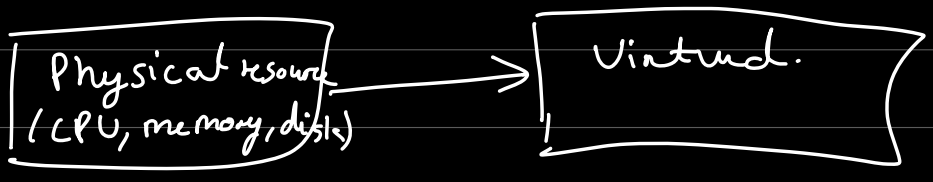
OS three easy pieces

Introduction

- What is OS? - Software making it easy to run programs, share memory, interaction with devices.
  - Early names supervisor or master control program

- How does OS do it?
  - Virtualization



  - Many programs running - sharing CPU
  - Many programs concurrently accessing their own data and instructions - sharing memory
  - Many programs access devices - share disk, ...

- Virtualizing a CPU

Running multiple programs

`./cpu A & ./cpu B & ./cpu C`

(Ref. book for code - simple loop printing input with delay)

Even on a single CPU computer you can't guarantee that programs will in seq. "&" ⇒ triggers process in background and then gives control back, "&&" ⇒ will run seq.

Picking which program to run from paused state depends on something called policy.

- Virtualizing Memory

Each process has its own private virtual address space that the OS maps to physical

memory

- Concurrency

Shared counter updated by multiple threads is a problem since the operation of increment in shown program is not atomic in nature. i.e. 3 instructions - load, increment & store run.

- Persistence

Non-volatile storage ⇒ hard-disk, SSDs

Files ⇒ open, write, close - system calls

Filesystem is responsible to figure out where, how and track the available space on disk etc.

* Design Goals

- High performance - minimize the overheads of the OS

- Protection - Isolating processes

System call | Procedure call

Hardware privilege level ⇒ Kernel mode | User mode

- Special hardware instruction ⇒ "trap" moves user ⇒ kernel mode
  Plus "trap-handler" gets control

- Access to hardware
  * initiate I/O request
  * make more memory available to program
- "return-from-trap" back

to application