CS 455 / 655: Introduction to Computer Networks

Last updated Thursday, December 7, 2017

Fall 2017
Prof. Abraham Matta

Computer Science, Boston University
office: MCS 294A. Tel: (617) 358-1062
e-mail: matta AT bu DOT edu

Important Dates: Last Day to DROP Classes without a 'W' grade is October 10, 2017. Last Day to DROP Classes with a 'W' grade is November 10, 2017.

TIME & PLACE Tuesdays and Thursdays 2-3:15pm. LSE B01 (Life Science and Engineering Building, 24 Cummington Mall).

COURSE DESCRIPTION The course introduces the underlying concepts and principles of computer networks. It presents the building blocks of a network and how these blocks fit together. The course emphasizes the design and implementation of network software (protocols) that transforms raw hardware into a richly functional communication system. Real networks (such as the Internet, Ethernet, Wi-Fi) will be used as examples to reinforce the concepts and demonstrate various protocols. The course also covers applications such as electronic mail, the World Wide Web, and P2P file sharing.

CS 455/655 is the first course in a two-semester networking sequence. The second course in the sequence, CS 556, covers advanced topics in greater depth (including multimedia networking), and provides more analytical and experimental experience with the design and implementation of network systems.

Note that security is exclusively covered in CS 558 but foundations of networking and protocol design discussed in CS 455/655 provide the basis for understanding some of the vulnerabilities of existing networks and also how to design networks that are inherently more secure.

LEARNING OUTCOMES In this course, students learn concepts, principles, and protocols used in computer networks, with the Internet as a case study. They learn how to design and implement protocols at many levels of the network architecture and across different timescales. Topics include: distributed inter-process communication; performance measurements; multiplexing; error and flow control; routing; media access control; etc. They understand protocol correctness and performance evaluation via statistically reliable measurements and discrete-event simulations. They also become familiar with the Wireshark tool, a network traffic sniffer, and use it to deeply understand the operation of many Internet protocols (HTTP, DNS, TCP, NAT, IP, Ethernet, ARP, etc.)

PREREQUISITES The course is lectured to both senior undergraduate and graduate students.

Graduate CS-655 students are expected to complete additional readings/assignments – should we need to discuss this additional material face-to-face, we will do so during a few of the CS 655 discussion sections. CS-655 students are also encouraged to participate in the Network Reading Group (NRG) meetings regularly held on Mondays in MCS 148– please subscribe to NRG and follow announcements from the NRG webpage.

1. Solid programming skills in a high-level language (such as C, Java, Python) are required. During the lectures I will mostly use C or C-like code, which is more suitable for exposing low-level system details and achieving higher performance in real implementations. Also, if we provide a skeleton of code in some language as part of an assignment, you will be

expected to incorporate your code in that language to implement specific networking functions. In this case, the code is mostly straightforward and does not need knowledge of advanced features of the language.

2. A rudimentary understanding of algorithms and their mathematical foundations (CS 112, CS 131) is required.
3. A rudimentary understanding of computer architecture and operating systems (CS 210) is required.
4. A basic understanding of queuing and probabilistic models, discrete-event simulation and how to collect statistically reliable performance metrics (CS 350) is also helpful.

You are expected to already have the background to read and understand code, write and debug reasonably large (1000-line) programs, and learn new syntax and apply it without much difficulty. You are also expected to learn new tools/programs and run them to test and analyze network protocols. If you are in doubt of your background, please talk to the instructor.

CLASS SCHEDULE (tentative)

<table>
<tr><th colspan="2">Dates
Tue Thu</th><th>Topics</th><th>Readings*</th></tr>
<tr><td>9/5</td><td>9/7</td><td rowspan="2">Overview: introduction to communications connectivity (links and nodes, LANs, WANs, internets, multiplexing, end-to-end channels, performance), standards, protocol architectures (OSI, TCP/IP)</td><td rowspan="2">Chapter 1 (read 1.1-1.5)</td></tr>
<tr><td>9/12</td><td>9/14</td></tr>
<tr><td>9/19</td><td>9/21</td><td rowspan="2">Applications: application programming interface (sockets), client-server, Email (SMTP, MIME, POP, IMAP), Web (HTTP, cookies, caches), DNS, P2P (file sharing, Skype), ~~video streaming and Content Distribution Networks~~</td><td rowspan="2">Chapter 2 (skip 2.6)</td></tr>
<tr><td>9/26</td><td>9/28</td></tr>
<tr><td>10/3</td><td>10/5</td><td rowspan="4">Transport services, protocols: UDP, TCP, basics of reliable communication (stop-and-wait, go-back-n, selective-repeat), flow control (sliding window), end-to-end challenges (TCP reliability and flow control), connection management, …
[[No lecture on 10/10. Substitute Monday Schedule]]</td><td rowspan="5">Chapter 3 (skip 3.6.2 & 3.7.2)</td></tr>
<tr><td>[redacted]</td><td>10/12</td></tr>
<tr><td>10/17</td><td>10/19</td></tr>
<tr><td>10/24</td><td></td></tr>
<tr><td></td><td>10/26</td><td>Midterm exam</td></tr>
<tr><td>10/31</td><td>11/2</td><td>… adaptive retransmission
Congestion control: TCP (AIMD, slow start, fast retransmit and recovery, fairness)</td></tr>
<tr><td>11/7</td><td>11/9</td><td>Routing: packets vs. virtual circuits, distance-vector routing (loops), link-state routing</td><td>Chapters 4 & 5 (skip 4.2, 5.3, 5.6, 5.7)</td></tr>
<tr><td>11/14</td><td>11/16</td><td rowspan="3">Internetworking: routers, IP datagrams, fragmentation and reassembly, global addressing, forwarding, address resolution (ARP), IPv6, scalability (classless routing, NAT, hierarchical routing), intra- and inter-domain routing protocols (RIP, OSPF, BGP), policy (path-vector) routing, ~~network management (SNMP)~~
Only CS 655: software-defined networking (OpenFlow)
[[No class on 11/23. Thanksgiving Recess 11/22-11/26]]</td><td rowspan="3">(read 4.4 & 5.5, only CS-655 as related to lab work)</td></tr>
<tr><td>11/21</td><td>[redacted]</td></tr>
<tr><td>11/28</td><td></td></tr>
<tr><td></td><td>11/30</td><td rowspan="3">Point-to-point links, LANs: encoding, framing, error detection (parity, CRC), MAC protocols (Ethernet CSMA/CD), internetworking (spanning-tree switches, forwarding), ~~virtualization (VLANs, MPLS), data center networking~~
Wireless: challenges, elements, characteristics (hidden terminals, signal fading), 802.11 Wi-Fi (CSMA/CA, RTS/CTS), mobility

Wrap-up: Last lecture on 12/12</td><td>Chapter 6 (skip 6.2, 6.3.4, 6.4.4, 6.5, 6.6)</td></tr>
<tr><td>12/5</td><td>12/7</td><td rowspan="2">Chapter 7 (read 7.1, 7.3.1-7.3.4)</td></tr>
<tr><td>12/12</td><td></td></tr>
<tr><td colspan="4">Final Exam scheduled on Saturday, 12/16, 3-5pm, LSE B01</td></tr>
</table>

* *Additional CS-655 readings/assignments will be noted elsewhere.*

** ***I will be away attending three technical meetings on 10/10, 10/26-10/27, and 11/3-11/4, but the TF(s) will provide coverage.***

OFFICE HOURS Mondays 2-3:30pm and Thursdays 3:30-5pm, or by appointment arranged by email. MCS 294A.

The purpose of the office hours of the Instructor and Teaching Fellows is to answer specific questions or clarify specific issues. Office hours are not to be used to fill you in on a class you skipped or to explain entire topics. Please come to class!

TEACHING FELLOWS

Name: Qiaobin Fu
Email: qiaobinf AT bu DOT edu (preferred); Phone: (xxx) xxx-xxxx
Office Hours: Tuesdays 4:30-6pm and Wednesdays 10-11:30am, or by appointment arranged by email. @ The computing lab (EMA 302).

Name: Xingchen (David) Zhou
Email: xczhou AT bu DOT edu (preferred); Phone: (xxx) xxx-xxxx
Office Hours: Mondays 3:30-5pm and Wednesdays 4:30-6pm, or by appointment arranged by email. @ The computing lab (EMA 302).

Grading (except for the final exam) is done by a number of class graders, under the direct supervision of the Teaching Fellows and the Instructor. If you have an issue with a grade (homework or exam), please contact the Teaching Fellow. Only if the issue is not resolved to your satisfaction, please contact the Instructor. Grades must be appealed within one week of receipt.

TUTORING HELP You can get extra help during the TF tutoring hours scheduled in the computing lab (EMA 302). Note that terminal assistants in the computing lab are not supposed to help with course material, but to maintain the lab environment.

REQUIRED TEXTBOOK Computer Networking: A Top Down Approach (7th edition), James F. Kurose and Keith W. Ross, Pearson, 2016. You may want to check the cheaper rental option on sites like Barnes & Noble and Amazon. The Instructor will also leave a hard copy of the textbook on reserve in the Science & Engineering Library for a 2-hour loan.

ONLINE RESOURCES We will be using Piazza for class discussion. The system is highly catered to getting you help fast and efficiently from classmates, the TF, and the Instructor. Rather than emailing questions to the teaching staff, you are encouraged to post your questions on Piazza. If you have any problems or feedback for the developers, email team@piazza.com.

Find our class page at: https://piazza.com/bu/fall2017/cs455655/home. All course material will also be accessible through Piazza. You should regularly visit Piazza for online references and up-to-date information regarding readings, assignments, exam-related material, announcements, *etc.* You can use the discussion board for asking questions and seeking clarifications, whether from the Instructor, Teaching Fellow, or classmates.

For grade reporting only, we will make use of Blackboard Learn: https://learn.bu.edu

GRADING POLICY There will be one midterm exam and one final exam. Both exams will be closed books and notes. Also, there will be 3-4 written homeworks, 4-5 hands-on labs, and 3-4

programming assignments. You will have 1-2 weeks to complete each assignment. Final grades will be approximately based on the following policy:

- MIDTERM EXAM (25%)
- ASSIGNMENTS (50%): Written (15%), Labs (20%), Programming (15%)
- FINAL EXAM (25%)

Unless otherwise specified, the grading of programming assignments will be generally based on the following policy:

- Program: works correctly (50%); in-line documentation (10%); design quality (10%)
- Design document: how it works (5%); tradeoffs and extensions (5%)
- Testing document: complete compilation instructions (5%); thoroughness of test cases (5%)
- Submission: design & testing documents (10%)

We will use the CS Linux machines (csa1, csa2 and csa3) to grade your lab/programming assignments. Although you may use your own machine, it is your responsibility to ultimately port your assignment to our CS machines to make sure they are graded correctly.

Each assignment will have a due date. If late submissions are allowed for an assignment, there will be a 10% penalty per day for late submissions. But, no late assignments will be accepted after one week from due date, and the last day to submit any late assignments is Thursday 12/7. Extensions may be granted only for religious holidays and certified medical reasons.

No incompletes will be given, except for reasons of dire illness shortly before the end of the course, and only if a significant amount of work has been completed (*e.g.,* attending lectures, handing in most assignments, and attending the midterm).

Attendance is important: I will depart from the textbook and its flow on occasions, and I will not provide backup lecture notes on certain additional details that I will cover in class, so it is imperative that you attend all lectures and take careful notes.

ACADEMIC HONESTY *Assignments must be completed individually. Discussion of issues in network systems is encouraged, but representing the work of another person as your own is expressly forbidden. This includes "borrowing", "stealing" or "buying" programs/solutions or parts of them (whether in printed or electronic form) from others. We may use an automated plagiarism checker. Cheating will not be tolerated under any circumstances. Handing in your own work a day or two late will affect your grade far less than turning in a copy of someone else's work on time!*

See the CAS Academic Conduct Code, in particular regarding plagiarism and cheating on exams. A student suspected of violating this code will be reported to the Academic Conduct Committee, and if found culpable, the student will receive a grade of "F" for the course.

CS 455 / 655: Introduction to Computer Networks

Last updated Thursday, December 7, 2017

Fall 2017
Prof. Abraham Matta

Computer Science, Boston University
office: MCS 294A. Tel: (617) 358-1062
e-mail: matta AT bu DOT edu

ASSIGNMENTS

Dates are tentative, until the assignment is actually posted.

Check submission guidelines below.

All assignments are to be completed individually. See Academic Honesty on the syllabus.

Additional CS-655 readings/assignments will be announced. These include additional labs or extensions to programming assignments, in which case students may be allowed to work in teams of two.

Tentative Assignment Schedule at a Glance

Assignment	Posted Date	Due Date
Lab 1 (Getting Started with Wireshark)	**Wed 9/6**	**Thu 9/14**
HW 1 (Getting Started, Chapter 1)	**Wed 9/13**	**Thu 9/21**
PA 1 (Sockets Programming)	**Wed 9/20**	**Thu 10/5**
Lab 2 (Wireshark HTTP)	**Wed 9/27**	**Thu 10/12**
HW 2 (Apps & Transport, Chapters 2 & 3) (NO LATE SUBMISSIONS WILL BE ACCEPTED)	**Wed 10/4**	**Thu 10/19**
MIDTERM on Thu 10/26		
PA 2 (Error Control Protocol)	**Wed 10/25**	**Thu 11/9**
PA 3 (Routing Protocol)	**Wed 11/8**	**Mon 11/20**
THANKSGIVING BREAK: Wed 11/22 – Sun 11/26		
HW 3 (Routing, Chapters 4 & 5)	~~**Wed 11/15**~~ **Tue 11/21**	**Thu 11/30**
Lab 3 (Wireshark Ethernet & ARP) (NO LATE SUBMISSIONS WILL BE ACCEPTED)	**Wed 11/29**	**Thu 12/7**

DISCUSSION SECTIONS

The main purpose of the discussion sections is to give you the chance to discuss and ask questions, in a smaller setting under the TF guidance, about specific concepts covered in the lectures or about the assignments, and to allow you to get started on the hands-on Wireshark[1] labs.

For CS 655, some of the discussion sections will discuss additional material, including hands-on GENI[2] labs.

[1] Wireshark is a "sniffer" that allows you to observe network traffic so you can understand better how application and network protocols work by observing their operation live!
[2] GENI (Global Environment for Network Innovations) is a wide-area experimental testbed.

In addition to full lab reports (as shown in the assignment schedule above), you may be asked to submit shorter reports on other hands-on labs by the end of discussion sections.

Each student attends one of the discussion sections held on Fridays in the CS Teaching lab, room EMA 304.

Tentative Discussion Schedule

Date	Topic
Week 1 (Fri 9/8)	Wireshark Introduction
Week 2 (Fri 9/15)	HW1 discussion (*GENI Intro for 655*)
Week 3 (Fri 9/22)	PA1 discussion / sockets
Week 4 (Fri 9/29)	Wireshark HTTP & DNS
Week 5 (Fri 10/6)	HW2 discussion (*GENI Scheduling for CS 655*)
Week 6 (Fri 10/13)	HW2 discussion
Week 7 (Fri 10/20)	Midterm review
Week 8 (Fri 10/27)	PA2 discussion
Week 9 (Fri 11/3)	Wireshark TCP (*GENI TCP for CS 655*)
Week 10 (Fri 11/10)	PA3 discussion
Week 11 (Fri 11/17)	HW3 discussion / Wireshark IP & NAT (*GENI IP & NAT for CS 655*)
Week 12 (Fri 11/24)	***No sections (Thanksgiving Recess)***
Week 13 (Fri 12/1)	Wireshark Ethernet & ARP
Week 14 (Fri 12/8)	Final review

Guidelines on Submissions of Assignments

Assignments will be available on-line.

Your homework solutions and hard copies of your programming/lab documentation/reports are to be handed in using the slotted homework (drop-off) inbox (located at the bottom), labeled "CS 455" or "CS 655", in the hall outside of MCS 137 by 1:00 pm on the day they are due (unless otherwise specified). Late assignments must be time-stamped by a CS Office Staff and left in the Teaching Fellow's mailbox (neither in the "CS 455/655" slotted homework inbox nor the Instructor's mailbox). Do not hand in your assignment in the class or during office hours. Do not hand in your assignment by slipping it under the office door of the Instructor.

Graded assignments will be available for pick-up during your discussion section or during the TF's office hour.

If you believe that there is a chance that your assignment will be lost by the course staff, then here are two ways you can protect yourself: (1) make a copy of your assignment before handing it in, and have the CS office *time-stamp* your copy of the assignment; or (2) "gsubmit" an electronic version of your submission (see "gsubmit" notes below) – you can type and submit your write-up in text or PDF, or if handwritten, you can scan your write-up and submit a PDF. The timestamp on your electronic submission should also indicate submission by the due date/time. *Claims for "lost" assignments will be considered only if accompanied by a time-stamped (hard or electronic) copy of what you handed in. There are no exceptions to this rule.*

Check for assignments regularly. Start early!

Programming Assignments

You will be required to submit an electronic copy of your code, in addition to both electronic and hard copies of supporting documentation and any requested written report. See guidelines for electronic submissions below.

General Guidelines on What to Submit

Unless otherwise specified, the program you submit should work correctly and be documented. You should submit an electronic copy of the following:

1. **Program:** a program listing containing in-line documentation (i.e., comments).
2. **Design document:** a separate file (a page or so) describing the overall program design, a verbal description of "how it works", and design tradeoffs considered and made. Also, describe possible improvements and extensions to your program (and sketch how they might be made).
3. **Testing document:** a separate file describing how to run your program. Specify the steps that must be followed to successfully run your program. Also, describe the tests you ran on your program to convince yourself that it is indeed correct. Also, describe any cases for which your program is known not to work correctly.

It is fine to submit all the above documentation in one README file; given you have clear subtitles. To save trees, you are required to submit a hard copy only of the above supporting documentation and any requested written report, but not of your program listing/code.

If a programming assignment asks you to follow a specific design, then you don't have to describe that design in your documentation, but you must still describe possible improvements and extensions.

HOW TO SUBMIT AN ELECTRONIC COPY (*Only plain ASCII files!*)

To submit your assignments, use the gsubmit program from your csa account. Submit only your text and source files (*e.g.*, README file and source .java, .c or .py files), but not compiled files (e.g.. .class, .o or .pyc files). Do **not** email your assignments.

gsubmit is an electronic file submission engine, which will submit files or directories of files to the grader so they can be marked.

Every file submitted by a given student for a given assignment should have a unique file name. If a file is submitted with a duplicate name, it will either overwrite the file or generate an error message.

To make it easy for the grader to find the files relating to a specific assignment, all files for each assignment should be stored in a subdirectory called lab1, lab2, hw1, hw2, pa1, pa2, etc. and the entire directory should be submitted.

To submit an assignment:

- Create a subdirectory, say "pa#" where # is the assignment number. This is done using the mkdir command: *e.g.*, **mkdir pa5**
- Copy all files necessary for that assignment into the new subdirectory, using the cp command: *e.g.*, **cp prog1.java pa5**
  Be sure to copy only the files you need to submit into this subdirectory.

- Use **gsubmit** to submit the entire subdirectory: **gsubmit cs455 -cp pa5**
  If submission is successful, a status message will be printed.

To submit a file to an already-submitted subdirectory:

If you only submitted part of the assignment and would like to add another file:

- To submit a file README.txt to subdirectory pa5, type (at the prompt):
  gsubmit cs455 -cp README.txt pa5

To resubmit a file:

- To resubmit a file prog1.java in subdirectory pa5, first un-submit (i.e., remove) the file:
  gsubmit cs455 -rm pa5/prog1.java
- Then resubmit it:
  gsubmit cs455 -cp prog1.java pa5

List all files which you have submitted:

- To list all files that you have submitted, type: **gsubmit cs455 -ls**

Looking at a file which has already been submitted:

- To look at a file that has already been submitted, type: **gsubmit cs455 -cat pa5/prog1.java**
- You can store this in a file foo.java by typing **gsubmit cs455 -cat pa5/prog1.java > foo.java**

Where do submitted files go?

Each student who submits an assignment has a subdirectory created to hold his/her files, in a directory for the specified course. This is called the student's "submission spool directory".

Note that cs655 students should specify cs655 instead of cs455.

How can the grader tell when a file has been submitted?

Every gsubmit command is automatically logged in a log file, along with a time stamp.

For further information:

Note - The information in this document is taken from the gsubmit man page.
For further information, type: **man gsubmit**
