CPS 214: Computer Networks

Slides by Adolfo Rodriguez

Paper Evaluations

- 1 page *maximum* evaluation of reading for each class
- Evaluations submitted *in advance* of class from course Web page
- Describe:
  - Biggest contribution of the paper
  - Most glaring problem(s) with the work
  - What the work implies for networking research
- Graded on 0-3 scale, standards increase progressively
- Can skip ~1/4 of the evaluations
- May institute pop quiz policy

Programming Assignments

- Work in groups of 1-2 (recommend 2)
  - Cannot work with the same person on more than one assignment
  - Contact me with concerns
- Each assignment consists of:
  - Code
  - Write-up that discusses what you learned and how to compile/use the code
- Grading criteria will be made available 1 week before the deadline

Term Project

- *Aim high!*
  - The best projects can result in publication in top conferences
- Work in groups of 2 (may reuse a partner)
  - Talk to me about different size groups
  - Best to have others to keep things moving
- List of suggestions will be available from course Web page
- Schedule of milestones
  - Description of interests (groups formed) by Feb 12
  - Topic chosen by Feb 26

General Goals

- Gain background in networking (+ distributed systems)
  - Textbook
- Understanding of issues in networking
  - Study of relevant research papers
  - Discuss issues in and out of class
- Develop the skills to perform research in this area
  - Three programming assignments
  - Term project (work in teams)
- Develop writing/presentation skills
  - End of term mini-conference
  - Project term paper

Non-Goals

- Teach the basics of systems programming
  - CPS 110/CPS 108 (or equivalent) are prerequisites
  - Some knowledge of distributed programming
  - Background reading available
  - Learn everything about Sockets programming
- 75 minute soliloquies
  - Lectures should be interactive
  - Leverage our setting
- Insulate the professor from the students
  - Schedule an appointment
  - Drop by when I'm there

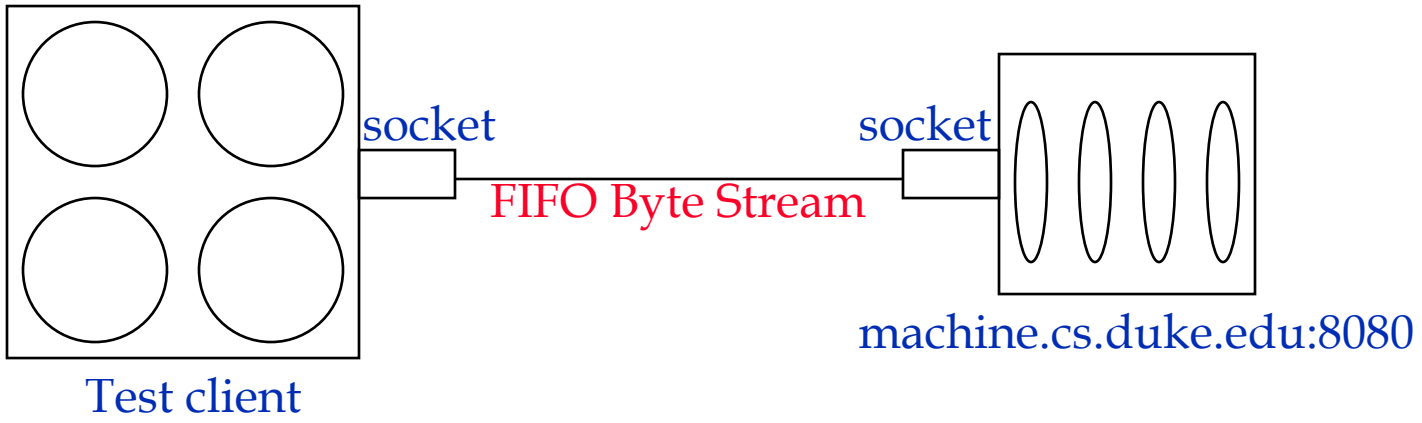
Your Goals

- To influence course content, present *your* goals for the class
- What would you like to learn?
- What do you think is important?
- Schedule time to meet with me to discuss how you think the class is progressing

Programming Assignment 1

- Build an HTTP web server
  - Full description on the web page
  - Implement in C/C++/Java (preference for C/C++)
- Evaluate your server under varying workloads
- Extra credit: multiple programming models
- Form groups of 2
  - All the code should be your own
- Due date: February 6

Background



- Test client and web server
  - Multi threaded versus multi process
  - Pooling strategies
- Event driven web server

Sockets API

- Creating a socket

  int socket(int domain, int type, int protocol)

  domain = AF_INET, AF_UNIX

  type = SOCK_STREAM, SOCK_DGRAM

- Passive Open (on server)

  int bind(int socket, struct sockaddr *addr, int addr_len)

  int listen(int socket, int backlog)

  int accept(int socket, struct sockaddr *addr, int addr_len)

  int select(int n, fd_set *readfds, fd_set *writefds, fd_set *exceptfds, struct timeval *timeout);

Sockets API

- Active Open (on client)

  int connect(int socket, struct sockaddr *addr,
  int addr_len)

- Sending/Receiving Messages

  int send(int socket, char *msg, int mlen, int flags)

  int recv(int socket, char *buf, int blen, int flags)

- How does TCP socket differ from UDP socket?
  - sendto/recvfrom

Server/HTTP Protocol

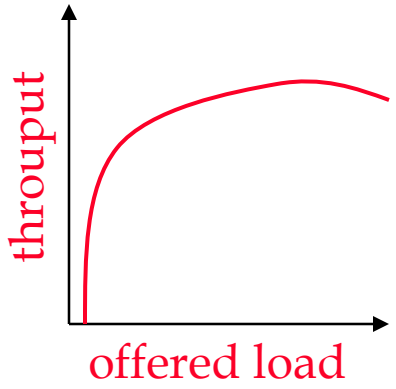
- HTTP Server
  - Creates a socket (*socket*)
  - *Bind*s to an address
  - *Listen*s to setup accept backlog
  - Can call *accept* to block waiting for connections
  - Can call *select* to check for data on multiple socks
- Requests (hand off to separate thread? separate process?)
  - GET /index.html HTTP/1.0\n
    <optional body, multiple lines>\n
    \n

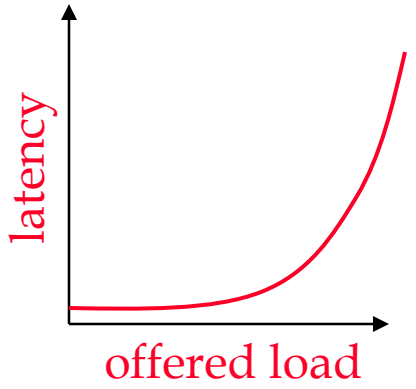
Performance Evaluation

- Subject web server to varying levels of *offered load*
  - A parameter to your test app specifies how many clients are simultaneously requesting the same resource
  - Vary the size of the requested resource
  - As a function of offered load

    Measure throughput (requests/sec, mbits/sec)

    Measure latency





Course Outline

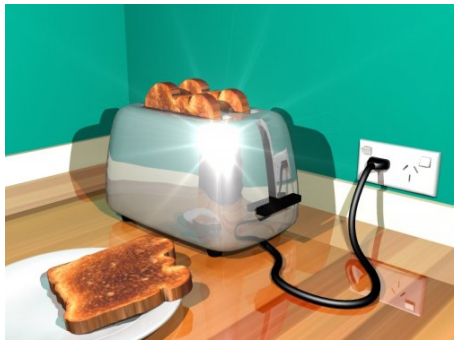
- Introduction and Packet-Switched Networks
  - What is underneath the host-to-host communication abstraction?
- Data-Link Layer
- Internetworking
  - Not all computers are directly connected
- End-to-End Protocols
  - E.g., provide the abstraction of a reliable byte-stream over error-prone, packet-switched network
- Applications

Course Outline (cont.)

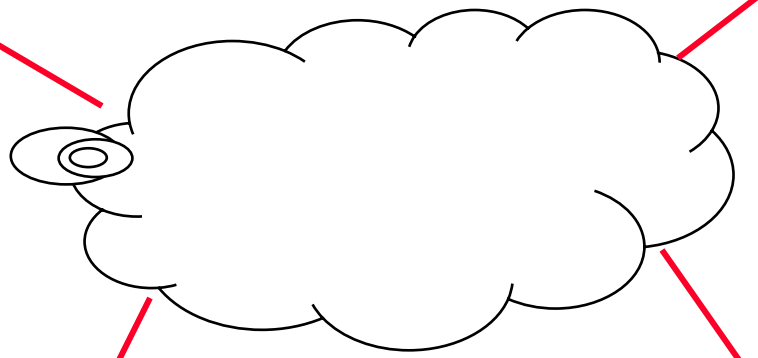
- Peer to peer networks
  - An old idea with new applicability?
- Overlay networks
  - Related to multicast
  - Application-layer technique for efficient data delivery
  - Used in Content Distribution Networks
- Security
- Wireless Networks
  - Mobility/embedded processors
- Current challenges
  - Resource allocation/reservations (admission control)

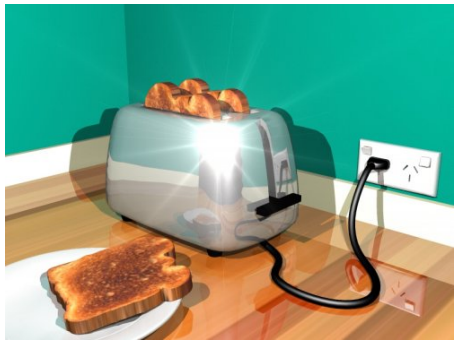
Course Goal: Scalable, Arbitrary Communication

Course Goal: Scalable, Arbitrary Communication



Course Goal: Scalable, Arbitrary Communication





Challenges to Achieving Universal Communication

- How to connect computers
  - Cannot have all-to-all connections
- How to *name* and *locate* computers
  - Billions of computers: translate name into physical location
- Routing
  - Transmitting messages from one computer to another
- Software/Protocols
  - Not just send messages, must agree on format and interpretation
- Reliability
  - Networks drop, corrupt, and reorder messages

➔ Common challenge is *scalability*

Challenges on the Horizon

- Computers embedded in all devices (toasters, library books, etc.)
  - Desktop processors less than 5% of the market
  - Does routing/congestion control scale to all processors?
  - Will they all run TCP?
- Internet telephony
  - Data traffic surpasses voice
  - Can the Internet provide 24/7 reliability? (e.g., mission critical applications)
  - Fundamental differences in data vs. voice networks

    Will there be a convergence?

Challenges on the Horizon

- Widespread use of ADSL and cable modems
  - Today, modems limit Internet use
  - What if lots of TCP cheats? (TCP as game theory)
- Web Services
- Mobility/wireless support
  - How to route packets to a wireless host
  - How to *name* them in the first place
  - Emerging technologies: Bluetooth, HomeRF
- What happens when bandwidth and computation become free?

New Directions

- Peer to peer (P2P)
  - Leverage the dark matter of the computing universe
  - Napster
  - Kazaa
- Overlay Networks
  - An instance of P2P
  - Computation in end-systems
  - Builds on best-effort Internet
  - Delivers rich application-semantics

    Multicast

    QoS

New Directions

- Internet evolution:
  - 1970's: telnet, email
  - 1980's: email, ftp
  - 1990's: email, http
  - 2000's: email, instant messaging, http, P2P, wireless
  - 2010's: email, http?, video mail?, virtual reality?, ???
- How are things different?
  - What support will we need in the network?
  - How much will the Internet scale?