Virtualization and Cloud Computing Workshop

bd NOG
Bangladesh Network Operators Group

Days and Topic

Virtualization & Cloud Computing

Workshop:
May 5 - 8, 2018

Venue:
Sheikh Hasina Software Technology Park, Jessore

01 Introduction, Virtualization & Linux Basic

02 KVM, Libvirt, Open vSwitch

03 KVM with LizardFS,LXC

04 LXC with ZFS

Virtualization Primer

NSRC

Virtualization types

- Hardware virtualization
- OS virtualization
- Network virtualization
- Storage virtualization

... we're interested in all of them!

Virtualization vs Emulation

- **Virtualization** is the concept of *dividing* available resources into smaller, independent units
- **Emulation**: using software to *simulate* hardware you do not have
- Complementary concepts
  - *Virtualize* a Server, making it appear as multiple smaller *virtual* machines...
  - Use *Emulation* to simulate individual hard drives, network card, displays, on each *virtual* machine

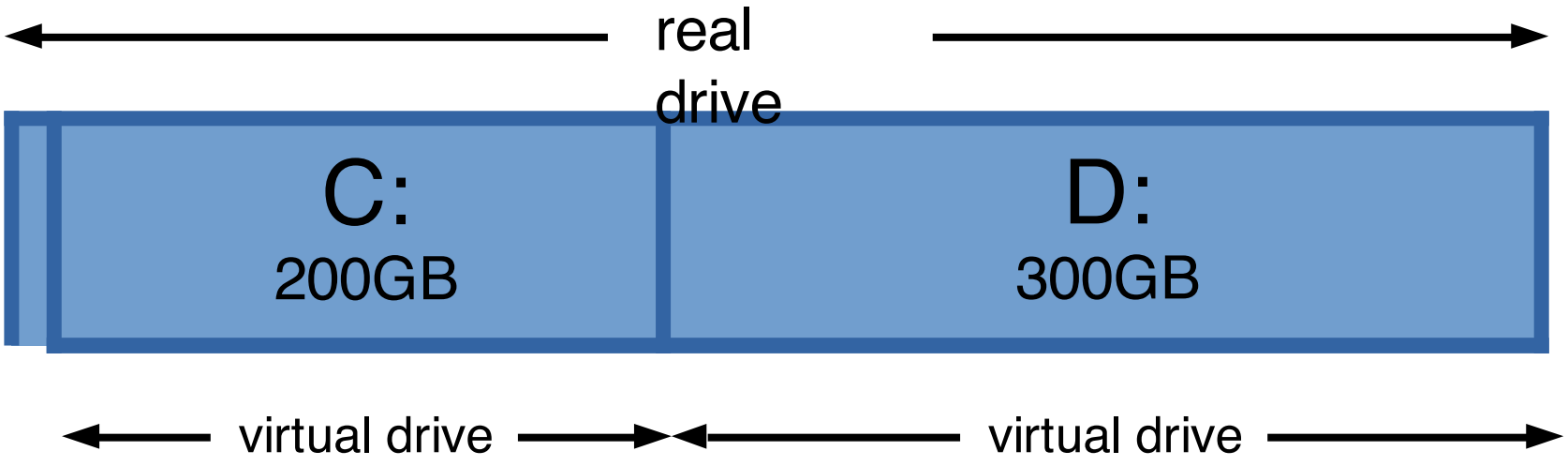
Benefits

- Consolidation
  - Most systems are under-utilized, especially the CPU is idle for much of the time
  - Do more work with less hardware
  - Reduced space and power requirements
- Management
  - Less hardware inventory to manage
  - Concentrate your resilience efforts
  - Increased isolation between services
  - Abstract away (hide) differences in hardware

Benefits

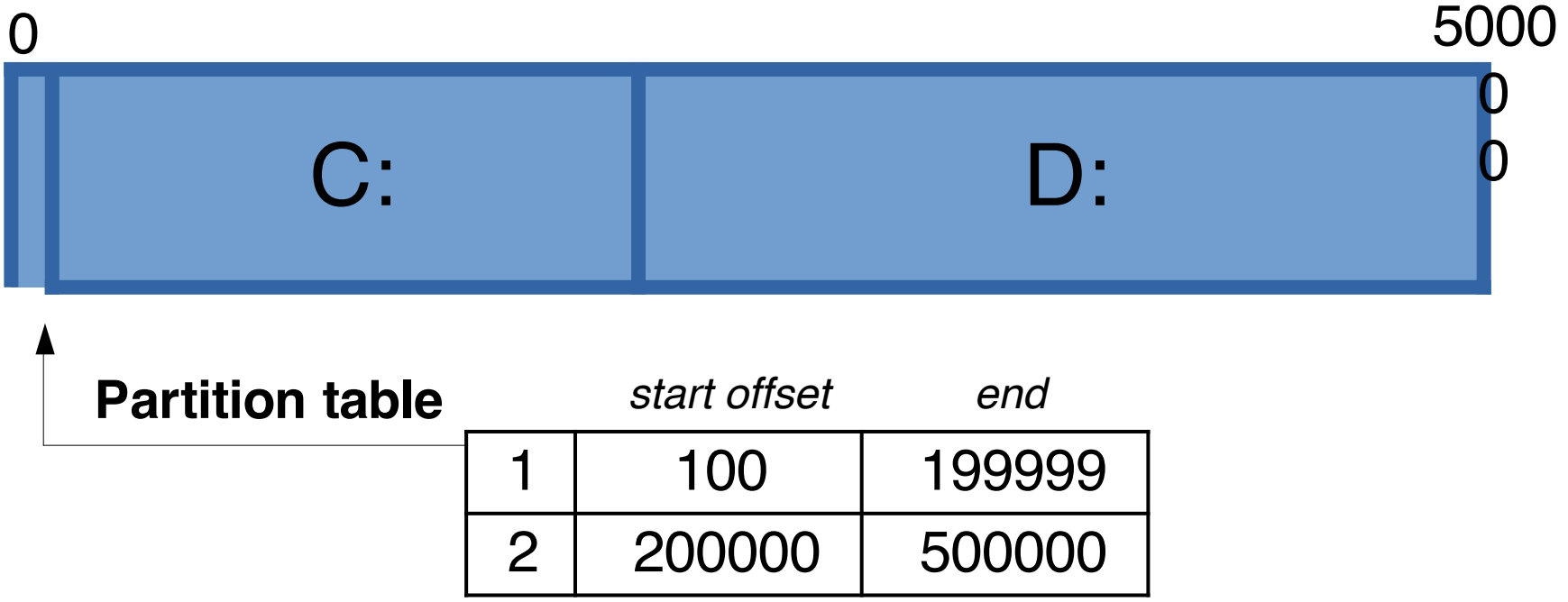
- Flexibility
  - Grow systems on demand (e.g. allocate more CPU or RAM where it is needed)
  - Create new services quickly without having to install new hardware every time
  - Dynamically create and destroy instances for testing and development
- New capabilities
  - Snapshot/restore, cloning, migration, …
  - Run different OSes on the same machine at once

Virtualization: a familiar example



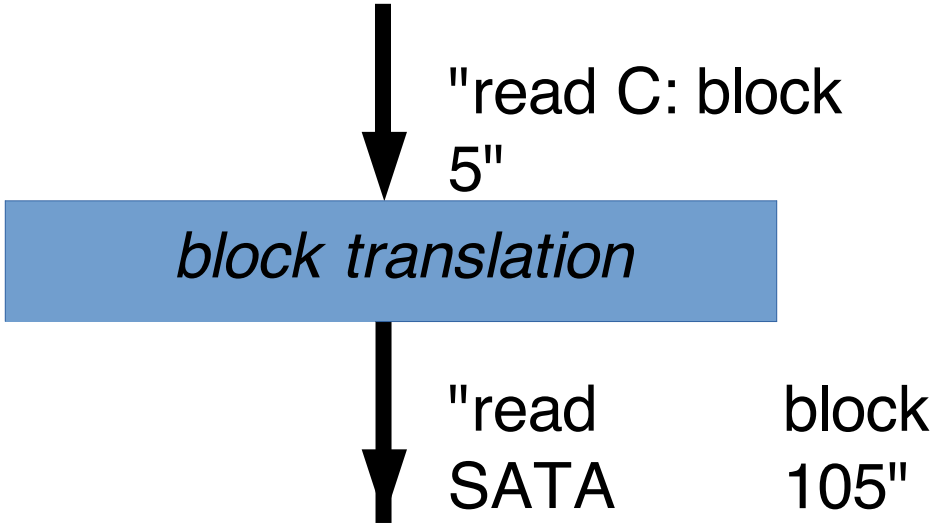
- Who has not seen this before?!
- Like having two (or more) hard drives
  - you get to choose the sizes
- *Why* is this useful?

How does partitioning work?



- Partition table is an example of *metadata*
- When the OS wants to access the Nth block, the real disk access is block (N+offset)

Implementation: translation layer

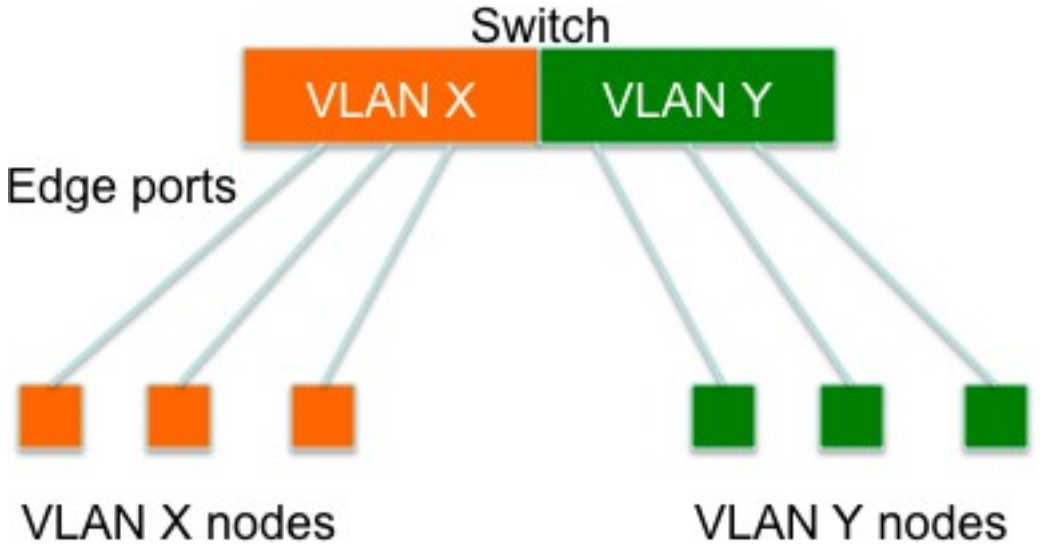


- Very simple and fast: just add offset
- Data is contiguous on disk
- Moving/resizing a partition can require copying all the data on the disk :-(

Another example

- Virtualize a switch: VLANs
  - like dividing a switch into separate switches
- Benefits:
  - can keep traffic separate (broadcast domains)
  - can create VLANs and how they are assigned to ports, purely through software configuration
  - can combine VLANs onto a single cable and split them out again (tagging/trunking)

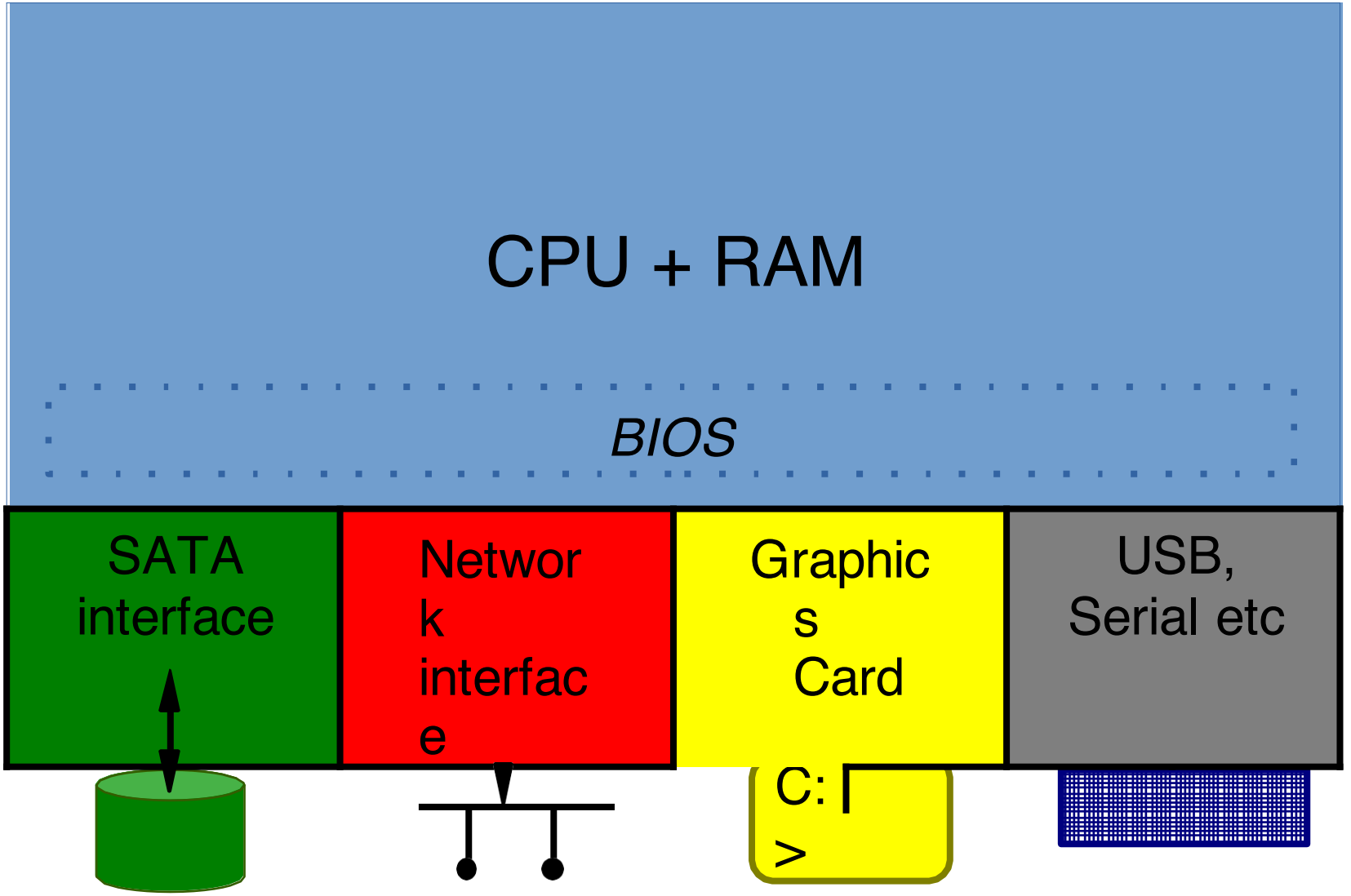
VLANs



Emulation

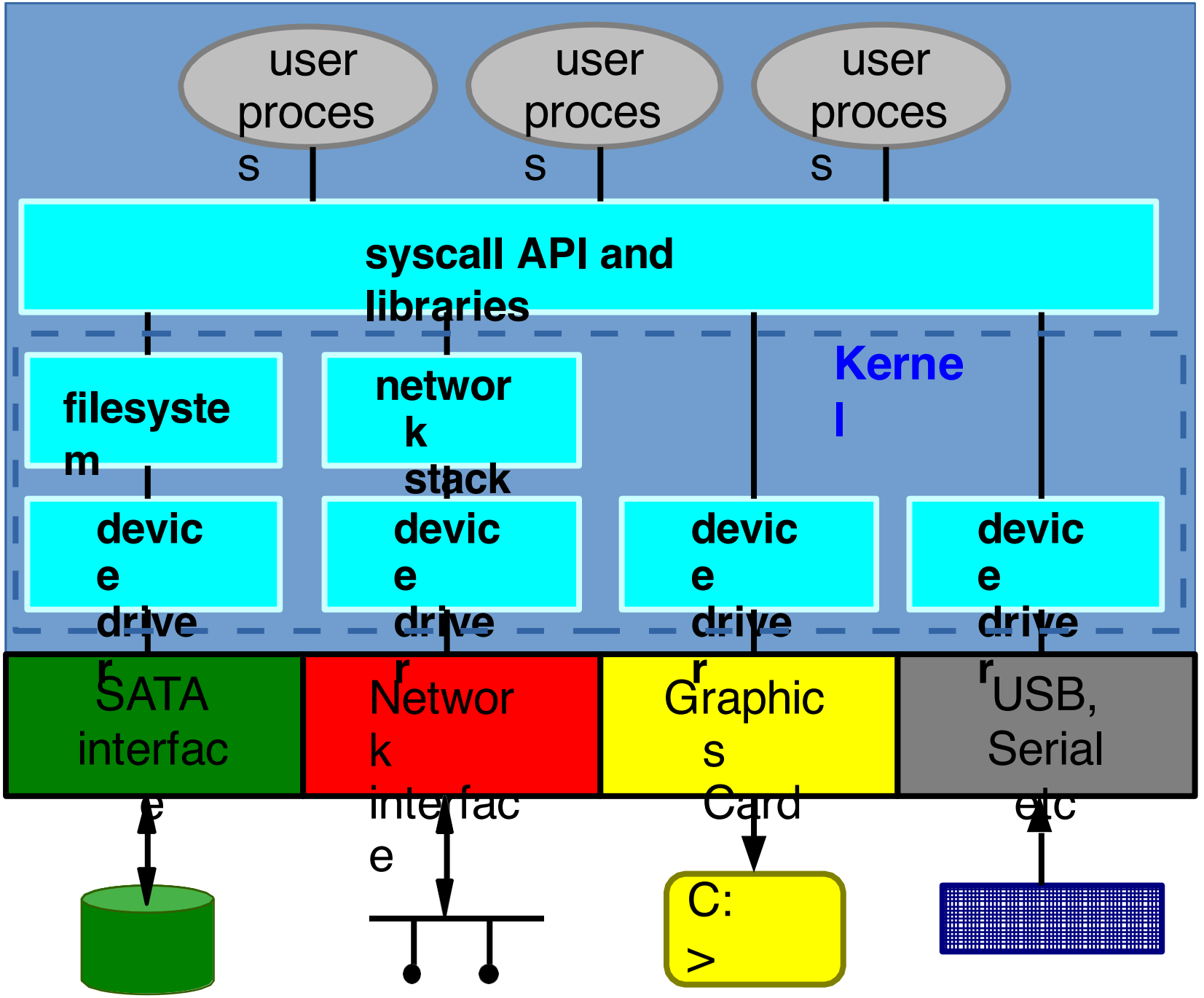
- In software, you can simulate the behaviour of a device which doesn't exist
- Example: emulation of a CD-ROM drive using an ISO file
  - a request to read block N of the (virtual) CD-ROM drive instead reads block N of the ISO file
  - similar to partition mapping
- You can simulate any hardware - including the CPU or an entire system!

What's in a PC?


CPU + RAM
BIOS
SATA interface
Networ k interfac e
Graphic s Card
USB, Serial etc
C:\ >_

Boot up sequence

- A small program (the BIOS) runs when machine is switched on
- It uses the hardware to load an operating system
  - boot from hard drive, USB/CD-ROM, network...
- Modern operating systems then ignore the BIOS from that point onwards
- The next slide shows a machine after it has booted up (simplified)

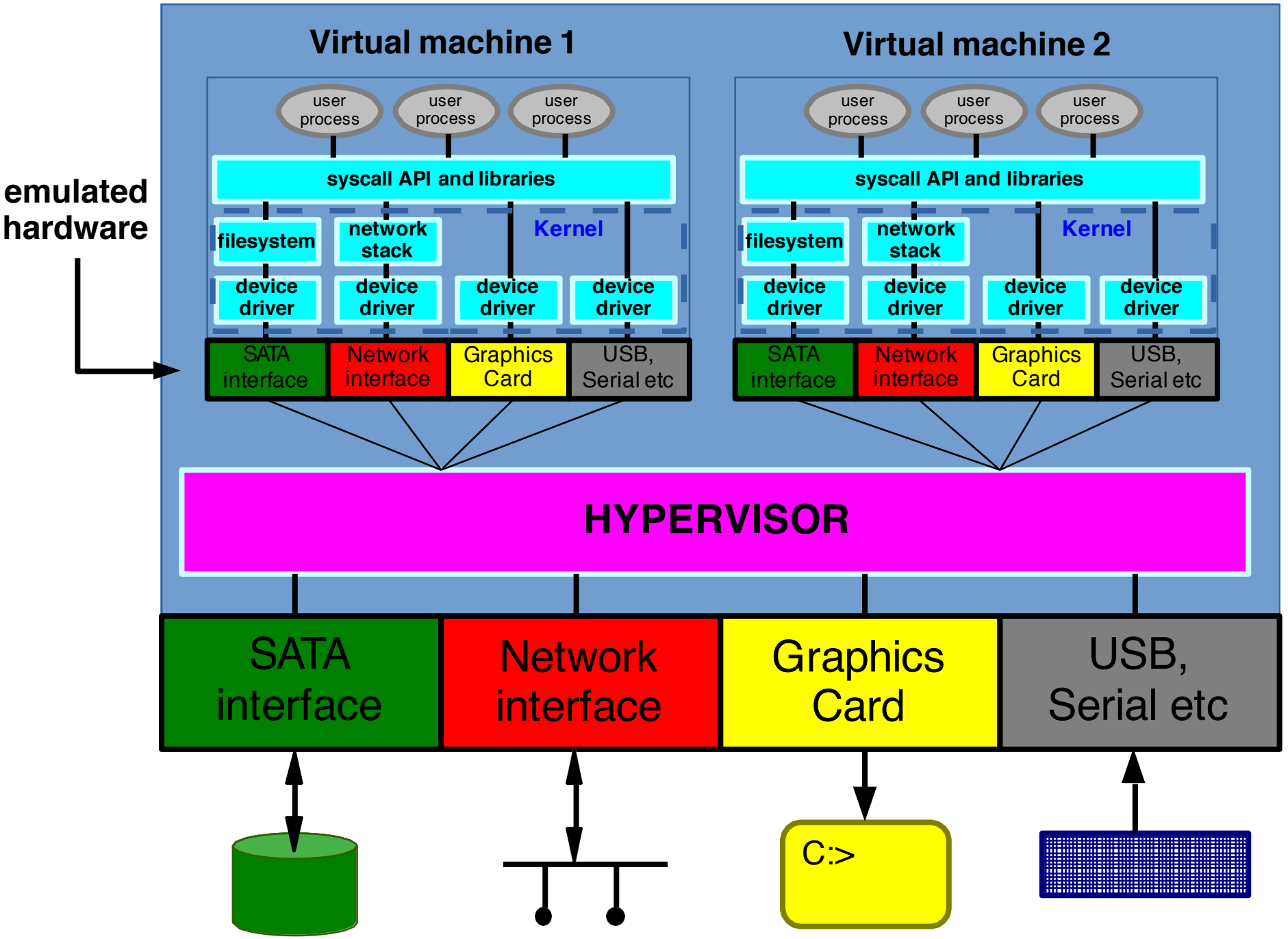

user process
user process
user process
syscall API and libraries
Kernel
filesystem
network stack
device driver
device driver
device driver
device driver
SATA interface
Network interface
Graphics Card
USB, Serial etc
C:>

Points to note

- The device drivers in the OS interact with the hardware
- User processes are forbidden by the OS from interacting directly with the hardware
  - the OS configures protection mechanisms to enforce this

What we need

- To emulate a PC we must emulate all the components of the PC
  - hard disk interface, network card
  - graphics card, keyboard, mouse
  - clock, memory management unit etc
- We want multiple instances to co-exist and not be able to interfere with each other
  - access to memory must also be controlled
- The software to do this is called a *hypervisor*

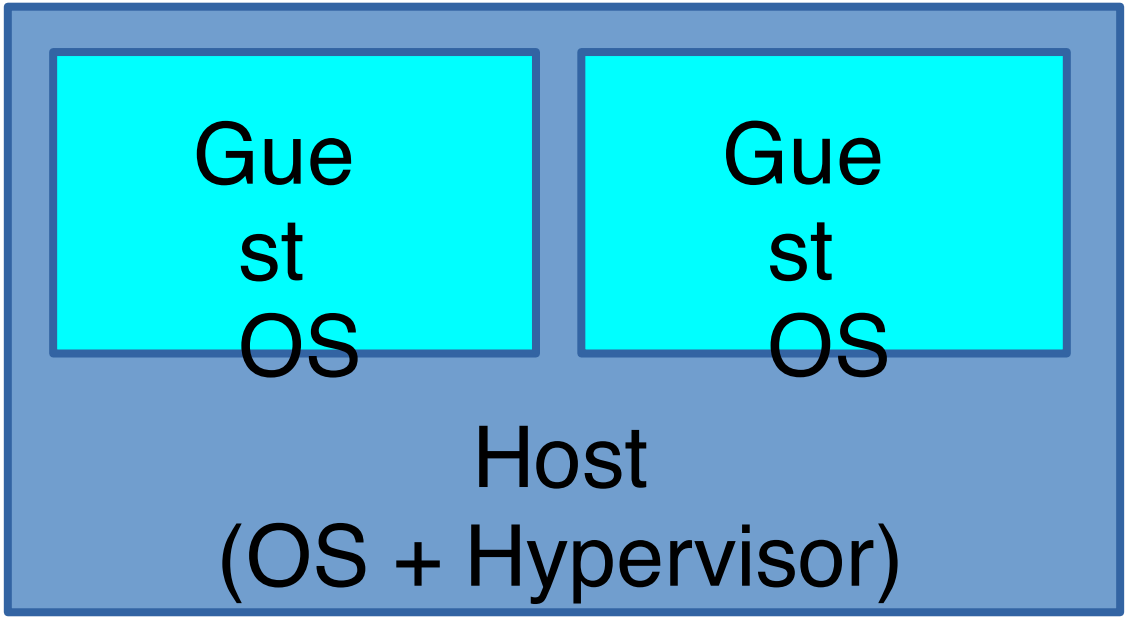

Virtual machine 1
Virtual machine 2
user process
user process
user process
syscall API and libraries
filesystem
network stack
Kernel
device driver
device driver
device driver
device driver
SATA interface
Network interface
Graphics Card
USB, Serial etc
emulated hardware
HYPERVISOR
SATA interface
Network interface
Graphics Card
USB, Serial etc
C:>

Virtual Machines

- Each emulated PC is a "virtual machine"
- Hypervisor allocates some real system RAM to each VM, and shares the CPU time
- Hypervisor emulates other hardware, e.g. disk and network interfaces
- Within each VM you can boot an operating system
- Full hardware virtualization means different VMs can be running different OSes

Virtualization terminology

- The host is the machine running the emulation
- The guest is the emulated (virtual) machine
- One host could be running many guests



The Hypervisor

- Note that the Hypervisor itself is a component of an operating system *
  - It needs device drivers, a filesystem, a network stack for remote management, etc
- So there is a host OS for the hypervisor, plus guest OSes

* Even so-called "bare-metal" or "Type 1" Hypervisors include a cut-down operating system

To wrap up...

- These are the basics of Virtualization
- We'll spend more time later talking about other kinds of virtualization that *don't* require emulating hardware, or booting a *virtual PC* at all. We call this *lightweight* virtualization.

Questions ?