

Data Communication and Networks

Overview

Instructor: Jay Lepreau, lepreau@cs.utah.edu, <http://www.cs.utah.edu/~lepreau>

4548 MEB (enter through 4560 in East Penthouse), ext. 1-4285.

Office hours: Wed 11:30–12:30, 2:30–3:30. Earlier slot is preemptible by faculty meetings.

Time: Tu–Th 2:00–3:20

Place: EMCB 102

Teaching Assistants: Sachin Goyal (sgoyal@cs); office hours in EMCB 210:

Mon/Wed: 11:00–13:00, Fri 15:00–17:00.

Abhijeet Joglekar (abhijeet@cs, half-time); office hours Mon 4:00–5:00, Tue/Thu 3:30–4:30.

Text: *Computer Networks: A Systems Approach* by Peterson and Davie, 2nd edition. Full information, including cheaper online sources, is on the course Web page. Supplementary material on sockets programming will be made available on the Web page. In addition, many of you may want to purchase a small but good book on sockets programming: Donahoo and Calvert, *TCP/IP Sockets in C: Practical Guide for Programmers*, Morgan-Kaufman, 2001, ISBN 1-55860-826-5. To help you decide, the TA's will have a copy for your perusal during office hours.

Overview of assignments and grading: Your grade will be based on a combination of programming projects, written homeworks, and exams. The probably breakdown of points from each type of assignment is as follows (subject to change):

- Exams (2): 50%
- Projects (3): 34%
- Homeworks (2): 16%

Computer Accounts: If you do not have a CADE account, let the TAs know as soon as possible so that one can be created for you.

Electronic class information: Copies of the course handouts, lecture notes, project software, and course schedule will be made available under the class Web page, <http://www.cs.utah.edu/classes/cs5480/>. Look at this page; everything you need will be linked from here. You are expected check it at least once a week for new information.

Other sources of information include:

- teach-cs5480@cs sends mail to myself and the TAs
- cs5480@cs sends mail to the *entire class*, including myself and the TAs—*use sparingly*
- /home/handin/cs5480 directory on the CADE lab machines.

Mailing list: It is essential to join the class mailing list immediately by sending email to majordomo@cs with `subscribe cs5480` in the body. The subject line is ignored by majordomo—do not put your `subscribe` command in the subject and expect it to work.

Course Description

This course introduces the basics of networking, ranging from sending bits over wires, to the Web and distributed computing. However, we focus on the internetworking ground between these two extremes, emphasizing distributed, large-scale, heterogeneous networks such as the Internet. The goal of the course is to give students an appreciation of the fundamental challenges of networking, design strategies of proven value, and common implementation technologies. We will cover the following topics: framing, error correction, packet switching, multi-access (Ethernet), addressing and forwarding (IP), distance vector and link state routing, queuing and scheduling, reliable transport, congestion control (TCP), quality of service, naming (DNS), and security. If there is time we may include more on networking issues for multimedia.

My measure of success for this course will be the extent to which you leave it with a solid intuition about how all the elements that compose a modern computer network operate and interoperate, and the extent to which you have enough specific knowledge to apply this intuition to real problems. After you have completed this course, you should be better able to evaluate the impact of new network hardware and protocols in a variety of settings, develop new (or modify existing) network designs to optimize behavior for particular applications or environments, and understand how seemingly unrelated design features can either greatly improve or reduce network performance. As such, we will not dwell on any specific collection of networking hardware and software systems, but will instead concentrate on the issues that are common to most modern network architectures.

You should read the text associated with each lecture before the corresponding material is covered in lecture. Sometimes I will cover different and additional material, or present similar information differently. I may make my lecture notes available before class so that you can print them out and have them available during the lectures, which will improve the efficiency of your note taking.

Finally, because you will not really learn the material unless you *do* something with it, i.e., homeworks and projects, this course will involve a number of programming projects in addition to homeworks. The projects will be progressively more rigorous and time consuming, but should be rewarding in terms of improving your understanding of the material. *Be sure to get started on the projects as soon as I hand them out!*

Grading

Projects

This course involves substantial programming in C, on Unix. If you are not comfortable with programming, you should take this course some other quarter.

The projects are being refined, so the following is subject to change. The current plan is to assign three projects, as follows:

- **Simple client-server** (8%): For this assignment, you will be asked to implement both the client and server for a simple web-like system. It will familiarize you with the basics of programming with sockets. This assignment will be an *individual* (not group) assignment.
- **Congestion and flow control** (10%): *This assignment in particular is subject to change.* For this assignment, you will be asked to implement congestion and flow control mechanisms ranging from the simplest (stop and wait) to something akin to the mechanisms used by a modern TCP implementation. This assignment will be *individual* (not group) assignment.
- **Complex client-server** (16%): For this assignment, you will be asked to implement a complex peer-to-peer system, complete with multiple simultaneous connections between client-

server pairs combined with a non-trivial distributed application. This assignment will be a *group* assignment, with two people per group.

While implementing a programming assignment, you are encouraged to discuss the project with other members of the class, the TAs, or myself. However, you *may not share code or specific solutions* or *use code from outside sources!* By this I mean you can discuss the problem, possible solutions, basic algorithms, and general approaches for attacking the problem, but *you are responsible for writing all of your own code!* Do not look at another student's code or show another student your code. Similarly, when giving or receiving assistance, it is not legal to write out specific algorithms in C/C++ that will solve the problem. Exchanging code or specific solutions is *cheating*, and will be dealt with harshly, as described below. Source code will be checked for similarity to other solutions.

The idea is that I do not care how you come to understand the problem and how to solve it, but once you have the background necessary to solve it, you must provide your own solution. After all, what I care about is that you gain the knowledge, not how you gain it. If a fellow student is able to make something more clear to you than I can, or is simply the only person available at 2am while you are working on your project in the CADE lab, then feel free to learn from them. What I do *not* want is for you to mooch the solution off of another student or the Web without gaining the knowledge of how to solve the problem (and related ones), and then “fake it” by turning in a solution that you did not derive yourself.

Homeworks

In addition to the projects, there will be two written homework assignments worth 16% of your final course grade (8% each).

Exams

There will be two written exams, one after approximately eight weeks and one during finals week. The second exam will primarily test you on the material covered after the first exam, but will be somewhat cumulative. Both exams will be in-class, open book, and open notes. The midterm exam will be worth 20% of your final grade, while the final exam will be worth 30%.

Due Dates and Late Policy¹

When a programming project is marked as due on a particular day, that means that you must turn in *all source and executable files (including makefiles, header files, etc.) that might be necessary to recreate your project* and a README file containing the external documentation using the “handin” program (described in a later handout) by *midnight* that night (i.e., at the end of that day). **Do not forget to turn in all of the .h files or your makefile – if we cannot recreate what you did, we cannot grade it.** My late policy for programming assignments is quite strict. You will receive a 10% penalty for each day (*or portion thereof*) that the assignment is past due, up to the lesser of {three days, when we discuss the project in class}—after that you will receive no credit. Extensions will be granted only in extreme, documented circumstances (serious illness with a doctor's note, jury duty, ...). It is unfair to the rest of the class if I give certain individuals extensions, so plan accordingly and start on your assignments early!

¹By in-class vote, the class chose the midnight and 5:00pm times.

When a non-project homework is marked as due on a particular day, that means that you must hand it in at the CS front office by 5:00pm on the day that it is due. A similar 10% deal applies to homeworks.

If you will miss an exam, you *must* tell me ahead of time. Make up exams will typically be quite a bit tougher than the original exam.

You are granted a total of two “free” late days that can be applied to any one homework or project, or split among two of them, without penalty.

Final Grades

The final grades are expected to be *curved*, which means that your grade will be based on your performance relative to your classmates rather than based on a strict 90-80-70-60 policy. Historically, roughly 30% of the final grades will be A’s of some kind, 30% will be B’s, 30% will be C’s, and 10% will be D’s or E’s, but the ratios vary considerably from year to year.

CS6480

Students registered for the graduate-level CS6480 will be assigned additional homework problems and additional exam questions (without extra time), and will be graded under a separate curve.

Cheating

Cheating is a *serious* problem, and I will not tolerate cheating of any form! There are strict limits on the extent to which you can collaborate with your fellow students, or use material from students who took this course in previous years. You may talk to each other, but draw the line at copying code or homework assignments. Specifically, *do not even look at someone else’s code or homework assignment!* When in doubt about whether a particular action is OK, ask me. The default punishment for any form of cheating is that you will receive an “E” in the course, although the specific punishment will depend on the particular circumstances. More serious punishments are possible for particularly egregious cases. Since this is a required course for CS/CE graduates, being caught cheating will delay your graduation by at least a year (and odds are high that I will be teaching it again when you retake it). Put simply - don’t cheat. It can ruin your life, and your grade is not *that* important. Dealing with cheaters is one of the most unpleasant parts of my job, and I would prefer to not have to do it this quarter.

Quoting from Article XI of the Student Code of the University of Utah (Proscribed Conduct):

The following conduct is proscribed and upon violation of such prescriptions a student shall be subject to one or more of the sanctions specified in Section 12.05. ...

A. Academic dishonesty in all its forms including, but without being limited to cheating on tests, plagiarism, and collusion.

The meaning of “cheating on tests” should be obvious. “Plagiarism” includes turning in somebody else’s work as your own (e.g., copying a homework solution or project). “Collusion” involves working with other people when you are not supposed to do so. See the Student Code for more details.

Acknowledgements

Thanks to John Carter and others for portions of the course material.