Chapter 33: Virtual Machines

- Virtual Machine Structure
- Virtual Machine Monitor

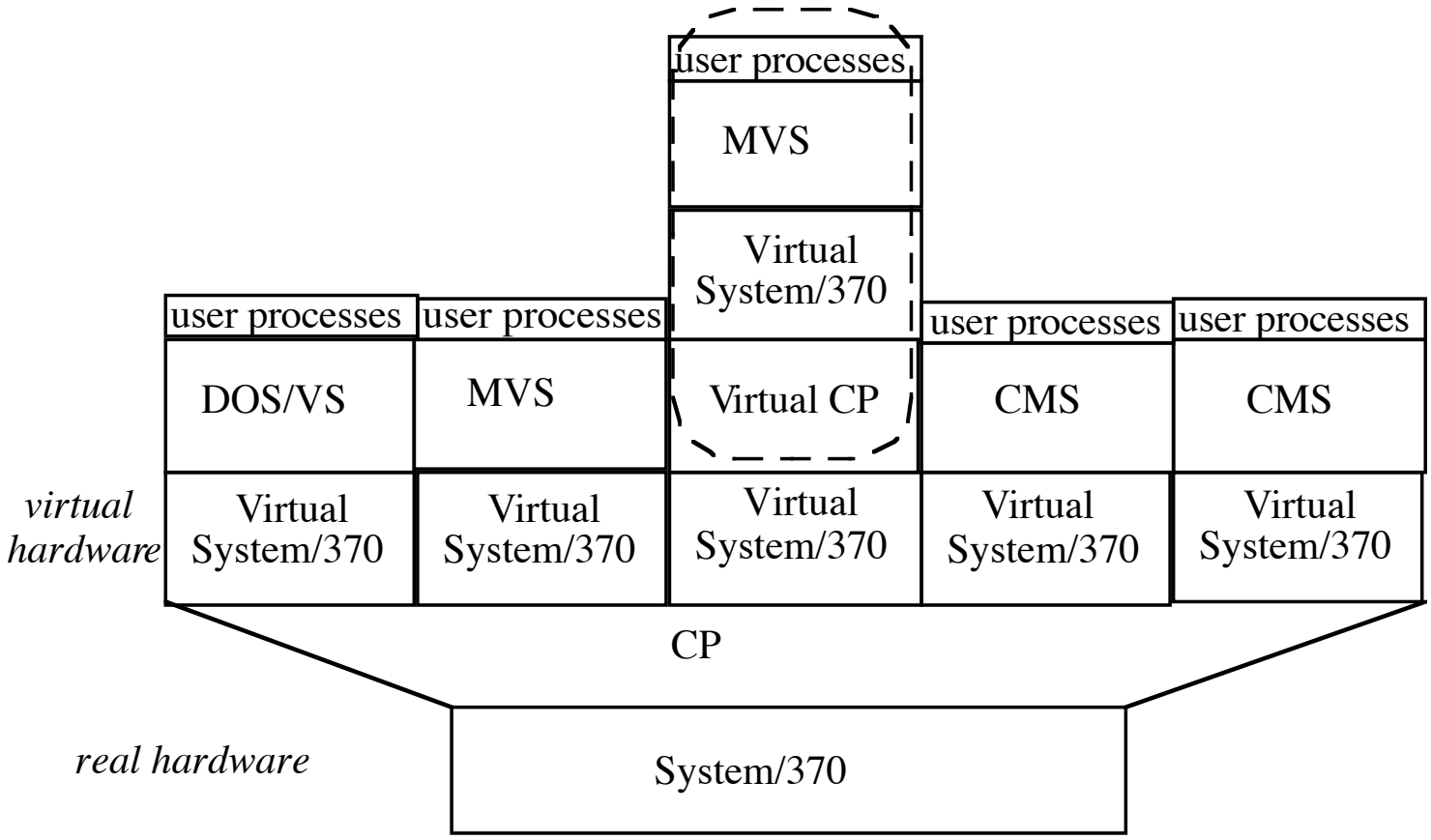
Overview

- Virtual Machine Structure
- Virtual Machine Monitor
  - Privilege
  - Physical Resources
  - Paging

What Is It?

- *Virtual machine monitor* (VMM) virtualizes system resources
  - Runs directly on hardware
  - Provides interface to give each program running on it the illusion that it is the only process on the system and is running directly on hardware
  - Provides illusion of contiguous memory beginning at address 0, a CPU, and secondary storage to *each* program

Example: IBM VM/370



Adapted from Dietel, pp. 606–607

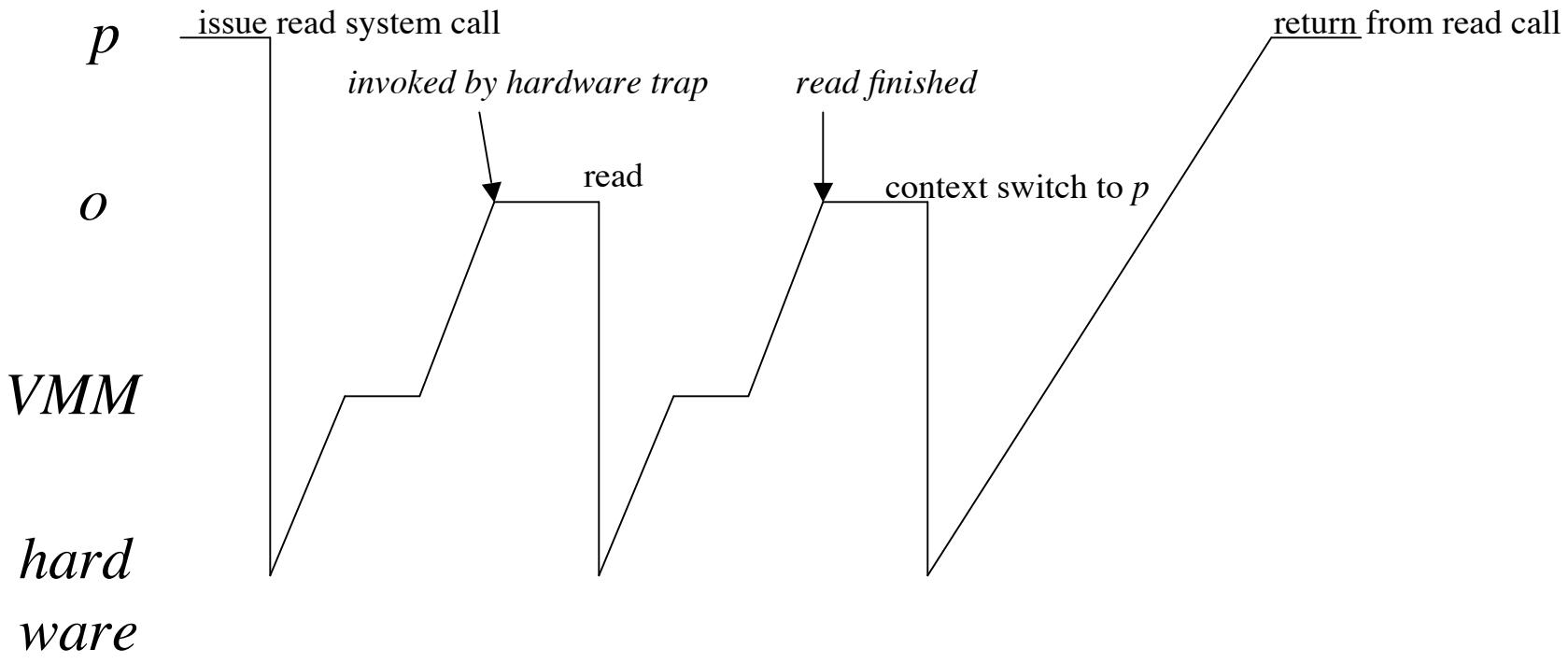
Privileged Instructions

1. VMM running operating system *o*, which is running process *p*
   - *p* tries to read—privileged operation traps to hardware
2. VMM invoked, determines trap occurred in *o*
   - VMM updates state of *o* to make it look like hardware invoked *o* directly, so *o* tries to read, causing trap
3. VMM does read
   - Updates *o* to make it seem like *o* did read
   - Transfers control to *o*

Privileged Instructions

4. *o* tries to switch context to *p*, causing trap
5. VMM updates virtual machine of *o* to make it appear *o* did context switch successfully
   – Transfers control to *o*, which (as *o* apparently did a context switch to *p*) has the effect of returning control to *p*

Privileged Instructions



Privilege and VMs

- *Sensitive instruction* discloses or alters state of processor privilege
- *Sensitive data structure* contains information about state of processor privilege

When Is VM Possible?

- Can virtualize an architecture when:
    1. All sensitive instructions cause traps when executed by processes at lower levels of privilege
    2. All references to sensitive data structures cause traps when executed by processes at lower levels of privilege

Example: VAX System

- 4 levels of privilege (user, supervisor, executive, kernel)
  - CHMK changes privilege to kernel level; sensitive instruction
    - Causes trap *except* when executed in kernel mode; meets rule 1
  - Page tables have copy of PSL, containing privilege level; sensitive data structure
    - If user level processes prevented from altering page tables, trying to do so will cause a trap; this meets rule 2

Multiple Levels of Privilege

- Hardware supports *n* levels of privilege
  - So each VM must appear to do this also
- But only VMM can run at highest level
  - So *n*–1 levels available to each VM
- VMs must virtualize levels of privilege
  - Technique called *ring compression*

Example: VAX/VMM

- VMM must emulate 4 levels of privilege
  - Cannot allow any VM to enter kernel mode, and thereby bypass VMM
  - But VAX/VMS requires all four levels!
- Virtualize executive, kernel privilege levels
  - Conceptually, map both to physical executive level
  - Add VM bit to PSL; if set, current process is on VM
  - VMPSL register records PSL of running VM
  - All sensitive instructions obtain info from VMPSL or trap to VMM, which emulates instruction

Another Approach

- Divide users into different classes
  - Control access to system by limiting access of each class
- Example: IBM VM/370 associates various commands with users
  - Each command associated with *user privilege classes*
    - Class G ("general user") can start VM
    - Class A ("primary system operator") can control system accounting, availability of VMs, etc.
    - Class "Any" can access, relinquish access, to VM

Physical Resources and VMs

- VMM distributes these among VMs as appropriate
- Example: minidisks
  - System to run 10 VMs using one disk
  - Split disk into 10 minidisks
  - VMM handles mapping from (virtual) minidisk address to physical disk address

Example

- VM's OS tries to write to a disk
  - Privileged I/O instruction causes trap to VMM
  - VMM translates address in I/O instruction to address in physical disk
  - VMM checks that physical address in area of disk allocated to the VM making request
    - If not, request fails; error returned to VM
  - VMM services request, returns control to VM

Paging and VM

- Paging on ordinary machines is at highest privilege level
- Paging on VM is at highest virtual level
  - Handled like any other disk I/O
- Two problems:
  - On some machines, some pages available only from highest privilege level, but VM runs at next-to-highest level
  - Performance

First Problem

- VM must change protection level of pages available only from highest privilege level to appropriate level
- Example:
  - On VAX/VMS, kernel mode needed for some pages
  - But VM runs at executive mode, so must ensure only virtual kernel level processes can read those pages
  - In practice, VMS system allows executive mode processes to elevate to kernel mode; no security issue
  - But … executive mode processes on non-VM system cannot read pages, so loss of reliability

Second Problem

- VMM pages: transparent to VMs
- VMs page: VMM handles it as above
  - If lots of VM paging, this may cause significant delay
- Example: IBM VM/370
  - OS/MFT, OS/MVT access disk storage
    - If jobs depend on timings, delays caused by VMM may affect results
  - MVS does that and pages, too
    - Jobs depending on timings could fail under VM/370 that would succeed if run under MVS directly