Virtual Machines

Virtual Machines

- A software illusion of another hardware machine
- Virtual servers, virtual RAM, virtual CPU
- Use real hardware to implement the virtual hardware
    - Ex. Instructions for the real CPU to run the virtual CPU

Virtual Machine Benefits

- Fault isolation
- Security
- To use a different OS
- To provide better controlled sharing of the hardware

Virtual Machine Fault Isolation

- OS must never crash
- Crashing a virtual machines operating system
- Correctness requirements can be relaxed
- Similar advantages for faults that could damage devices

Better Security

- OS is supposed to provide security for processes
- OS also provides shared resources, such as the file system and IPC channels
- Virtual machine need not see the real shared resource
- VM in other virtual machines are harder to reach and possibly damaged

Using a different OS

- On Windows you can run Linux
- Windows has one call interface and Linux has another one
    - System calls on Linux won't work on Windows
- If you have a virtual machine running Linux on top of the real machine running Windows

Sharing a machine’s resource

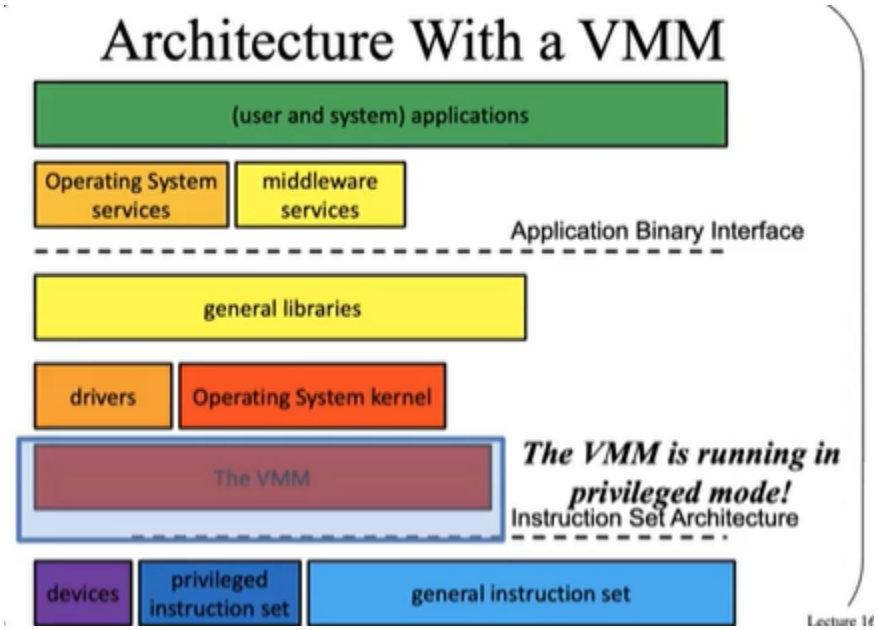
- OS can control how to share resources among processes
- Guaranteeing allocation of resources is hard
- Easier to get the entire VM and get a set allocation of resources
    - The processes running it in doesn’t steal resources from other virtual machines
    - Important for cloud computing

Running Virtual Machines

- Easy if they have the same ISA
- Difficult to do otherwise
    - Use limited direct execution
    - Run as much of the VM directly on the CPU

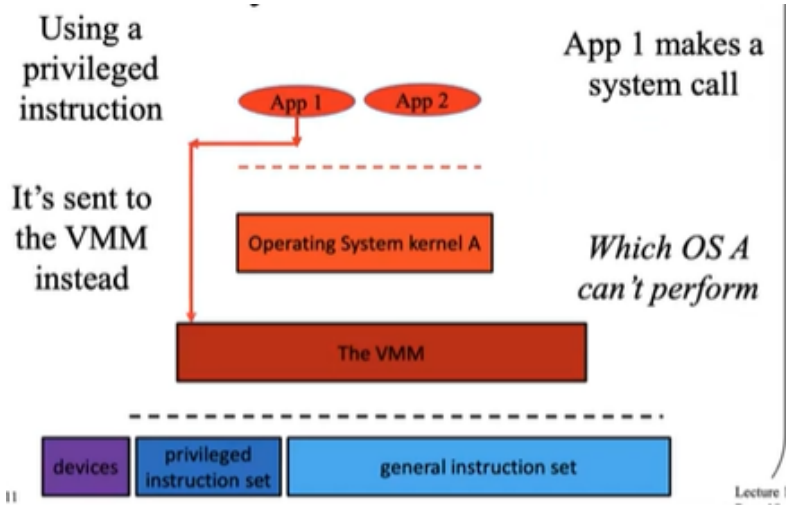
Hypervisor (VMM)

- Controller that handles all virtual machines running on a real machine
- When necessary, trap form the VM to the VMM
    - VMM performs the trap instruction by calling an OS kernel
    - Returns after trap instruction finished, return to limited direct execution
        - Similar to process sys call to OS

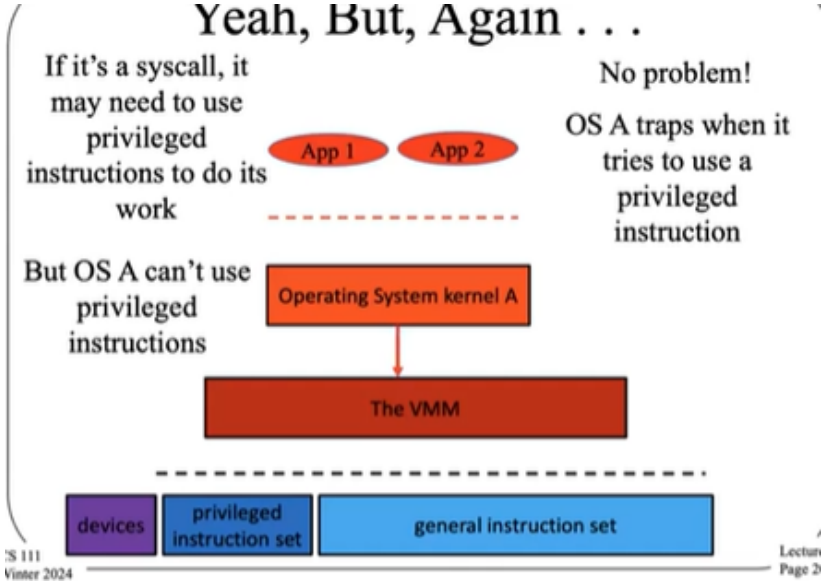


- Look up in the trap table where to run the sys call, where the VMM handles it

Making syscalls on VM



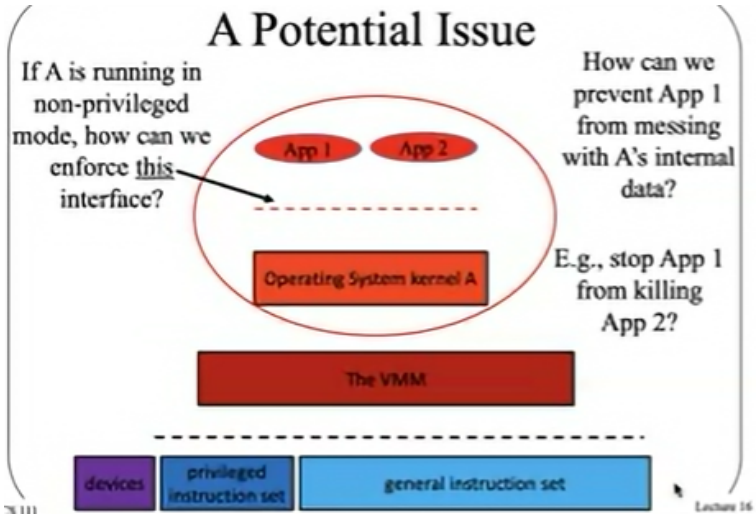
- Trap instructions is a privileged instructions, which the OS kernel can't perform
- VMM performs the trap instruction, which attempts to get the OS kernel to handle it
- OS kernel can't handle syscalls which are privileged instructions, so it returns to the VMM



- VMM performs the privileged instruction then returns to the non privileged OS kernel

VMM privileged instructions

- VMM does not necessarily run the privileged instruction
    - There are other operations that the VMM is also running
- VMM controls what happened
    - Even though the OS thinks the VM is in control



Separation of App and OS

- We don't want the apps to run privileged instructions such as interfering with other apps resources

- OS A thinks its in control and has a segregated virtual memory to App 1 and App 2
- Key tech for doing this is managing page tables and CPU registers
- OS A has no control over registers, but VMM does
- VMM doesn't know anything about the page tables that OS A handles

Virtualizing Memory

- Virtual OS thinks is has physical memory
    - Provides virtual memory addresses to its processes
    - Handles virtual to physical translations

- VMM has machine addresses (genuine location in RAM)
    - Translates physical addresses within a single VM
    - Still using the same paging hardware

VM Syscall Summary

1. TLB miss causes a trap
2. Can't run it on the OS A kernel, so the VMM catches the trap
3. VMM has no instructions on how to execute the trap instructions
    a. VMM has no idea what is in the page table since OS A set up the page table
4. VMM invokes OS A to do the translation
    a. These aren't privileged instructions, so the OS kernel can run it
    b. OS attempts to install the physical page for X into the TLB, but cna perform the privileged operation
5. Traps to VMM, which receives the physical address from OS kernel A
    a. VMM does its own translation and stores the data into the TLB
6. App reruns the instruction that caused the TLB miss

- The "physical" address isn't actually the physical space
- Machine address is the real address (these are the real page frame addresses)

VMM TLB misses

- TLB misses are much more expensive
    - Since we move between privileged and unprivileged mode
    - Results in overhead which takes time
- Need extra paging data structures in the VMM
- Virtual machines suffer from performance

Improving VMs

- Add special hardware
    - Some CPUs have features to make virtualization CPU and memory cheaper

- Paravirtualization
    - Basic VM approach assumes the guest OSes in VMs don't know about virtualization
    - Paravirtualization involves changing the code in the guest OS to make it match the VMM
    - Improvements to OS an make virtualization cheaper

Virtual Machines and Cloud Computing

- Cloud computing is about sharing hardware among multiple customers
- Cloud provider sells/rents computing power to customers
- Cloud providers need a lot of customers, so selling VMs is done

VMs in the Cloud

- Cloud provider benefits from making the most efficient use of the hardware
    - More customers on the same amount of hardware = more profit
- If a customer doesn't use the full power of a machine, then you can give part of it to another customer
- Therefore there needs to be strong isolation within a VM

Solution to VMs on the Cloud

- Everyone runs on a VM
- Customers may have many virtual machines to handle large jobs
- Some customers virtual machines share physical machines with other customer's VMs
- Customer's work loads fluctuate

Efficiently placed VMs

- More physical nodes and many more VMs
- Reduces to a bin packing algorithm
- Tends to be a NP-hard problem

VM Use Cases

- Allow for experimentation not easily performed on real hardware
- Allow basic servers to safely divide their resources
- Allow greater flexibility in the software your computer can run