Virtual Machines
Part 1: 54 years ago
It’s 1964 …

The Beatles appear on the Ed Sullivan show

- IBM wants a multiuser time-sharing system

- TSS project
  - large, monolithic system
  - lots of people working on it
  - for years
  - total, complete flop

- CMS
  - single-user time-sharing system for IBM 360

- CP67
  - virtual machine monitor (VMM)
  - supports multiple virtual IBM 360s

- Put the two together …
  - a (working) multiuser time-sharing system
Virtual Machines

![Diagram of Virtual Machines]

- Applications
  - OSa
  - Virtual Machine
- Applications
  - OSb
  - Virtual Machine
- Applications
  - OSc
  - Virtual Machine

Virtual Machine Monitor

Hardware
Why?

• Structuring technique for a multi-user system
• OS debugging and testing
• Multiple OSes on one machine
• Adapt to hardware changes in software
• Server consolidation and service isolation
User vs. Privileged Mode

• Privileged mode
  – may run all instructions, access all registers
  – for example:
    - modify address translation for virtual memory
    - access and control I/O devices
    - mask and unmask interrupts
    - start and stop system clock

• User mode
  – may run only “innocuous” instructions
  – may access only normal registers
How?

• Approach 1
  – system has “normal” scheduler and virtual memory
  – its processes run in privileged mode
How?

• Approach 2
  – system has “normal” scheduler and virtual memory
  – its processes run an emulator of the real machine
How?

• Approach 3
  – system has “normal” scheduler and virtual memory
  – its processes execute user-mode code directly, but run the emulator when going into privileged mode
How?

• Approach 4
  – system has “normal” scheduler and virtual memory
  – its processes execute non-privileged instructions directly, but emulate privileged instructions
How?
Requirements

• A virtual machine is an efficient, isolated duplicate of real machine
Sensitive Instructions

• Control-sensitive instructions
  – affect the allocation of resources available to the virtual machine
  – change processor mode without causing a trap

• Behavior-sensitive instructions
  – effect of execution depends upon location in real memory or on processor mode
Privileged Instructions

• Cause a fault in user mode
• Work fine in privileged mode
Theorem (!)

- For any conventional third-generation computer, a virtual machine monitor may be constructed if the set of sensitive instructions for that computer is a subset of the set of privileged instructions.
The (Real) 360 Architecture

• Two execution modes
  – supervisor and problem (user)
  – all sensitive instructions are privileged instructions
• Memory is protectable: 2k-byte granularity
• All interrupt vectors and the clock are in first 512 bytes of memory
• I/O done via channel programs in memory, initiated with privileged instructions
• Dynamic address translation (virtual memory) added for Model 67
Real Interrupts and Traps

handler address
handler address
handler address
handler address
handler address
handler address
handler address
handler address
handler address
handler address
Virtual Interrupts and Traps

(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
(handler address)
### Actions on Real 360

<table>
<thead>
<tr>
<th></th>
<th>User mode</th>
<th>Privileged mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>non-sensitive instruction</td>
<td>executes fine</td>
<td>executes fine</td>
</tr>
<tr>
<td>errant instruction</td>
<td>traps to kernel</td>
<td>traps to kernel</td>
</tr>
<tr>
<td>sensitive instruction</td>
<td>traps to kernel</td>
<td>executes fine</td>
</tr>
<tr>
<td>access low memory</td>
<td>traps to kernel</td>
<td>executes fine</td>
</tr>
</tbody>
</table>
# Actions on Virtual 360

<table>
<thead>
<tr>
<th></th>
<th>User mode</th>
<th>Privileged mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>non-sensitive instruction</td>
<td>executes fine</td>
<td>executes fine</td>
</tr>
<tr>
<td>errant instruction</td>
<td>traps to VMM; VMM causes trap to occur on guest OS</td>
<td>traps to VMM; VMM causes trap to occur on guest OS</td>
</tr>
<tr>
<td>sensitive instruction</td>
<td>traps to VMM; VMM causes trap to occur on guest OS</td>
<td>traps to VMM; VMM verifies and emulates instruction</td>
</tr>
<tr>
<td>access low memory</td>
<td>traps to VMM; VMM causes trap to occur on guest OS</td>
<td>traps to VMM; VMM verifies and emulates/translates access</td>
</tr>
</tbody>
</table>
Quiz

Can a VMM (supporting other virtual machines) run on a virtual machine?

a) yes, no problem
b) it requires some changes to a VMM for it to run on a virtual machine
c) no, can’t be done
Virtual Devices?

- Terminals
  - connecting (real) people
- Networks
  - didn’t exist in the 60s
  - (how did virtual machines communicate?)
- Disk drives
  - CP67 supported “mini disks”
  - extended at Brown into “segment system”
- Interval timer
  - virtual or real?
Coping

- Invent new devices
  - recognized by VMM as not real, but referring to additional functionality
    - e.g., mini disks
- Provide new VM facilities not present on real machine
  - e.g., Brown segment system
  - special instructions on VM to request service from VMM
    - sort of like system calls (supervisor calls on 360), but ...
      - hypervisor calls
        - 360 had an extra, unused privileged instruction
          - the diagnose instruction