CSE 486/586 Distributed Systems Introduction

Steve Ko
Computer Sciences and Engineering
University at Buffalo

First Things

- Instructor
  - Steve Ko
  - 9th year at UB
- TAs
  - Harishankar Vishwanathan
  - Archita Pathak
  - Yuyang Chen
  - Hanbin Zhang
- Add/drop
  - Help your fellow students and make up your mind by Thursday (and drop if that's your decision).
  - On Friday, if necessary, I will increase the cap
  - Please don't email me for force registration until Friday night.

Welcome to CSE 486/586

- Why do you want to take this course?
- Some positive feedback of this course…
  - *"(CSE 486/586) didn't only helped with understanding the concepts involved, but have also always given me something cool and interesting to talk about in interviews."*
  - *"I am actually learning new things."*
  - *"(CSE 486/586) literally got me a job."*
- Some negative feedback of this course…
  - *"Projects are a bit too much on the difficult side."*
  - *"The midterm came almost out of nowhere."*
  - *"Stay away at all cost!"*
- Are you ready? ;-)

Building a Distributed System

- *"The number of people who know how to build really solid distributed systems…is about ten"*
  - Scott Shenker, Professor at UC Berkeley
- The point: it's hard to build a solid distributed system.
- So, why is it hard?...but first of all…

What is a Distributed System?

What is a Distributed System?

- *A distributed system is a collection of entities with a common goal, each of which is autonomous, programmable, asynchronous and failure-prone, and which communicate through an unreliable communication medium.*
- This will be a working definition for us.

Why Is It Hard to Build One?

- Scale: hundreds or thousands of machines
  - Google: 4K-machine MapReduce cluster
  - Yahoo!: 4K-machine Hadoop cluster
  - Akamai: 70K machines distributed over the world
  - Facebook: 60K machines providing the service
  - Hard enough to program one machine!
- Dynamism: machines do fail!
  - 50 machine failures out of 20K machine cluster per day (reported by Yahoo!)
  - 1 disk failure out of 16K disks every 6 hours (reported by Google)
- As we will learn, these come with:
  - Concurrent execution, consistency, etc.

OK; But Who Cares?

- This is where all the actions are!
  - What is the two biggest driving forces in the computing industry for the last 7-8 years?
  - It's the cloud!
  - And smartphones!
  - They are distributed!
  - (And there's also machine learning, robotics, etc.)
- Now --- it's all about distributed systems!
  - Well…with a bit of exaggeration… ;-)

OK, Cool; How Am I Going to Learn?

- Textbook
  - Main: Distributed Systems: Concepts and Design, 5th Edition (Coulouris, Dollimore, Kindberg, Blair)
  - Optional: Distributed Systems: Principles and Paradigms, 2nd Edition, (Tanenbaum, Van Steen)
- Lectures
- (Non-graded) HW assignments
- Programming assignments
- Exams

What Am I Going to Build?

- A "starter" project: PA1
  - This will be out today and due next week Friday.
- A distributed key-value storage (based on Amazon Dynamo) on Android in multiple stages
- Individual submission

Important Policies

- Late submissions only allowed for one day
  - 20% penalty
  - The deadlines are on Friday, and we don't count weekends, so technically you have 3 more days.
- Regrading
  - If requested, the entire work will be regraded
- No "I"
- No makeup exam
- No grade negotiation

I Have a Confession to Make…

- I have a split personality disorder.
  - Jekyll
  - Hyde
- Most of you (I expect) will just see my Jekyll's side. If you…
  - work with good ethics,
  - respect others on Piazza, during office hours, etc.,
  - follow class and submission rules,
  - and generally use common sense and are a good citizen in the class.
- Some of you might see my Hyde's side. If you…
  - copy other people's code or exams,
  - try to negotiate your way in the class,
  - generally are not such a good citizen in the class.

Academic Integrity Policies

- Academic integrity: exams, HW, and code
  - Copying others' code: no
  - Copying from other sources (the Web, books, etc.): get permission
  - Exceptions: http://developer.android.com (copy freely, but mark clearly that you copied)
  - http://stackoverflow.com (generally OK to see how things get done; but **do not copy and paste**.)
  - If found, the incident will be reported to the university.
- Will use an automatic similarity checker.
  - When similar submissions are found, both will get an F for the entire semester.
- Please be careful when using an online code repository, e.g., GitHub, BitBucket, etc.

How Can I Reach the Teaching Staff?

- Steve: 113F Davis
  - Lectures (MWF 1:00 pm-1:50 pm)
  - Office hours (TBD)
- TAs
  - Office hours: Posted on Piazza
  - Please do not expect that the TAs will stay more than the announced office hours.
- Use Piazza (http://piazza.com/class), instead of email, mailing list, blog, etc.
  - The teaching staff will not have any activity during weekends and holidays.
  - Signup link: http://piazza.com/buffalo/spring2019/cse486586
- http://www.cse.buffalo.edu/~stevko/courses/cse486/spring19/

For Undergraduates

- We do have recitations.
- But it's just like office hours, dedicated for undergraduates.

Background Required

- You must have some background in different topics.
- OS concepts
  - Threads, processes, synchronization (e.g., locks, semaphores), etc.
- Networking concepts
  - IP, DNS, NAT (e.g., private IPs vs. public IPs), TCP, etc.
- System programming experiences
  - Programming experiences with sockets, processes, threads, synchronization primitives, file I/O, etc.
  - Experiences with setting up environment variables, using regex, scripting (e.g., bash, python, etc.)
- Programming environment
  - Linux or Mac

Background Check: PA1

- Programming Assignment (PA) 1
  - Use this as a background check.
  - If you can finish this in a week all by yourself, then you are ready to take this class.
  - See for yourself!
  - Due on next Friday (2/8) 11:59:59 am.
- SimpleMessenger on Android
  - Overall, need to implement a chatting app.
  - Need to set up the Android programming environment.
  - Need to use sockets.
  - Need to understand the code provided.
  - Need to read Android tutorials and understand them.
  - Need to understand and use Android APIs.

What Exactly Am I Going to Learn? Distributed Systems 10 Questions!

- Course goal: answering 10 questions on distributed systems
  - At the end of the semester, if you can answer only 10 questions about distributed systems, you'll probably get an A.
  - Easy enough!
- What are those questions?
  - Organized in 6 themes
  - 1~2 questions in each theme
  - A few (or several) lectures to answer each question

Theme 1: Hint

Theme 1: Communications

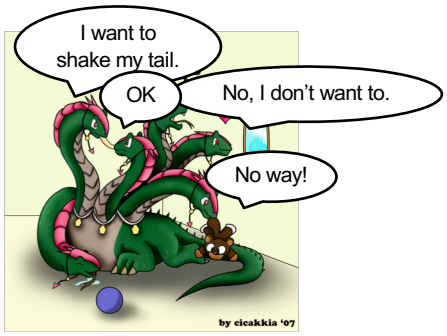
- Q1: how do you talk to another machine?
  - Networking basics
- Q2: how do you talk to multiple machines at once?
  - Multicast
- Q3: can you call a function/method/procedure running in another machine?
  - RPC

Theme 2: Hint

Theme 2: Concurrency

- Q4: how do you control access to shared resources?
  - Distributed mutual exclusion, distributed transactions, 2-phase commit, etc.

Theme 3: Hint



Theme 3: Consensus

- Q5: how do multiple machines reach an agreement?
  - Time & synchronization, global states, snapshots, mutual exclusion, leader election, paxos
- Bad news: it's impossible!
  - The impossibility of consensus

Theme 4: Hint

Theme 4: Storage Management

- Q6: how do you locate where things are and access them?
  - DHT, DFS

Theme 5: Hint

Theme 5: Non-Byzantine Failures

- Q7: how do you know if a machine has failed?
  - Failure detection
- Q8: how do you program your system to operate continually even under failures?
  - Replication, gossiping

Theme 6: Hint

Theme 6: Byzantine Failures

- Q9: how do you deal with attackers?
  - Security
- Q10: what if some machines malfunction?
  - Byzantine fault tolerance

Acknowledgements

- These slides heavily contain material developed and copyrighted by Indranil Gupta at UIUC.
- The material was originally developed for courses CS425/CSE424/ECE428 at UIUC.