Virtual Machine Overview

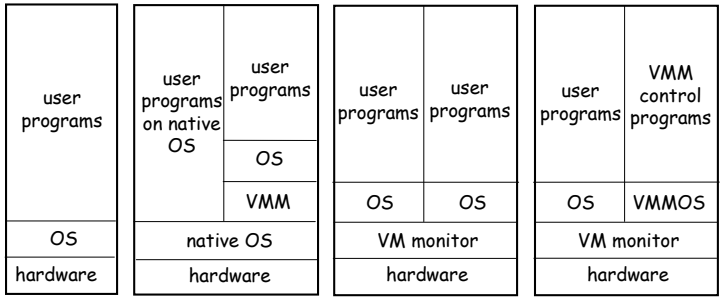
Kai Shen

Dept. of Computer Science, University of Rochester

Virtual Machines

- Virtual machine architecture
  - Virtualization: A piece of software that provides an interface *identical* to the underlying bare hardware.
    - the upper-layer software has the illusion of running directly on hardware
    - the virtualization software is called virtual machine monitor
  - Multiplexing: It may provide several virtualized machines on top of a single piece of hardware.
    - resources of physical computer are shared among the virtual machines
    - each VM has the illusion of owning a complete machine
- Trust and privilege
  - the VM monitor does not trust VMs
  - only the VM monitor runs in full privilege
- Compared to an operating system
  - VM monitor is a resource manager, but not an extended machine

Virtual Machine Architectures

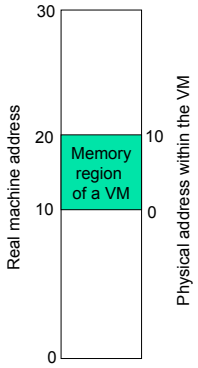


Why Virtual Machine?

- Allow flexible management of "machines" at software level
  - experimenting with new architecture
  - debugging an OS
  - checkpointing and migrating all state on a machine
- Enhanced reliability and security
  - VM monitor much smaller than OS, therefore:
  - the full privileged code base (VM monitor) is small
  - the trusted code base (VM monitor) is small
- Strong isolation between VMs
  - fault and resource isolation

Virtualization Challenges

- CPU virtualization
  - how to switch out/in a VM?
- Memory virtualization
  - VM physical memory address may not be real machine address
  - a VM's memory access must be restricted
- I/O virtualization
  - I/O targets maybe virtual
  - some I/O accesses must be restricted



Virtualization Approach - Interpretation

- Do not directly run VM code ⇒ Interpretation
  - inspect each instruction in software and realize its intended effects using software
  - Nachos VM does this
- CPU virtualization
- Memory virtualization
- I/O virtualization
- Problem: too slow!

Virtualization Approach – Direct Execution

- Directly executing VM code to attain high speed
- CPU virtualization
  - VM monitor catches timer interrupts and switches VM if necessary
- I/O access virtualization
  - cause a trap to VM monitor, which processes appropriately
  - extra overhead is not too bad
- Memory virtualization
  - a trap at each memory access is not a very good idea
  - How?

Memory Virtualization Under Direct Execution (protected page table)

- From the VM OS's view, the page table contains mapping from virtual to VM physical addresses
- For proper operation, the page table hooked up with MMU must map virtual to real machine addresses
- VM OS cannot directly access the page table
  - each page table write is trapped, for a translation (the physical address field is translated from VM

Memory Virtualization Under Direct Execution (shadow page table)

- VM OS maintains virtual to VM physical (V2P) page table
- VM monitor
  - maintains a VM physical to machine (P2M) mapping table
  - combines V2P and P2M table into a virtual to machine mapping table (V2M)
  - supplies the V2M table to the MMU hardware
- Page table updates
  - any VM change on its V2P page table must be trapped by VM monitor
  - VM monitor modifies V2M table appropriately

Virtual Machine Transparency

- Sometimes you want OS work differently from original:
  - for correctness:
    - e.g., non-faulting access to privileged state
  - for performance:
    - e.g., batched multiple monitor traps through a single explicit monitor call
    - e.g., a quick check of whether a monitor call is necessary
- Less-than-full transparency (para-virtualization):
  - modified OS runs within VM
  - Example: Xen
- Full transparency (perfect virtualization):
  - stock OS (without change) can run within VM
  - Example: VMware
  - binary translation for unvirtualizable instructions