CS 2690: Cloud and Datacenter Operating Systems — Intro Lecture

Instructor: Deepti Raghavan

Thursday, September 5

Course Website

- Class website: https://brown-csci2690.github.io/classwebsite-fall24/
- Also accessible via: https://cs.brown.edu/courses/csci2690

Agenda

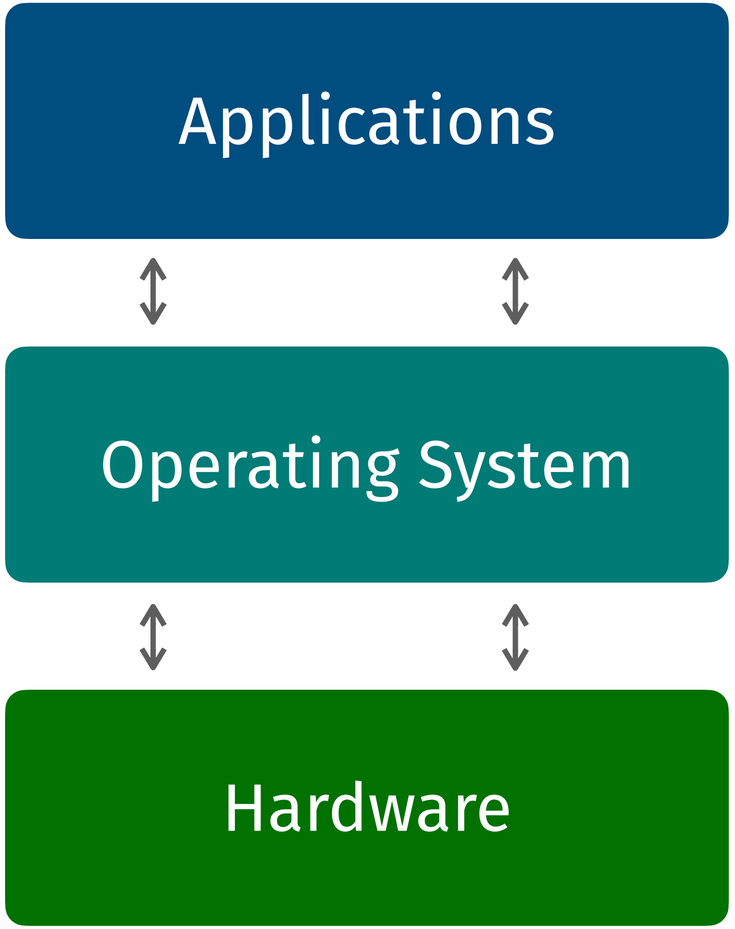
- Brief Introduction
- Why Datacenter and Cloud OS?
    - Brief overview of what “datacenter” and “cloud computing” means*
    - How computing is different inside datacenters and in the cloud
    - Challenges for datacenter systems & challenges for cloud systems
    - Disclaimer: class is about **2/3 on datacenters**, and **1/3 on cloud**
        - And mostly research on how to support cloud and cloud workloads, not how to use the cloud
- Course logistics
- Overview of Cloudlab & Hints for Warmup Assignment

About Me

- Assistant Professor in CS (I am new to Brown!)
- Research focus: Improving performance in networked & machine learning systems
  - With a focus on efficient communication abstractions

What is an Operating System?

- Software that:
  - Manages hardware on machine and user-facing software (e.g., applications)
  - Provides services to programs (so it is easier to write application)
  - Acts as interface between hardware and applications

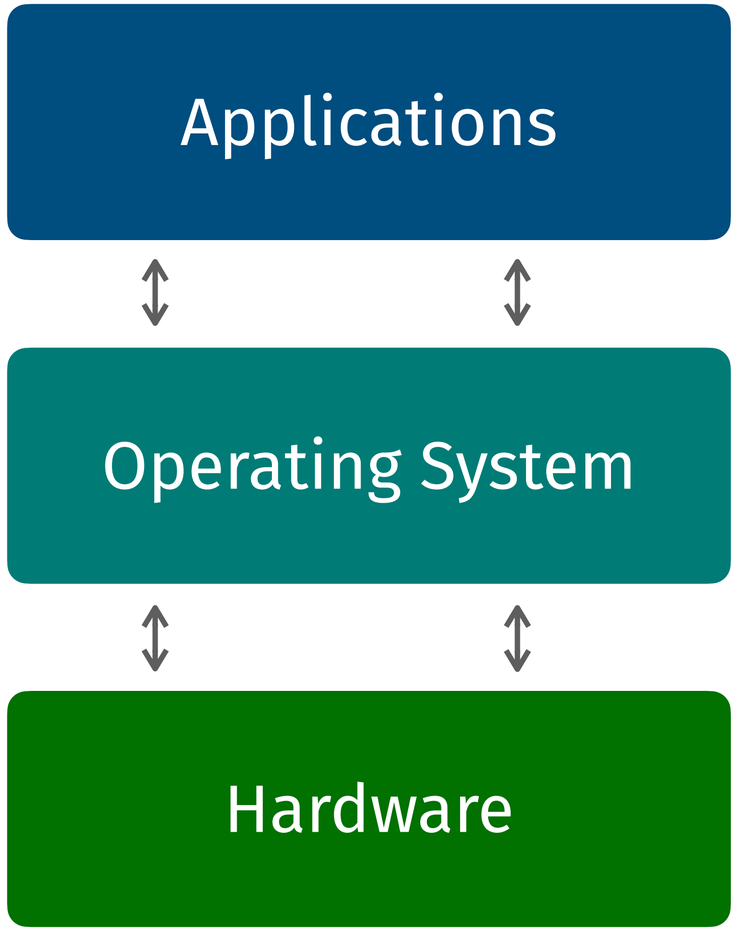


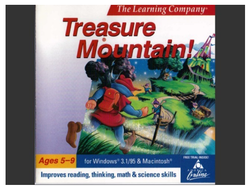
Slide design and content from: https://amyousterhout.com/cse291-fall23/slides/L1-Intro.pdf

Example OS: Mac OS 8

You would have learned about this in a 1000-level OS course!

- Released in 1997
- What does it provide?
  - Processes/Threads
  - Storage
  - Virtual Memory
  - File System
  - Network Stack





Mac OS 8

CD drive
Floppy disk drive

Specs:
2 180 MHz CPUs
16-32 MB Memory
10 Mbit/s Ethernet

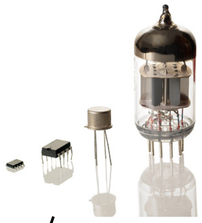
Slide design and content from: https://amyousterhout.com/cse291-fall23/slides/L1-Intro.pdf

https://en.wikipedia.org/wiki/Power_Macintosh_9500#/media/File:PowerMac_9500_132_front.jpg
https://www.macintoshrepository.org/6069-treasure-mountain-
https://en.wikipedia.org/wiki/Netscape_Navigator#/media/File:Netscape_icon.svg

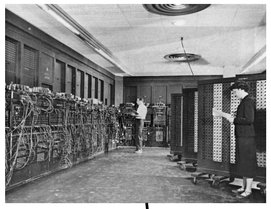
What is a datacenter?

- Space with:
  - Servers
  - Communication Systems (so servers can talk to each other and to public internet)
  - Storage systems
  - Cooling infrastructure
  - Power infrastructure

A mini history lesson (not to scale)



ENIAC, 1946



Transistors start to replace vacuum tubes

Minicomputers

PCs

Hyper scalar datacenters (google, belgium)

World Wide Web and Dot-com bubble

1940 1960 1980 2000 2020

VMWare releases workstation (1999)

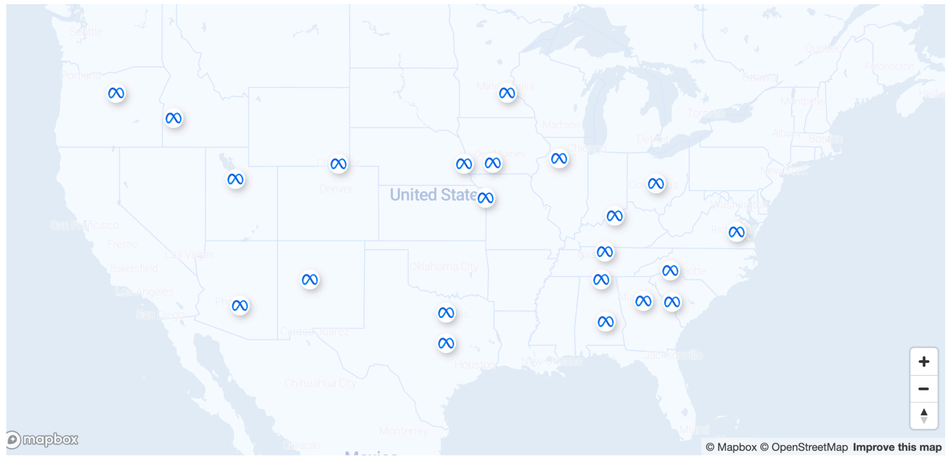
AWS starts selling compute/ storage (2006)

https://www.gocertify.com/articles/who-invented-the-computer-tradic
https://en.wikipedia.org/wiki/ENIAC
https://en.wikipedia.org/wiki/Minicomputer#/media/File:Data_General_Nova_SN_1.agr.JPG
https://en.wikipedia.org/wiki/Apple_II_(original)
https://www.google.com/about/datacenters/gallery/

Slide design and content from: https://amyousterhout.com/cse291-fall23/slides/L1-Intro.pdf

Where are datacenters today? What do they look like?

- At least 8000 datacenters worldwide
- Many in US! (~2000)
- Globally, they consume 2% of energy use
  - Usually built near clean energy sources (solar, wind, etc.)
- Contains lots of cooling infrastructure



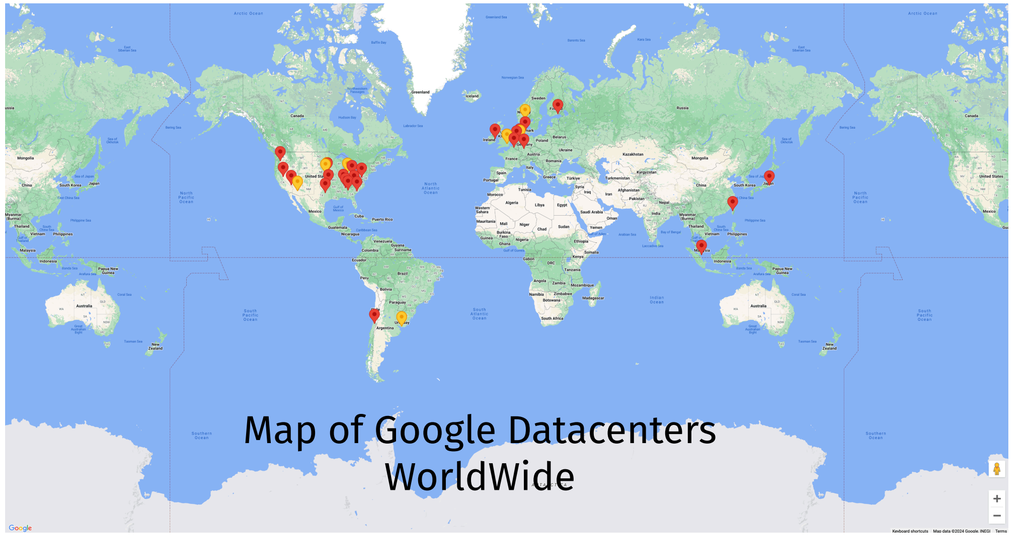
Map of Meta datacenters in the US

Google Datacenter in Belgium

Solar Panels

Wind Mills

Google Datacenter in the Netherlands

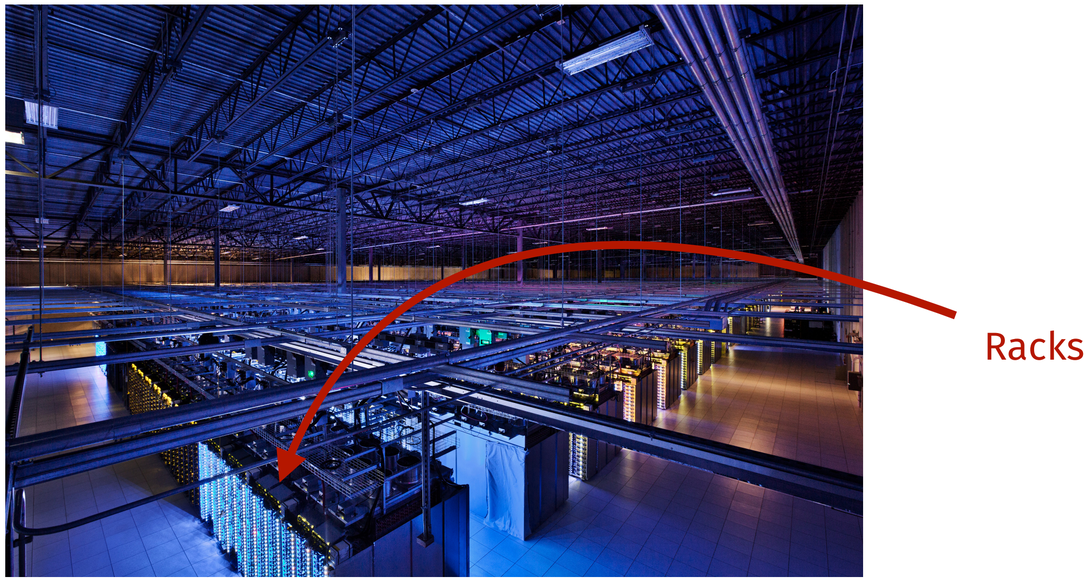


https://datacenters.atmeta.com/

https://www.google.com/about/datacenters/gallery/

What does the inside of a datacenter look like?

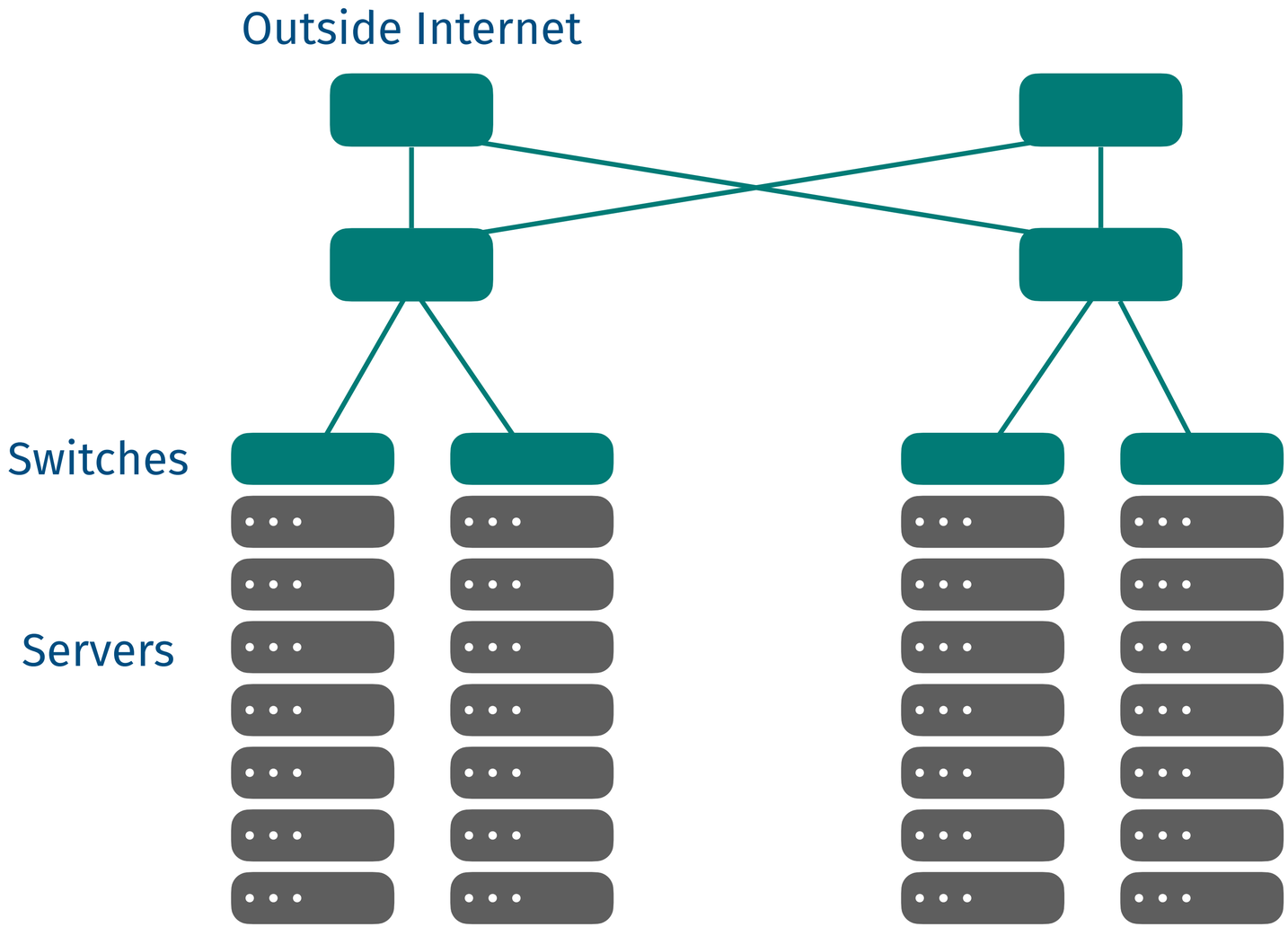
- Servers arranged into racks
- Various cables handle both power and networking



Slide design and content from: https://amyousterhout.com/cse291-fall23/slides/L1-Intro.pdf

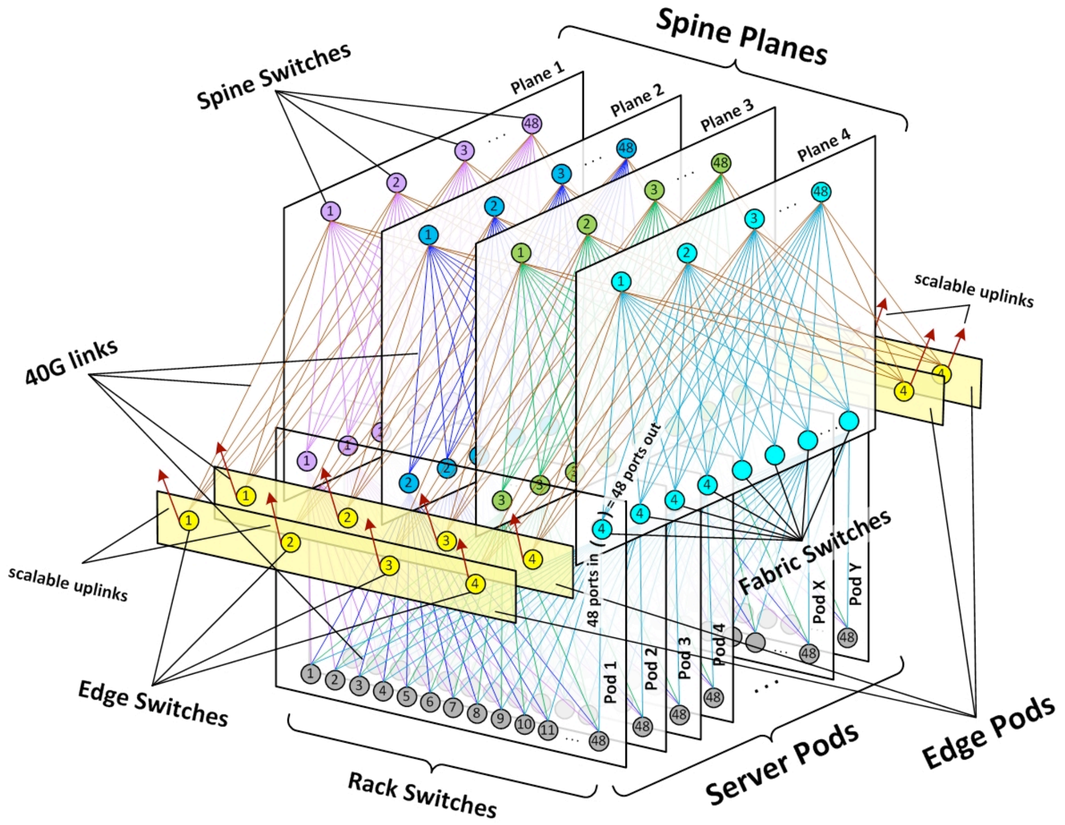
https://www.google.com/about/datacenters/gallery/

How might we connect datacenter racks?



Slide design and content from: https://amyousterhout.com/cse291-fall23/slides/L1-Intro.pdf

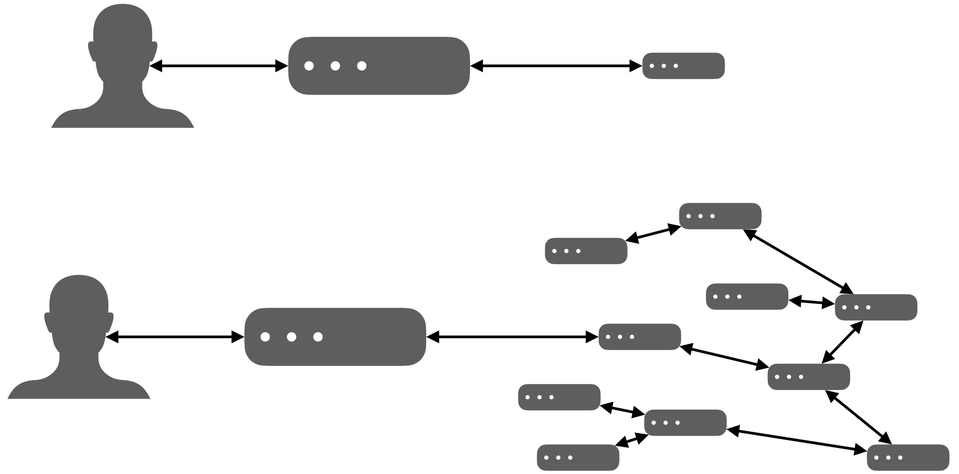
Networking inside datacenters is complex

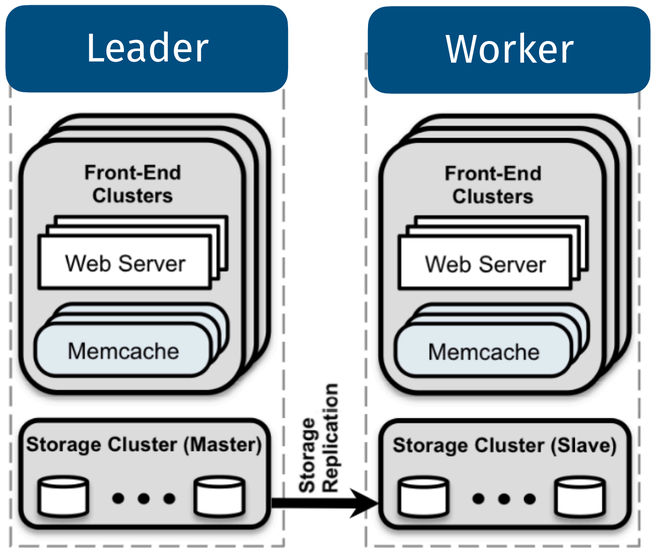


https://engineering.fb.com/2014/11/14/production-engineering/introducing-data-center-fabric-the-next-generation-facebook-data-center-network/

Why are datacenter systems today different? 1: More Complex Apps

- 1990s: static web pages served by a single server
- 2000s: more online services — tens to hundreds of servers
  - Social networks (early Facebook, myspace), Search (Google)
- 2020: hundreds to thousands of servers involved
  - A simple online query will result in requests to many many servers



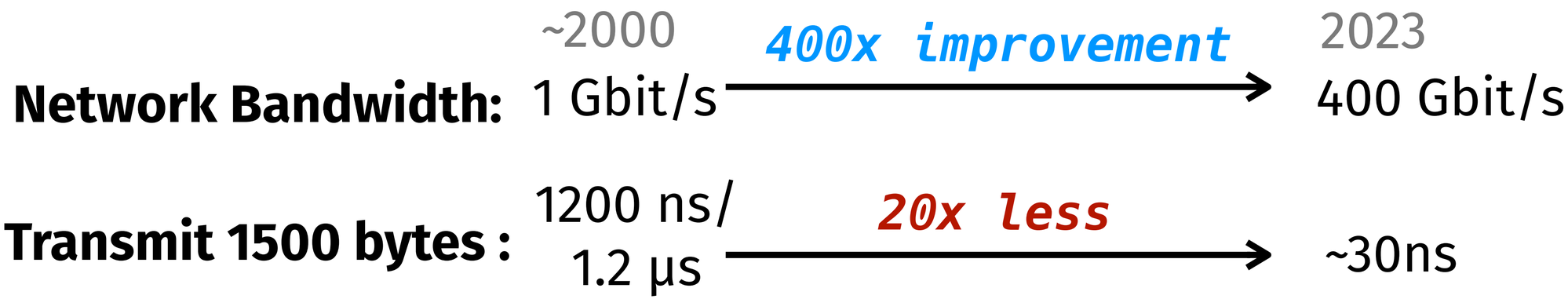


} Memcached service at Facebook (small component of larger application)

Slide design and content from: https://amyousterhout.com/cse291-fall23/slides/L1-Intro.pdf

https://www.usenix.org/system/files/conference/nsdi13/nsdi13-final170_update.pdf

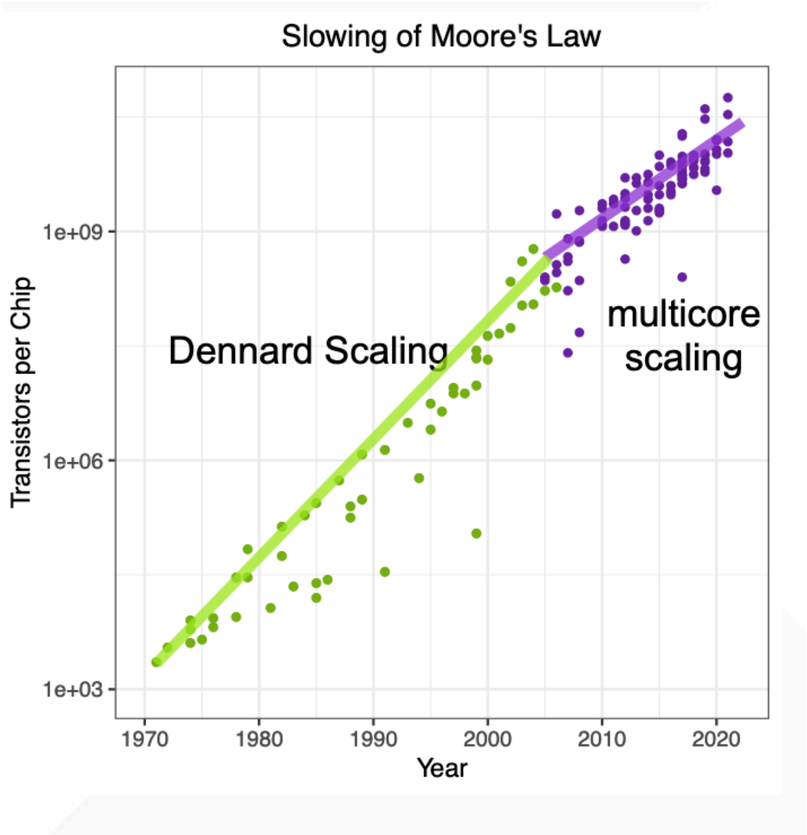
Why are datacenter systems today different? 2: Faster Networks



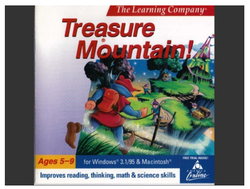
- Networks have gotten much faster:
  - If network is a pipe, think the pipe is both much fatter and water flows through faster:
- Latency has **also** decreased (transmission + switching + propagation delay):
  - From 10s of **MILLI**seconds to 10s of **MICRO**seconds

Why are datacenter systems today different? 3: End of Moore's Law

- Moore's Law: every X time (1.5-2 years), the number of transistors per chip doubles
- Dennard Scaling: as transistors get smaller, they are faster and consume less power
  - Until about 2006, when we hit the max of about ~4 GHz clock frequency for CPUs
- Combined: free performance for CPUs as time goes on
- Now: not the case…
  - Multicore
  - non-CPU accelerators for specialized workloads (GPU, TPU, NIC)



What new challenges do datacenters have to deal with?



Mac OS 8

Specs:

2 180 MHz CPUs

16-32 MB Memory

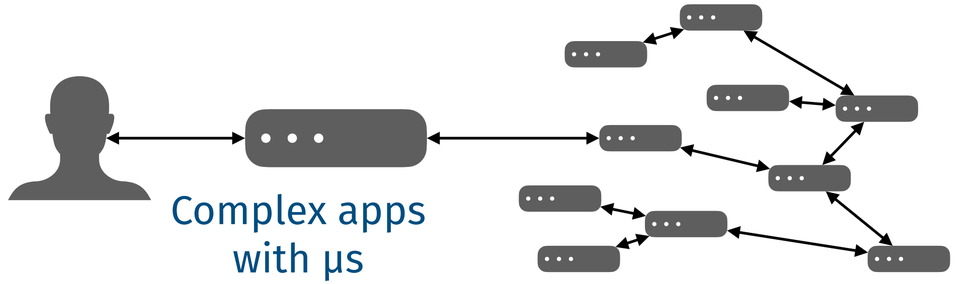
10 Mbit/s Ethernet

10s of ms of latency

Applications

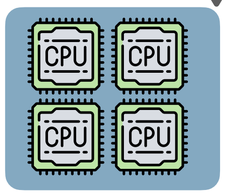
Operating System

Hardware



Complex apps with µs requirements!

1/2-2/3 of the papers in this class!



Multiple cores (128-256!)

Non-CPU accelerators

Faster networks

Slide design and content from: https://amyousterhout.com/cse291-fall23/slides/L1-Intro.pdf

https://nvidianews.nvidia.com/news/nvidia-blackwell-platform-arrives-to-power-a-new-era-of-computing

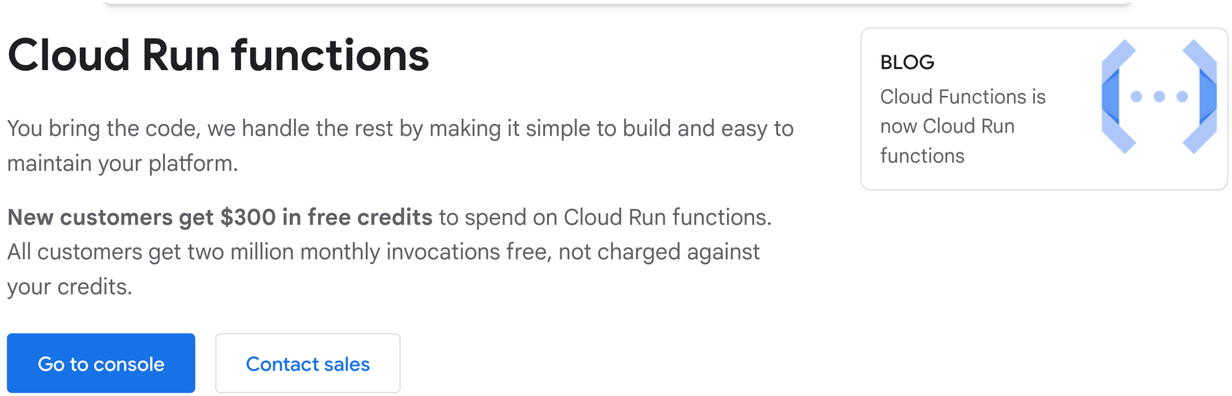
DEMO

What is “cloud computing” (in the context of this class)

- The idea of computing as a utility,
  - Can be public or private:
    - Outsourced to third party (cloud provider like Amazon AWS)
    - Internal org handles infrastructure (e.g., OIT runs compute cluster at Brown)
  - IaaS: Infrastructure as a Service — rent compute, storage
  - PaaS: Platform as a Service — services like managed Map Reduce/Spark
  - SaaS: Software as Service — email/github

Newer model: FaaS (function as a service)

- Run functions in an isolated container triggered by events
- Used by: web apps (bookstore app), stream processing analytics, etc.
  - Newer use cases: video encoding, highly-parallel compilation



Why Use the Cloud

- Users
  - Pay-as-you use/go (if apps have variable utilization, can only pay when resources are in use and scale up/down)
  - Elasticity (can get 1000 CPUs for 1 hour rather than 1 CPU for 1000 hours) — for highly parallel tasks
  - Don’t need to manage hardware
- Cloud providers
  - Costs amortized over many customers
  - Can deploy changes to many customers and get faster feedback

Systems challenges for Cloud Computing (focus of this class)

- Programming Model:
  - Data consistency across functions (FaaS)?
  - How to specify dependencies (FaaS)?
- Performance:
  - Elastic scale up and down
  - Cold-start times (FaaS)
  - Scheduling across functions, VMs, memory
- Security:
  - Virtualization & efficient virtualization techniques
- Fault tolerance:
  - Application State (correctness)
  - Availability

Tour of the Class Topics (subject to change)

- Class 1: Multicore
- Class 2: OS structure/OS abstractions
- Classes 3-4: High Performance I/O (networking, storage)
- Classes 5-7: Kernel Bypass & new networking interfaces
- Classes 8-10: CPU scheduling
- Classes 10-13: Virtualization (includes guest lecture!)
- Classes 14-16: Memory
- Classes 17-18: Distributed Programming in Cloud
- Classes 19-20: Serverless Programming Models
- Class 21: Sustainability
- Classes 22-23: Machine Learning Systems/Inference

Course Structure

- Paper Reading / Reading Responses (30 %):
- Class Discussion Lead & Class Participation (25 %)
- Warmup Assignment (5 %)
- Semester Long Project (40 %)

Logistics

- Class website: https://brown-csci2690.github.io/classwebsite-fall24/
  - All information is posted here
  - Reading questions will be posted as google forms linked in the calendar
- Canvas: https://canvas.brown.edu/courses/1097148
  - Warmup assignment starter code posted here
  - Used to collect warmup assignment code/response
  - Used to collect project deliverables (proposal, paper)

Paper Reading/Responses

- For each class, 1-2 papers assigned
  - **4-5 hours** of reading a week!
- Reading response due at **5 PM the previous day**
  - I will grade these with check, check plus and check minus and to email/post feedback on your response
  - It may also be seen by the discussion leader to help aid their preparation
  - **Response != summary**
    - Sometimes a specific question, sometimes a more open-ended prompt
- Can miss up to 3 reading responses over semester, no questions asked, don't need to tell us

Class Discussion Lead

- Once (or more) either alone or in pairs (depends on enrollment), lead class discussion
- Share list of discussion points **3 days in advance**; I will suggest changes (optionally can also meet with me)
  - Thursday at 12 PM for Tuesday class, Monday at 12 PM for Thursday class
- Formal slides/presentation NOT required
- Do NOT summarize the paper — similar to the reading responses, we want a critique/discussion

How to read a paper

- Lots of advice for how to read systems papers
- Just read it, start to end, in depth
- Then think:
  - What problem are they solving? — 1-2 sentences
  - Replay their motivation — does the motivation and use case make sense?
  - Replay how solution works for simple example — can you? If not, go back to step 1.
  - Anything you don't understand? (OK to discuss with others and on Ed!)
- Once you are confident you understand:
  - Is problem important? Ambitious? Hard? Long shelf life?
  - Is solution effective? Under what conditions does it break?
  - What other approaches are possible?

Advice from: https://cs268.io/assets/lectures/cs268-2023-L1.pdf

Questions to think about when reading a paper

- What problem is being addressed?
- Do you believe problem is/was important? Consider context (time, target users)
- What is solution's main insight (nugget)?
- Do you think solution is a good one? Explain strengths and weaknesses
- Does paper actually solve stated problem?
- Is evaluation realistic and believable?
- Do you think work will be impactful? What kind of impact?
- (As a researcher) What weaknesses does paper have, how can it be improved?
- (As a researcher) Can you apply ideas from the paper to other domains? How?

Advice from: https://cs268.io/assets/lectures/cs268-2023-L1.pdf and https://amyousterhout.com/cse291-fall23/slides/L1-Intro.pdf

Class Participation

- Come to class ready to discuss the paper (even if you skipped the response)!
- Obviously it is ok to miss class in case of sickness, other extenuating circumstances (and please let me know how I can be of help)
- Frequent absences (including coming without having read or without participating) will be penalized

Warmup Assignment

- Already posted; due **Tuesday September 17**
  - It is 5% of your grade, but you must do it to be enrolled in the class
- Not meant to be stressful! Please see me in OH if you are having issues.

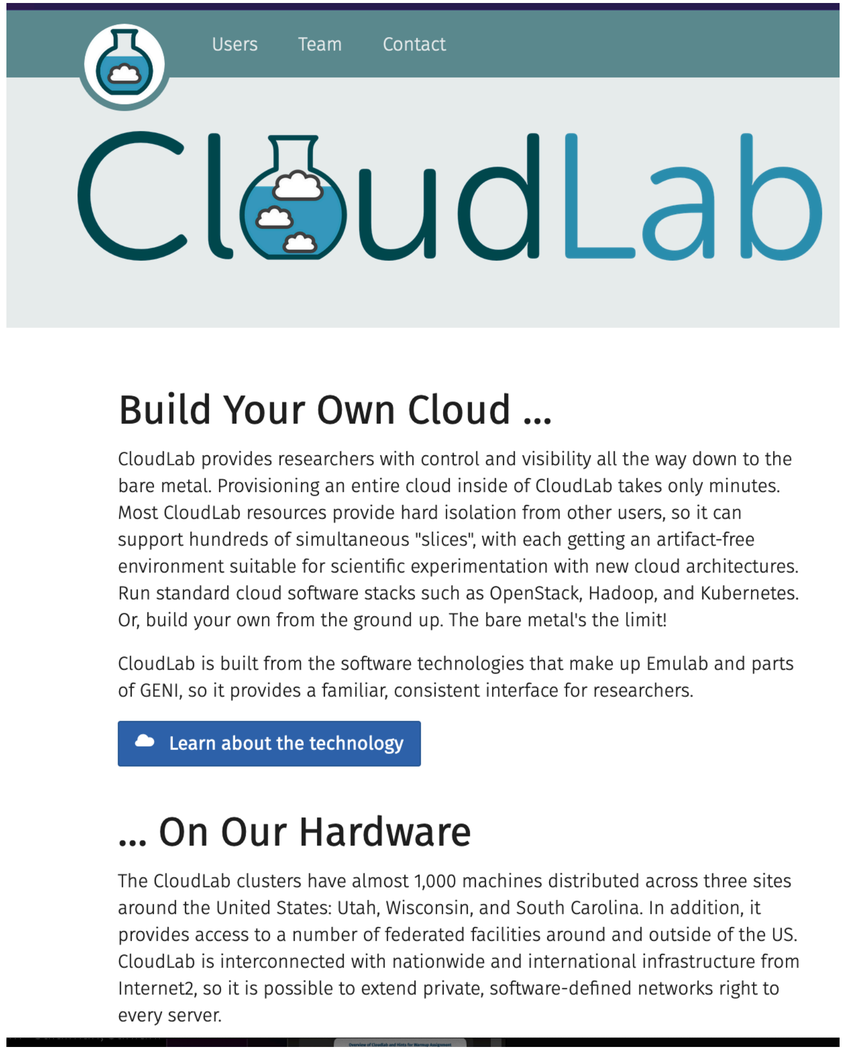
Semester-Long Project

- Open-ended group project consisting of an implementation, some kind of evaluation, and a few deliverables:
  - Proposal (Due 9/30)
  - Week of 9/30 - 10/1: Mandatory initial check in
  - Week of 10/28 - 11/1: Mandatory mid-semester check in
  - Week of Dec 10: Presentations
  - Fri. Dec 13: Final report due
- More info on website about logistics, using external research projects for the class, sharing projects with other seminars
- Potential Project Ideas & more specific info about proposal will be posted before Tuesday's class

Collaboration Policy

- You are encouraged to discuss readings, reading questions and the warmup assignments with others (including those who might not be in the class). However:
  - All answers you submit should be written by you; your response should not be copied from anyone else and not be generated automatically using AI.
  - You can discuss implementations of the warmup assignment with others, but you must write code and responses to the questions yourself.
- Understand your work! We reserve the right to quiz you about your work to ensure you understand your responses and implementation.
- Additionally, you must cite all sources (people, website, papers) that you consult as a part of your work
- You may use AI to help you write code (only code) for your research project.
  - But, you must acknowledge this in your writeup and mention how you used AI

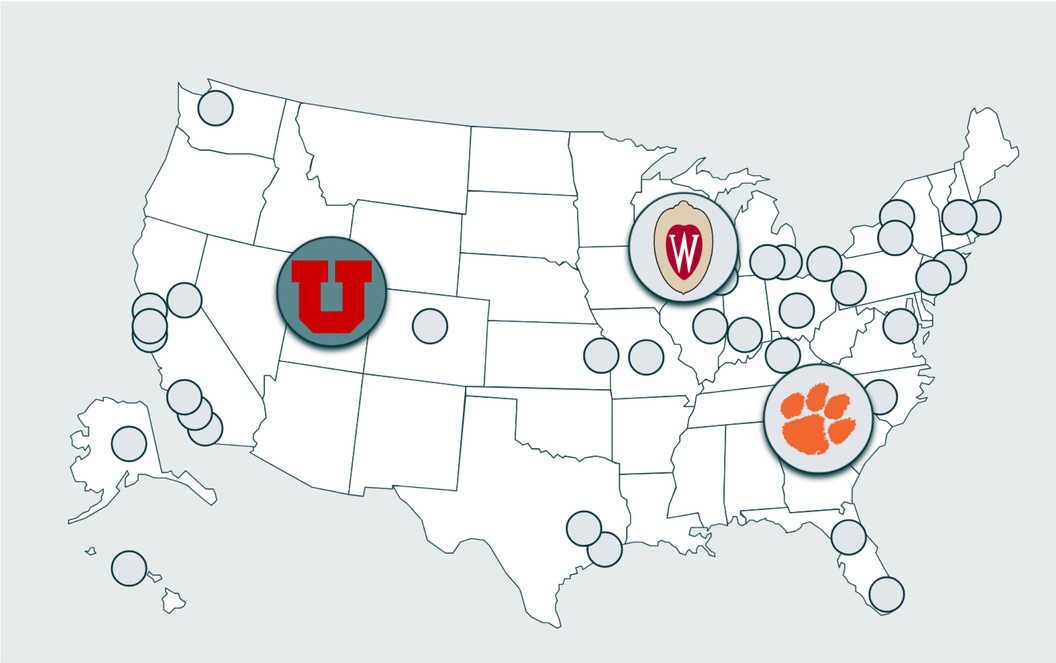
Overview of Cloudlab and Hints for Warmup Assignment



What is cloudlab?

- A testbed for systems experimentation (free for research and class use!), with a focus on cloud computing and datacenter computing
- Goals:
  - Fully customizable systems:
    - Modify storage, networking, virtualization
  - Reproducible (repeatable) research:
    - Uniform Performance
    - Bare metal (not a VM!)

Where is cloudlab?



- Grey nodes are from GENI, an older testbed also now available through cloudlab
- Run by amazing folks at University of Utah, University of Wisconsin Madison, and Clemson University

What hardware can we get?

m400	270 nodes (64-bit ARM)
CPU	Eight 64-bit ARMv8 (Atlas/A57) cores at 2.4 GHz (APM X-GENE)
RAM	64GB ECC Memory (8x 8 GB DDR3-1600 SO-DIMMs)
Disk	120 GB of flash (SATA3 / M.2, Micron M500)
NIC	Dual-port Mellanox ConnectX-3 10 GB NIC (PCIe v3.0, 8 lanes (one port available for experiment use)

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 (one port available for experiment use)

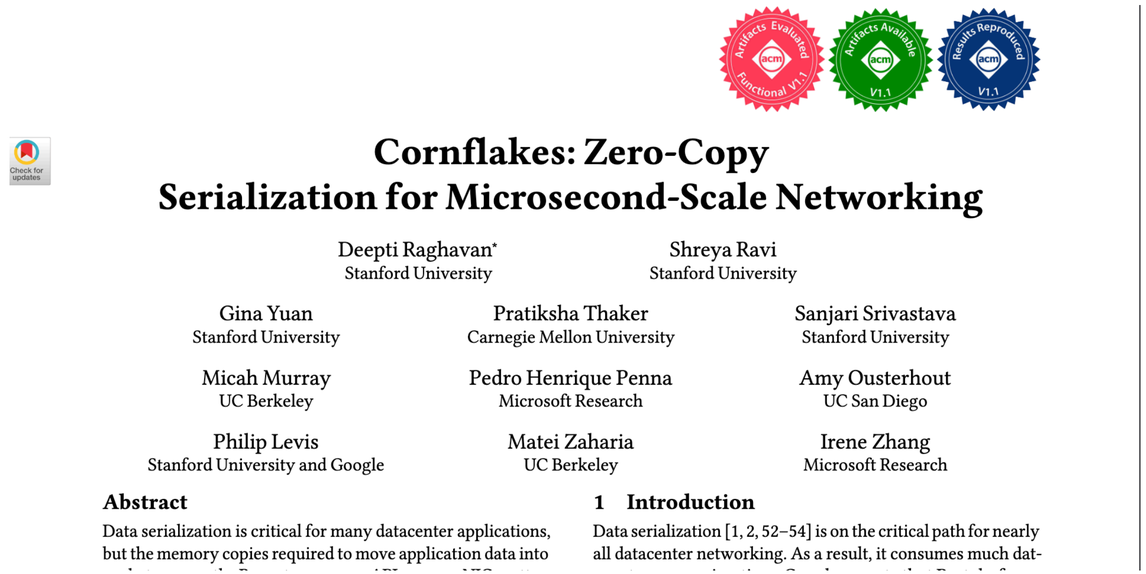
- Standard CPUs, storage memory NICs
- Specialized hardware:
  - Persistent Memory (Intel Optane)
  - SmartNICs
  - Fast Mellanox NICs (100 Gbps)
- More details: https://docs.cloudlab.us/hardware.html

How does one use Cloudlab?

- More detailed instructions on warmup assignment page
- Step 1: Create “Project Profile”:
  - Server configuration (1 or more)
  - Network topology
  - OS image, optional installation scripts
- Step 2: Instantiate profile
  - Start immediately; can also make reservations
  - Stop experiment when done
  - Will terminate automatically after 16 hours (I have lost data due to this!)

Why should you use cloudlab?

- It’s free! Great way to get access to machines for project or independent research
- Customizable, specialized hardware you may not have access to
- Reproducible (others can follow same instructions and get same results):



Cornflakes: Zero-Copy Serialization for Microsecond-Scale Networking

Deepti Raghavan* Stanford University

Shreya Ravi Stanford University

Gina Yuan Stanford University

Pratiksha Thaker Carnegie Mellon University

Sanjari Srivastava Stanford University

Micah Murray UC Berkeley

Pedro Henrique Penna Microsoft Research

Amy Ousterhout UC San Diego

Philip Levis Stanford University and Google

Matei Zaharia UC Berkeley

Irene Zhang Microsoft Research

Abstract

Data serialization is critical for many datacenter applications, but the memory copies required to move application data into

1 Introduction

Data serialization [1, 2, 52–54] is on the critical path for nearly all datacenter networking. As a result, it consumes much dat-

When would cloudlab not be great for research?

- Need many nodes — hundreds to thousands of nodes (hard to get availability)
- Need many locations or specific locations
  - Vs. paid public cloud is located all over the world
- Need real cloud users

To Learn More About Cloudlab:

- https://www.usenix.org/system/files/atc19-duplyakin_0.pdf

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 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-

Lastly, some hints for the warmup assignment

- Remember the structure of ethernet, IP and UDP frames
  - Provided here even if you have not taken 1680
- For warmup assignment, you must write code that **“flips” a UDP packet** in a high-performance networking stack

```
struct rte_ether_hdr {
  struct rte_ether_addr d_addr;
  struct rte_ether_addr s_addr;
  rte_be16_t ether_type;
 }
```

```
struct rte_ipv4_hdr {
 uint8_t version_ihl;
 uint8_t type_of_service;
 rte_be16_t total_length;
 rte_be16_t packet_id;
 rte_be16_t fragment_offset;
 uint8_t time_to_live;
 uint8_t next_proto_id;
 rte_be16_t hdr_checksum;
 rte_be32_t src_addr;
 rte_be32_t dst_addr;
}
```

```
struct rte_udp_hdr {
  rte_be16_t src_port;
  rte_be16_t dst_port;
  rte_be16_t dgram_len;
  rte_be16_t dgram_cksum;
 }
```

Taken from: https://www.cs.nmt.edu/~risk/TCP-UDP%20Pocket%20Guide.pdf