Operating Systems

24. Virtualization

Paul Krzyzanowski

Rutgers University

Spring 2015

Virtualization inside the OS

- Memory virtualization
  - Process feels like it has its own address space
  - Created by MMU, configured by OS
- Storage virtualization
  - Logical view of disks “connected” to a machine
  - External pool of storage
- CPU/Machine virtualization
  - Each process feels like it has its own CPU
  - Created by OS preemption and scheduler

Storage Virtualization

Logical Volume Management

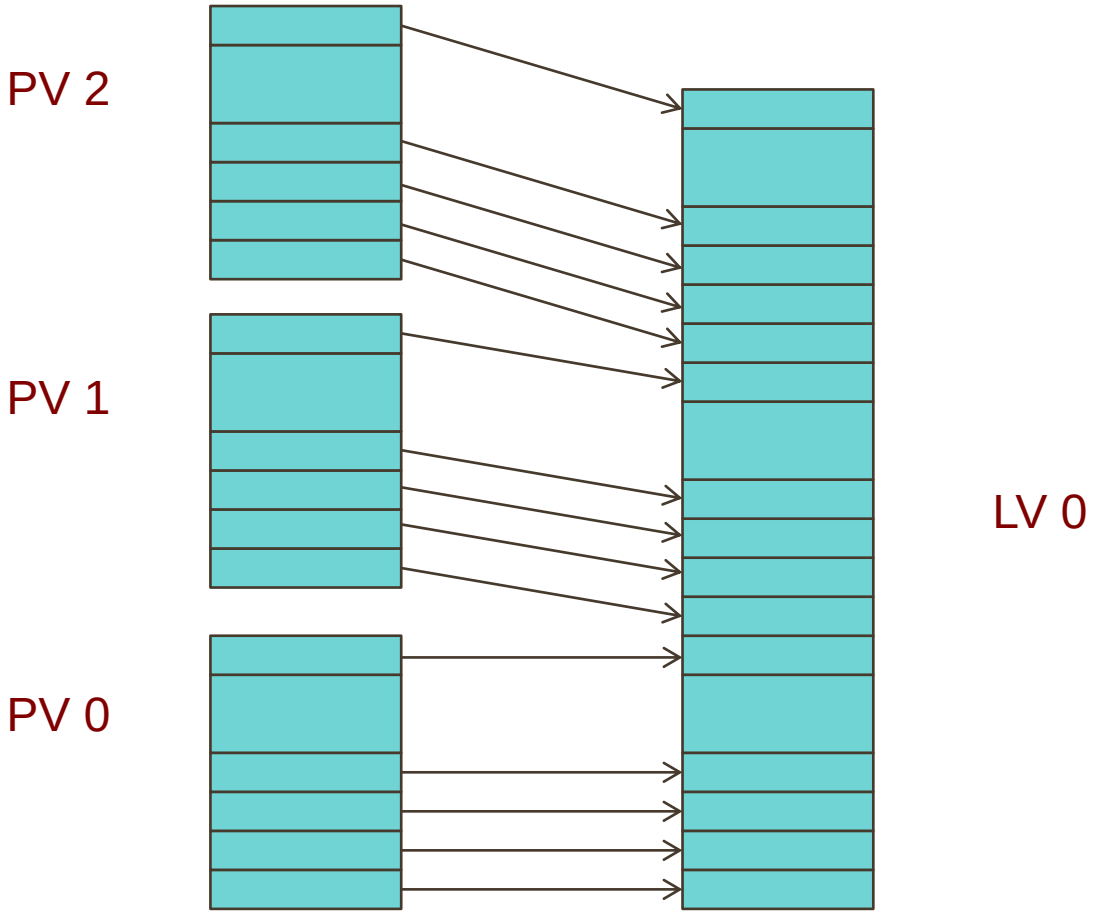
- Physical disk
  - Divided into one or more Physical Volumes
- Logical partitions – *Volume Groups*
  - Created by combining Physical Volumes
    - May span multiple physical disks
  - Can be resized
  - Each can hold a file system

Mapping Logical to Physical data

- Storage on physical volumes is divided into clusters (misnamed *extents*): fixed-size chunks
- Logical volume defined and managed by mapping of logical extents to physical extents
- Logical Volume Manager (LVM) takes care of this mapping

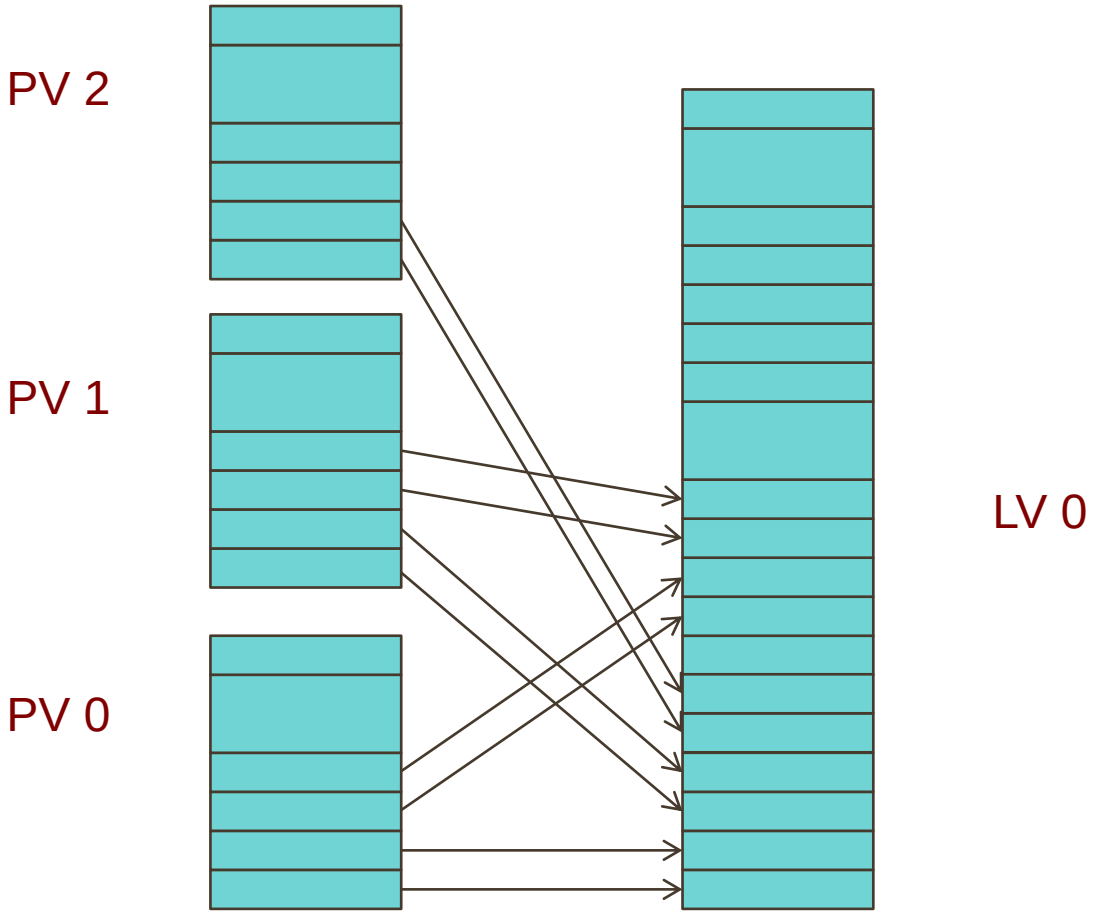
LVM Linear Mapping

Concatenate multiple physical disks to create a larger disk



LVM Striped Mapping

Groups from alternate physical volumes mapped to a logical volume.
N physical extents per stripe. Improve bandwidth of file transfers



Advantages

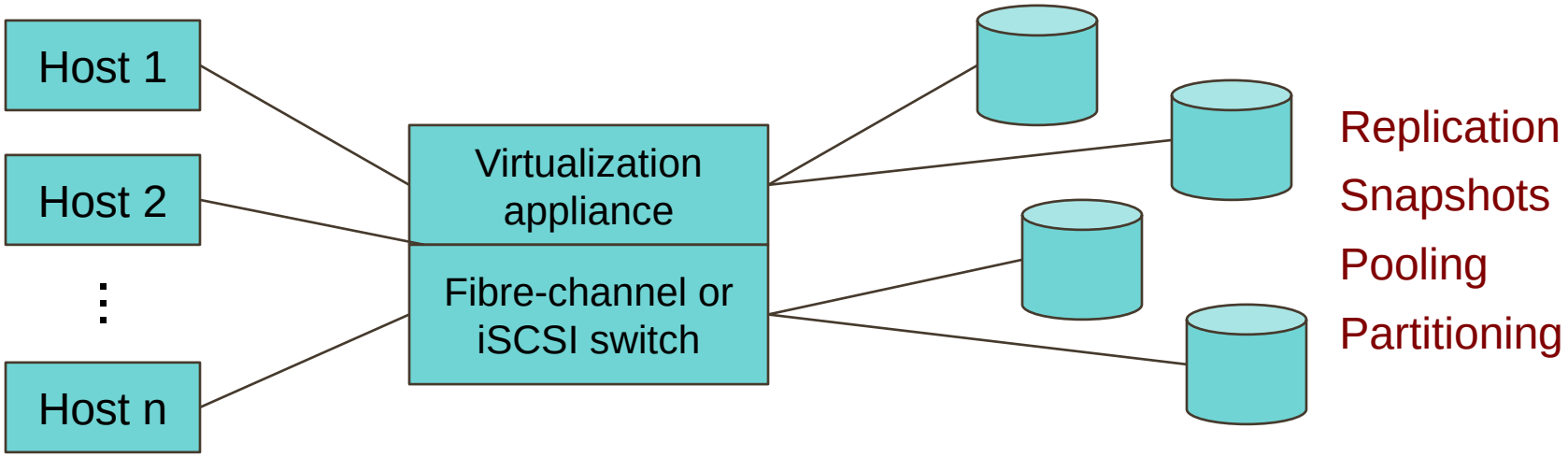
- Logical disks can be resized while mounted
  - Some file systems (e.g., ext3 on Linux or NTFS) support dynamic resizing
- Data can be relocated from one disk to another
- Improved performance (through disk striping)
- Improved redundancy (disk mirroring)
- Snapshots
  - Save the state of the volume at some point in time.
  - Allow backups to proceed while the file system is being modified

Storage Virtualization

- Dissociate knowledge of physical disks
  - The computer system does not manage physical disks
- Software between the computer and the disks manages the view of storage
- Virtualization software translates read-block / write-block requests for logical devices to read-block / write-block requests for physical devices

Storage Virtualization

- Logical view of disks “connected” to a machine
- Separate logical view from physical storage
- External pool of storage



Processor Virtualization

Virtual CPUs (sort of)

What time-sharing operating systems give us

- Each process feels like it has its own CPU & memory
  - But cannot execute privileged instructions (e.g., modify the MMU or the interval timer, halt the processor, access I/O)
- Illusion created by OS preemption, scheduler, and MMU
- User software has to “ask the OS” to do system-related functions.

Process Virtual Machines

- CPU interpreter running as a process
- Pseudo-machine with interpreted instructions
  - 1966: O-code for BCPL
  - 1973: P-code for Pascal
  - 1995: Java Virtual Machine (JIT compilation added)
  - 2002: Microsoft .NET CLR (pre-compilation)
  - 2003: QEMU (dynamic binary translation)
  - 2008: Dalvik VM for Android
  - 2014: Android Runtime (ART) – ahead of time compilation
- Advantage: run anywhere, sandboxing capability
- No ability to even pretend to access the system hardware
  - Just function calls to access system functions
  - Or “generic” hardware

Machine Virtualization

Normally all hardware and I/O managed by one operating system

- Machine virtualization
  - Abstract (virtualize) control of hardware and I/O from the OS
  - Partition a physical computer to act like several real machines
    - Manipulate memory mappings
    - Set system timers
    - Access devices
  - Migrate an entire OS & its applications from one machine to another

- 1972: IBM System 370

Machine Virtualization

An OS is just a bunch of code!

- Privileged vs. unprivileged instructions
- Regular applications use unprivileged instructions
  - Easy to virtualize
- If regular applications execute privileged instructions, they trap
- VM catches the trap and emulates the instruction
  - Trap & Emulate

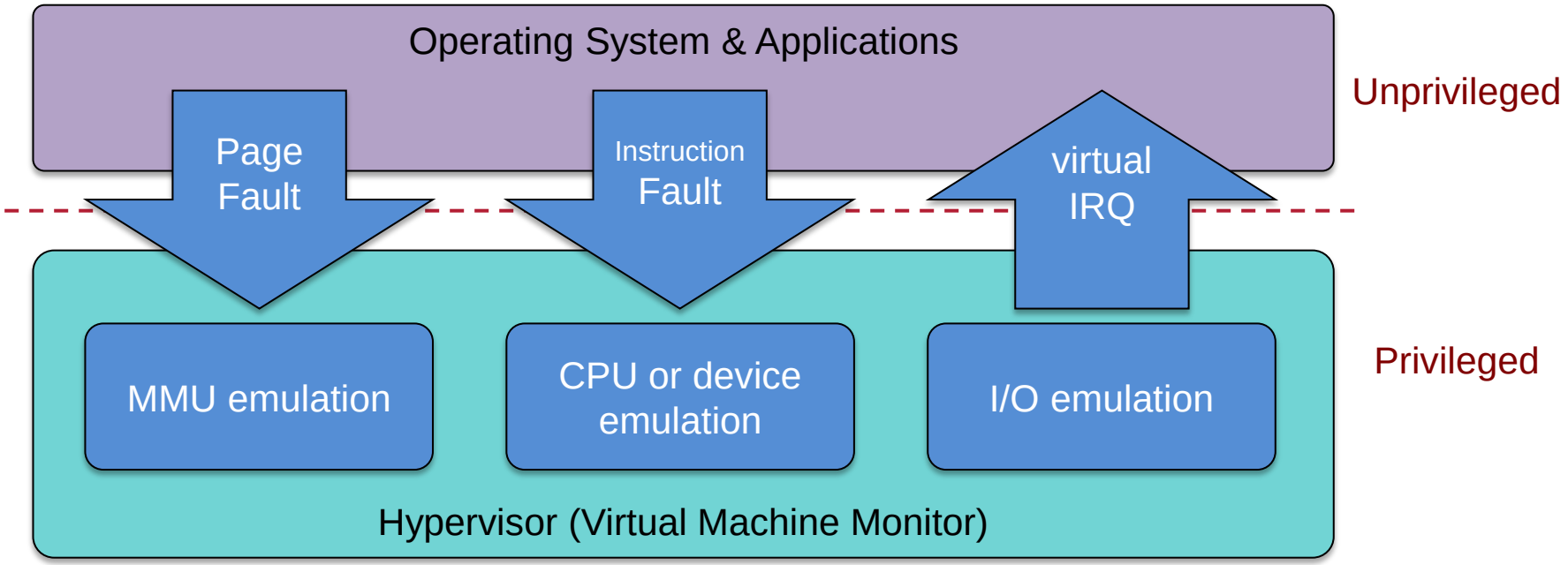
Hypervisor

- Hypervisor: Program in charge of virtualization
  - Aka Virtual Machine Monitor
  - Provides the illusion that the OS has full access to the hardware
  - Arbitrates access to physical resources
  - Presents a set of virtual device interfaces to each host

Hypervisor

Application or Guest OS runs until:

- Privileged instruction traps
- System interrupts
- Exceptions (page faults)
- Explicit call: VMCALL (Intel) or VMMCALL (AMD)



The End