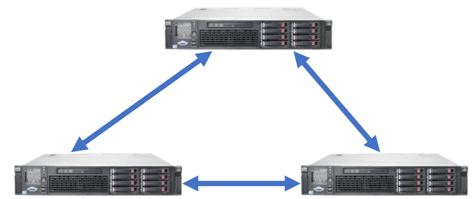
Distributed Systems Intro and Course Overview

COS 418 + 518: (Advanced) Distributed Systems
Lecture 1

Mike Freedman & Wyatt Lloyd

Distributed Systems, What?



1) Multiple computers
2) Connected by a network
3) Doing something together

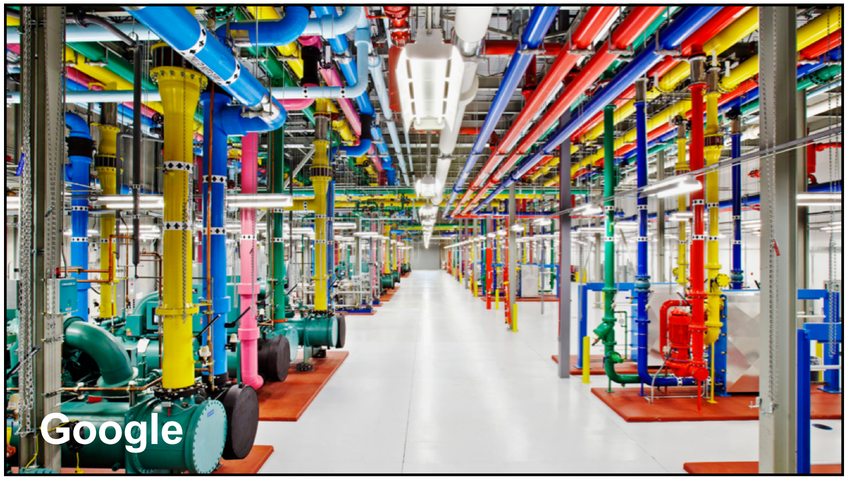
Distributed Systems, Why?

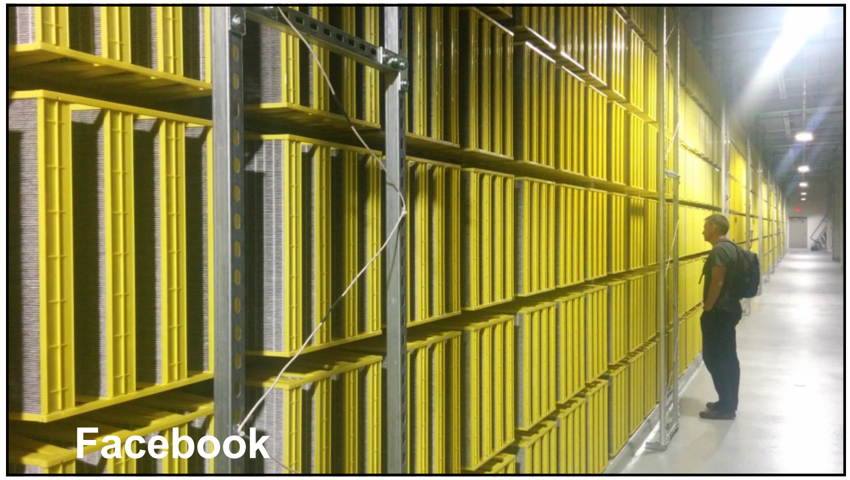
- Or, why not 1 computer to rule them all?
- Failure
- Limited computation/storage/…
- Physical location

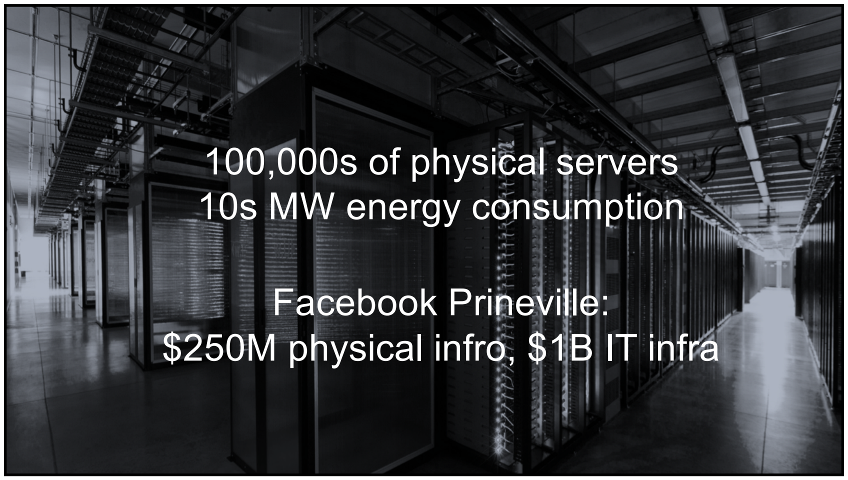
Distributed Systems, Where?

- Web Search (e.g., Google, Bing)
- Shopping (e.g., Amazon, Walmart)
- File Sync (e.g., Dropbox, iCloud)
- Social Networks (e.g., Facebook, Twitter)
- Music (e.g., Spotify, Apple Music)
- Ride Sharing (e.g., Uber, Lyft)
- Video (e.g., Youtube, Netflix)
- Online gaming (e.g., Fortnite, DOTA2)
- …

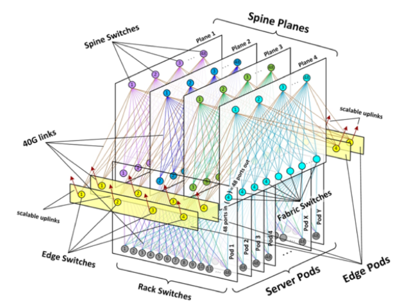
“The Cloud” is not amorphous







Everything changes at scale



"Pods provide 7.68Tbps to backplane"

Distributed Systems Goal

- Service with higher-level abstractions/interface
  - e.g., file system, database, key-value store, programming model, …
- Hide complexity
  - Scalable (scale-out)
  - Reliable (fault-tolerant)
  - Well-defined semantics (consistent)
- Do “heavy lifting” so app developer doesn’t need to

Decisions matter: Layering & Naming

- Abstractions everywhere: Layers partition the system
  - Each layer solely relies on services from layer below
  - Each layer solely exports services to layer above
- Interface between layers defines interaction
  - Hides implementation details
  - Layers can change without disturbing other layers

Decisions matter: Layering & Naming

- **Host names:** www.cs.princeton.edu
  - Mnemonic, variable-length, appreciated *by humans*
  - Hierarchical, based on organizations
- **IP addresses:** 128.112.7.156
  - Numerical 32-bit address appreciated *by routers*
  - Hierarchical, based on organizations and topology
- **MAC addresses** : 00-15-C5-49-04-A9
  - Numerical 48-bit address appreciated *by adapters*
  - Non-hierarchical, unrelated to network topology

Decisions matter: Layering & Naming

- **Host names:** www.cs.princeton.edu
  - Domain: registrar for each top-level domain (eg, .edu)
  - Host name: local administrator assigns to each host
- **IP addresses:** 128.112.7.156
  - Prefixes: ICANN, regional Internet registries, and ISPs
  - Hosts: static configuration, or dynamic using DHCP
- **MAC addresses:** 00-15-C5-49-04-A9
  - Blocks: assigned to vendors by the IEEE
  - Adapters: assigned by the vendor from its block

Research results matter: NoSQL

Dynamo: Amazon's Highly Available Key-value Store

Giuseppe DeCandia, Deniz Hastorun, Madan Jampani, Gunavardhan Kakulapati, Avinash Lakshman, Alex Pilchin, Swaminathan Sivasubramanian, Peter Vosshall and Werner Vogels

Amazon.com

Research results matter: Paxos

The Chubby lock service for loosely-coupled distributed systems

Mike Burrows, *Google Inc.*

Research results matter: MapReduce

MapReduce: Simplified Data Processing on Large Clusters

Jeffrey Dean and Sanjay Ghemawat

jeff@google.com, sanjay@google.com

Google, Inc.

hadoop

Spark

kafka

Flink

S4 distributed stream computing platform

STORM

Course Organization

Course structure

- Joint ugrad (418) + grad (518) course: first of kind
- Why / how do they differ?
  - 418 + 518: Both attend same lectures. Everybody needs background, few get it elsewhere
  - 418
    - Precepts (review/understanding material), TA led
    - Programming assignments
  - 518:
    - Recitations (paper reading + discussion), faculty led
    - Semester-long project

Learning the material: People

- Professors Mike Freedman & Wyatt Lloyd
- Teaching Assistants: Carlo Rosati, Jeffrey Helt, Jennifer Lam, Suriya Kodeswaran, Yue Tan
- Lab Assistants (for programming assignments)
- Main Q&A forum: www.piazza.com
  - No anonymous posts or questions, can DM instructors
  - Office hours (TAs and LAs) posted on Piazza
  - Setting expectation: TAs will monitor/respond to Piazza 1-2 times per day in a burst of activity

Learning the Material: Lectures!

- Lectures: MW 10-10:50 in CS 104
- Attend lectures and precepts and take notes!
  - Lecture slides posted day/night before
  - Recommendation: Print slides & take notes
  - Not everything covered in class is on slides
  - You are responsible for everything covered in class
- No required textbooks
  - Links to Go Programming textbook and two other distributed systems textbooks on website

418 specifics

Grading

- Five assignments (10% each)
  - 90% 24 hours late, 80% 2 days late, 50% >5 days late
  - **Three** free late days (we'll figure which one is best)
- Two exams (50% total)
  - Midterm exam before fall recess (25%)
  - Final exam during exam period (25%)

Weekly recitations (Friday)

- Supporting materials for class
  - Go programming
  - Problem solving around lecture topics
  - Things to think about for assignments
- Taught by TAs (rotation on weekly basis)

Assignment 1 (in three parts)

- Learn how to program in Go
  - Basic Go assignment (due 9/19)
  - "Sequential" Map Reduce (due 9/26)
  - Distributed Map Reduce (due 10/03)

Warnings

This is a 400-level course, with expectations to match.

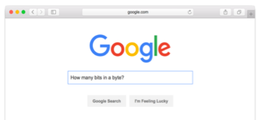
Warning #1:
Assignments are a LOT of work

- Assignment 1 is purposely easy to teach Go. Don't be fooled.
- Last year they gave 3-4 weeks for later assignments; many students started 3-4 days before deadline. **Disaster**.
- Distributed systems are hard
  - Need to understand problem and protocol, carefully design
  - Can take 5x more time to debug than "initially program"
- Assignment #4 builds on your Assignment #3 solution, i.e., you can't do #4 until your own #3 is working! (That's the real world!)

Warning #2:
Software engineering, not just programming

- COS126, 217, 226 told you how to design & structure your programs. This class doesn't.
- Real software engineering projects don't either.
- You need to learn to do it.
- If your system isn't designed well, can be *significantly* harder to get right.
- Your friend: test-driven development. We'll supply tests, bonus points for adding new ones

Warning #3:
Don't expect 24x7 answers



- Try to figure out yourself
- Piazza not designed for debugging
  - Utilize right venue: Go to office hours (TAs or LAs)
  - Send detailed Q's / bug reports, not "no idea what's wrong"
- Instructors are not on pager duty 24 x 7
  - Don't expect response before next business day
  - Questions Friday night @ 11pm should not expect fast responses. Be happy with something before Monday.
- Implications
  - Students should answer each other (+ it's worth credit)
  - Start your assignments early!

Policy: Write Your Own Code

Programming is an individual creative process. At first, discussions with friends is fine. When writing code, however, the program must be your own work.

Do not copy another person's programs, comments, or any part of submitted assignment. This includes character-by-character transliteration but also derivative works. Cannot use another's code, etc. even while "citing" them.

Writing code for use by another or using another's code is academic fraud in context of coursework.

Do not publish your code e.g., on github, during/after course!

Policy: Write Your Own Code

Programming is an individual creative process. At first, discussions with friends is fine. When writing code, however, the program must be your own work.

Do not copy another person's programs, comments, or any part of submitted assignment. This includes character-by-character transliteration but also derivative works. Cannot use another's code, etc. even while "citing" them.

Writing code for use by another or using another's code is academic fraud in context of coursework.

Do not publish your code e.g., on github, during/after course!

Don't Plagiarize!

518 specifics

Grading

- Semester-long project (40% total)
- Recitation participation (30% total
- Two exams (30% total)
  - Midterm exam before fall recess (15%)
  - Final exam during exam period (15%)
    - Mostly same midterm/final as 418, without Go/programming assignment related questions.

Recitations / paper readings

- One paper that everybody reads
  - Discuss paper at length in recitation
  - Be prepared: We'll cold-call students!
- Friday recitations: 1:30 – 2:30, 2:30 – 3:30 pm
  - Mandatory: Will record attendance + participation
- This Friday: Butler Lampson (Turing Laureate): "Hints for Computer System Design"

Course Project

- Groups of 2 per project
- Project schedule (to be posted online)
  - Team selection
  - Project proposal
  - Finalized project proposal
  - Interim project presentation
  - Final project presentation
  - Final project report published on Medium:
    https://medium.com/princeton-systems-course

Course Project: Options

- **Choice #1: Reproducibility**
  - Select paper from class or paper on related topic
  - Re-implement and carefully re-evaluate results
  - See detailed proposal instructions on webpage
- **Choice #2: Novelty** (less common)
  - Must be in area closely related to 518 topics
  - We will take a **narrow** view on what's permissible
- Both approaches need working code, evaluation

Topics Preview

Fundamentals

- Lectures
  - Network communication and Remote Procedure Calls
  - State in Network File Systems & the Web
  - Time, logical clocks
  - Vector clocks, distributed snapshots
- Precepts
  - Lots of Go
  - Mapreduce (assignment 1)

Eventual Consistency and Scaling Out

- Lectures
  - Eventual consistency and Bayou
  - Peer-to-peer systems and Distributed Hash Tables
  - Scale-out key-value storage and Dynamo
- Precepts
  - More Go
  - Distributed snapshots (assignment 2)

Replicated State Machines

- Lectures
  - Replicated State Machines, Primary-Backup
  - Reconfiguration and View Change Protocols
  - Consensus and Paxos
  - Leader Election and RAFT
- Precepts
  - Viewstamped replication
  - RAFT (Assignments 3,4)

Strong Consistency and Scaling Out with Transactions

- Lectures
  - Strong consistency and the CAP Theorem
  - Scalable Causal Consistency
  - Atomic commit and Concurrency Control
  - Spanner (Concurrency control + Paxos!)
  - The SNOW Theorem and Systems
- Precepts
  - Consistency
  - Concurrency control

Various Topics

- Lectures
  - Blockchains
  - Big data processing
  - Cluster scheduling and fairness
  - …
- Precepts
  - Big data systems