Goals for Today’s Class

- **EE 122 overview**
  - Goals of the course
  - Structure of the course
  - Instructor & TAs
  - Prereqs & assignments
  - Course grading
  - Academic policies
- **What makes networking challenging**
  - The fundamental issues we must grapple with to build a global Internet

What You Learn in This Course

- **Insight**: key concepts in networking
  - Protocols
  - Layering
  - Resource allocation
  - Security
  - Naming
- **Knowledge**: how the Internet works
  - Internet architecture
  - IP protocol suite
  - Applications (Web, e-mail, P2P, …)
- **Skill**: network programming
  - Socket programming
  - Designing and implementing protocols

What This Course Is and Isn’t

- **EE122 comes in two flavors:**
  - **Spring offering**: taught by EE faculty
    - More emphasis on diverse link technologies, wireless & mobility, communication theory & simulation
  - **Fall offering**: taught by CS faculty
    - More emphasis on Internet technology, applications, practice & empiricism / hands-on
  - Differences aren’t huge, though
  - **My particular emphasis**:
    - Network architecture, naming, resource allocation
    - Overlays, peer-to-peer applications

Structure of the Course (1st Half)

- **Start at the top**
  - Protocols: how to structure communication
  - Sockets: how applications view the Internet
- **Then study the “narrow waist” of IP**
  - IP best-effort packet-delivery service
  - IP addressing and packet forwarding
  - And how to build on top of the narrow waist
  - Transport protocols (TCP, UDP)
  - Domain Name System (DNS)
  - Applications (Web, email, file transfer)
  - Looking underneath IP
  - Link technologies (Ethernet, bridges, switches)

Structure of the Course (2nd Half)

- **How to get the traffic from here to there …**
  - Routing (intradomain, interdomain)
  - Glue (ARP, DHCP, ICMP)
  - … in a way that’s both **efficient** and **stable**
    - How much data to send without clogging the sender (flow control) or the network (congestion control)
    - With some assurance (quality of service) … or not
  - **How to control network traffic …**
    - Enforcing policy
    - Defending against attacks
  - … and scale it to potentially huge structures
  - Peer-to-peer & overlays
**Instructor**
- Ion Stoica ([istoica@cs.berkeley.edu](mailto:istoica@cs.berkeley.edu))
  - Research focus
    - Network architectures
    - Tracing & debugging distributed systems
    - Overlay & p2p networks
  - [http://www.cs.berkeley.edu/~istoica/](http://www.cs.berkeley.edu/~istoica/)
  - Office hours M 1-2PM & W 2-3PM in 413 Soda
  - Phone: 643-4007
  - Email preferred!

**Interact!**
- Inevitably, you won’t understand something(s) …
  - that’s my fault, but you need to help
- Come to office hours, request an appointment, communicate by e-mail
  - We are here to help, including general advice!
  - TAs first line for help with programming problems
- Give us suggestions/complaints/feedback as early as you can
- What’s your background? Tell us at
  - [http://tinyurl.com/5rsdx5](http://tinyurl.com/5rsdx5)

**TAs**
- Lucian Popa ([popa@eecs.berkeley.edu](mailto:popa@eecs.berkeley.edu))
  - Office hours: W 1:30-2:30, F 11:30-12:30pm in 413 Soda
  - And by appointment
  - Section: W 12-1pm (299 Cory) & F 10-11am (293 Cory)
- David Zats ([dzats@eecs.berkeley.edu](mailto:dzats@eecs.berkeley.edu))
  - Office hours: TT, 2:30-3:30 (location TBA)
  - And by appointment
  - Section: M 2-3pm (237 Cory), T 10-11am (299 Cory)
- Another TA will join next week

**Course Materials**
- Textbooks
    - Note, we jump around in it a lot
  - Recommended & on reserve:
  - Web site: [http://inst.eecs.berkeley.edu/~ee122/](http://inst.eecs.berkeley.edu/~ee122/)
    - Updated frequently, including lecture slides
    - Note: if you are following the slides during lecture, please don’t use them to answer questions I ask
  - Mailing list: ee122@lists.berkeley.edu
    - Sign up: [https://calmail.berkeley.edu/manage/list/listinfo/ee122@lists.berkeley.edu](https://calmail.berkeley.edu/manage/list/listinfo/ee122@lists.berkeley.edu)

**Brighten Godfrey**
- [pbq@cs.berkeley.edu](mailto:pbq@cs.berkeley.edu)
- 2008 graduate
- Will join UIUC in Fall 2009 as Assistant Professor
- Will help with teaching several lectures

**Class Workload**
- Four homeworks spread over the semester
  - Strict due dates (no slip days!)
  - Unless otherwise specified, deadlines are before lecture starts
- Two projects
  - Chat application (teams of two)
    - Includes socket programming, client/server
  - Dynamic DNS (teams of two)
    - You design and implement a DNS server
  - C (or C++) required
  - Deadlines 11:50PM
  - These are extensive undertakings, particularly the second
- Exams
  - Midterm: Monday, October 13 in class, 4-5:30PM
  - Final: Thursday, December 18 location 277 Cory, 8-11AM
  - Closed book, open crib sheet
Prerequisites

- CS 61A, 61B
  - Programming, data structures, software engineering
  - Knowledge of C or C++
- Math 53 or 54
  - In fact, we'll be relatively light on math, though your algebra should be very solid, you should know basic probability, and you'll need to be comfortable with thinking abstractly
- Background material will not be covered in lecture. TAs will spend very little time reviewing material not specific to networking

Grading

<table>
<thead>
<tr>
<th>Component</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homeworks</td>
<td>20% (5% each)</td>
</tr>
<tr>
<td>Projects</td>
<td>40% (20+20)</td>
</tr>
<tr>
<td>Midterm exam</td>
<td>15%</td>
</tr>
<tr>
<td>Final exam</td>
<td>25%</td>
</tr>
</tbody>
</table>

- Course graded to mean of B
  - Relatively easy to get a B, harder to get an A or a C
  - 10% A, 15% A-, 15% B+, 20% B, 15% B-, 15% C+, 10% C
  - A+ reserved for superstars (1 or 2 per class)
  - Mean can shift up for an excellent class
  - For which the TAs have significant input

No Cheating

- Cheating means not doing the homework by yourself
- Fine to talk with other students about homeworks outside of class
  - Concepts, not specifics
  - No copying, no Google, etc
  - If you're unsure, then ask
- We will do automated similarity detection on assignments.

What do this two have in Common?

- First printing press
  - Key idea: splitting up text in individual components
  - E.g., lower, upper case letters
- Bible: first mass produced book
- Johann Gutenberg (1398-1468)
- The Internet
  - Both lower the cost of distributing information

The ARPANet

- Paul Baran
  - RAND Corp, early 1960s
  - Communications networks that would survive a major enemy attack
- ARPANet: Research vehicle for “Resource Sharing Computer Networks”
  - 2 September 1969: UCLA first node on the ARPANet
  - December 1969: 4 nodes connected by phone lines
Networking: Actually Not Boring

- How hard can it be?
- You just string a wire (or other signaling path) between two computers …
- … first one pushes bits down the link …
- … and the second one gets them up. Right?
- Where does it get tricky? What are the challenges?

Why Networking Is Challenging

- Fundamental challenge: the speed of light
- Question: how long does it take light to travel from Berkeley to New York?
- Answer:
  - Distance Berkeley → New York: 4,125 km
  - Traveling 300,000 km/s: 13.75 msec
Fundamental Challenge: Speed of Light

- Question: how long does it take an Internet "packet" to travel from Berkeley to New York?
- Answer:
  - For sure ≥ 13.75 msec
  - Depends on:
    - The route the packet takes (could be circuitous!)
    - The propagation speed of the links the packet traverses
      - E.g., in optical fiber light propagates at about 2/3 C
    - The transmission rate (bandwidth) of the links (bits/sec)
      - and thus the size of the packet
    - Number of hops traversed (store-and-forward delay)
    - The "competition" for bandwidth the packet encounters (congestion). It may have to sit & wait in router queues.
  - In practice this boils down to: ≥ 40 msec

Why Networking Is Challenging (con't)

- Fundamental challenge: components fail
  - Network communication involves a chain of interfaces, links, routers and switches ...

Fundamental Challenge: Speed of Light

- Question: how many cycles does your PC execute before it can possibly get a reply to a message it sent to a New York web server?
- Answer:
  - Round trip takes ≥ 80 msec
  - PC runs at (say) 3 GHz
  - 3,000,000,000 cycles/sec * 0.08 sec = 240,000,000 cycles
  - Thus,
    - Communication feedback is always dated
    - Communication fundamentally asynchronous

Examples of Network Components

![Network Components Diagram](image)

Why Networking Is Challenging (con't)

- Fundamental challenge: components fail
  - Network communication involves a chain of interfaces, links, routers and switches ...
  - ... and all of which must function correctly.
- Question: suppose a communication involves 50 components which work correctly (independently) 99% of the time. What’s the likelihood the communication fails at a given point of time?
- Answer: success requires that they all function, so failure probability = 1 - 0.99^50 = 39.5%.
- So we have a lot of components, which tend to fail ...
  - ... and we may not find out for a looong time
Why Networking Is Challenging (con’t)

- **Challenge:** enormous dynamic range
  - Round-trip times (latency) vary 10 µsec’s to sec’s ($10^6$)
  - Data rates (bandwidth) vary from kbps to 10 Gbps ($10^7$)
  - Queuing delays inside the network vary from 0 to sec’s
  - Packet loss varies from 0 to 90+%%
  - End system (host) capabilities vary from cell phones to supercomputer clusters
  - Application needs vary enormously: size of transfers, bidirectionality, need for reliability, tolerance of jitter
- Related challenge: very often, there is no such thing as “typical”. Beware of your “mental models”!
  - Must think in terms of design ranges, not points
  - Mechanisms need to be adaptive

Why Networking Is Challenging (con’t)

- **Challenge:** different parties must work together
  - Multiple parties with different agendas must agree how to divide the task between them
- Working together requires:
  - Protocols (defining who does what)
    - These generally need to be standardized
  - Agreements regarding how different types of activity are treated (policy)
- Different parties very well might try to “game” the network’s mechanisms to their advantage

Why Networking Is Challenging (con’t)

- **Challenge:** incessant rapid growth
  - Utility of the network scales with its size
  - Fuels exponential growth (for more than 2 decades!)
  - Adds another dimension of dynamic range …
    - … and quite a number of ad hoc artifacts
Why Crooks Matter for Networking

- They (and other attackers) seek ways to misuse the network towards their gain
  - Carefully crafted “bogus” traffic to manipulate the network’s operation
  - Torrents of traffic to overwhelm a service (denial of service) for purposes of extortion / competition
  - Passively recording network traffic in transit (sniffing)
  - Exploit flaws in clients and servers using the network to trick into executing the attacker’s code (compromise)
- They do all this energetically because there is significant $$$ to be made

Why Networking Is Challenging (con’t)

- Challenge: you cannot reboot the Internet!
  - Everyone depends on the Summary

- Networking is about design in the presence of challenges/constraints:
  - Not akin to e.g. programming languages / compilers
    - Which have well-developed theories to draw upon
  - Much more akin to operating systems
    - Abstractions
    - Tradeoffs
    - Design principles / “taste”
- Next lecture: types of networks, protocols
  - Read through 1.3 of the Kurose/Ross book
  - Subscribe to the mailing list: ee122@lists.berkeley.edu
  - Take survey at: http://tinyurl.com/5r6dx5