CS-438 Decentralized Systems Engineering

Week 2

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

CS-438: Decentralized Systems Engineering

Course Introduction and Basic Concepts

What is a Decentralized System?

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

An appealing but simplistic definition is a *system that works although no one's in charge*.

More precise definitions:

- *Distributed system*: a system of multiple computers (nodes) communicating over a network.
- *Decentralized system*: a distributed system in which different nodes or subsets of the network are owned or controlled by different people, organizations, or interests.

Some examples

- Centralized distributed systems
  - Google
  - Yahoo
  - Facebook
- Decentralized distributed systems
  - E-mail
  - UseNet
  - IRC
  - BitTorrent
  - Tor
  - Bitcoin
  - Ethereum

Course Goals

- Understand the fundamental challenges inherent in designing and building decentralized systems.
- Get a feel for the (limited) body of techniques and solutions we currently have for meeting these challenges.
- Examine a number of real, past and present systems, how they work, and how and why they succeeded or failed.
- Solidify this knowledge by applying it to the construction of a small, but working and usable, decentralized system.

Lecture 1: Course introduction

- Course goals
- **Course logistics**
  - Tentative schedule and topic outline
  - Course web sites and tools
  - Programming exercises overview

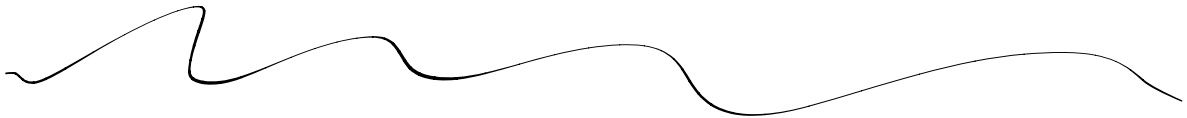
Tentative Course Syllabus

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Week	Content	Week	Content
1	*No lecture*	8	Replication and consensus
2	Course introduction	9	Sybil attacks and defenses
3	UseNet and gossip	10	Blockchains and cryptocurrencies
4	Flooding search and routing	11	Anonymous communication
5	Structured search and compact routing	12	Smart contracts
6	Cryptographic tools	13	Advanced blockchain architectures
7	Distributed storage	14	Decentralized democracy

Important course sites and tools

- Moodle for CS-438
  - https://moodle.epfl.ch/course/view.php?id=15483
  - Announcements and discussion forum
  - Slides, exercises, solutions
  - Project descriptions
- Contact address
  - Please E-mail the instructors at: cs438@groupes.epfl.ch
- Lecture style
  - Questions and discussion always welcome and encouraged!
  - Most of the course material will be explained with hand-drawn notes for interactivity.
  - Snapshots of in-lecture notes will be posted to Moodle after each lecture



Exercises and grading

- Weekly workload
  - Lectures: 2h - Mondays 10:15am-12:00pm in INF 1
    - Zoom link for remote participation: see Moodle
  - TA-guided Q&A sessions: 2h - Fridays 3:15pm-5:00pm in INJ 218
    - Q&A, help with architecting your system and understanding the requirements
    - *The TAs will not debug your code for you.*
  - Self-guided work sessions: 2h - Mondays 1:15pm-3:00pm in INF 1
    - Room open to hack, discuss designs/problems, and interop-test with colleagues
  - Homework: 3h-10h depending on your understanding of the concepts and programming skill
  - *No sharing or copying code; everyone implements their own Peerster!*
- Grading structure:
  - Labs: 60% of grade
  - Project: 40% of grade

Programming Exercise Structure

- Three exercise sets over the semester
  - Do not expect to do them last minute
  - Do not skip a part. Successive exercises build on the previous ones.
- TA-guided Q&A sessions
  - Discussion on design choices and how to architect your system
  - The TAs will not directly debug or fix your code for you!
- Main problem-solving and programming to be done “on your own”

End of semester project

- Idea will be developed during the semester
- Groups of 2 or 3
  - Every member must have a sub-project representing a readily-distinguishable contribution
  - Together the sub-projects should integrate into a running project
  - Each member will be asked question on his sub-project and evaluated independently
- Must be a project in the scope of the course
  - E.g. adding anonymity to peerster
  - Many potential topics will present themselves throughout the lectures and exercises
  - You're welcome to come up with your own; ***ask*** if you're uncertain it's in-scope

Project Schedule (tentative)

- Project proposal by student teams: week 5 (approx)
- Feedback on proposals: week 7 (approx)
- Project proposal refinement: week 8 (approx)

Why study decentralized systems?

- **Direct applicability**: only a decentralized system can solve computing problems in which there is no common authority everyone trusts.
- **Indirect applicability**: real life is full of decentralized systems — but unlike other relevant fields (econ, soc-sci, pol-sci), we get to build and not just study them.
- **Intellectual challenge**: making decentralized systems work reliably and securely is often fundamentally more difficult than centralized systems, because we get to make fewer simplifying assumptions — particularly about security.

Why Build a Decentralized System?

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

- Sometimes a basic requirement: e.g., federated or "business-to-business" systems in which there is no natural trusted authority.
- Decentralized system components/subsets may be more autonomous, and hence more available or reliable in the face of disconnection or network partition, if they aren't dependent on regular communication with a central authority.
- Successful decentralized systems can become vibrant, diverse evolutionary ecosystems when alternative, competing node implementations can coexist and interoperate but evolve independently (e.g., IRC clients, P2P clients).
- This diversity can in turn make a decentralized system less vulnerable to attack or infection: only a subset of the complete system is likely to be vulnerable to any given attack or infection.

Why might we prefer centralized systems?

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

- **Management simplicity**: it's much simpler to manage an army of obedient slaves; “managing” a decentralized system is akin to herding cats.
- **Security model simplicity**: a lot of hard security challenges become easy when you can just ask a central authority who we assume (rightfully or not) to be trustworthy.
- **Efficiency**: A centralized management and security model can allow for simpler and more efficient algorithms, less overhead due to crypto, etc.
- **Version compatibility**: one can limit the number/variety of software versions that exist in a centralized system, but a decentralized node might have to (try to) interoperate with any past, future, or independently developed software/hardware combination on another node.

Major General Topics and Applications

- Communication: messaging, chat, voice/video
- Decentralized search and data mining facilities
- Collaboration mechanisms (e.g., wiki)
- Social networking
- Deliberation, peer review (voting, reputation systems)
- Blockchains, cryptocurrencies, and smart contract systems

This course inherently has a heavy security component — which we approach not primarily from a formal or cryptographic perspective, but from a pragmatic systems perspective.

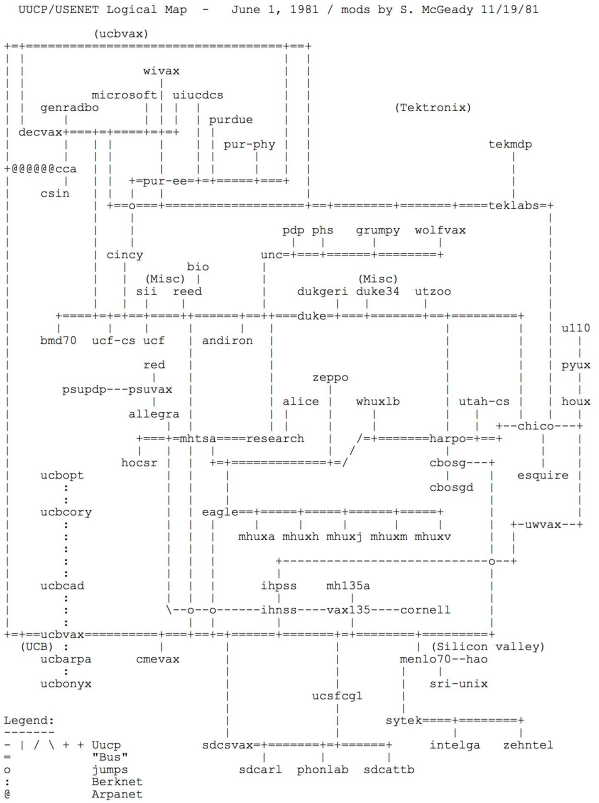
Recurrent issues and themes

- Identity (real, sybil) versus location
- Information integrity and privacy
- Behavior accountability
- Denial-of-service
- Protocol efficiency, in the normal case and under load or attack

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Gossip and USENET

Early UUCP/USENET Map



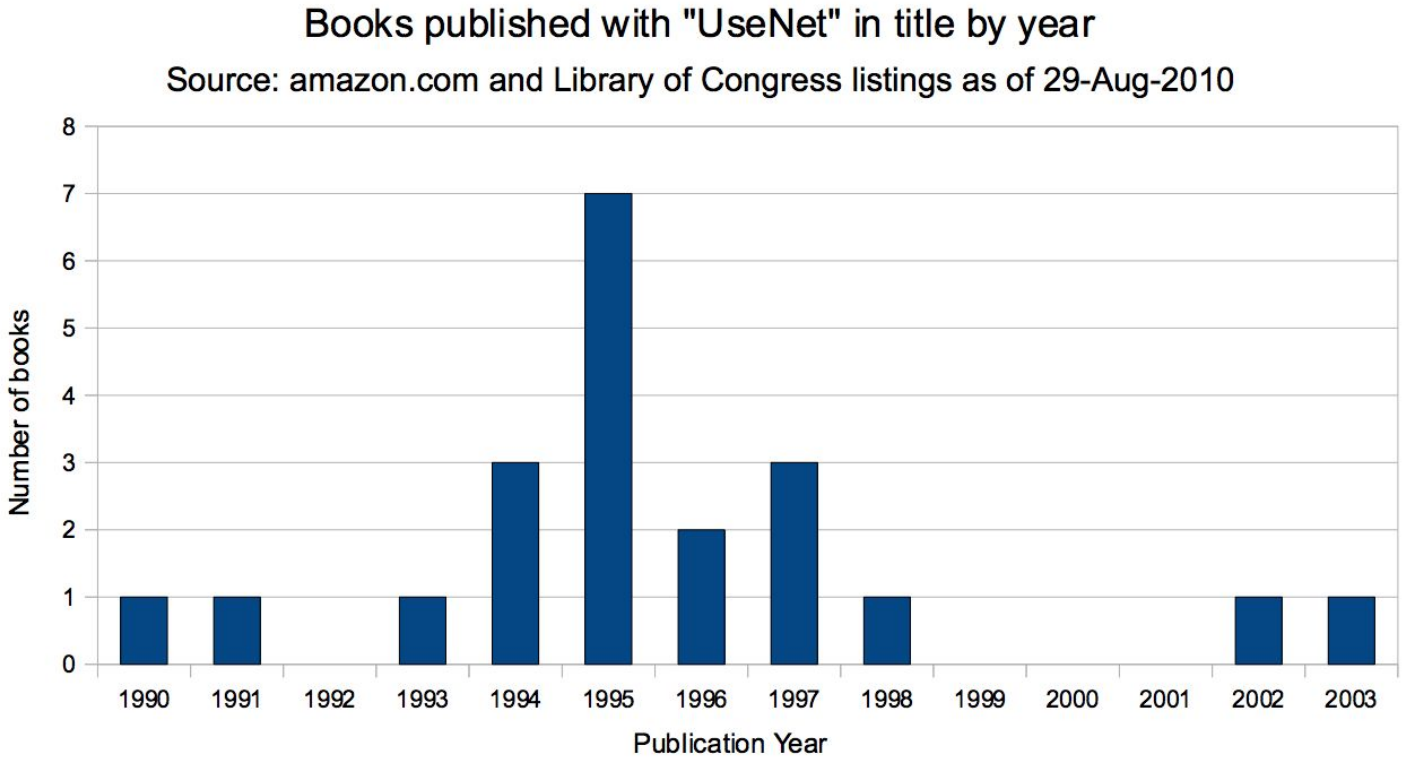
1979 - created

1981

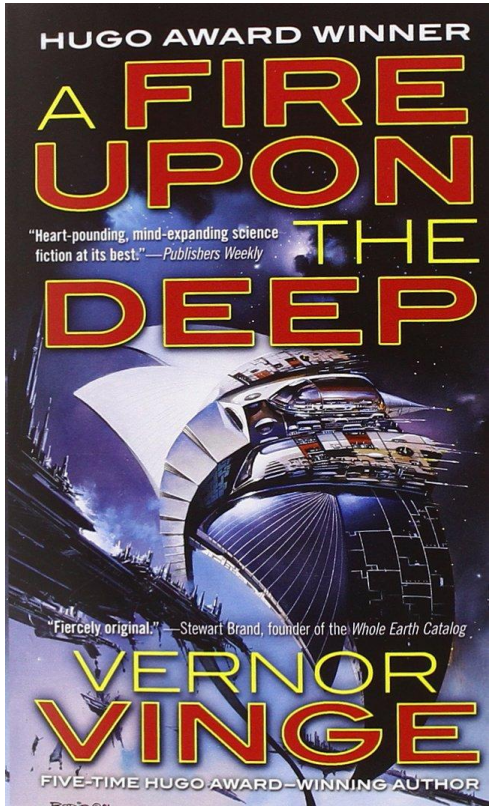
1986 - ARPAnet / Internet

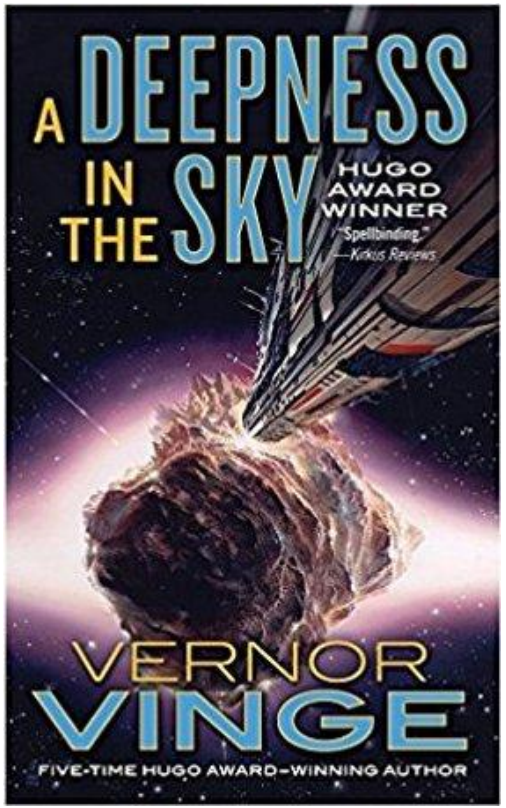
1987 - "The Great Renaming"

The Life and Death of USENET



Even inspired an “Intergalactic USENET”





Example USENET message