UC San Diego

CSE 291: Operating Systems in Datacenters

Amy Ousterhout

Sept. 28, 2023

UC San Diego

Agenda for Today

- Brief introductions
- Introduction to OS in Datacenters
- Course logistics
- Questions to ask when reading a paper
- CloudLab overview

Introductions

A bit about me:

- Amy Ousterhout

"oh"-"stir"-"howt"

- Please call me Amy!
- Assistant Professor in CSE
- Research focus: resource efficiency in datacenters

A bit about you!

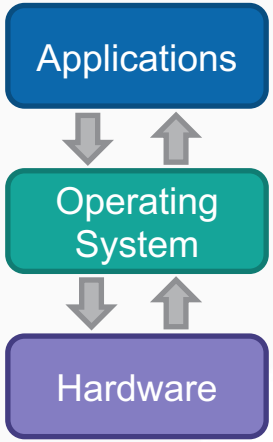
- Your name
- Undergrad/Masters/PhD
- Why are you interested in operating systems and datacenters?

UC San Diego

Operating Systems in Datacenters

What is an Operating System?

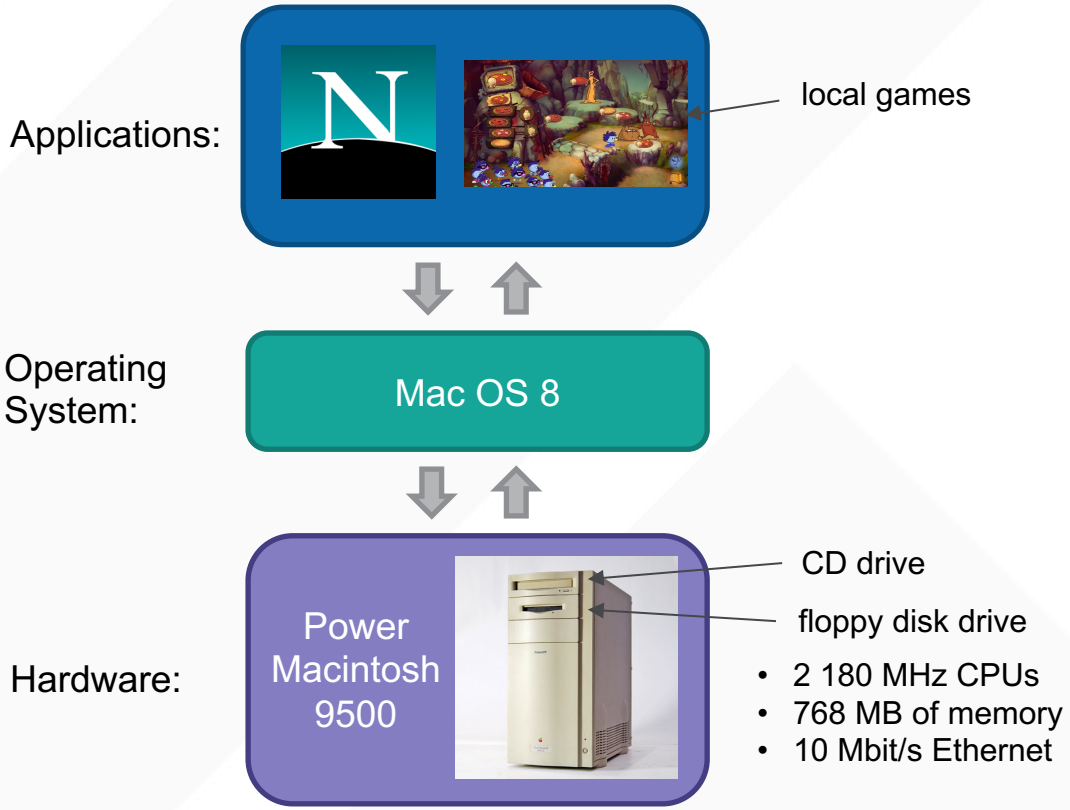
- System software that:
  - Manages computer hardware and software
  - Provides services to computer programs
- Acts as the interface between hardware and applications



Example Operating System: Mac OS 8

- Released in 1997
- Key OS features:
  - Processes and threads
  - Storage (disks)
  - Virtual memory
  - File system
  - Network stack

main components of an undergrad OS course

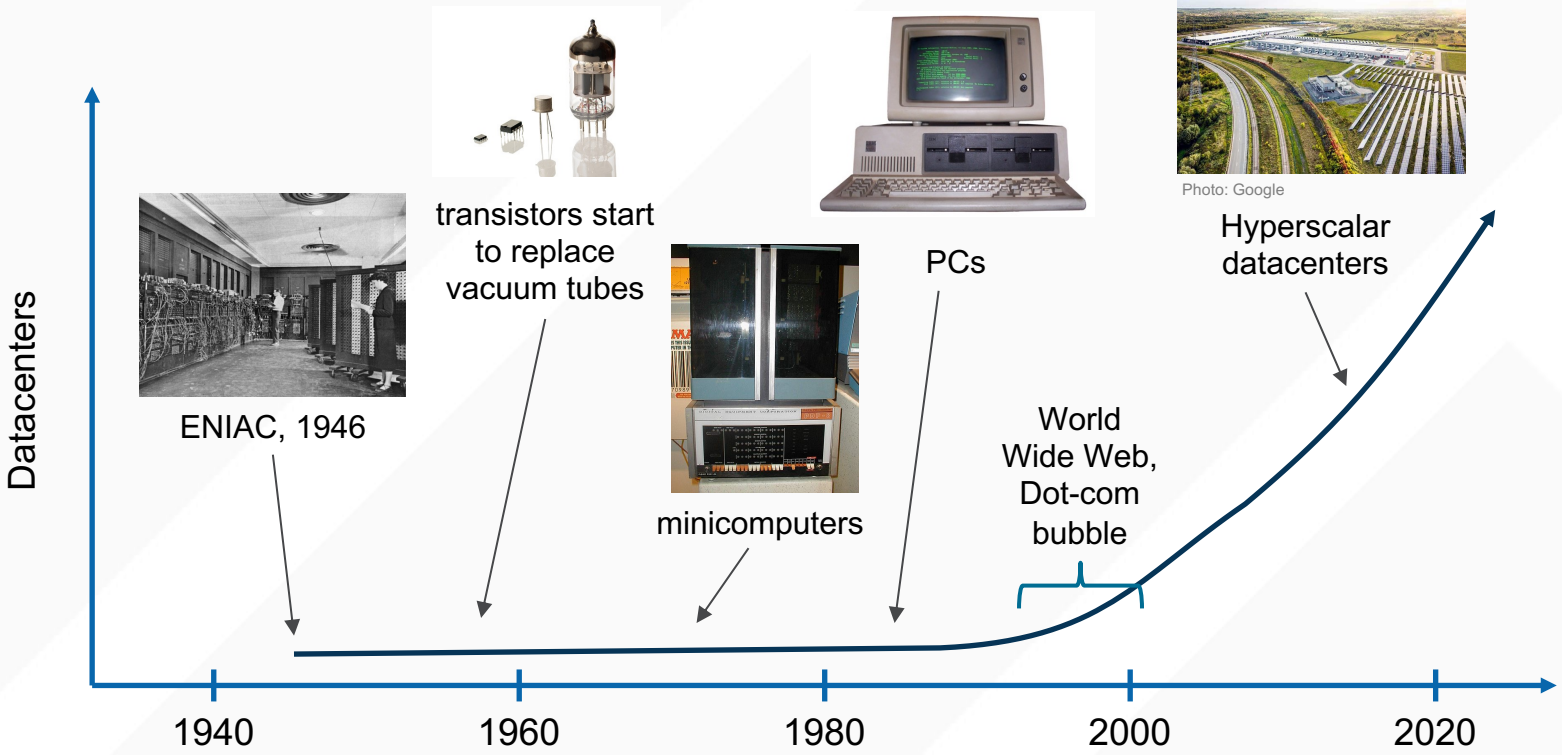


https://commons.wikimedia.org/wiki/File:PowerMac_9500_132_front.jpg
https://www.usatoday.com/story/money/2015/08/05/zoombinis-game-reboot/30615099/

What is a Datacenter?

- Dedicated space that contains:
  - Computers
  - Communication systems
  - Storage systems

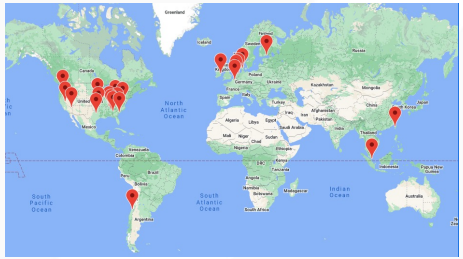
History of Datacenters



https://www.gocertify.com/articles/who-invented-the-computer-tradic

Datacenters Today

- Over 8,000 datacenters globally
- Over 2,600 datacenters in the U.S.
- Huge energy consumers – almost 2% of global energy use
  - Usually built near energy sources (hydroelectric, wind, solar)



Google's Datacenter Locations

Amazon AWS Locations

Photo: Google

Google datacenter

Datacenters from the Outside

Google datacenter in Oklahoma

Cooling systems

Meta datacenter in Texas

Power infrastructure: windmills and power lines

Microsoft datacenter in Wyoming

https://commons.wikimedia.org/wiki/File:Google_Mayes_County_P0004991a.jpg
https://www.datacenterknowledge.com/microsoft/microsoft-creates-blueprint-green-data-center-design
https://www.dallasnews.com/business/real-estate/2017/05/03/facebook-s-fort-worth-data-center-opening-this-week-is-getting-bigger/

Inside a Datacenter

- Servers arranged into racks
- Each server has power and network cables

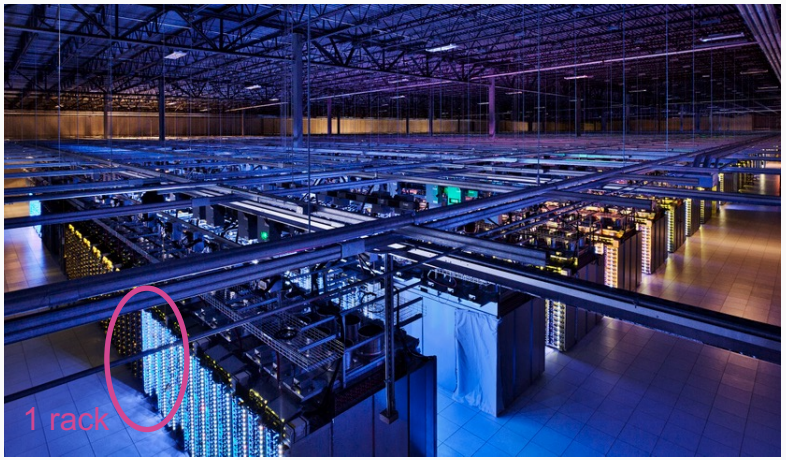


Photo: Google

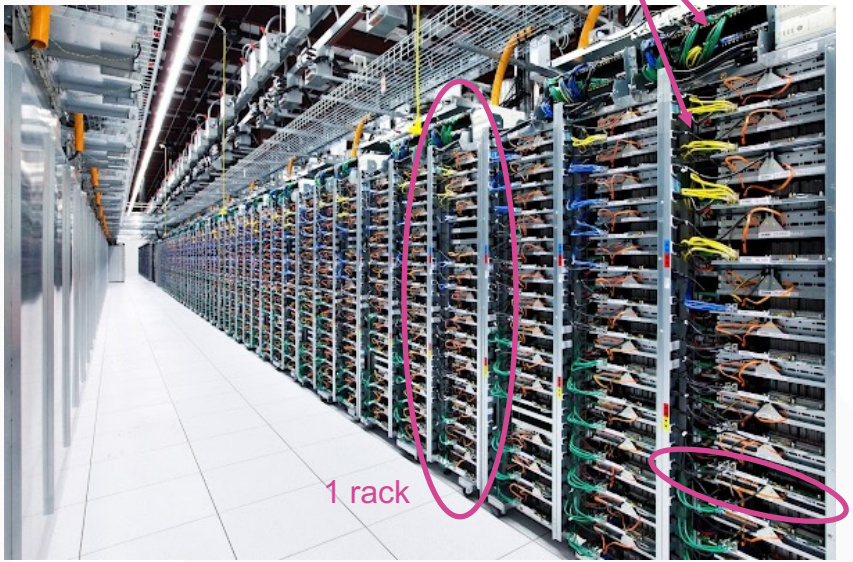
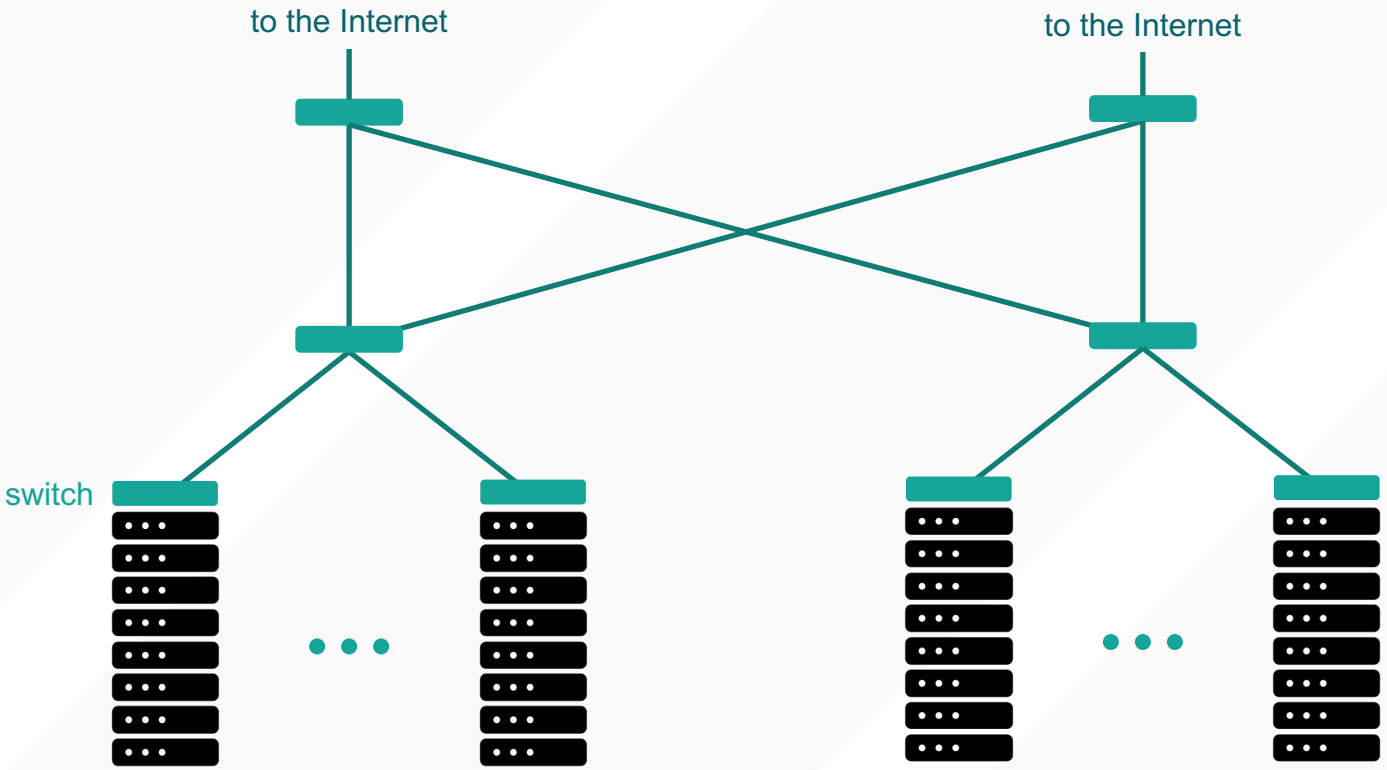
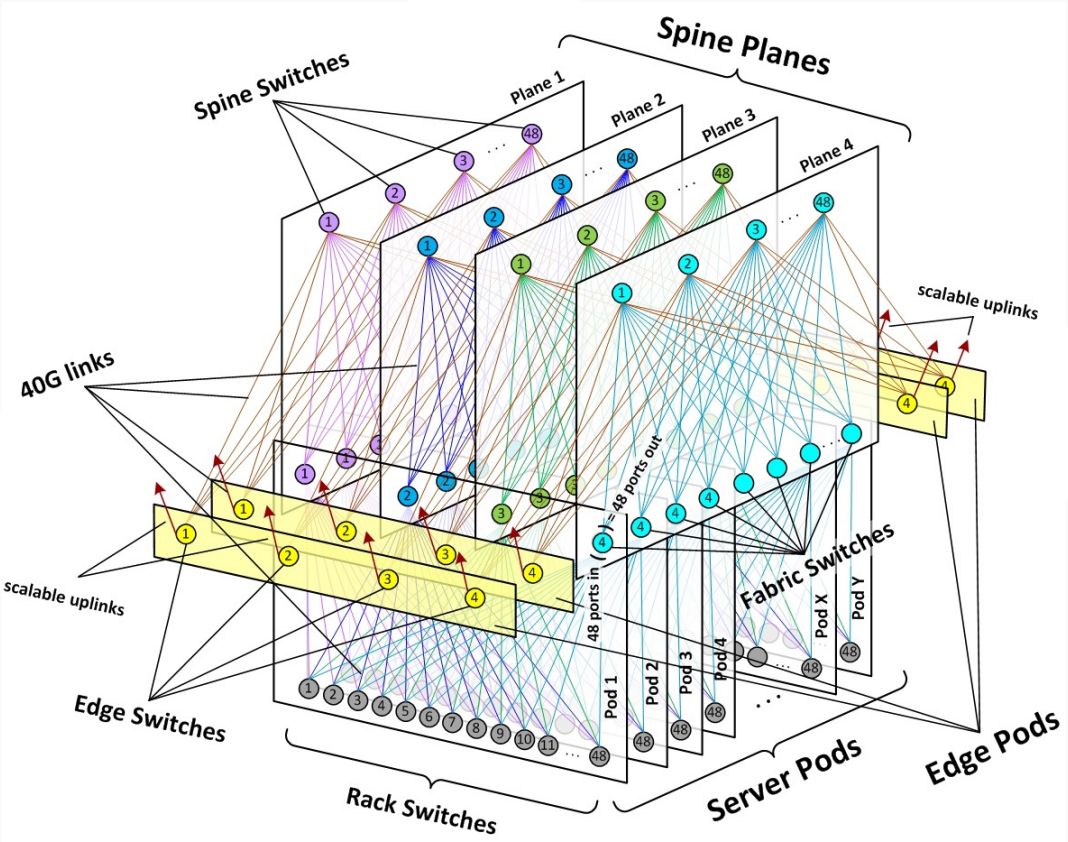


Photo: Google

Datacenter Networks - Simplified



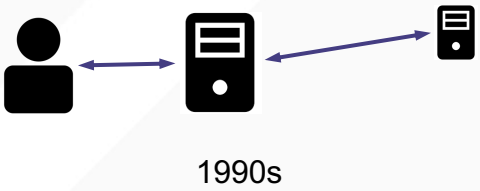
Datacenter Networks – In practice



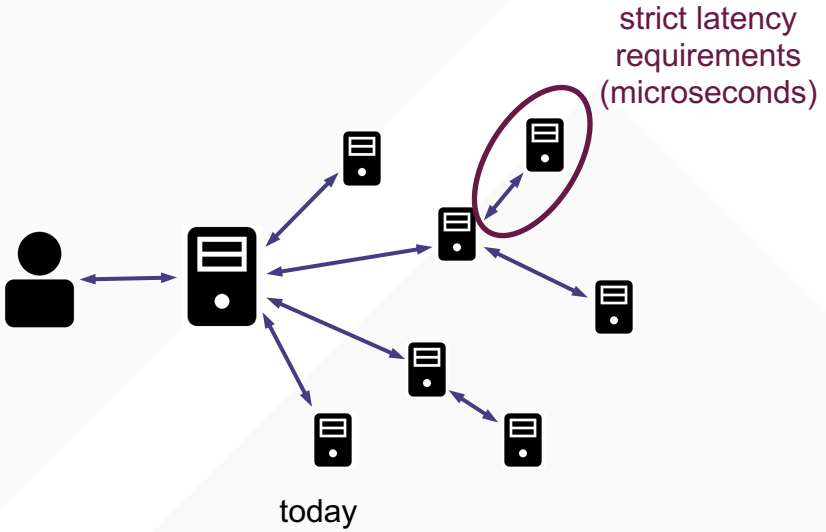
https://engineering.fb.com/2014/11/14/production-engineering/introducing-data-center-fabric-the-next-generation-facebook-data-center-network/

Trend #1: Increasingly Complex Applications

- 1990s: static web pages served by a single server
- 2010: tens to hundreds of servers involved
  - Web search, social networks, etc.
- 2020: hundreds to thousands of servers involved



1990s

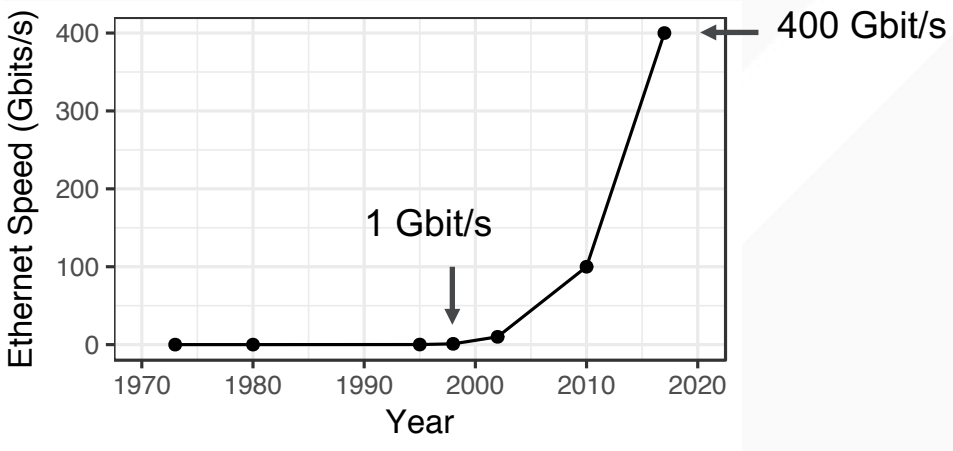


today

Trend #2: Faster Networks

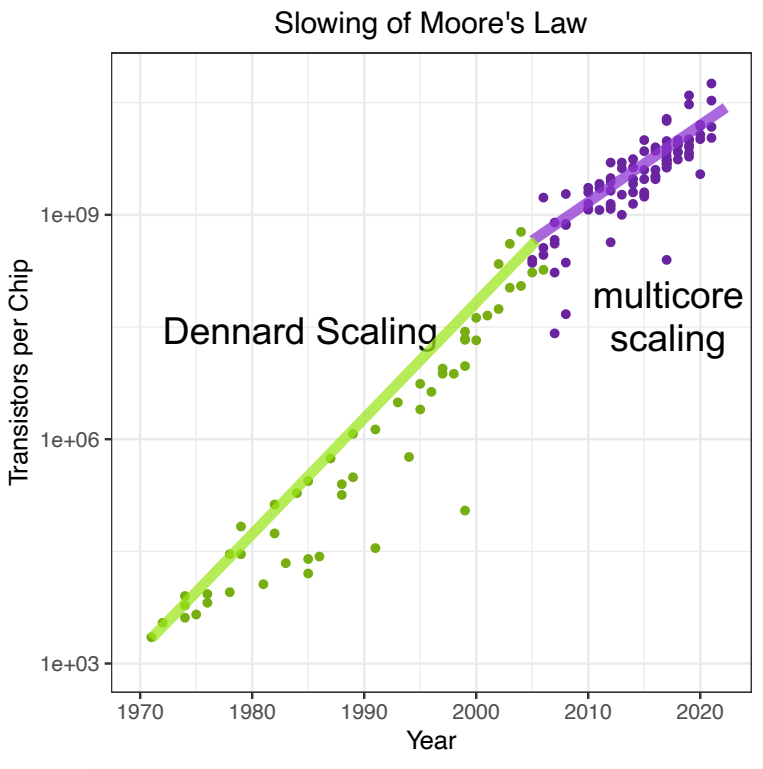
- Network bandwidth has increased 400x
- Network latencies have decreased too
  - Network latency = transmission + propagation + switching
  - Transmit 1500 bytes at 1 Gbit/s: 12 µs
  - Transmit 1500 bytes at 400 Gbit/s: 30 ns

servers are expected to process large amounts of network traffic very quickly



Trend #3: End of Moore's Law

- Increasing demand for compute
- Faster CPUs every few years!
- But, Moore's Law is ending
- Consequences:
  - More cores per server (multicore)
  - Move tasks to hardware with custom accelerators

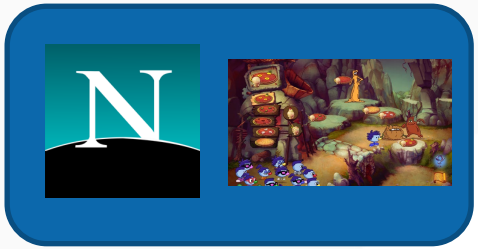


Operating Systems Requirements in Datacenters

Applications:

Operating System:

Hardware:



Applications tolerate ms-scale overheads

Applications demand <= µs-scale overheads

Mac OS 8

This course!

Power Macintosh 9500

1990s

1-2 CPU cores

CD drive

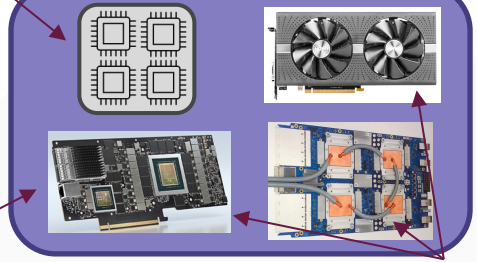
floppy disk drive

Low I/O bandwidth

- 2 180 MHz CPUs
- 768 MB of memory
- 10 Mbit/s Ethernet

I/O latency: 10s of ms

Multicore CPUs



I/O: hundreds of Gbits/s, µs latency

datacenter today

GPUs, TPUs, SmartNICs

https://commons.wikimedia.org/wiki/File:RX580_Nitro.jpg
https://upload.wikimedia.org/wikipedia/commons/b/be/Tensor_Processing_Unit_3.0.jpg
https://www.fierceelectronics.com/electronics/nvidia-claims-dpu-performance-record

Challenges Imposed on OS by Datacenters

- New hardware
  - Multicore
  - Fast networks
  - Heterogeneity (GPUs, TPUs, SmartNICs)
- New applications
  - Large and complex
  - Expect extremely low latency
- This course: how should Operating Systems adapt?

UC San Diego

Course Logistics

Overview

- Graduate-level research-focused course
- The goals of the course are:
  - To learn about recent OS techniques to address datacenter challenges
  - To practice reading and discussing research papers
  - To conduct a research project

Readings

- We will read 1-2 papers per class
  - Everyone should read the papers ahead of time
- Come prepared to discuss!
- Class format
  - Brief overview of each topic
  - Paper discussions

Reviews

- Submit a short review about each paper
- Due by 11:59 pm the evening before
- Google form for submitting reviews will be posted on Canvas
- Share your **opinions** of the paper, such as:
  - Strengths or weaknesses of the key contribution
  - An idea for how to extend the paper
  - What kind of impact you think this paper might have
- For example:
  - “I thought the technique in section 3.4 was a surprisingly simple yet effective solution!”
  - “I’m not sure if this approach can handle a situation in which…”
  - “I think this paper could have a big impact on industry because…”
- Do not include a summary of the paper

Leading a Discussion

- Each student will lead 1 paper discussion
  - Either individually or in pairs
- Preparation:
  - Read the paper as usual
  - Outline your discussion
  - Share with instructor at least 48 hours before discussion
- No need for slides

Research Project

- Open-ended research project
- Can work alone or in groups of 2-3 students
- You choose the topic
  - Broadly related to OS in datacenters
  - Implementation, experimental, algorithmic, theoretical
- How to pick a topic?
  - Propose your own idea
  - I will suggest some ideas for how to find a topic
  - Continue an existing research project
- Computing platform
  - I recommend CloudLab

Research Project Components

- ~1-page proposal, due 10/20
- I will meet with you throughout the quarter to check-in
- Project presentations, in-class 12/5 and 12/7
- ~6-page project write-up, due 12/14

Warm-Up Assignment

- Goals:
  - Show you how to use CloudLab
  - Give you some experience with RDMA and DPDK

Grading

- There are no exams
- 15% paper reviews
- 15% class participation
- 10% discussion lead
- 10% warm-up assignment
- 50% research project

Course website

https://amyousterhout.com/cse291-fall23

Date	Topics	Papers
Th 9/28	Course overview, intro to CloudLab	
Tu 10/3	Multicore	Multikernel (SOSP '09)
Th 10/5	Network stacks	IX (OSDI '14), XDP (CoNEXT '18)
Tu 10/10	RDMA and RPCs	FaRM (NSDI '14)
Th 10/12	RDMA and RPCs	eRPC (NSDI '19), PRISM (SOSP '21)
Tu 10/17	no class	
Th 10/19	Congestion control	Homa (SIGCOMM '18), Swift (SIGCOMM '20)
Tu 10/24	CPU scheduling	Killer Microseconds (CACM '17), Shenango (NSDI '19)
Th 10/26	CPU scheduling	ghOSt (SOSP '21)
Tu 10/31	Performance diagnosis	NSight (NSDI '22), Fathom (SIGCOMM '23)
Th 11/2	NIC Interfaces	Ensō (OSDI '23)
Tu 11/7	Datacenter tax	Accelerometer (ASPLOS '20)
Th 11/9	SmartNICs	AccelNet (NSDI '18), iPipe (SIGCOMM '19)
Tu 11/14	GPUs and TPUs	TensorFlow (OSDI '16)
Th 11/16	FPGAs	Coyote (OSDI '20)
Tu 11/21	Disaggregation	LegoOS (OSDI '18), Memory disaggregation (SOSR '23)
Th 11/23	Thanksgiving holiday	
Tu 11/28	Memory management	Llama (ASPLOS '20)
Th 11/30	Miscellaneous topics	OS Verification (HotOS '23), Reducing Embedded Carbon (HotCarbon '23)

Academic Integrity

- You are welcome to discuss this class with others
- You should write all text for this class yourself
  - Paper reviews
  - Warm-up assignment
  - Project proposal
  - Project write-up
- No assistance from other humans or forms of AI when writing text
- But, you may use AI to help you write code for your research project
  - You must acknowledge this in your write-up

UC San Diego

Questions to Ask When Reading a Paper

High-Level Questions

- What is the problem?
  - Why is it important?
- What is the solution?
  - What is new about the solution?
- Which parts did you not understand?

More Detailed Questions

- Solution
  - What is their approach?
  - What are the key components and how important is each one?
  - Did the paper solve the problem?
  - Are there limitations? How fundamental are they?
- Evaluation
  - How did they evaluate their work?
  - Are the experiments realistic (testbed, workloads, etc.)?
  - Do they demonstrate that the solution works?
- Impact
  - Do you think this work will be impactful? Why?
  - What kind of impact do you think it will have?

More Detailed Questions

- Authors
  - Who are they and why did they write this paper now?
- Extensions
  - Useful for you to think about as a researcher!
  - What weaknesses does the paper have/how could it be improved?
  - Could you apply these ideas to other problems or in other domains?

UC San Diego

CloudLab Overview

Platforms for Experimentation

- Private compute resources
- Public clouds

- PlanetLab (2002-2020)
- Emulab
- Geni
- Mass Open Cloud (since 2013)
  - Enables experimentation with real users
- CloudLab (since 2014)

focus on networks and distributed systems

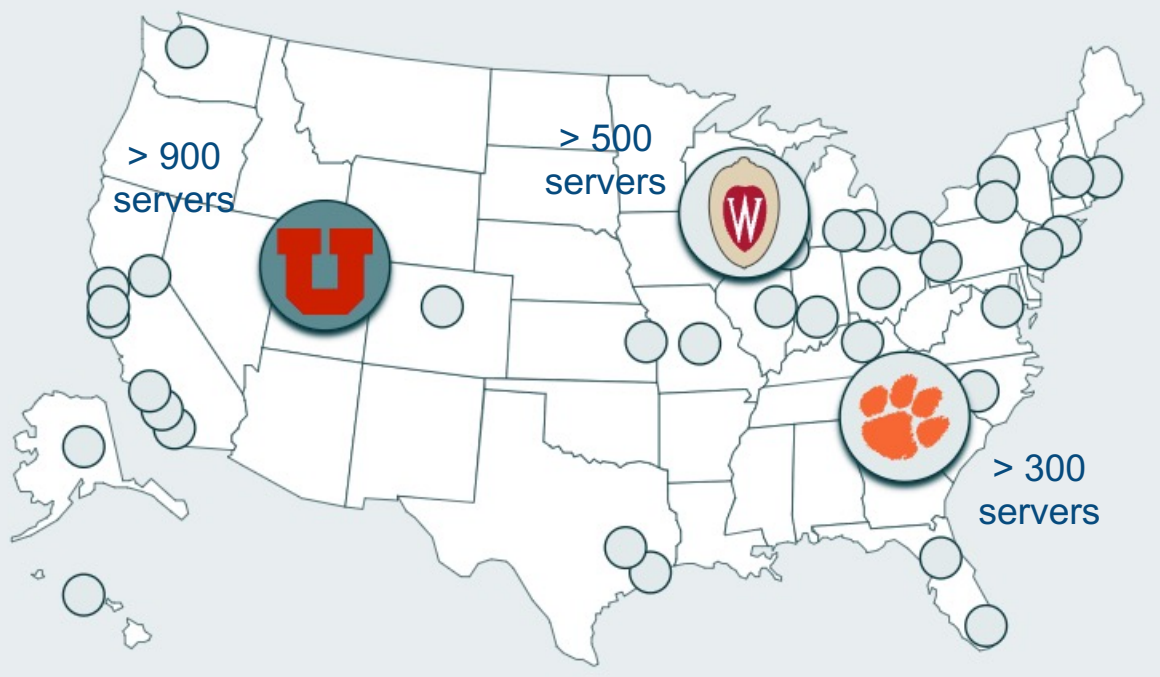
impacted

CloudLab Goals

- Customization
  - Modify storage, virtualization, networking
- Repeatable research
  - Bare metal
  - Uniform performance

What is CloudLab?

A testbed for research on cloud computing



https://www.usenix.org/sites/default/files/conference/protected-files/atc19_slides_duplyakin.pdf

Hardware in CloudLab

- Standard CPUs, memory, storage, NICs
- Specialized hardware: Intel Optane, GPUs, 100 Gbit/s NICs, SmartNICs
- Details at: `https://docs.cloudlab.us/hardware.html`

Warm-up assignment:

m510	270 nodes (Intel Xeon-D)
CPU	Eight-core Intel Xeon D-1548 at 2.0 GHz
RAM	64GB ECC Memory (4x 16 GB DDR4-2133 SO-DIMMs)
Disk	256 GB NVMe flash storage
NIC	Dual-port Mellanox ConnectX-3 10 GB NIC (PCIe v3.0, 8 lanes

Other options:

r7525	15 nodes (AMD EPYC Rome, 64 core, 512GB RAM, 2 x GPU)
CPU	Two 32-core AMD 7542 at 2.9GHz
RAM	512GB ECC Memory (16x 32 GB 3200MHz DDR4)
Disk	One 2TB 7200 RPM 6G SATA HDD
NIC	Dual-port Mellanox ConnectX-5 25 Gb NIC (PCIe v4.0)
NIC	Dual-port Mellanox BlueField2 100 Gb SmartNIC
GPU	Two NVIDIA GV100GL (Tesla V100S PCIe 32GB)

Who pays for CloudLab?

- National Science Foundation
- Free to use for research and educational purposes

How to Use CloudLab

- Create a ”Project Profile” which specifies:
  - The configuration of 1 or more servers
  - Network connectivity between them
  - Software to run on servers
- Instantiate your Project Profile to create an experiment
  - Start immediately or make a reservation
  - Stop your experiment when you’re done
  - It will terminate after 16 hours

Why use CloudLab?

- Cost – cheaper than buying and maintaining your own resources or using public clouds
- Flexibility – can try out different computing resources for short periods of time
- Customization – tune the hardware
- Reproduceable research

runs on CloudLab

AIFM: High-Performance, Application-Integrated Far Memory

Zhenyuan Ruan Malte Schwarzkopf † Marcos K. Aguilera ‡ Adam Belay
MIT CSAIL *†Brown University* *‡VMware Research*

Abstract. Memory is the most contended and least elastic resource in datacenter servers today. Applications can use only local memory—which may be scarce—even though

Throughput [accesses/sec]	*64B object*	*4KB object*
Paging-based (Fastswap [6])	582K	582K
AIFM	3,975K	1,059K

What kind of research is CloudLab not ideal for?

- Large scale – requiring hundreds or thousands of nodes
- Locations
  - More than a few locations
  - Specific locations
- Real cloud users

To Learn More About CloudLab (optional)

The Design and Operation of CloudLab

Dmitry Duplyakin *Robert Ricci* *Aleksander Maricq* *Gary Wong* *Jonathon Duerig*
Eric Eide *Leigh Stoller* *Mike Hibler* *David Johnson* *Kirk Webb*
*Aditya Akella** *Kuangching Wang*† *Glenn Ricart*‡ *Larry Landweber** *Chip Elliott*§
Michael Zink¶ *Emmanuel Cecchet*¶ *Snigdhaswin Kar*† *Prabodh Mishra*†

University of Utah *University of Wisconsin †Clemson University
‡US Ignite §Raytheon ¶UMass Amherst

Given the highly empirical nature of research in cloud computing, networked systems, and related fields, testbeds play an important role in the research ecosystem. In this paper, we cover one such facility, CloudLab, which supports systems research by providing raw access to programmable hardware, enabling research at large scales, and creating a shared platform for repeatable research.

We present our experiences designing CloudLab and operating it for four years, serving nearly 4,000 users who have run over 79,000 experiments on 2,250 servers, switches, and other pieces of datacenter equipment. From this experience, we draw lessons organized around two themes. The first set comes from analysis of data regarding the use of CloudLab: how users interact with it, what they use it for, and the implications for facility design and operation. Our second set of lessons comes from looking at the ways that algorithms used "under the hood," such as resource allocation, have important—and sometimes unexpected—effects on user experience and behavior. These lessons can be of value to the designers and operators of IaaS facilities in general, systems testbeds in particular, and users who have a stake in understanding how these systems are built.

1 Introduction

CloudLab [31] is a testbed for research and education in cloud computing. It provides more control, visibility, and performance isolation than a typical cloud environment, enabling it to support work on cloud architectures, distributed systems, and applications. Initially deployed in 2014, CloudLab is now heavily used by the research community, supporting nearly 4,000 users who have worked on 750 projects and run over

and more. CloudLab staff take care of the construction, maintenance, operation, etc. of the facility, letting users focus on their research. CloudLab gives the benefits of economies of scale and provides a common environment for repeatability.

CloudLab differs significantly from a cloud, however, in that its goal is not only to allow users to build applications, but entire clouds, from the "bare metal" up. To do so, it must give users unmediated "raw" access to hardware. It places great importance on the ability to run fully observable and repeatable experiments. As a result, users are provided with the means not only to *use* but also to *see*, *instrument*, *monitor*, and *modify* all levels of investigated cloud stacks and applications, including virtualization, networking, storage, and management abstractions. Because of this focus on low-level access, CloudLab has been able to support a range of research that cannot be conducted on traditional clouds.

As we have operated CloudLab, we have found that, to a greater extent than expected, "behind the scenes" algorithms have had a profound impact on how the facility is used and what it can be used for. CloudLab runs a number of unique, custom-built services that support this vision and keep the testbed operational. This includes a resource mapper, constraint system, scheduler, and provisioner, among others. CloudLab has had to make several trade-offs between general-purpose algorithms that continue to work well as the system evolves, and more tailored ones that provide a smoother user experience. The right choices for many of these trade-offs were not apparent during the design of the facility, and required experience from the operation of the facility to resolve.

The primary goal of this paper is to provide the architects of large, complex facilities (not only testbeds, but other IaaS-type facilities as well) with lessons from CloudLab's design

For Tuesday

Multikernel

- Read the paper
- Submit a review by 11:59 pm on Monday

The Multikernel: A new OS architecture for scalable multicore systems

Andrew Baumann,* Paul Barham,† Pierre-Evariste Dagand,‡ Tim Harris,† Rebecca Isaacs,†
Simon Peter,* Timothy Roscoe,* Adrian Schüpbach,* and Akhilesh Singhania*

*Systems Group, ETH Zurich

†Microsoft Research, Cambridge ‡ENS Cachan Bretagne

Abstract

Commodity computer systems contain more and more processor cores and exhibit increasingly diverse architectural tradeoffs, including memory hierarchies, interconnects, instruction sets and variants, and IO configurations. Previous high-performance computing systems have scaled in specific cases, but the dynamic nature of modern client and server workloads, coupled with the impossibility of statically optimizing an OS for all workloads and hardware variants pose serious challenges for operating system structures.

We argue that the challenge of future multicore hardware is best met by embracing the networked nature of the machine, rethinking OS architecture using ideas from distributed systems. We investigate a new OS structure, the *multikernel*, that treats the machine as a network of independent cores, assumes no inter-core sharing at the lowest level, and moves traditional OS functionality to a distributed system of processes that communicate via message-passing.

We have implemented a multikernel OS to show that the approach is promising, and we describe how traditional scalability problems for operating systems (such as memory management) can be effectively recast using messages and can exploit insights from distributed systems and networking. An evaluation of our prototype on multicore systems shows that, even on present-day ma-

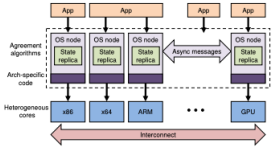


Figure 1: The multikernel model.

Such hardware, while in some regards similar to earlier parallel systems, is new in the general-purpose computing domain. We increasingly find multicore systems in a variety of environments ranging from personal computing platforms to data centers, with workloads that are less predictable, and often more OS-intensive, than traditional high-performance computing applications. It is no longer acceptable (or useful) to tune a general-purpose OS design for a particular hardware model: the deployed hardware varies wildly, and optimizations become obsolete after a few years when new hardware arrives.

Moreover, these optimizations involve tradeoffs specific to hardware parameters such as the cache hierarchy, the memory consistency model, and relative costs of lo-