CS-438  
Decentralized systems engineering

Ford Bryan Alexander

Summary
A decentralized system is one that works when no single party is in charge or fully trusted. This course teaches decentralized systems principles while guiding students through the development and testing of their own decentralized system incorporating messaging, encryption, and blockchain concepts.

Content
Topics this course covers include:

- Addressing, Forwarding, Routing. Peer-to-peer communication.
- Consistency Models. Disconnected operation, eventual consistency, conflict resolution.

Keywords
distributed systems, decentralized systems, security, privacy, anonymity, cryptography, gossip, consensus, swarming, blockchain, cryptocurrency

Learning Prerequisites
Required courses

• COM-208 Computer networks

Recommended courses

• CS-206 Parallelism and concurrency
• COM-301 Computer security
• CS-323 Introduction to operating systems

Important concepts to start the course

Students must already be highly competent at programming and debugging in a high-level systems programming language such as Java, C#, or Go. Programming exercises will be in Go, but students already well-versed and experienced in programming with comparable systems languages should be able to pick up Go during the course.

Students should have both solid foundational knowledge of how networks function, and some experience actually writing network programs, e.g., TCP/IP programming using the Sockets API.

Learning Outcomes

By the end of the course, the student must be able to:

• Implement systems via hands-on coding, debugging, and interoperability testing
• Design practical distributed and decentralized systems

Teaching methods

Lectures: The course's lectures will present and discuss challenges, known techniques, and open questions in decentralized system design and implementation. Lectures will often be driven by examination of real decentralized systems with various purposes in widespread use the past or present, such as UseNet, IRC, FreeNet, Tor, BitTorrent, and Bitcoin. Throughout the course we will explore fundamental security and usability challenges such as decentralized identification and authentication, denial-of-service and Sybil attacks, and maintenance of decentralized structures undergoing rapid changes (churn).

Labs: During the semester, students will develop a small but usable peer-to-peer communication application that reflects a few of the important design principles and techniques to be explored in the course, such as gossip, social trust networks, distributed hash tables, consensus algorithms, and cryptocurrencies. The labs will designed so that solutions can initially be tested individually on private, virtual networks running on one machine, then tested collectively by attempting to make different students' solutions interoperate on a real network.

Warning: This course is extremely programming-intensive. Students should be strong and confident in their programming skills in general, and be willing to spend substantial time outside of class debugging difficult distributed concurrency bugs and other challenges. TAs will be available to help at the exercise sessions, but they will not solve your problems or debug your code for you.

Expected student activities

Students will be expected to attend lectures to understand the concepts needed for the course, but the main workload will be actual hands-on programming assignments, which the students will perform on their own during the first part of the course and optionally in small teams during the final project-oriented part of the course.

Assessment methods
• Lecture attendance: 10%
• Programming assignment grading (evaluating both function and student documentation): 50%
• Final project grading (accounting for both scope, appropriateness, and follow-through in implementation quality and documentation): 40%

Resources

Moodle Link

• https://go.epfl.ch/CS-438