CMSC 417: Computer Networks

Neil Spring

Spring 2014

Instructor Neil Spring

E-mail nspring@cs (include "417" in your subject line)

Class TuTh 11:00-12:15 CSIC 1122

Office hours TBA AVW 4133

TA Vasileios Lekakis: lex@cs (include "417" in your subject line)

Undergrad TA Eric Jeney: jeney@terpmail (include "417" in your subject line)

TA Office hours TBA in CSIC Linuxlab

Web `http://www.cs.umd.edu/class/spring2014/cmsc417/`

Piazza `http://piazza.com/umd/spring2014/cmsc417/home`

Textbook Peterson & Davie, *Computer Networks*, any ed.

Supplement Donahoo & Calvert, TCP/IP Sockets in C

1 Goals of this course

The goal of this course is to prepare you to use and develop the software of networks. We will learn about the architectural features of networked software, the problems protocols solve, and the problems left for end-host software to address. At the end of this class, you should be able to design and build a simple but complete network stack.

2 Summary

This course will use an unconventional, problem-focused organization that largely ignores the "layer" of each solution. It is a new and relatively-untested approach to organizing course content.

This course will cover the four main problems of networking:

error detection and reliability – framing, error correcting codes, retransmissions, flooding, routing.

sharing and multiplexing – CSMA, flow control, congestion control, quality of service, traffic engineering.

growth and evolution – layering, multicast, caching, addressing, naming, overlays.

cooperation and competition – security, attacks, pricing, inter-domain connectivity.

Unfortunately, no textbook uses this organization. The layer-based organization, in my experience, confuses more than it organizes. The unfortunate consequence is that it will sometimes be up to you to determine which sections of the book are appropriate.

3 Prerequisites

Experience in CMSC412 (operating systems) may help you.
CMSC351 – Algorithms.
CMSC330 – Programming Languages.

You must know what a function pointer is and how it is used. Find a book on C today if you do not.

You must know how to write basic data structures or have basic data structures on-hand: queues (heaps if you're advanced) and hash tables are necessary.

You should understand basic issues of concurrency. That includes the interactions between non-blocking sockets, user-level and kernel-level threads, locking, etc. Too many students seem to think that forking a thread will solve a simple problem without creating many more.

4 Grading

4.1 Participation: 4%

In a class so large, I can't expect each of you to speak; participation here is a negative grade, if I think you're doing poorly and it's your own fault for not being engaged with the material, you won't get the participation bump.

Sitting in the back row does not correlate with good participation scores.

Participation on Piazza counts so long as you're answering questions.

4.2 Quizzes: 4%

There will be quizzes. Mostly this is a reward for attending class and paying attention. The lowest quiz score will be dropped, including missed quizzes.

4.3 Course Notes: 4%

You'll work with another student in the class to prepare and format notes for one lecture this semester, to be shared with other students in the class. Your task will be to take more complete notes than my lecture buffer, organize information to help study, and answer questions about the notes online.

I will post the schedule online. You may ask to be reassigned given a reasonable excuse. I will ask for volunteers to take the second lecture.

4.4 Homework: 7%

Homework assignments are (usually) on the site. They are not of uniform weight.

4.5 Midterm Exams: 28%

4.6 Final Exam: 18%

The midterm and final exams will mix multiple choice, simple matching, short answer and long answer questions. The midterms will consume lecture slots, the final during finals week as scheduled by the university. The exams will be longer than will allow all of you to finish the entire exam. You will have to learn and study before the exam.

Multiple choice often connotes "easy," but most answers on my exams are superficially plausible and correct answers depend on understanding the material and vocabulary of the class. I will often ask "which of the following is *not*" questions, which are generally harder.

4.7 Programming Assignments: 35%

My programming assignments are not over-specified. They are meant to be exciting for those who want to play, but may be frustrating for those who want clear milestones and supplied test cases.

I will teach coding concepts in the programming assignments as if you are implementing in C. C is the one true language. C is beautiful. C is the language of networks. C function calls correspond directly to many system calls; the system calls involved form the sockets API. Other languages attempt to hide errors that occur in the network and do so poorly.

You will have the option of completing later programming assignments in Ruby.

5 Lateness

All programming assignments can be turned in electronically.

There are no late deadlines. Manage your time.

Homework assignments are to be turned in at the beginning of class. Turn them in once at the end of class before I leave, and I'll look the other way. Turn them in electronically a few hours after class, also okay, once. If you can't make class, send the homework by email to both TA and prof.

6 Administrative Cruft

I dislike this section greatly, but codifying each of these policies is important for keeping myself sane and making clear what my expectations are. I'd much prefer a section that said "treat me with respect and I'll do the same for you;" this section is intended mostly for those who would hope to game the system. Note that I copied verbatim some of these passages; I hope you appreciate irony.

6.1 Excused absences

Students claiming a excused absence must apply in writing and furnish documentary support (such as from a health care professional who treated the student) for any assertion that the absence qualifies as an excused absence. The support should explicitly indicate the dates or times the student was incapacitated due to illness. Self-documentation of illness is not itself sufficient support to excuse the absence. An instructor is not under obligation to offer a substitute assignment or to give a student a make-up assessment unless the failure to perform was due to an excused absence. An excused absence for an individual typically does not translate into an extension for team deliverables on a project.

6.2 Religious observances

I have made an effort to avoid deadlines 4/14-16, 18 for Passover and Good Friday. Please inform me in the first or second week of class of religious observances that will interfere with your ability to complete assignments on time.

6.3 Disability Support

(Although the people who need this already know it. . .)

Any student eligible for and requesting reasonable academic accommodations due to a disability is requested to provide, to the instructor in office hours, a letter of accommodation from the Office of Disability Support Services (DSS) within the first two weeks of the semester.

6.4 Honor code

The University of Maryland, College Park has a nationally recognized Code of Academic Integrity, administered by the Student Honor Council. This Code sets standards for academic integrity at Maryland for all undergraduate and graduate students. As a student you are responsible for upholding these standards for this course. It is very important for

you to be aware of the consequences of cheating, fabrication, facilitation, and plagiarism. For more information on the Code of Academic Integrity or the Student Honor Council, please visit `http://www.studenthonorcouncil.umd.edu/whatis.html`.

6.5 What constitutes cheating?

(And now back to stuff I wrote.)

Copying other assignments, looking over someone's shoulder in the lab, emailing function code, using google to find a code fragment without understanding, looking for code in other people's directories, pulling code printouts off printers, not working with your partner, alternating assignments with your partner, and in any other way attempting to gain a grade without learning.

Note: cheating goes both ways; leaving someone your code because you want to help is just as bad as borrowing someone else's code. We can tell when code looks and acts too similar to be independent work; we can't (easily) tell which of two implementations was the original.

6.6 What constitutes legal collaboration?

Interaction via mailing list or discussion of problem and code solutions governed by the Gilligan's Island rule.[1] Collaborative interoperability testing, experimenting with whether two independent implementations can talk to each other, is also permitted, so long as code is not shared.

Using google is OK. Using wikipedia is encouraged. If you find a particularly good solution on either, please cite it; there is no penalty for citing sources and I'm more likely to consider answers that disagree with textbook or lecture legitimate. If you find a question far too easy because an answer is present on-line, please let me know.

This policy applies to all course assignments. Explicitly, it is not permitted to collaborate on homework assignments. If your answer is not your own, it must be cited (wikipedia, google). If you have questions, post to the forum. If you learned something through a discussion with another student, cite.

[1] You understand the concept only if you can watch one half-hour complete episode of Gilligan's Island and still retain the concept. You may then begin coding with your newfound knowledge safe that it is your own work. Without the thirty minute pause, it is not your work.

This schedule subject to change.

Tue Jan 28	Intro, syllabus, layering, 216 review: file descriptors (sockets), etc. P0 assigned. *Read: Chapter 1: 1.3-1.4*
Thu Jan 30	Philosophy: Robustness principle, End-to-end argument; Byte ordering, socket operations, packet structures. Sockets in C. *Read: Chapter 5: 5.2.1*
Mon Feb 3	H1 due.
Tue Feb 4	Quiz (C content: printf, pointers, sizeof, for loops, preprocessor, const). Phy layer basics: Manchester, 4B/5B *Read: Chapter 2: 2.1-2.2*
Thu Feb 6	Framing: sentinel, stuffing, clock; Error detection: parity, checksum, crc; Error correction. Fragmentation. *Read: Chapter 2: 2-3-2.4*
Fri Feb 7	P0 (simple client) due.
Mon Feb 10	H2 due.
Tue Feb 11	Quiz (Socket operations). Reliability, sequence numbers. Fast path/slowpath. Waterfall diagram. *Read: Chapter 5.2.6*
Thu Feb 13	Bittorrent intro. Sliding window. Silly window syndrome. Delayed ack. Nagle's algorithm. P1 assigned. *Read: Chapter 9: 9.4.2 (bittorrent)*
Fri Feb 14	P1 (concurrent server) due.
Mon Feb 17	H3 due.
Tue Feb 18	Connection setup and teardown. State diagram. RTT and RTO. Nack. *Read: Chapter 5.2.3*
Thu Feb 20	Addressing: v4, CIDR, RFC 1918, MAC, v6. ARP. Hierarchical and flat. *Read: Chapter 4.1, 4.3.2*
Fri Feb 21	P2 (bittorrent tracker query) due
Tue Feb 25	Hubs, bridges/switches, forwarding database. Spanning tree protocol. *Read: Chapter 3.2*
Thu Feb 27	Kernel network stack structure.
Mon Mar 3	H4 due
Tue Mar 4	Routers, Distance vector routing, Forwarding vs. routing. *Read: Chapter 4.2*
Thu Mar 6	Split horizon, Count-to-infinity. OSPF, Areas. *Read: Chapter 4.2.2, 4.2.3*
Fri Mar 7	P2 (geekos ethernet) due
Mon Mar 10	H5 due
Tue Mar 11	Handling sockets with callbacks and state machines. BGP - stubs, transit, prefer customer, no valley, path vector, localpref, prepending. *Read: Chapter 4.3.3*
Thu Mar 13	Bittorrent handshake, nonblocking connect. Midterm review.
	Spring break

Tue Mar 25	Midterm 1
Thu Mar 27	midterm review, project review.
Tue Apr 1	Sharing: TDMA, FDMA..., Token ring, CSMA/CD, CSMA/CA *Read: Chapter 2.6*
Thu Apr 3	Binary exponential backoff. Hidden terminal. RTS/CTS. Exposed terminal. 802.11 DIFS. *Read: Chapter 2.8*
Fri Apr 4	P3 (bittorrent hanshake acceptance) due
Tue Apr 8	AIMD, slow start, congestion avoidance. cwnd, ssthresh. fast retransmit, fast recovery. H5 (big) assigned *Read: Chapter 6.3 (TCP, AIMD, fast rxmit)*
Thu Apr 10	Tahoe, Reno, Vegas. Tcpdump. RED, ECN *Read: Chapter 6.4.2 (RED) 6.4.3 (Vegas)*
Fri Apr 11	P3 (bittorrent hanshake acceptance and creation) due
Tue Apr 15	Congestion control review. Ack division, optimistic ack.
Thu Apr 17	DNS: A, PTR, NS, CNAME. Iterative and recursive queries. Zones and labels. *Read: Chapter 9.1.3 (DNS)*
Mon Apr 21	H6 (packet trace analysis) due
Tue Apr 22	Security concepts. Diffie Hellman. BAN logic. *Read: Chapter 8.3*
Thu Apr 24	Security miscellany: web of trust, firewall, IDS, fail-closed, revocation list, syn flood, zombie. *Read: Chapter 8.5*
Mon Apr 28	H7 due
Tue Apr 29	Midterm 2
Thu May 1	Multicast, ack implosion, digital fountain, DHTs. *Read: Chapter 4.4, 9.4*
Fri May 2	P4 (bittorrent downloading complete) due
Tue May 6	Final review
Thu May 15	Final Exam 08:00 AM