UC San Diego

# CSE 291: Operating Systems in Datacenters

Amy Ousterhout

Sept. 22, 2022

# Agenda for Today

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

# Introductions

A bit about me:

- $\underbrace{\text{Amy Ousterhout}}_{\text{“oh”-“stir”-“howt”}}$
- Please call me Amy!
- Assistant Professor in CSE
- Research focus: resource efficiency in datacenters

Course TA:

- Anil Yelam

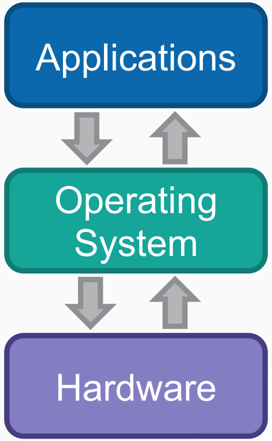
A bit about you!

- Your name
- Your background
- Why are you interested in operating systems and datacenters?

# 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

Applications:

"Logical Journey of the Zoombinis"

Operating System:

Mac OS 8

Hardware:

Power Macintosh 9500

CD drive

floppy disk drive

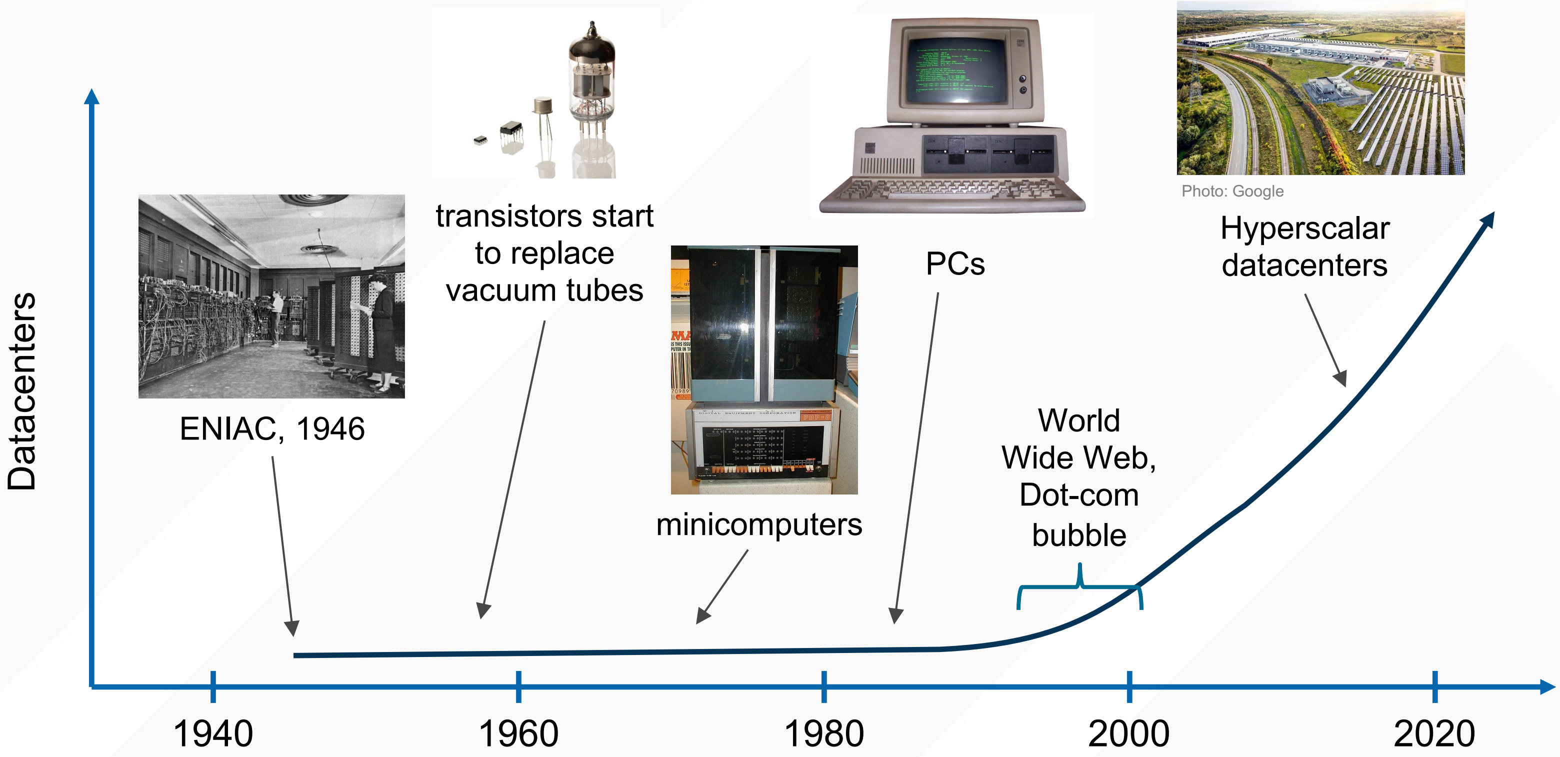
- 2 180 MHz CPUs
- 768 MB of memory
- 10 Mbit/s Ethernet

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

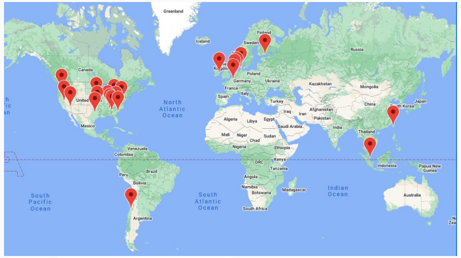
# History of Datacenters




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

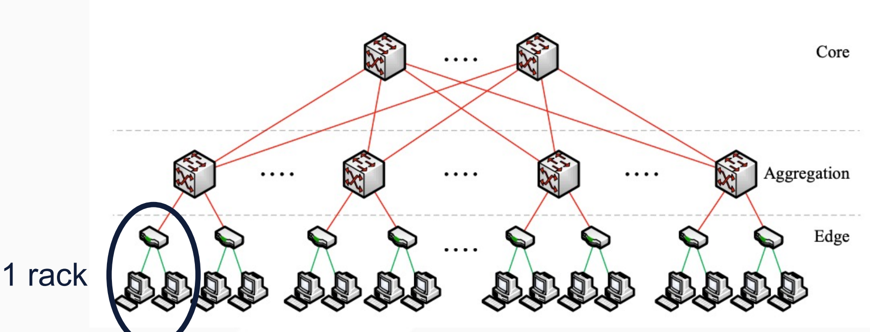
Photo: Google

Google datacenter

# Inside a Datacenter

- Servers arranged into racks
- Racks connected by a hierarchical network topology

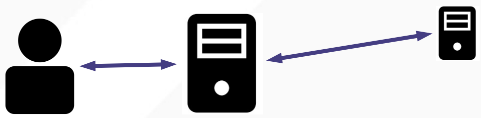

Photo: Google



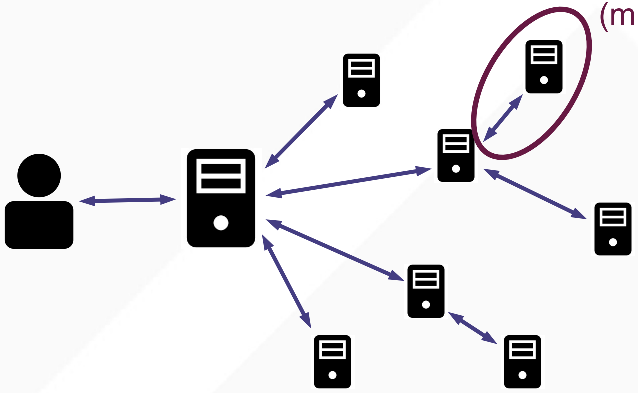

A Scalable, Commodity Data Center Network Architecture, SIGCOMM, 2008

# 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

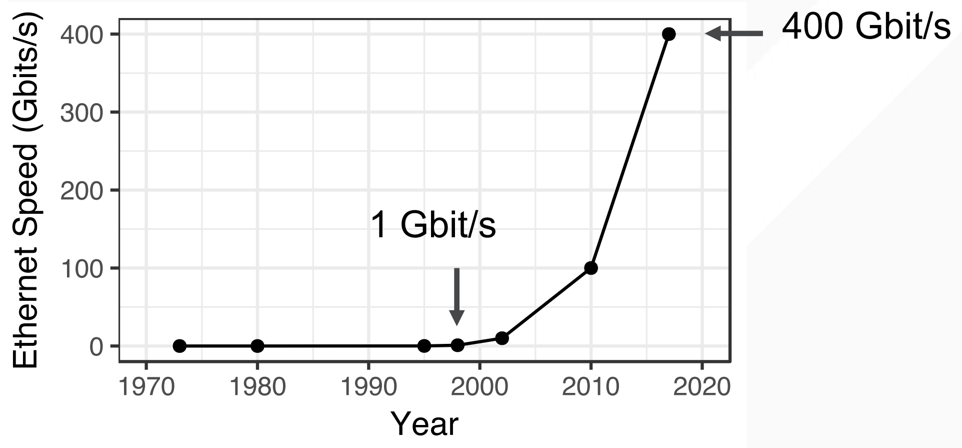



today

# Trend #2: Faster Networks

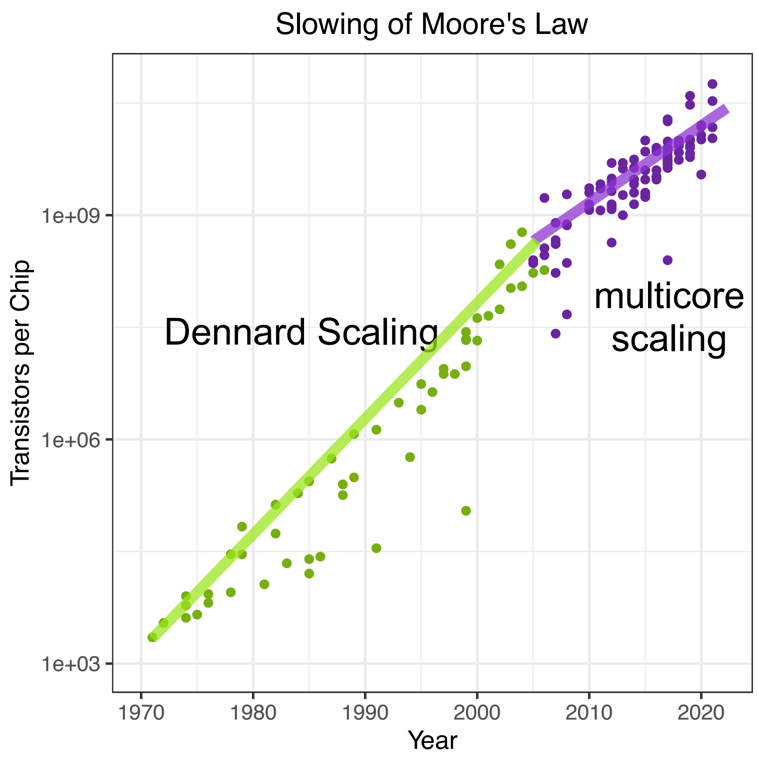
- Network bandwidth has increased 400x
- Network latencies have decreased too
  - Network latency = transmission + propagation + switching
  - Transmit 1500 bytes at 1 Gbit/s: 12 µs
  - Transmit 1500 bytes at 400 Gbit/s: 30 ns

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:

Applications tolerate ms-scale overheads

Applications demand µs-scale overheads

Operating System:

Mac OS 8

This course!

Hardware:

Power Macintosh 9500

1-2 CPU cores

CD drive

Low I/O bandwidth

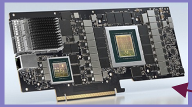
floppy disk drive

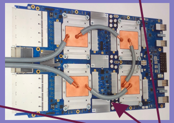
- 2 180 MHz CPUs
- 768 MB of memory
- 10 Mbit/s Ethernet

I/O latency: 10s of ms

1990s

Multicore CPUs





I/O: hundreds of Gbits/s, µs latency

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?

# 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 and Reviews

- We will read 1-2 papers per class
  - Everyone should read the papers ahead of time
  - Submit a short review by 11:59 pm the evening before
  - Details about reviews will be on Canvas coming soon!
- Come prepared to discuss!
- Class format
  - Brief overview of each topic
  - Paper discussions

# Leading a Discussion

- Each student will lead 1 paper discussion
- Preparation:
  - Outline your discussion
  - Share with instructor at least 2 days 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
- Computing platform
  - I recommend CloudLab

# Research Project Components

- ~1-page proposal, due 10/20
- Project presentations, in-class 11/29 and 12/1
- ~6-page project write-up, due 12/8
- We will meet with you throughout the quarter to check-in
- More details will be posted on Canvas

# 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% project

# Course website

https://amyousterhout.com/cse291-fall22

| Date | Topics | Papers |
|---|---|---|
| Th 9/22 | Course overview | |
| Tu 9/27 | Multicore, intro to CloudLab | Multikernel (SOSP '09), CloudLab (ATC '19) - only first 2 sections |
| Th 9/29 | Network stacks | IX (OSDI '14), XDP (CoNEXT '18) |
| Tu 10/4 | RDMA and RPCs | FaRM (NSDI '14) |
| Th 10/6 | RDMA and RPCs | eRPC (NSDI '19), PRISM (SOSP '21) |
| Tu 10/11 | Congestion control | Homa (SIGCOMM '18), Swift (SIGCOMM '20) |
| Th 10/13 | CPU scheduling | Killer Microseconds (CACM '17), Shenango (NSDI '19) |
| Tu 10/18 | CPU scheduling | ghOSt (SOSP '21) |
| Th 10/20 | Performance diagnosis | NSight (NSDI '22), Collie (NSDI '22) |
| Tu 10/25 | Datacenter tax | Warehouse-scale computers (ISCA '15) |
| Th 10/27 | SmartNICs | AccelNet (NSDI '18) |
| Tu 11/1 | SmartNICs | iPipe (SIGCOMM '19), nanoPU (OSDI '21) |
| Th 11/3 | GPUs | PTask (SOSP '11) |
| Tu 11/8 | TPUs | TensorFlow (OSDI '16) |
| Th 11/10 | FPGAs | AmorphOS (OSDI '18), Coyote (OSDI '20) |
| Tu 11/15 | Disaggregation | LegoOS (OSDI '18) |
| Th 11/17 | Memory management | Llama (ASPLOS '20) |
| Tu 11/22 | Memory management | TLB shootdowns (EuroSys '20) |

# For Tuesday

Multikernel

- Regular paper discussion
- Do submit a review

CloudLab

- To learn more about what CloudLab is
- First 2 sections only
- No need to submit a review

## 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-

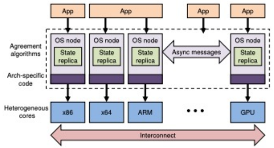



Figure 1: The multikernel model.

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-

## 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.

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,

# 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?

# Summary

- CSE 291: Operating Systems in Datacenters
- Focus on learning about new OS techniques in datacenters
- Read and discuss research papers
- Undertake a class project