Virtualization and Cloud Computing

Content taken from the following:
Various online sources; some images from wikipedia.org and openclipart.org
What is "the cloud"?

- A. The world's current fastest supercomputer
- B. The internet of things
- C. An internet service provider
- D. Other people's computers
- E. A novel HPC architecture
Problem

- Distributed systems are now ubiquitous
  - It’s hard to provide any software service at a modern scale from a single server
    - (Although if you can, you SHOULD!)
  - Many companies don’t need/want to manage hardware
    - High up-front costs, security vulnerabilities, etc.
  - Solution: abstraction!
    - In particular, abstracting away the hardware
      - Sometimes software too
    - Usually referred to as virtualization
Virtualization

- **Virtual environment**: abstract machine *(guest)* implemented on top of another (sometimes physical) machine *(host)*
  - Requires some kind of interpretation layer
- **Various goals**:
  - *Emulation*: run programs designed for one architecture on another
  - *Isolation*: run programs in a sandbox
  - *Scalability*: spawn/destroy instances dynamically
  - *Automation*: reduce tedium and mistakes during deployment
  - *Reproducibility*: suspend/resume snapshots or configurations
The Unix Users Group uses a tool called Ansible to more quickly set up and maintain VM installations. What is this an example of?

- A. Emulation
- B. Isolation
- C. Scalability
- D. Automation
- E. Reproducibility
Virtualization

• A PC gamer uses the DOSBox software package to run the original SimCity game on a modern Windows 11 machine. What is this an example of?
  - A. Emulation
  - B. Isolation
  - C. Scalability
  - D. Automation
  - E. Reproducibility
Virtualization

- An online programming contest judge system creates a new, separate environment for every submission to avoid security issues. What is this an example of?
  - A. Emulation
  - B. Isolation
  - C. Scalability
  - D. Automation
  - E. Reproducibility
Virtualization

• Various levels
  - Circuits / CPU (microcode emulating machine code)
  - Storage (e.g., RAID)
  - Networks (e.g., NAT or overlays)
  - Runtime environment (e.g., Java VM or Microsoft .NET)
  - Operating system (e.g., Docker)
  - Full desktops (e.g., QEMU, VMware or VirtualBox)
Hypervisors

- **Native** hypervisors (“type 1”)
  - Run directly on the host’s hardware in kernel mode
  - Sometimes as part of a general-purpose OS
  - Examples: VMware ESX, Microsoft Hyper-V, Oracle VM Server, Xen

- **Hosted** hypervisors (“type 2”)
  - Runs as a process inside the host OS
  - Often hardware-accelerated (e.g., Intel VT-x or AMD-V)
  - Examples: VMware Workstation, VirtualBox, QEMU
  - Sometimes called an emulator if it virtualizes a different architecture
    - Example: Project 4 in CS 261 is a Y86-64 emulator for x86-64
Windows: 3.1, 95, and 10 on 8.1

Image courtesy of Mike Ripley (JMU Infrastructure and Database Support)
OS-level virtualization

- **Container**: isolated user space for a program and its dependencies
  - Multiple user spaces implemented at the kernel level
  - Alternative descriptions:
    - Virtual memory extended to files and libraries
    - Sandboxed, lightweight, app-specific VMs that run natively (no guest OS)
    - “Packages” for a single program's file system
  - **Performant**: minimal overhead vs. running natively
  - Examples: `chroot`, `FreeBSD jail`, `Docker`, `Apptainer/Singularity`
• Which of the following statements is true?
  - A. A hosted hypervisor always has less overhead than a native hypervisor.
  - B. Native hypervisors pose fewer security hazards than hosted hypervisors.
  - C. An emulator will always run a program slower than the original hardware.
  - D. A Docker container will always run faster than a VirtualBox VM with comparable configurations.
  - E. Just-in-time-compiled (to x86) Java code will generally run slower than bytecode on the Java VM.
Cloud computing

- **Cloud computing**: technically, it’s more nuanced than just “other people’s computers”

[Image: There is NO CLOUD, just other people’s computers]

https://fsfe.org/contribute/spreadtheword#nocloud
Cloud computing

- Essential characteristics (from NIST definition*)
  - On-demand self-service for provisioning
  - Broad network access for availability
  - Resource pooling for independence
  - Rapid elasticity for scaling
  - Measured service for transparency

- Examples: Amazon Web Services, Google Cloud Platform, Microsoft Azure, Rackspace

*https://doi.org/10.6028/NIST.SP.800-145
Cloud computing

- **Service models (from NIST definition*)**
  - Software as a Service (SaaS)
  - Platform as a Service (PaaS)
  - Infrastructure as a Service (IaaS)


* https://doi.org/10.6028/NIST.SP.800-145
Cloud computing

• Deployment models (from NIST definition*)
  - Private (single organization)
  - Community (multiple organizations)
  - Public (open to general public)
  - Hybrid (combination of above)

Image from https://en.wikipedia.org/wiki/Cloud_computing

* https://doi.org/10.6028/NIST.SP.800-145
Everything as a service (EaaS/XaaS)

- Analytics as a service (AaaS)
- Backend as a service (BaaS)
- Communication as a service (CaaS)
- Containers as a service (CaaS)
- Content as a service (CaaS)
- Data platform as a service (dPaaS)
- Desktop as a service (DaaS)
- Function as a service (FaaS)
- Games as a service (GaaS)
- Hardware as a service (HaaS)
- Integration platform as a service (iPaaS)
- IT as a service (ITaaS)
  ...
- Workspace as a service (WaaS)
- Hybris as a service (YaaS)
- Zenoss as a service (ZaaS)
Cloud computing

- "Serverless" computing
  - FaaS: Function as a Service (another layer of abstraction!)
  - Pay for compute time, not a particular host or VM
  - There's still a server, but the user doesn't interact with it directly
  - Code must be written using a supported language
  - Amazon Lambda, Google Functions

[Diagram showing the process of uploading code to AWS Lambda, setting up code to trigger from other AWS services, HTTP endpoints, or in-app activity, and how Lambda runs your code only when triggered, using only the compute resources needed.]

https://aws.amazon.com/lambda/
Cloud engineering

- Emerging/developing field
  - Combines computer system engineering (EE), software engineering (CS), and computer information systems (business)
  - Focus on IaaS/PaaS/SaaS/FaaS applications
    - Often with a “big data” focus
  - Goals: performance, scalability, security, reliability
  - Challenge: integrating multiple solutions and layers
  - First IEEE International Conference on Cloud Engineering (IC2E) in March 2013
Future prediction

• Will cloud computing become the dominant model of computation within the next 5-10 years?
  - A. Yes, within five years
  - B. Yes, within ten years
  - C. No, but it will eventually
  - D. No, on-premises computing will always remain the dominant model of computing