Distributed Systems
CPSC 416
Winter 2018

Course: January 3 - April 6, 2018

Jan 3, 2018 Lecture (first class!)
Course staff

- Ivan Beschastnikh, instructor
- TAs
  - Anna Zheltukhina (u)
  - Renato Costa (g)
  - Gleb Naumenko (g)
  - Matthew Do (g)
Logistics

• 2016: 77 students (open-ended project)

• 2017: 117 students (assignment hell)

• 2018: 160 students (assignments + projects)

• 4 full TAs

• 2 easier (individual) assignments, one (group) defined project, one (group) open-ended project
Logistics

• Everything on the website, updated continuously:
  https://ugrad.cs.ubc.ca/~cs416/

• Use Piazza for all course-related communication

• 5 hrs office hours/week
• Learning goals
• Go programming language (start learning!)
• Schedule (a work in progress)
  • Assignment 1 due Jan 15 (week from Monday)
• Exam (‘just’ a final)
• Advice for doing well
  • learn Go (a must to pass the course)
  • don’t hack, engineer
  • choose team, wisely
  • reach out on Pizza/email for help.
• Collaboration guidelines
Assignment 1: Proof of work fortune
Distributed system examples

• YouTube

• Videos are **replicated** (multiple machines host the same video)

• **Scalable** wrt. client requests for videos (internally **elastic** — can throw more machines at the service to have it scale out further)
Distributed system examples

• DropBox (or google drive)
  • **Replicated** content across personal devices
    • Supports **disconnected operation** (can work while disconnected, and synchronize when re-connected)
    • Maintaining data **consistent** across devices
  • Supports sharing; **access control** policies (security!)
Distributed system examples

• NASDAQ

• **Transactions** (e.g., ACID semantics from databases). Many DBMS concepts apply to distributed systems!

• Strong **consistency** and **security** guarantees (otherwise people would not trust it with money)
Some D.S. challenges

• Synchronizing multiple machines (protocol complexity)

• Performance (how do you define/measure it?)

• Maintaining consistency: strong models (linearizable) to weak models (eventual) of consistency

• Failures: machine failures (range: failure stop to byzantine); network failures (just a few: disconnections/loss/corruption/delay/partitioning)

• Security (how to prevent malicious control of a single host in a system escalating into control of the entire system?)
For Friday

• Install Go on your personal machine

• Work through Tour of Go! and other tutorials.

• **Practice Go!**