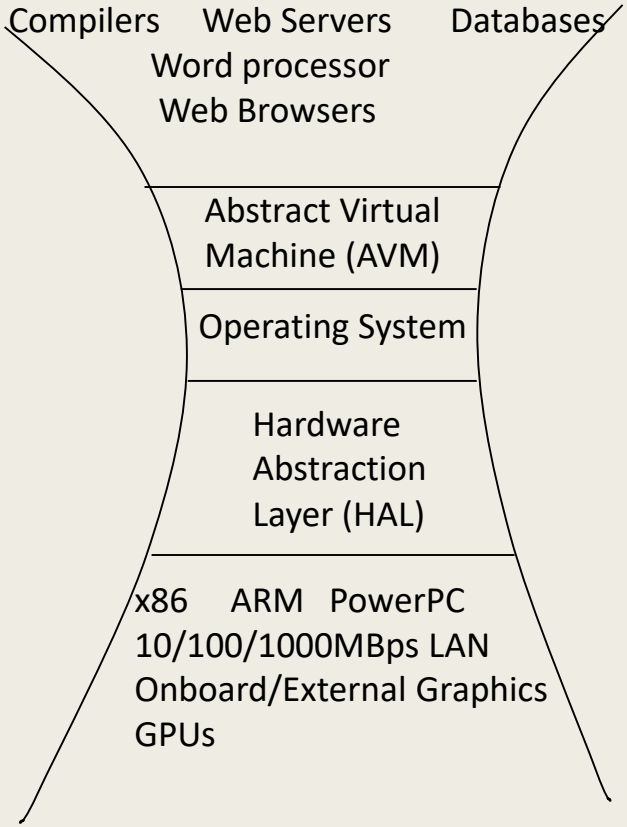
- **Reliability**
  - *Does the system do what it is supposed to do?*
  - *Are all your files consistent after a restart?*
  - *Challenging in a heterogeneous app ecosystem (bugs, viruses etc)*
- **Availability**
  - *What portion of the time is the system working?*
    - Does your system restart all the time?
  - *Can you think of a system that is reliable but not available, or vice versa?*

OS Challenges

- **Security**
  - *Can the system be compromised by an attacker?*
  - *No large system can be completely secure*
    - Bugs
    - Developer backdoors
    - Programs still need to interact (complete fault isolation not possible)
- **Privacy:**
  - *Data is accessible only to authorized users*
    - I cannot access your CSE network drives
- An OS needs both a ***security policy*** and an ***enforcement mechanism***

OS Challenges

- **Portability**
  - *Came from the concept that application programs have to be independent of hardware modification*
    - Example: An app should keep working if you change the graphics card
  - *But portability also applies to app-OS and OS-h/w interface*
  - *Applications change*
    - OS provides Application Programming Interface (API)
  - *Hardware changes*
    - OS keeps working when new devices are added
    - Uses Hardware Abstraction Layer (HAL) to avoid commitment to h/w details



OS Challenges

- **Performance**
  - *Latency/response time*
    - How long does an operation (e.g., 1GB file copy) take to complete?
  - *Throughput*
    - How many operations can be done per unit of time?
  - *Overhead*
    - How much extra work (e.g., for context switching, memory allocation, scheduling) is done by the OS?
  - *Fairness*
    - How equal is the performance received by different users/apps?
  - *Predictability*
    - How consistent is the performance over time?

Challenges in Modern OSs

- Smart Phones
  - *Responsiveness, security, battery life related to CPU cycles*
- Embedded Systems
  - *Reliability, efficiency, security*
- Web Servers
  - *Supporting billions of requests/sec efficiently and safely*
- Virtual Machines
  - *Low overhead and proper h/w virtualization*
- Server Clusters
  - *Hide the clustering details from application programs*

Challenges in Tomorrow's OSs

- Existing challenges would be more critical
  - *OSs controlling future self driving cars, or traffic lights need to have absolutely reliability, security, and efficiency*
- The future of OSs is intertwined with that of emerging computing hardware
  - *Giant-scale data centers*
  - *Increasing numbers of processors per computer*
  - *Newer portable devices*
  - *Very large-scale storage*