CPS 512: Distributed Systems
Spring 2020

Class Meetings
TTh 1:25 – 2:40 in D106 LSRC

Instructor
Jeff Chase
Office hours: after class in D306 LSRC, or by appointment, or try a drop-in.

Teaching Assistant
Jiyao Hu

CPS 512 is an advanced course in distributed and networked systems. This offering of CPS 512 will focus on core concepts in distributed systems, using geo-distributed mega-services in the cloud as a motivation and driving example. Well-designed cloud applications are layered above common service platforms that handle the hard problems: tracking groups of participating servers (views), distributing state and functions across a group, coordinating control and ownership of data, managing consensus, and recovering from server and network failures. The course focuses on the design of these service platforms and their abstractions.

Although the course covers the fundamentals, the emphasis is on practical technologies and their limitations. This course includes an important software technology component. We will use the DSLabs projects from UW-Seattle, with an optional independent project component.

Topics. The course divides loosely into three parts (although we jump around a bit). The first part covers basic challenges and abstractions, focusing on the Google mega-service infrastructure stack, ranging from the network to cluster management, geo-replicated service structure, and storage. In the second part, we dive deeper into foundational distributed systems topics that underlie these systems: distributed transactions, geo-replication, logical time and causality, eventual consistency with vector clocks, views and leader election, and consensus. The third part shifts focus to secure Internet-scale systems with multiple identities and federation, showing how cryptosystems are used to manage naming, identity, authorization, blockchains, and crypto-currency (bitcoin). We also introduce trust logic as a formalism for building secure networked systems, and use it to represent the Internet security architecture and cloud access control. A detailed plan of topics is available on the course web [PDF]. Here is a rough overview:

- **Internet as a distributed system.** IP-LANs, DNS, routing.
- **Services and RPC.** Reliable RPC; NFS; caching, handling failures.
- **Foundations of mega-services.** Sharding, load spreading, state storage; elastic infrastructure.
- **The datacenter stack.** Linearizability, RPC revisited.
- **Storage abstractions.** Consistency; atomicity; transactions.
- **Causality and logical time.** Weak orderings and their application.
- **Consensus.** Coordination services, CAP, and CALM.
- **Distributed transactions.** Transactional concurrency control and deadlock.
- **Managing trust.** Byzantine failures; blockchains; governance.

Preparation. You should be comfortable with the fundamentals of operating systems and networks, and consider yourself a strong student and a good programmer. You will be programming in Java. You should be comfortable with Unix concepts and the Unix command interface.

Readings. There is no required textbook. Readings for this course consist of tutorials, surveys, and research papers written by researchers in networked systems. The readings are available through the course website. This course does not emphasize reading of research papers: although there are research papers to read, and you are expected to wrestle with them, the purpose of the readings is to support the concepts presented in lectures. The intent is to “distill” core concepts out of the readings, and build a solid
grounding in those concepts without getting lost in the details. Exams cover only the material discussed in class and represented on the lecture slides.

**Base workload.** In addition to the readings, there are four assigned programming assignments ("labs"), one midterm exam, and a final exam. The labs may be done individually or in groups of 2-3 (the first lab is individual). You may replace Lab 4 with an independent project of your choosing, in accordance with guidelines presented in class. In addition, there may be in-class quizzes and other exercises out of class. Quizzes are averaged into your exam score (time-weighted), and exercises are ungraded but may be testable on the exams.

**Calendar.** Here are the dates for Spring 2020:

- Jan 27 (M) Lab #1: Reliable RPC
- Feb 10 (M) Lab #2: Primary/backup replication
- Mar 5 (F) Lab #3: Consensus replication
- Mar 19 (Th) Midterm exam
- Apr 9 (F) Lab #4: Sharded key/value store, or project (plus two-week penalty-free extension)
- Apr 30 (Th) Final exam (9AM-noon)

**Late work.** Late work receives a penalty of up to 5% per day depending on circumstances. It is much better to do the work and hand it in late than to receive a zero on an assignment.

**Assistance.** We will provide online assistance through Piazza: see the course web. Please post your questions there. Anonymous posting is allowed: please maintain a high standard of civility. The instructor and TA hold regular office hours (see the course web) and are available at other times by arrangement. Drop-bys are welcome as time allows. If you are having trouble or just want to talk, please visit!

**Attendance and participation.** Attendance or lack of attendance in class/recitation is not recorded. However, it is expected that you will attend and participate actively. In particular, you should prepare questions or opinions about the reading, and I may ask you to speak in class. We may also have short written quizzes during class, which are factored (time-weighted) into your exam grade. You should expect a short quiz in every class, but we will often skip them.

**Grading.** The semester grade is determined from your exam/quiz grades (50%) and lab/project work (50%). I may make adjustments of up to half a letter grade for participation/engagement and other factors. Additional information about grading policies, project, and exams is available on the course web.

**Policy on collaboration for CPS 512.** The Duke Community Standard applies in all aspects of this course: we value your honor and your honesty. Collaboration on lab work and project work is encouraged. Help each other. However, any work you turn in must be your own, and you may be called upon to explain (alone) your choices and approaches in more detail. You may incorporate public software into your assigned lab work and course project to a reasonable extent, but not so much as to undermine the educational purpose and spirit of the project. You must acknowledge any sources of your ideas and software when they are not your own, and you must disclose in advance, without any specific request, any sources you used. Do not use code from a student who took the course in a previous semester or who completed related labs in another course or at another school. All students should understand that we have software that flags copied code with a high degree of certainty and precision. (The tools do not differentiate the makers from the takers.) No assistance of any kind is acceptable during exams.