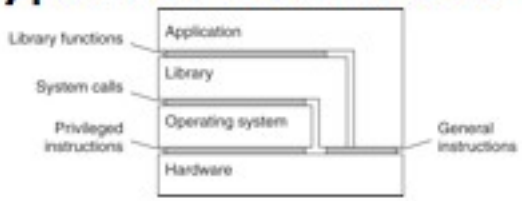
CMPSCI 377: Operating Systems
Guest Lecture: David Irwin

Virtualization and Cloud Computing
Room 142

Virtualization

- Virtualization: extend or replace an existing interface to mimic the behavior of another system.
  - Introduced in 1970s: run legacy software on newer mainframe hardware
- Handle platform diversity by running apps in

Types of Interfaces



- Different types of interfaces
  - Assembly instructions
  - System calls
  - APIs
- Depending on what is replaced / mimiced, we obtain different forms of

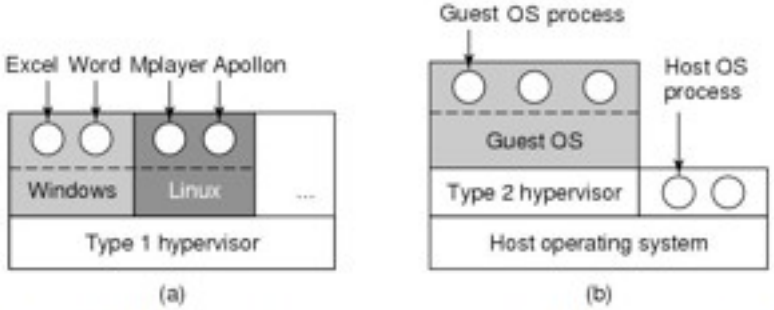
Types of Virtualization

- Emulation
  - VM emulates/simulates complete hardware
  - Unmodified guest OS for a different PC can be run
    - Bochs, VirtualPC for Mac, QEMU
- Full/native Virtualization
  - VM simulates "enough" hardware to allow an unmodified guest OS to be run in

Types of virtualization

- Para-virtualization
  - VM does not simulate hardware
  - Use special API that a modified guest OS must use
  - Hypercalls trapped by the Hypervisor and serviced
  - Xen, VMWare ESX Server
- OS-level virtualization
  - OS allows multiple secure virtual servers to be run
  - Guest OS is the same as the host OS, but appears isolated
    - apps see an isolated OS
  - Solaris Containers, BSD Jails, Linux Vserver
- Application level virtualization
  - Application is gives its own copy of components that are not shared
    - (E.g., own registry files, global objects) - VE prevents conflicts
  - JVM, Rosetta on Mac

Types of Hypervisors



- Type 1: hypervisor runs on "bare metal"
- Type 2: hypervisor runs on a host OS
  - Guest OS runs inside hypervisor
- Both VM types act like real hardware

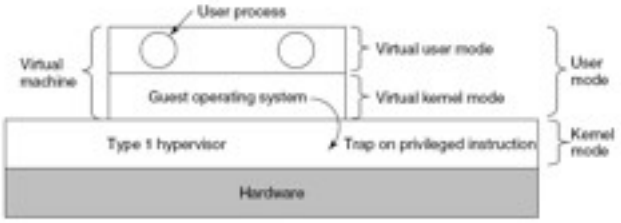
How Virtualization works?

- CPU supports kernel and user mode (ring0, ring3)
  - Set of instructions that can only be executed in kernel mode
    - I/O, change MMU settings etc -- sensitive instructions
  - Privileged instructions: cause a trap when executed in kernel mode
- Result: type 1 virtualization feasible if sensitive instruction subset of privileged instructions
- Intel 386: ignores sensitive instructions in user mode
  - Can not support type 1 virtualization
- Recent Intel/AMD CPUs have hardware support
  - Intel VT, AMD SVM
    - Create containers where a VM and guest can run
    - Hypervisor uses hardware bitmap to specify which inst should trap

x86 virtualization isn't straightforward

- x86 instruction set contains 17 sensitive, unprivileged instructions
  - Sensitive register instructions: read/write sensitive registers and memory locations, e.g.,clock/interrupt registers
    - SGDT, SIDT, SLDT
    - SMSW
    - PUSHF, POPF
  - Protect system instructions, i.e., reference the storage protection system, memory or address relocation system
    - LAR, LSL, VERR, VERW
    - POP
    - PUSH
    - CALL, JMP, INT n, RET

Type 1 hypervisor

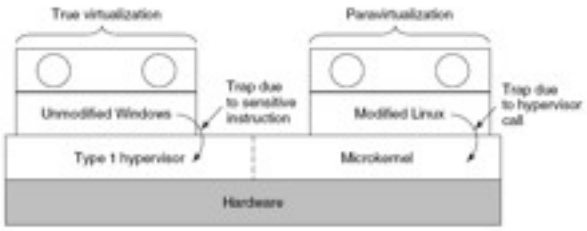


- Unmodified OS is running in user mode (or ring 1)
  - But it thinks it is running in kernel mode (virtual kernel mode)
  - privileged instructions trap; sensitive inst-> use VT to trap
  - Hypervisor is the "real kernel"

Type 2 Hypervisor

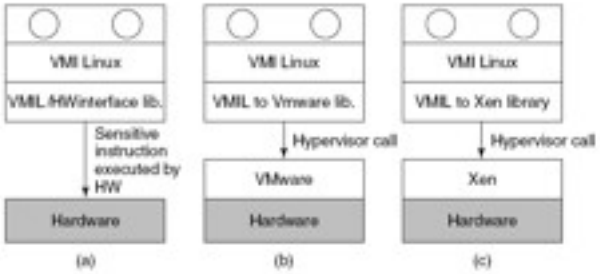
- VMWare example
  - Upon loading program: scans code for basic blocks
  - If sensitive instructions, replace by Vmware procedure
    - Binary translation
  - Cache modified basic block in VMWare cache
    - Execute; load next basic block etc.
- Type 2 hypervisors work without VT

Paravirtualization



- Both type 1 and 2 hypervisors work on unmodified OS
- Paravirtualization: modify OS kernel to replace all sensitive instructions with hypercalls
  - OS behaves like a user program making system calls

Virtual machine Interface



- Standardize the VM interface so kernel can run on bare hardware or any hypervisor

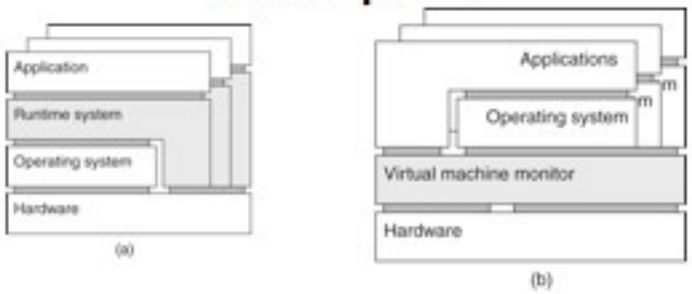
Memory virtualization

- OS manages page tables
  - Create new pagetable is sensitive -> traps to hypervisor
- hypervisor manages multiple OS
  - Need a second shadow page table
    - Virtual → Physical
    - Physical → Machine
  - OS: VM virtual pages to VM's physical pages
  - Hypervisor maps to actual page in shadow page table
  - Two level mapping
  - Need to catch changes to page table (not

I/O Virtualization

- Each guest OS thinks it "owns" the disk
- Hypervisor creates "virtual disks"
  - Large empty files on the physical disk that appear as "disks" to the guest OS
    - Hypervisor converts block # to file offset for I/O
  - DMA need physical addresses
    - Hypervisor needs to translate

Examples



- Application-level virtualization: "process virtual machine"
- VMM /hypervisor

Virtual Appliances & Multi-

- Virtual appliance: pre-configured VM with OS/ apps pre-installed
  - Just download and run (no need to install/ configure)
  - Software distribution using appliances
- Multi-core CPUs
  - Run multiple VMs on multi-core systems
  - Each VM assigned one or more vCPU
  - Mapping from vCPUs to physical CPUs

New Topic: Data Centers & Cloud Computing

- Data Centers
- Cloud Computing

Data Centers

- Large server and storage farms
  - Used by enterprises to run server applications
  - Used by Internet companies
    - Google, Facebook, Youtube, Amazon…
  - Sizes can vary depending on needs

Data Center Architecture

- Traditional: applications run on physical servers
  - Manual mapping of apps to servers
    - Apps can be distributed
    - Storage may be on a SAN or NAS
  - IT admins deal with "change"
- Modern: virtualized data centers
  - App run inside virtual servers; VM mapped onto physical servers
  - Provides flexiblility in mapping from virtual to physical resources

Virtualized Data Centers

- Resource management is simplified
  - Application can be started from preconfigured VM images / appliances
  - Virtualization layer / hypervisor permits resource allocations to be varied dynamically
  - VMs can be migrated without application down-time

Workload Management

- Internet applications => dynamic workloads
- How much capacity to allocate to an application?
  - Incorrect workload estimate: over- or under-provision capacity
  - Major issue for internet facing applications
    - Workload surges / flash crowds cause overloads
    - Long-term incremental growth (workload doubles every few months for many newly popular apps)
  - Traditional approach: IT admins estimate peak workloads and provision sufficient

Dynamic Provisioning

- Track workload and dynamically provision capacity
- Monitor -> Predict -> Provision
- Predictive versus reactive provisioning
  - Predictive: predict future workload and provision
  - Reactive: react whenever capacity falls short of demand
- Traditional data centers: bring up a new server
    - Borrow from Free pool or reclaim under-used server
- Virtualized data center: exploit virtualization

Energy Management in Data Centers

- Energy: major component of operational cost of data centers
  - Large data centers have energy bills of several million $.
  - Where does it come from?
    - Power for servers and cooling
- Data centers also have a large carbon footprint
- How to reduce energy usage?
- Need energy-proportional systems
  - Energy proportionality: energy use proportional to

Energy Management

- Many approaches possible
- Within a server:
  - Shut-down certain components (cores, disks) when idling or at low loads
  - Use DVFS for CPU
- Most effective: shutdown servers you don't need
  - Consolidate workload onto a smaller # of servers
  - Turn others off
- Thermal management: move workload to cooling or move cooling to where workloads are
  - Requires sensors and intelligent cooling systems

Container-based Data

- Modular design
- No expensive buildings needed
- Plug and play: plug power, network, cooling vent

Example: Container DC

- Courtesy: Dan Reed, Microsoft
  - Talk at NSF workshop
- Benefits of MS Gen 4 data ctr
  - Scalable
  - Plug and play
  - Pre-assembled
  - Rapid deployment
  - Reduced construction

Cloud Computing

- Data centers that rent servers/ storage
- Cloud: virtualized data center with self-service web portal
- Any one with a "credit card" can rent servers
- Automated allocation of servers

- Use virtualized architecture

Cloud Models

- Private clouds versus Public Clouds
  - Who owns and runs the infrastructure?

- What is being rented?
  - Infrastructure as a service (rent barebone servers)
  - Platform as a service (google app engine)
  - Software as a service (gmail, online backup, Salesforce.com)

Pricing and Usage Model

- Fine-grain pricing model
  - Rent resources by the hour or by I/O
  - Pay as you go (pay for only what you use)
- Can vary capacity as needed
  - No need to build your own IT infrastructure for peaks needs

Amazon EC2 Case Study

- Virtualized servers
  - Different sizes / instances
- Storage: Simple storage service (S3)
  - Elastic block service (EBS)
- Many other services
  - Simple DB
  - Database service
  - Virtual private cloud