ACM SIGCOMM

Workshop on Computer Networking: Curriculum Designs and Educational Challenges

August 20, 2002

ACM SIGCOMM 2002 CONFERENCE
PITTSBURGH PA ~ AUGUST 19-23

Final Program

ACM SIGCOMM Workshop on Computer Networking: Curriculum Designs and Educational Challenges

August 20, 2002

9:00-9:15 Welcome

9:15-10:30 **Panel 1: Undergraduate Curriculum**
Co-Chairs: Shawn Ostermann (Ohio U.)
Panelists: Russell Clark (Georgia Tech), Ralph Droms (Cisco),
Michael Greenwald (University of Pennsylvania), Dave
Morgan (Fidelity), Craig Partridge (BBN)

10:30-11:00 Break

11:00-12:15 **Panel 2: Laboratory Courses**
Chair: Jorg Liebeherr (U. Virginia)
Panelists: Ann Burroughs (Humboldt State University), Magda
El Zarki (University of California at Irvine), Doug Comer
(Purdue University), Gene Longo (Cisco), Nick McKeown
(Stanford University).

12:15-1:15 Lunch

1:15-2:30 **Panel 3: Graduate Curriculum:**
Chair: Jim Kurose (U. Massachusetts)
Panelists: Ken Calvert (University of Kentucky), Scott
Jordan
(University of California, Irvine), Raj Yavatkar (Intel),
T. Znati (U. Pitt./NSF)

2:30-3:00 Break

3:00-4:00 **Break-out sessions**

4:00-5:00 **Report back and discussion**

5:00 **Wrap up**

Workshop organizers:

J. Kurose (U. Massachusetts, kurose@cs.umass.edu)
J. Liebeherr (U. Virginia, jorg@cs.virginia.edu)
E. Nemeth (U. Colorado, evi@cs.colorado.edu)
S. Ostermann (Ohio U., sdo@cs.ohiou.edu)
T. Ott-Boisseau (CAIDA, theresa@sdsc.edu)

Acknowledgement:

Travel support for some participants has been provided by the National Science Foundation Division of Advanced Networking Infrastructure and Research, ANIR-0226866.

White Papers

2002 ACM SIGCOMM Workshop on Computer Networking: Curriculum Designs and Educational Challenges

The Undergraduate Computer Networking Course

John P. Abraham
University of Texas – Pan American
Edinburg Texas

The Computer Networking course originally was offered as a topics course under the title of Data Communication. We taught the course every two to three years until 1991 and used the textbook "Data and Computer Communications" by William Stallings. In 1992 I proposed a senior level course in computer science called Computer Networking. Since it was not a required course, we offered this course every two years with an enrollment of 25 to 30 students. The demand for the course increased over the years and now we offer it every other semester. The course is still not required for a degree in Computer Science. I have been using the textbook "Computer Networks" by Andrew Tanenbaum. Since the third edition of this book was edited in 1996 it was rapidly becoming out of date and I am changing the textbook this year.

The most recent syllabus is attached to the end of this paper. Student comments indicate that they do not like the heavy programming requirement. I am thinking of reducing programming assignments next year and including more hands on work such as configuring a redundant network with several routers and switches. Each year the student projects included a Novell, Windows, and Unix network. I will continue doing that.

One of the problems in teaching networking is the lack of time to cover all the OSI layers in one semester. I can only cover from layers 1 to 4. I am thinking of switching over completely to TCP/IP protocol suite. Another problem is not having a permanent laboratory for Networking. Every time I teach the course I have to gather all the necessary hardware and software, reserve a room and issue keys to students. It would be interesting to see how others handle these problems.

CS 4345
COMPUTER NETWORKS

Dr. John P. Abraham

Office: Engineering Building 3.276. Office Telephone Number – 956-381-3550

Home Telephone # 956-686-5042. email jabraham@panam.edu (email is the best way to contact me.)
My web site: www.cs.panam.edu/abraham
TA: Shaily Kumar. Ms. Kumar will be in class on Tuesdays.
Ms. Kumar's Office hours: M T W T 1pm to 3 pm. Email: skumar1@panam.edu

Current Schedule

CSCI 4345	Telecommunications and Networking	TR 10:35-11:50 AM	Eng 1.290	Syllabus	**Assignments**
CSCI 6300	Foundations of Systems (Grad)	W 5:45-8:25 PM	Eng 1.290	Syllabus	**Assignments for OS** **Assignments for Architectur**
CSCI 6335.01	Adv Computer Architecture (Grad)	Th 5:45-8:25 pm	Eng 1.290	Syllabus	**Assignments**
CSCI 6390	Thesis or Project (Grad)	ARR	Eng 3.276		
Office Hours	Due to my heavy Teaching load, I ask you to please make an appointment.	Th 3:30-5:30 pm W 3:30-5:30 pm	Eng 3.276		Policy for Spring 02 ONLY: Please make an Appointment with The departmental Secretary
Office Hours	to see me. See my TA first.	TR 8:30-9:00 pm	Eng 3.276		To see me.

RequiredText book: Tanenbaum, A. S., 1996, Computer Networks, 3rd Ed., Prentice Hall, Upper Saddle River, New Jersey.

Reference:

- • *Data and Computer Communications* 5th edition, by William Stallings, Prentice Hall, 1997
- • *Internetworking with TCP/IP: Principles, Protocols, and Architecture* 3rd edition, by Douglas E. Comer, Prentice Hall, 1995

- • *Internetworking with TCP/IP: Design, Implementation, and Internals* 3rd edition, by Douglas E. Comer and David L. Stevens,Prentice Hall, 1999
- • *Internetworking with TCP/IP: Client-Server Programming and Applications* 2nd edition, by Douglas E. Comer and David L. Stevens,Prentice Hall, 1996
- • *UNIX Network Programming* by W. Richard Stevens, Prentice Hall, 1990

Expected Background:

Students are expected to have the CSCI 6300, CSCI 4335 or the equivalent. Students are also expected to be able to program in C or ++, or Java. Students who are not fluent in these topics should make up the deficiencies by homeworks and programming exercises.

Catalog description:

An introduction to data communication topics, including data transmission, encoding data link control, switching, network topologies, protocols, internetworking and data security. Examples of existing networks and network architectures are studied. Prerequisites: Operating systems or Computer Architecture or consent of instructor.

In addition to this, students will gain practical experience in setting up communication between computers, networking, and network management. Students will also install different peer-to-peer and client-server network software. They will also gain some experience on inter-networking.

Approximately 60% of the time will be spent in lecture and 40% in lab. All lab work will be done in groups of four to five students. Since all groups will have to give presentations, no schedule changes will be allowed. Choose your topic and schedule carefully.

Grading:

One Research Paper (Theory)	15% (10% paper, 5% oral presentation)
Written report on your project	10%
Practical Projects and programs	25%
Daily quizzes/questions	15% (student made)
Two exams	35%

For each day of class you will prepare 5-10 **multiple choice** questions (along with keys) over the material covered. Please bring two copies of the tests. These questions will be given as daily quizzes. 5 points will be given for good questions and 10 points for correct answers. Only multiple choice questions will be accepted. These quizzes may not be made up. Therefore, it is essential that you come to every class on time. The quizzes are given at the beginning of the class.

Assignments:

General instructions about programming: You may choose any of the following languages: C, C++, Pascal, Java, Visual Basic. If you would like to use another language please talk with me first. I will not give you any assistance with the programming assignments. You are welcome to talk with others in the class to get general ideas and algorithms, but may not view their source codes. Assignments are due at the beginning of the class. Late penalties: 1 day=10%, 2 days=20%, 1 week=30%, 2 weeks=50%, after two weeks I do not accept assignments.

General instructions about the research paper: Start working on the research paper beginning the second week of the semester. Submit a topic for approval by the 4th week. All research should be completed by the middle of February. Schedule for a 10 minutes presentation starting with the first week in March.

1. All students in this class should have sufficient working knowledge installing and removing interface cards, installing appropriate driver software, assigning IRQ, base address, etc. If you do not have enough experience in these matters, you need to build a computer with parts provided. All students need to write one page summarizing steps in building a computer or your practical experiences working with hardware.
Due Jan 22, 2002.

2. Write a program to send a file across a pair of serial or parallel ports. If you are using serial ports, you should be able to use modems or a direct cable. When writing program specifications please include a paragraph explaining how a direct cable should be constructed. Due Jan 29, 2002.

3. Write a program to send a file across Transport Service Access Points (TSAPs) also known as TCP ports or Sockets. Your program can select any non-privileged port (that is, the port number should be greater than 1024). Due February 12, 2002.

4. Paper Due Feb 14. For late grades no later than March 5h. Schedule for class presentations starting with Feb 19th.

5. Do a peer to peer network of your choice. Due March 19, 2002

6. Practical work: Install one of the following: Client-server network, router, print server, ISP provider, FTP server (write your own software), Mail server (write your own program), Internet radio, Distributed application, wirless networking or any other latest project as long as you clear it with me. Due April 11th. Start scheduling demonstrations starting on the 11th.

All assignments should be coordinated with my TA. The TA will be responsible for collecting, evaluating and grading your projects.

ACM SIGCOMM Conference
Workshop on Computer Networking: Curriculum Designs and Educational Challenges
August 19-23

Graduate Level Curriculum and Laboratory Courses for Computer Networking

Ehab Al-Shaer and Greg Brewster
School of Computer Science Telecommunication and Information Systems
DePaul University
Chicago, IL 60604
(ehab, brewster)@cs.depaul.edu

1. Introduction

In the past decade, there has been a significant progress in research and development of computer networking. As a result, the demand for researchers, engineers and practitioners in this field was remarkably increasing. Education institutions have realized the importance of offering up-to-date courses in various areas in networking in order to prepare graduate students to the market needs. School of Computer Science, Telecommunication and Information Systems (CTI) in DePaul University was one these universities that have responded to this demand by offering a Master Degree in Telecommunication and Data Communication for more than 14 years. The main goal of this degree is to give CS graduate students the breadth and depth knowledge of networking concepts and practice by offering basic and specialized courses, and by providing hands-on laboratory experience for various areas in networking. There are 8 full-time faculty members and about 240 graduate students in the Telecommunications and Data Communication division. Our goal of this paper is to share our experience in this area and acquire the community feedback as we go through curriculum evaluation and revision periodically. The paper describes the graduate curriculum and the laboratory environment in CTI. Then, we focus the study on of the most recent course designed in the curriculum and its related lab work.

2. Curriculum Overview

The degree is divided into two main tracks: (1) Standard telecommunications degree that focuses on voice communication and telephony technology, and (2) Data communications degree that focuses on the Internet technology and applications. The second one requires more C/C+ programming background but they both share common prerequisites including Java, discrete math, electric circuits, statistics, computer architecture and operating systems. Each track is divided into core phase and advanced phase. The core phase is common between both tracks. Figure 1 sketches the road map for both tracks in the networking curriculum. Course outline and description can be found at www.cs.depaul.edu.

In general, the core course covers the basic telecommunication systems, regulation, PBX and phone systems, transmission medium (including microwave radio and wireless), signals, modulation and encoding techniques, and basic data communication and transportation concepts like network topologies, medium access protocols, routing, reliable transportation and network application services. With the increasing awareness of networking concepts in undergraduate level, a plan is proposed to move the basic data communication course from core to prerequisites and introduce an advanced topic in operating systems that covers real-time and embedded system in the core. This proposed change is also shown in Figure 1.

The telecommunication students focus mainly on voice communication technical and administrative issues but they also required to take at least two courses from the data communication field to have some depth in data communication. In both tracks, the telecom and data communications practicum courses are required. These courses provide extensive hands-on experience, as 100% of the course is a laboratory work. Data communications students, in this course, are assigned individual and group projects that ranges from simple network installation and configuration using Linux, Windows, Novell and LAN switches, to high-end/core routers configuration and troubleshooting for WAN using CISCO routers.

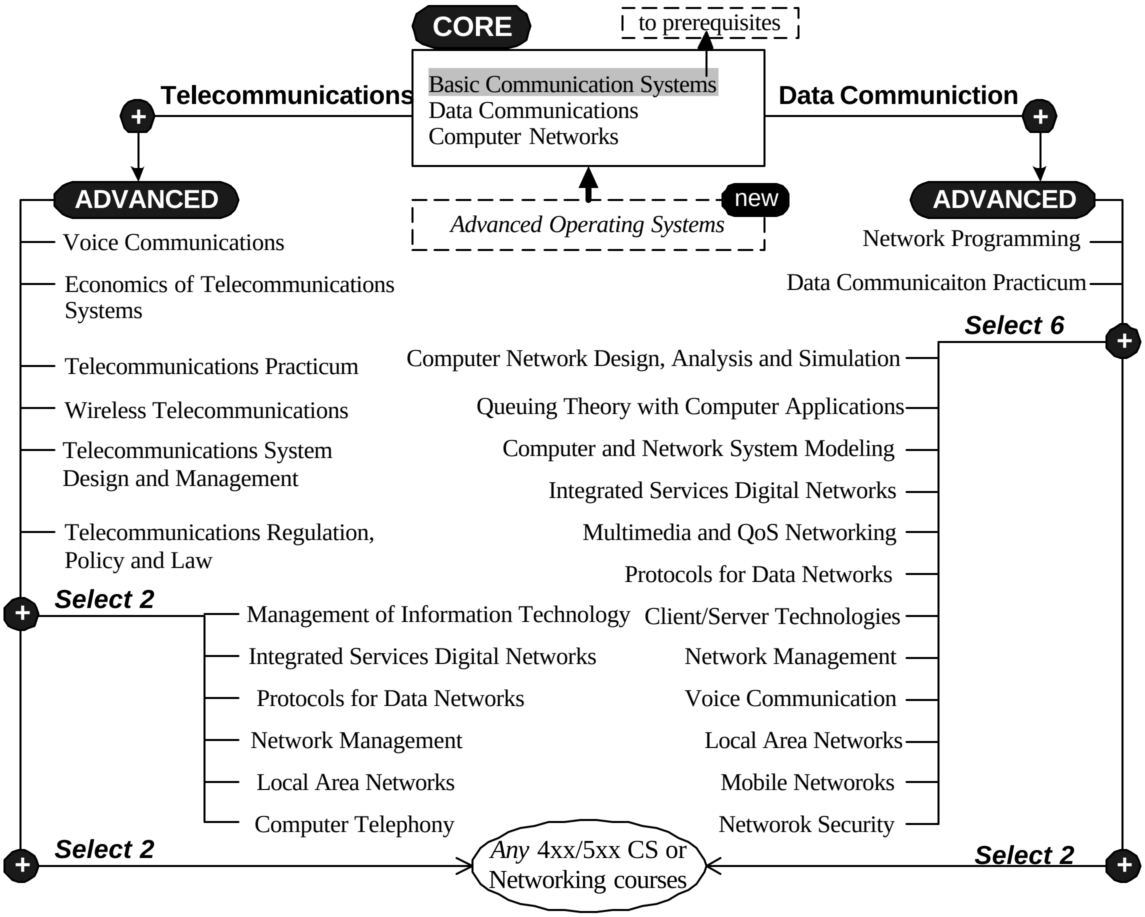

CORE
to prerequisites
Basic Communication Systems
Data Communications
Computer Networks
Telecommunications
Data Communiction
new
Advanced Operating Systems
ADVANCED
Voice Communications
Economics of Telecommunications Systems
Telecommunications Practicum
Wireless Telecommunications
Telecommunications System Design and Management
Telecommunications Regulation, Policy and Law
Select 2
Management of Information Technology
Integrated Services Digital Networks
Protocols for Data Networks
Network Management
Local Area Networks
Computer Telephony
Select 2
ADVANCED
Network Programming
Data Communicaiton Practicum
Select 6
Computer Network Design, Analysis and Simulation
Queuing Theory with Computer Applications
Computer and Network System Modeling
Integrated Services Digital Networks
Multimedia and QoS Networking
Protocols for Data Networks
Client/Server Technologies
Network Management
Voice Communication
Local Area Networks
Mobile Networoks
Networok Security
Any 4xx/5xx CS or Networking courses
Select 2

Figure 1: CTI Graduate Network Curriculum

However, telecommunication students use a special telephony lab that contains a digital AT&T PABX and a telephony network. Our future plans include developing a VoIP lab for both tracks for learning VoIP (H.323 and SIP-based) system administration, management and programming. The Network Programming course is made a required course in the data communication curriculum because (1) it enables students to use and interact with the Internet protocols (TCP/UDP, IP, ICMP and IGMP) and services (DNS, Telnet, FTP ..etc) that he studied in the core, and (2) the network programming experience is important for the student projects in many other advanced courses such as Multimedia Networking, Network Management, Network Security and Client/Server Technologies. Example of student projects in these courses includes developing video conferencing systems and media streaming servers (in Multimedia Networking), distributed management system using SNMP++ (in network management) and gateway proxy firewalls (Network Security) and so on. Such projects require an extensive interaction with Internet protocols programming. The advanced courses material and projects were designed carefully to combine the theory and practice in the field. Many specialized laboratories in various networking areas were established to serve this goal. The following are existing and planned computer network laboratories in CTI:

- Multimedia Networking Lab (networked multimedia software development)
- Network Telephony Lab (digital PABX interconnection and programming)
- VoIP Lab (SIP and H.323 nodes: installation, management and programming)-- *planned*
- Network Administration Lab (Cisco, Linux, windows and Novell: configuration, troubleshooting and network management tools)
- Wireless and Mobile Networking Lab -- *planned*
- General LAN lab (simulation, modeling, monitoring)

3. Multimedia Networking Course Outline in CTI

Multimedia networking is becoming one of the most important fields in networking. Many universities started teaching this topic in the PhD and graduate curriculum. Our design outline of this course went through number of revisions for number of years. The course covers both multimedia networking and software development of networked multimedia systems such as video conferencing, media streaming, interactive games, application sharing, ..etc. Here is the outline of the course as of May 2002:

1. **Introduction to Distributed Multimedia Systems:** requirements, digital and analog signals, encoding and decoding, traffic Characteristics, Mbone applications/tools.
2. **Media Compression Methods:** basic coding, Video Compression (JPEG, MPEG, H261, H.263, Wavelet), Audio Compression (PCM, Mlaw, G.72X and GSM), real-time multimedia transmission and recovery, media streaming.
3. **Real-time Networked Multimedia Systems Development:** audio development kit and examples (Sun and Microsoft), silence detection, audio mixing, XIL video library and examples, video for Windows with examples, JMF overview and examples
4. **Media Streaming Development:** RTSP concept, OpenRTSP, adaptive streaming techniques, RTP/RTCP libraries, LiveMedia development kit examples, JMF streaming.
5. **Integrated Services:** architecture and service model, RSVP, packet scheduling Disciplines in the Internet, Evaluation.
6. **Differentiated Services:** concepts assured and expedited services, packet classification, packet dropping techniques, active queue management, and performance evaluation.
7. **Multiprotocol Label Switching (MPLS) and Internet Engineering:** concepts, network provisioning, optimization problem, QoS routing, multipath load sharing.
8. **Multimedia Session Protocols:** VoIP models and architectures, standards protocols, RTP/RTCP, H32x, SDP/SAP, SIP, Media Filtering, media scaling, floor control issues
9. **Multicasting:** concepts, multicast routing (Intra- and Inter-domain), reliable multicast: sender and receiver-initiated, tree-based and ring-based protocols, congestion control.
10. **Multimedia Applications:** application-level framing, audio/video conferencing, video servers, applications sharing, Web multimedia applications, and interactive multiplier games.

3.1. Course Projects and Laboratory Work

The course contains 2 assignments and 3 programming projects. The following is a short description of the lab work in a quarter. The full description of the course outline, assignments, projects and slides are found at www.mnlab.cs.depaul.edu/~ehab/Courses/TDC573. Student project snapshots can also be found at http://www.mnlab.cs.depaul.edu/mnlab/tdc573-w02.htm.

Number	Title	Short Description
Assignment#1	Networked Multimedia Tools	Students are asked to experiment with the following multimedia tools to appreciate this technology sdr, vic, vat, wb, MICE, Netmeeting, Mimaze interactive game ..etc to create private session and to join public sessions as well. Questions will be asked about the session control mechanisms of these tools
Project#1	Audio conferencing	Designing a unicast (TCP and UDP) real-time Internet audio conferencing tool using various audio encoders. Evaluate the quality of different encoder and tune buffers to improve quality (less delay). Use
Project#2	Video programming	Using XIL in Solaris, Video for Windows, or JMF, impalement a video transport program over UDP
Project#3	Multicast Real-time Video Conferencing, or	Integrate Project#1 and #2 to implement a complete video conferencing tool that includes multicast/group communication, simple floor/session control, silence detection, and audio mixing – use libraries provided in class
	Multicast Media Streaming using MPEG and H.263	Integrate Project#1 and #2 to implement a media streaming server that supports multicast, RTSP, RTP, simple session control, MPEG and H.263 – use libraries provided in class
Assignment#2	Best effort Vs. differentiated services	Using NS, create mixed FTP, WWW and CBR traffic in a simple network. Measure the delay and throughput (drop ratio) when "best effort" (of drop tail and SFQ) is used and when diffserv (with different marking and RED parameters) is used.

Table 1. TDC573: Multimedia Networking Projects and Assignment

An Undergraduate Networked Systems Laboratory

Maurice Aburdene, Dan Hyde, Xiannong Meng, John Janntzi, Brian Hoyt
Bucknell University

Ralph Droms
Cisco Systems

This document describes a new and innovative undergraduate networked systems laboratory, which supports both instruction and research in the Electrical Engineering and Computer Science departments at Bucknell University. The laboratory facilities accommodate study of several computer networking hardware and software technologies, computer systems and organization.

The Networked Systems Laboratory (NSL) provides new opportunities for undergraduate instruction and supports faculty and undergraduate research computer network systems. The laboratory enables students to experiment "under the hood" of computer network systems much like a mechanic of a car. It is important that students have the experience of taking out and replacing components of the operating system or swap components of the computer network. The laboratory facility has the flexibility to allow students to experiment with and explore the issues and challenges associated with networked computing systems and computing and communication structures. The hands-on experience with software and hardware will improve their understanding of the underlying principles and concepts in computer networks while better preparing them for employment or graduate studies**.**

The Networking Systems Lab includes fourteen computers, interconnected with a 100Mb/sec Fast Ethernet that provides reliable remote access to files and access to campus network. Figure 1 gives a schematic description of the laboratory facilities. The instructor and the student workstations include:

•Dell PC with Pentium III 866 MHZ and 256MB RAM
•Three Network Interface Cards (NICs)
    –Public and two private networks
•Two Operating Systems
    –Windows 2000
    –Red Hat Linux 7.2
•Two Active Keyboards/Mice
    –For Pair Programming
    –For Collaboration

The networking equipment is housed in a student-accessible closet in a nearby locked room and includes:

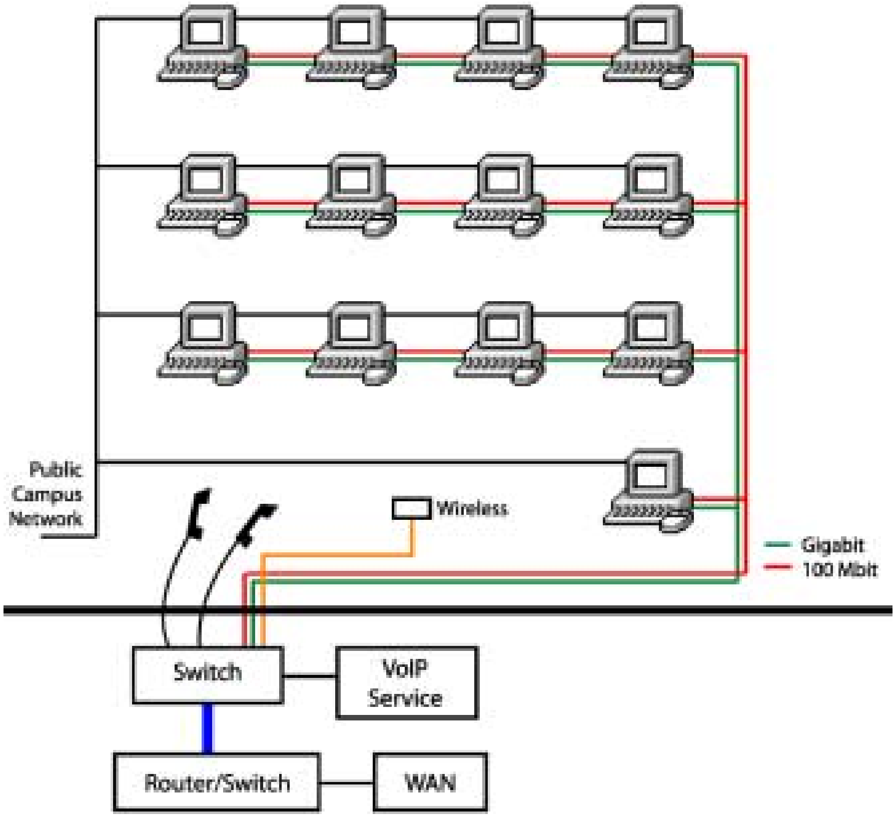
- Core Router - Cisco 4006 with:
  Gig fiber card
  100 mbps utp
  Two 100 mbps fiber
  Supervisor
  One spare slot

- Edge switches
  Three Cisco 3500 12 ports
  Three Cisco 1900
  One Cisco 3500 24 port Power
- Wide Area Network Routers
  Two Cisco 2500 WAN routers

IP Telephony
Cisco IP Phone Starter Kit

Figure 1: Network System Laboratory Configuration.

The fourteen computers also are connected to three experimental (private) network systems: 100MB/sec, gigabit Ethernet and wireless. Each computer has an interface and connection to each network, and is be able to support instruction and research projects involving any of the available networks.



We felt it was important that a fourteenth workstation be placed in the same room as the network equipment. This workstation allows faculty/staff to develop laboratory exercises or do research while the main NSL room is in use by others.

All of the private networks are connected to a firewall router, which has a connection to the campus IP network. This router is configured to filter traffic from the experimental networks, and to provide security and traffic management to the rest of the campus network. Also, there is a computer on the 100Mb/sec Ethernet network that provides file, download, control and other services to the experimental computers in the lab.

The network organization, router and experimental computers can be reconfigured to accommodate new network architectures and technologies. As network technologies evolve, new experimental networks will be added to the lab.

The experimental computers will be configured to run multiple operating systems, including Windows NT, Linux and Xinu (an open-source, Linux-like operating system). Linux and Xinu will be available for use by students in systems and networking courses, where they can make changes to the system and network interface software, download their modified system software to one of the experimental computers, and test the resulting system with the experimental network hardware.

Source and Reference

Maurice Aburdene, Dan Hyde, Xiannong Meng, John Janntzi, Brian Hoyt, Ralph Droms, “An Undergraduate Networked Systems Laboratory”, Proceedings of the 2002 American Society for Engineering Education Annual Conference & Exposition, Session 2258, (http://www.asee.org/conferences/caps/document2/2002-1121_Paper.pdf)

Computer Networking: Curriculum Designs and Educational Challenges

Humboldt State University
California State University System
Ann Burroughs
Computing Science Department

The Laboratory Component of a Networking Course

Background

Humboldt State University is relatively small (about 8000 students) and relatively rural. It draws its students locally but also from the San Francisco Bay area and the Los Angeles area. It is primarily a residential institution. While there are departments hosting graduate programs, the University delivers mostly undergraduate education, and is primarily a teaching, rather than a research, institution.

The Computing Sciences department has seven tenured/tenure-track faculty and about a half-dozen adjunct faculty. It offers degrees in Computer Information Systems (20 years) and Computer Science (1 year). There are approximately 150 majors.

Courses

The department offers two networking-related courses: Telecommunications and Network Design and Implementation. The latter class is meant to be primarily hands-on; the former is much more a survey class. While I have interest in the latter class, it is the former that is the focus of my attention.

Resources

Two years ago, Humboldt was awarded one of the first Internet Teaching Labs by CAIDA. As a result of that award, we were able to identify a space near to a teaching lab in which we could locate the routers. With departmental and college funding, we have added infrastructure (air conditioning, power, racks, hubs, switches, patch panels, tools…) until we have a small but quite capable laboratory amenable to a myriad of configurations for conducting a wide variety of networking-related learning experiences.

The Network Design and Implementation course has used this facility from its inception, and has grown laboratory exercises in concert with the development of the facility. This course is very hands-on oriented – for instance, students construct a LAN using a hub, listen to traffic, then change the hub for a switch and again listen to the traffic. See http://www.humboldt.edu/~mdh3/network/index.html for more information.

This summer, the University has renovated a nearby microcomputer laboratory so that each student station has a dual-boot device that attaches to the University's backbone, and a second dual-boot device that attaches only to the Internet Teaching Lab. This provides the isolation the University's network manager requires while still allowing remote access by a largish class (25 students) to the Internet Teaching Lab (ITL). (I cannot say enough good things about the vision and hard work of the college's Instructional Technology Consultants that have resulted in this facility.)

The Telecommunications Class

Telecommunications is a junior-level class requiring completion of most of the lower-division core. Over the years, it has been delivered with and without a laboratory/activity component. With no dedicated laboratory, it is certainly possible to do exercises using ping and traceroute, network simulators and similar tools. Nevertheless, it is hard to justify using class time in a closed lab under such circumstances. For several years, we have been delivering the class with no lab/activity component, consigning many exploratory activities to assignment status, which buys us an extra hour per week of lecture time. Even with the advent of the ITL, enrollment in this class precluded its use without remote access (and without air conditioning, which has only recently occurred!). With remote access provided, the class is being once again delivered with a weekly lab/activity component (2 hours lecture; 2 hours lab/activity).

Issues

It's difficult to cover absolutely requisite material in a one-semester survey course. What material now does not get covered because of the laboratory component?

Ideally, the lab is a learning environment where students discover knowledge. What, if any, topics can safely be left to this learning modality, without some sort of preliminary or subsequent lecture treatment? Is there some pre-lab/post-lab presentation that can be prepared and presented upon demand? If so, we can devolve topics from the lecture portion to the lab/activity portion of the course. Does this work? If not, is the lab component really necessary or really useful? Do students having the opportunity to explore in a "hands-on" way learn the material better?

Experience

An activity session conducted as a closed lab with an exercise that students perceive can be done anywhere and at any time is not successful. Assignments that anchor lecture material are, on the other hand, essential, given the scope of the course and are relatively successful. But the luxury of a closed lab in a context where the component under study (WANs, LANs, the Internet, security, …) can be simulated and explored has just become available. This fall, we'll do exercises on flow control and error detection, compression, routing, TCP/IP, LANs and security, about 10 in all. For now, I intend to use the lab as an anchor for lecture material. It will be much more ambitious to use the lab to deliver a topic independent of lecture support. I am hoping that less is more – that even though the lecture time will be reduced by 1/3, it will be more meaningful and more interesting because of the lab.

Future

One of the promises we made to the University when it decided to support the ITL with space and infrastructure was that we would reach out to other populations than traditional computing science students. I need to determine a set of stand-alone lab-based experiences that will carry by themselves sufficient networking principles that some of our alternate audiences can be served. These include the older student, the American Indian student, the education community, the Upward Bound student and perhaps even the liberal arts student.

Five Things To Leave Out of Graduate Computer Networks

Ken Calvert
Dept. of Computer Science, University of Kentucky

One of the biggest challenges in teaching a first course in graduate networking is finding a suitable starting point. Invariably the students come from a variety of backgrounds and have vastly different levels of experience. Our graduate course (CS 571) has our undergrad course (CS 471) as its only prerequisite; yet students come from a wide variety of foreign and domestic universities, and even if they have a networks course on their transcript, their understanding may not match the expectations.

Is it better to start from scratch, or begin with a brief review, or assume some level of understanding and plunge right in? Clearly each group of students is a little different, and some flexibility is necessary to accomodate the particular mix in any given offering. Course objectives are also a factor—is the goal to equip people for jobs with Cisco, or to lay a foundation for advanced courses and research, or to identify promising PhD students, or some combination of these or other goals? Ultimately, however, I believe that having a well-defined (and non-trivial) interface between a first graduate course and its prerequisite(s) is advantageous, independent of these other factors.

Such an interface makes it more likely that students have some idea how to think about networks when they begin the course. Obviously that helps by enabling students to deal with new topics more comfortably, and by adding flexibility, freeing up time to add breadth or depth of coverage. But what should the intrerface be? What should constitute the entrance requirements for "Grad Networks I"?

So here is a list of topics I believe should *not* need to be taught in "Grad Networks I". Every student should (of course I am talking about an ideal world here) enter the course with a solid grasp of these topics—that is, understand each problem and the standard solutions well enough to be able to recognize and apply them in new contexts. Probably your list would differ from mine. I include topics here because: (1) they can be taught effectively to undergraduates; (2) they are more or less self-contained; (3) they can be reinforced to the point of being "in one's fingertips" by means of hands-on demonstrations or lab exercises; and (4) they appear, in various forms, throughout much of networking, where they can become distracting details if students are unprepared.

Digital Channels. Students should be equipped with an adequate mental model of communication channels. For example: pushing colored marbles one at a time through a pipe. The rate at which marbles can be pushed through is limited; the length of the pipe determines how long it takes for a marble to emerge after it is put into the pipe. The number of marble colors determines how much information is carried by each one. This model conveys concepts like channel capacity, independence of bandwidth and latency, and entropy—concepts necessary for understanding congestion control, digital audio and video, medium access control protocols, compression, etc.

Framing and Encapsulation. The problem of delimiting message boundaries occurs at many layers, in many different contexts. The student should be able to recognize, name, and implement a variety of solutions involving different assumptions about available technologies (e.g. in-band or out-of-band marker information), and should understand the relationship between layering and encapsulation.

Errors and Error Detection. The student should be aware that marbles sometimes change color in the pipe, and should understand the implications of that fact. How does a receiver determine whether a message received is the same one that was sent? This problem should also be recognizable in many contexts, as should the fundamental paradigm of the solution: prior agreement on a property that should be possessed by all transmitted frames and that is unlikely to be preserved by errors. The student should know about codes, from block parity to CRC.

Addressing and Relaying. Students should have a good mental model of how intermediate systems relay packets of information by receiving them on one channel and transmitting them on another, and how structured identifiers are involved in this process. They should know about table-based methods including "switching" (lookup based on a local identifier) and "routing" (lookup based on a global identifier). If students have an abstract framework for thinking about the problem and solution of relaying, they are in a better position to study switching network design, longest-match lookup algorithms, routing protocols, and distributed hash tables.

Client-Server Protocols and Programming. The students should have written simple client and server programs, and should have implemented some simple protocol. Knowing the basics of how clients and servers work provides a backdrop for understanding how all kinds of real-world application protocols work, as well as overlay networks, web performance, etc. This also—depending on the particular protocol implemented—may increase the likelihood that students have some idea of how data is represented inside the machine.

I do not claim that this list is exhaustive; obviously it could be expanded (it is tempting to add basic queueing models, for example). I do think that omission of any of these topics from the interface will result in spending a significant amount of graduate course syllabus space to bring students up to speed. I would argue that these topics can fit comfortably in an undergraduate course, with plenty of room left over for other topics.

On the other hand, if students have solid these topics solidly in hand, the instructor of the graduate course is free to focus on whatever aspect is desirable, from system implementation and coding details, to large-scale network design. The question of remedial measures for deviations from the "ideal world" is left as an exercise for the reader/topic for discussion.

Hands-on Networking Laboratory Experiments Design

Liang Cheng

Department of Computer Science and Engineering, Lehigh University
19 Memorial Drive West, Bethlehem, PA 18015
liangcheng@ieee.org

***Abstract* – Hands-on networking laboratory experiments are complement components to classroom lectures of *Computer Networks*. A design of such lab sessions is presented here with the goal to help students in academic studying environment to gain some industry-oriented training and deepen their understanding of networking technologies learned from textbooks and classroom lectures. It includes network-device configuration, network topology design and setup, and network packet analysis and network troubleshooting.**

I. INTRODUCTION

Hands-on networking laboratory experiments are complement components to classroom lectures of *Computer Networks*. A design of such lab sessions is presented here with the goal as that, by learning from the networking laboratory experiments, students in academic studying environment gain some industry-oriented training and deepen their understanding of what they have learned from textbooks and classroom lecturing. The underlying rationale is that direct interactions with networking devices help students to "learn from the practice".

II. LAB EXPERIMENT DESIGN

The hands-on networking laboratory experiments are designed to include three parts: (*i*) network-device configuration, (*ii*) network topology design and setup, and (*iii*) network packet analysis and network troubleshooting.

A. Network-device Configuration

Routers and switches, such as Cisco 3620 and Catalyst 2916, can be configured both locally from console port and remotely from telnet. Thus this part of the lab experiments includes the following contents:

- Router/switch basic configuration from console port
- Device-orient command, such as Cisco IOS commands
- Password and identification
- Configuration backup and restore
- Device-oriented image copy, refresh, and upgrade
- Network interface configuration

B. Network Topology Design and Setup

Design and setup a basic local area network (LAN) and a wide area network (WAN) help students to apply their knowledge of networking technologies learned from the lectures.

- Subnet design and setup
- Virtual LAN (VLAN) design and setup
- Routing protocols in campus networking and WAN scenarios
- Serial link configuration and setup
- WAN design and setup
- BGP, firewall and NAT design and setup

The basic anchor network topology is shown as Figure 1.

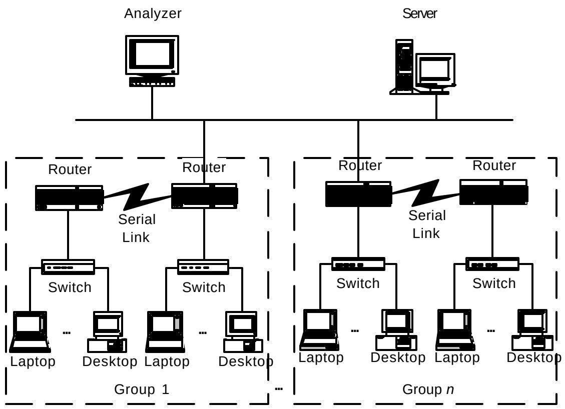


Figure 1. The basic anchor network topology for the lab design

C. Packet Analysis and Network Troubleshooting

Students gain direct sense of network protocol design and specifications by analyzing captured network packets, and deepen their understanding of what they have learned from troubleshooting network problems.

- Network packet capturing applications (e.g. ethereal)
- Troubleshooting TCP/IP Configuration
- Troubleshooting routing protocol
- Remote access
- Traffic management (e.g. weighted fair queuing)

III. RELATED PROGRAMS

There are a lot of networking testbeds and laboratory programs/experiments have been designed and built for teaching computer networking classes, such as Collaborative Advanced Internet Research Network (CAIRN) [1], Utah Network Testbed (Emulab) [2], X-Bone [3], and etc. The design presented here is for a customized lab-experimental sessions to help students gain some industrial-oriented training and deepen their understanding of networking technologies by direct interactions with network devices.

ACKNOWLEDGMENTS

This work is supported by the Department of Computer Science and Engineering and P .C. Rossin College of Engineering and Applied Science, Lehigh University.

REFERENCES

[1] Collaborative Advanced Internet Research Network (CAIRN), http://www.isi.edu/div7/CAIRN/

[2] Utah Network Testbed (Emulab), http://www.emulab.net/

[3] X-Bone, http://www.isi.edu/xbone

[4] L. Chappell, *Introduction to Cisco Router Configuration*, MacMillan Publishing Company, November 1998.

Hands-on Networking at Rockhurst University

John F. Cigas
1100 Rockhurst Road, Kansas City, MO 64110, cigas@acm.org

OVERVIEW

This paper describes the laboratory equipment and some of the Linux-based exercises used to integrate hands-on experimentation, observation, and measurement into our two networking courses. The first section describes the hardware and software used in the project. The next section gives an overview of the exercises already developed. The final section outlines some of the plans for future development.

HARDWARE AND SOFTWARE

The equipment for the laboratory consists of the following hardware:

- 16 workstation PCs, each with a 600 MHz processor, 128 Mb RAM and an 8-20Gb removable hard disk
- 5 server PCs, with a 600 MHz processor, 128 Mb RAM and 18 Gb SCSI RAID array, and tape backup unit
- 5 Cisco 2514 routers, each with two ethernet ports
- 5 P133 PCs, each with 16 Mb RAM and two network cards (for constructing homebrew routers)
- 5-10 10/100 Ethernet hubs and switches

The abundance of hubs and switches allows for multiple room configurations. Most exercises use mini-networks of one server and three client PCs, connected via a router to the room's backbone. A few exercises have all the computers using the room's backbone directly.

The exercises described below are all based on Linux (RedHat 7.1). We use the network packet analyzer *ethereal* extensively.

EXERCISES

Following are brief description of exercises developed and used in classes:

Single, shared network

- Look at ARP cache as you ping and get pinged
- Ping an actual IP address, a bogus address on same LAN, and an address on an unreachable network.
- Configure 4 computers to use IP addresses with a different network id, but still using same shared physical network.
- Configure a DHCP server, still using the same shared network.
- Examine and explain the chaos!

LANs and routing

- Create an isolated, functional LAN
- Configure a router to connect LAN to backbone using RIP
- Configure a DNS server, integrate into room's domain namespace
- Build a router from a PC running Linux Router Project (LRP)

Measurement and observation

- Configure *apache* and *inetd* (xinetd)
- Watch and explain web browser caching for static and dynamic pages
- Compare running *apache* standalone and from *xinetd*
- Examine plain telnet, ftp, and http traffic
- Examine encrypted traffic using ssh, sftp, https
- Obtain a new digital cerficate for the web server
- Configure service filters via xinetd and packet filters via netfilter/iptables

Additional exercises

These are exercises that we've tried once, but aren't fully developed yet:

- Configure IP subnets
- Examine network capacity by saturating the LAN
- Watch routing traffic and routing tables as routers are brought on-line and off-line.
- Configure routers to use OSPF instead of RIP
- Configure and watch remote file access via NFS

FUTURE WORK

Plans are being developed to add wireless cards and access points to the laboratory. There will be exercises in setting up and integrating wireless networks with wired networks, as well as allowing experiments in breaking messeges sent with the WEP (wired equivalent privacy) protocol. We are also looking at more exercises using a mixed Windows/Linux environment. Exercises can be obtained from the author or by checking http://www.cs.rockhurst.edu/~cigas/adminlab.

ACKNOWLEDGMENT

This work was supported in part by the National Science Foundation's Course, Curriculum, and Laboratory Improvement (CCLI) program, grant number DUE-9952454.

Position Paper: Networking Curricula and Laboratories

Douglas Comer
CS Department
Purdue University

Educational Objectives

A discussion of networking curricula should begin by listing objectives. The next paragraphs provide a few ideas.

Undergraduate objectives. The goal is acquaintance and breadth. A student should have broad familiarity with all aspects of networking: signals on wires, bits on signals, packets on bits, internets on packet networks, and applications on internets. A student should be able to state the purpose and function of fundamental hardware and software components and understand their role. The student should know the role of protocols and understand basic protocol layering. The student should be familiar with fundamental concepts and be able to relate them to technologies. The student should be able to write computer programs that use networks and internets.

Graduate objectives. The goal is mastery and depth. A student should be familiar with the literature, and should understand the intricacies of extant protocols. A student should know about the latest technologies, and be able to contrast and compare them by citing tradeoffs and limitations. A student should be able to devise ways to measure a networking system that identify and expose flaws or bottlenecks. A student should be able to design and build correct and efficient implementations of a protocol stack. A student should understand the difficulties inherent in building and deploying large-scale networks and distributed systems.

Undergrad Courses

Overview Of Networking And Internetworking. Many universities offer only one undergraduate networking course. To fulfill the objective of breadth, the course should be an overview that covers all aspects of networking from wires to applications. If a university offers multiple networking courses, an overview course provides an excellent prerequisite for each of the more advanced courses. Overview topics include: signals over media, bits over signals, packets over bits, internet datagrams over packets, transport protocols over datagrams, applications over transport protocols.

Network Programming. A course that explores the client-server paradigm, with emphasis on the design of clients and servers. Topics include: application protocols, the request-response paradigm, the socket API, server concurrency using threads and processes, multicast applications, super servers such as inetd, and middleware.

Internetworking. A course that covers the entire TCP/IP protocol suite. Topics include: address binding, the Internet Protocol and Internet address assignment, transport protocols, application protocols, and network management.

Network System Design. A course that exposes students to the internal structure of network systems such as bridges and routers. Topics include: software-based network systems, implementation of protocols in an operating system, intelligent and programmable I/O interfaces, switching fabrics, and network processors.

Web Technologies. A course that discusses the architecture of large-scale web sites. Topics include: local and global load balancing, address translation, proxy and reverse proxy caches, and content distribution networks.

Current Trends In Networking. Because networking continues to change, it may make sense to have a generic course that covers the latest technologies and trends. For example, this year such a course might include wireless networking or network security.

Graduate Courses

Although they cover some of the same general topics as undergrad courses, a graduate course should not repeat basic material; students who have not had basics should be required to take the undergraduate overview course.

Internetworking. The classic graduate course that explores internetworking in great depth, with emphasis on protocol design, alternatives, and tradeoffs. Topics include: relationships among naming, addressing, and routing; ways to accommodate heterogeneity; transport protocols; algorithms and techniques used to implement protocols.

Network System Design. A course that explores the design and implementation of network systems such as bridges, address translators, edge routers, core routers, firewalls, and TCP terminators. Topics include engineering tradeoffs, software and hardware architectures, and network processors.

Special topics courses that explore new areas. Because networking continues to change rapidly, a graduate curriculum should include special topics courses that change from year to year to accommodate new content and new research interests. Current special topics might include: Optical Networking (DWDM, relationship with packet switching, and consequences of lambda switching), Routing (a perennial tough subject), Wireless And Mobile Networks, and Design Of Large Scale Services (caching, coherence, etc).

Laboratories For Networking

Laboratories form an absolutely essential aspect of any networking curriculum because students learn by doing. Labs reinforce concepts presented in class, expose students to practical technologies, allow students to understand and appreciate details, and keep courses tied to reality.

It may seem that cost will prohibit some schools from having networking labs. Fortunately, current prices mean that every college and university can afford a minimal lab that consists of a set of PCs on a LAN connected to the Internet through a NAT box. Although the hardware and software available in a lab may limit the range of possible experiments, a small lab is better than no lab. Besides, it is possible to invent interesting and informative lab exercises for virtually any hardware environment. For example, [1] lists sets of possible lab equipment ranging from a single computer to a complex systems design lab, and shows a set of exercises that can be carried out with each set of equipment.

The Author's Laboratories

The author has been devising and using labs for over twenty years, and, along with many student volunteers, has built a variety of labs.†

The Author's Undergraduate Lab. The author teaches a 1-semester overview course for seniors. The current lab for the course contains a set of twenty workstations with extra NICs that students can connect to a hub or switch. The facilities allow students to measure throughput or capture and analyze packets (without compromising security on the production network). Students spend approximately half the semester making measurements and analyzing protocols (e.g., reassembling IP fragments), and approximately half the semester learning the socket API and building a concurrent server. Appendix 6 in [2] describes the facilities, and Chapter 10 in [1] provides further details. Chapters 11 through 13 in [1] give examples of the experiments that students perform.

The Author's Graduate Labs. The author teaches a graduate internetworking course and a graduate network systems design course. In the lab for the internetworking course, students work in teams of three or four to implement a software-based IP router. Each team implements ARP, IP, ICMP, UDP, and their choice of another facility such as multicast routing, NAT, or a VPN mechanism. The lab contains 24 workstations, 85 back-end computers (to which students can download an arbitrary image), and miscellaneous other equipment such as a load balancer, VLAN switches, and 802.11b hardware. See Chapters 19 and 21 of [1] for a description of the facilities, and Chapter 22 of [1] for a description of projects.

The author also teaches a graduate course on network system design. The lab consists of 24 network processors (20 from Intel; 2 from IBM). Students work in teams. Each team proposes an implementation project using the network processors. Chapter 21 of [1] describes the lab facilities, and Chapter 22 outlines a packet classification project.

Course Materials

Fortunately, texts and course materials are available for networking courses and labs. For example, [2] contains material for an undergraduate overview course, including a description of a simplified API; [1] gives a set of laboratory exercises to go with the course. [3] can be used for a senior or graduate level course in internetworking; [4] provides additional details about the implementation of TCP/IP. [5] contains material for a network programming course. Finally, a text is being written for a network processor course [6].

1. Comer, D., *Hands-on Networking With Internet Technologies,* Prentice Hall, 2002.

2. Comer, D., *Computer Networks And Internets,* 3rd edition, Prentice Hall, 2000.

3. Comer, D., *Internetworking With TCP/IP Volume 1: Principles, Protocols, And Architecture,* 4th edition, Prentice Hall, 2000.

4. Comer, D. and D. Stevens, *Internetworking With TCP/IP Volume 2: Design, Implementation, And Internals, 3rd edition, Prentice Hall, 1999.*

5. Volume 3: Client-Server Programming And Applications, LINUX/POSIX socket version, Prentice Hall, 2000.

6. Comer, D. and L. Peterson, *Network And Internet System Design With Network Processors,* (will appear in 2003).

†The author gratefully acknowledges support that has been received over the years from a variety of sources, including donations and grants from Intel, IBM, Cisco, AT&T, Sun Microsystems, Digital Equipment Corporation (now Hewlett Packard), and Lucent.

A Laboratory Course in Wireless and Mobile Systems Design

Luiz A. DaSilva (†), Scott F. Midkiff (†), and Ing-Ray Chen (‡)

† Bradley Department of Electrical and Computer Engineering
‡ Department of Computer Science
Virginia Polytechnic Institute and State University

Virginia Polytechnic Institute and State University (Virginia Tech) is a comprehensive land-grant university with its main campus in Blacksburg, Virginia and additional research and teaching facilities at the Alexandria Research Institute (ARI) and Northern Virginia Center (NVC). The ARI and NVC service the research, education, and training needs of industry and government in the metropolitan Washington, DC region.

The Electrical and Computer Engineering (ECE) and Computer Science (CS) departments have a history of collaboration in both teaching and research. For years, we have offered a number of graduate and undergraduate courses that focus on computer networks. Several of these are cross-listed by the two departments. The main course offerings are summarized in Figure 1.

This paper focuses on a new hands-on laboratory course on Wireless and Mobile Systems Design that will be offered for the first time in the Spring semester of 2003. Intel has recently awarded Virginia Tech a grant to support the development of this course and associated laboratory.

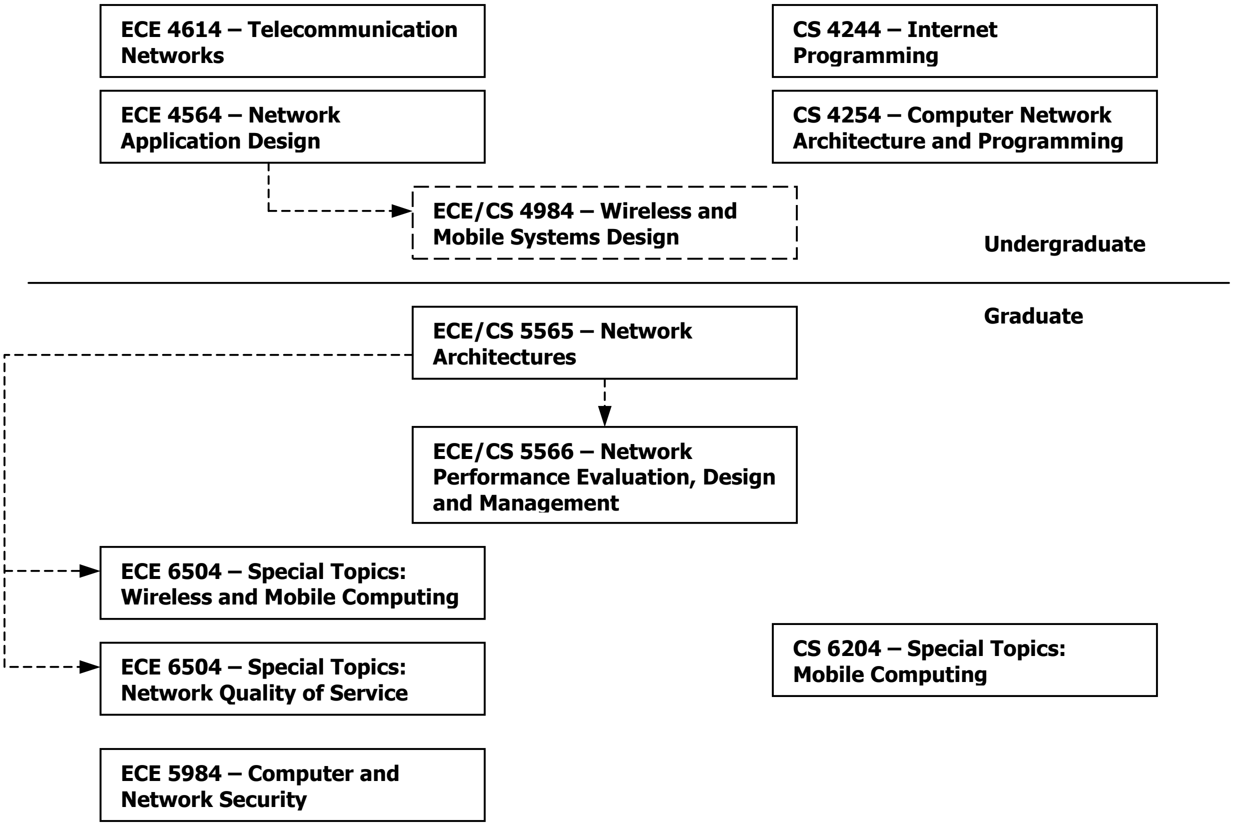

ECE 4614 – Telecommunication Networks
ECE 4564 – Network Application Design
CS 4244 – Internet Programming
CS 4254 – Computer Network Architecture and Programming
ECE/CS 4984 – Wireless and Mobile Systems Design
Undergraduate
Graduate
ECE/CS 5565 – Network Architectures
ECE/CS 5566 – Network Performance Evaluation, Design and Management
ECE 6504 – Special Topics: Wireless and Mobile Computing
ECE 6504 – Special Topics: Network Quality of Service
CS 6204 – Special Topics: Mobile Computing
ECE 5984 – Computer and Network Security

Figure 1 – Course offerings on computer networks at Virginia Tech. Dashed arrows indicate pre-requisites. ECE graduate students may apply a limited number of 4xxx-level courses to their degree.

Traditional networking courses (including our own) often take a layered approach following the Open Systems Interconnection (OSI) model, which we believe lacks the richness and complexity needed to architect and design effective wireless and mobile systems. Therefore, in this new course, we will adopt an integrated approach to the design and characterization of wireless and mobile systems.

As illustrated in Figure 2, our philosophy is to cover an appropriately selected vertical slice of topics that span mobile applications, middleware, mobile networking, and wireless networks and links. This is in contrast to the traditional approach of covering a broad horizontal slice (e.g., as in courses on local area networks or physical layer communications). We believe that this more integrated view of wireless and mobile systems is critical to the success of computer scientists and engineers designing and researching *any* aspect of such systems. Given the realities of prior background, the relative inflexibility of hardware as a design medium, and the time constraints of a three-credit hour class, we take a somewhat uneven slice across the topics. The course will emphasize characterization of the lower layers of wireless and mobile systems to understand the properties of and design constraints presented by wireless networks and wireless links. The course will emphasize design at the upper layers, specifically building applications using middleware and modifying core protocols to examine protocol architectures and to characterize the performance of alternative designs.

Several universities offer hands-on courses in wireline networking. Most of these courses consider the design and configuration of networks (e.g., planning addressing architectures and configuring routers). However, we know of only a few (with a notable example being Georgia Tech) where undergraduates get "under the hood" of TCP/IP and modify networking protocols for wireline systems. Our course follows a similar approach, but considers wireless and mobile systems.

We will leverage mobile devices and networks to create a laboratory environment that is more flexible and portable than a traditional laboratory. This will eliminate the need for continuous dedicated laboratory space (a traditional laboratory consumes space even if the course is not being taught) and will permit us to take the laboratory "on the road" for teaching workshops and short courses.

Students, working in groups of two, will be able to check out equipment such as laptops, palmtops and access points and work on design experiments at their own pace and location. Developing skills in multidisciplinary teamwork is one of the objectives of the course. A mix of ECE and CS graduate and undergraduate students, on the main campus and extended campus in Northern Virginia, will make for a heterogeneous student population with the potential to benefit from one another's strengths.

A number of integrated courses in wireless networking and mobile computing are emerging. These courses tend to be taught at the graduate or advanced graduate level and most do not include hands-on project work. Some courses are notable in their inclusion of design-oriented projects using handheld devices (such as courses at Maryland, Rutgers, and Florida). We believe our course to be innovative in that we will incorporate a more focused and structured laboratory component to enable the course to be taken successfully by upper-division undergraduate students.

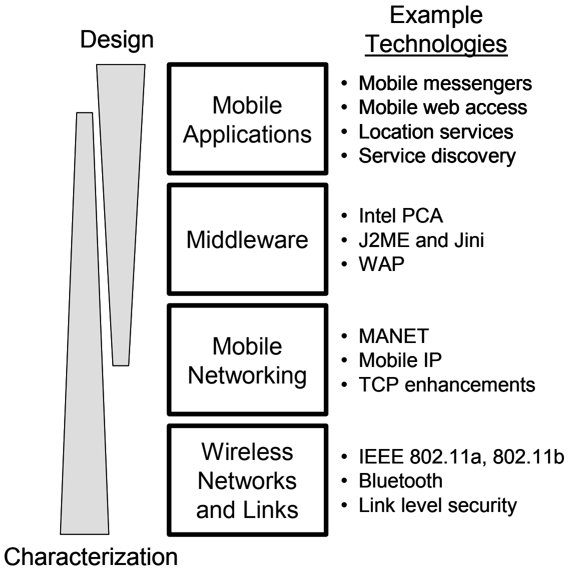


Figure 2 – An integrated approach to wireless and mobile systems design.

Offering a Hands on Computer Networking Course

Magda El Zarki
Dept. of ICS, UC Irvine
elzarki@uci.edu

A hands on computer networking course has tremendous appeal for students as a follow on course to an introductory course in networking. A lab based course enables students to put to practice many of the concepts that they are exposed to in their readings in a first course and that only really "gel" in their minds once they have had to actually use them in a real (versus a simulated or emulated) setting.

The Hands on Computer Networking course has been taught now for a number of years, at both the undergraduate and graduate level, and is filled to capacity each time it is offered. It has taken many years to refine the experiments, and, together with Jorg Liebeherr, we have worked on creating a set of experiments that are designed to walk the students through the material and introduce them to tools and utilities that are fundamental to networking. The goals of the course are to teach the students the inner workings of Internet networking (see Appendix for course outline). As such it explains the meaning of an IP address and the use of prefixes, the importance of ARP, the role of ICMP, the basics of IP forwarding, routing mechanisms and how they are implemented, the difference between TCP and UDP as transport protocols, the essence of NAT, etc.

Although many faculty recognize the benefits of teaching a hands on course for their students, there is always tremendous reluctance to offer a "lab" based course. The three main reasons why most institutions do not offer a lab course are: one, the time commitment involved in designing a comprehensive set of experiments amenable to being taught to a large number of students, two, the management and organization of a significant sized lab course and three, the cost of setting up a robust lab.

The lab course that we designed is geared to be taught in an "open" environment to a large number of students. The equipment is housed in racks in a supervised but otherwise open lab. The students frequent the lab at their own convenience, and work at their own pace with no time supervision. A teaching assistant is available offline, via email, to answer their queries. The teaching assistant is also required to spend a couple of hours a week in the lab for hands on assistance.

The equipment consists of fairly basic, inexpensive components: low end Cisco routers, low end PCs running Linux, and some hubs, power strips, and cables, all housed in 19" racks. The equipment is standalone (i.e., unattached to the dept. network) as the students are required to have root access for equipment configuration. The students are therefore required to save their lab data to removable storage for offline analysis and report writing.

As a faculty member, I am very familiar with the experiments and often demo some parts of the lab during the weekly lectures (I use a mobile "instructor" rack). I also hold part of my office hours in the lab to interact with the students and follow their progress.

The course requires the students to submit weekly lab reports. To test their understanding of the material, the course also requires that each student take both a midterm and a final. The students learn how to work in groups but also know that they have to actively participate in the experiments as their knowledge of the material will be tested in the exams. To prevent cheating (i.e. copying of lab reports from one quarter to the next) we swap NIC cards between racks and hosts. That way, the IP addresses and MAC addresses are different every quarter, making it very hard to do a simple "copy" of previous results.

In conclusion, the feedback from the students is what makes it all worthwhile. For the most part, the students are very appreciative of what they learnt during the labs and to hear them tell you that they finally understand the fundamentals of networking because of this course is extremely gratifying.

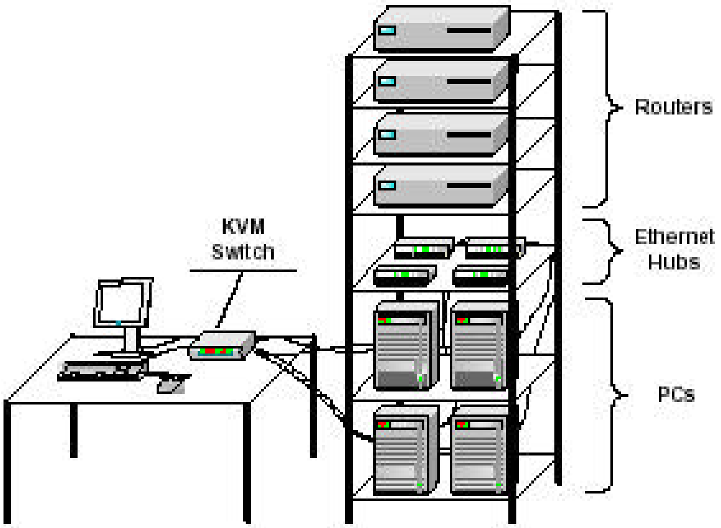
Appendix: Outline of Lab Course

Basic Material:

1. Getting Acquainted
    a. Overview of equipment
    b. Overview of Linux
2. Single Segment Networks
    a. Ethernet
    b. Transmitting and receiving packets
    c. ARP
    d. IP and ICMP
3. Multiple Segment Networks
    a. IP Forwarding
    b. PC and commercial routers
    c. Configuring routers
    d. Static routing
4. Dynamic Routing
    a. Static versus dynamic routing
    b. RIP
    c. OSPF
5. LAN Switching
    a. Transparent bridges
    b. Spanning Tree algorithm
6. Transport Protocols: TCP and UDP

Advanced Material:

7. Multicasting
8. NAT and DHCP
9. DNS



Lab Equipment

A Graduate-Level Networking Curriculum

Wu-chang Feng
OGI School of Science and Engineering at OHSU

Abstract— **The OGI School of Science and Engineering at OHSU offers exclusively graduate-level education to students and working professionals in the greater Portland metropolitan area. In order to keep up with the explosive growth in networking topics and the demands of an increasingly diverse population of students, we are currently updating our curriculum to better meet the educational mission of our university.**

I. Motivation

As a graduate-only school, OGI at OHSU services an unusually diverse set of students. In particular, the student population consists of non-degree part-time students, part-time masters students sponsored by companies such as Intel, full-time masters students, and full-time Ph.D. students. Courses are taught under the quarter system with each quarter lasting 10 weeks. Up until recently, the networking curriculum consisted of a single 10-week course designed to service all students. Inevitably, in a classroom filled with a range of students, this approach failed. Depending on the level it was taught at, it was either inadequate for networking students or inappropriately time-consuming for non-networking students.

The main problem is that inevitably the goals of each student in taking a networking course varies. Some students require only a cursory treatment of a wide breadth of topics. Some students require significant amount of practical training with the goal of applying the skills that they develop in their current or eventual job. Finally, some students require advanced, research-oriented material to better prepare themselves for a rigorous Ph.D. program. Motivated by this, OGI/OHSU is currently re-tooling its networking curriculum, as well as its operating system and security curriculum, to better fit the needs of students.

II. Curriculum

The curriculum developed splits the material into an introductory course, a practicum course, and a research seminar. The introductory course focuses mostly on basic conceptual material drawn from a wide variety of leading sources [1], [2], [3], [4]. Practical programming assignments are limited to simple client-server socket programming. The course material and slides are publicly available [5]. The practicum course focuses strictly on hands-on experience with building networks from the ground up. The course involves a large dose of low-level network programming and requires students to build routers and firewalls using the Intel IXP 1200, a modern, network processor platform [6]. Besides learning how networking devices are architected, the students will also learn basic network administration using Linux-based clients and servers. The course is scheduled to be taught in Spring of 2003, but initial versions of the course material have also been made available [7]. Finally, the research seminar course is a course effectively taught by the students themselves with limited in-class guidance from the instructor. Students are assigned groups of current, related research papers and are asked to synthesize and present an analysis and comparison of these papers. This allows them to develop develop critical thinking skills along with research presentation and formulation skills. Empowering the students was an immense success. The initial course garnered a course evaluation rating of 3.9 out of 4. The papers covered and all of the presentations are also available [8].

The curriculum development described is supported by the generous donations of Intel Corporation.

References

[1] Committee on the Internet in the Evolving Information Infrastructure, Ed., *The Internet's Coming of Age*, National Academy Press, 2001, `http://www.nap.edu/books/0309069920/html/`.

[2] J. Kurose and K. Ross, "Computer Networking: A Top-Down Approach Featuring the Internet," `http://www.awl.com/kurose-ross`, 1999.

[3] S. Seshan, "Computer Science 15-441: Fall 2002," `http://www-2.cs.cmu.edu/˜srini/15-441/F02/`, 2002.

[4] R. Govindan, "CSci 551: Fall 1998," `http://netweb.usc.edu/cs551f98/"`, 1998.

[5] W. Feng, "CSE524: TCP/IP Internetworking Protocols (Fall 2001)," Available via `http://www.cse.ogi.edu/˜wuchang`, 2001.

[6] E. Johnson and A. Kunze, "IXP1200 Programming," 2002.

[7] W. Feng, "CSE58X: Networking Practicum (Spring 2003)," Available via `http://www.cse.ogi.edu/˜wuchang`, 2002.

[8] W. Feng, "CSE581: Networkng Research Seminar (Winter 2002)," Available via `http://www.cse.ogi.edu/˜wuchang`, 2002.

Breadth and Experience in Education for Next Generation Networks

Victor S. Frost
Dan F. Servey Distinguished Professor
Director, Information and Telecommunications Technology Center
Joseph B. Evans
Charles E. Spahr Professor
Electrical Engineering and Computer Science
University of Kansas
2335 Irving Hill Road, Lawrence, Kansas 66045-7612
Phone: 785-864-4833, Fax:785-864-0387, E-mail: frost@eecs.ku.edu

Next generation networks will be complex and composed of a wide variety of technologies, including wireless, optical, and copper based transmission, Internet routing and switching, as well as sophisticated support for multimedia, security, and reliability. Engineers and computer scientist involved with these next generation networks will require a wide range of knowledge. Since about 1983 the University of Kansas has had a focus in Telecommunications Systems Engineering. By exposing the students to courses in random signal theory, digital communications (augmented with simulation experience), communications networks (augmented with simulation experience), and other courses the students learned the fundamentals of each of these areas. Though individual graduate research each student has developed specialize knowledge in a specific area, e.g., optical communications link modeling and design. Often the graduate research projects have been sponsored by industry.

Our experience since 1983 has shown that the following factors were significant in the success of our students:

- Breadth of knowledge gained through course work
- Hands on experience (either through simulation and/or with hardware, e.g., fiber transmission systems, radios, switches or IP routers) gained by means of course and/or graduate research projects.
- Experience and maturity gained by integrating the knowledge acquired from different courses to address one problem and then documenting the solution in a Master's thesis or project report.
- Industry sponsored research projects that created connections for integrating the needs of industry with evolution of our academic program.

While it is likely that the specifics of the technologies in future networks will change and the curriculum will evolve through the introduction of specific courses to reflect these advances, the basic elements required for the success of the students as they move on into their careers will remain constant. For example in the curriculum at the University of Kansas the first networking course has continually changed; reducing the material on ISDN and ATM while increasing the time devoted to IP. We have introduced an Internet routing course with an associated laboratory, where the students get hands on experience with state-of-the-art routers. Recently, we have introduced a two-semester sequence in information security, where the second semester is a practicum as well as a course on optical networks. Throughout the past 18 years we have maintained the emphasis on the breadth, hands on experience, an opportunity to integrate disparate knowledge, and industry involvement. We see these elements as critical to producing the computer scientist and engineers for the next generation of networks. We have organized our communication/networking classes into three focus areas: Internet Engineering, Principles of Communication Networks, and Telecommunication Systems Engineering

Internet engineering: The Internet Engineering area focuses on the application of technology and engineering principles to the design of Internet systems. The emphasis of this area is on IP services and characteristics, Internet protocols, IP routing protocols, information retrieval, and information security. Courses in this area often have associated laboratory experiences where students receive hands on knowledge concerning the configuration and operation of modern Internet systems. The principal areas of concentration for the faculty is on ambient/ubiquitous computing, Internet routing protocols, Internet based distributed applications, and information retrieval. The research resources include extensive Internet routers and high performance switches, a testbed array of networked computers for distributed systems as well as state-of-the-art network software design tools.

Principles of communication networks: The principles of communications networks area focuses on theory and evaluation of networks and systems, with a particular emphasis on network control, traffic management system optimization, modeling, and simulation. Integrated voice, data, and video systems are studied as well as networking using lightwave and wireless technologies. Courses consider the behavior of networks and systems using both analytical and simulation techniques. Topics in advanced courses include active networks, multimedia networks and optimization. The principal areas of concentration for the faculty are distributed performance measurement and modeling (tools, analysis techniques, simulation models, and accurate performance prediction), network control and management systems (traffic management, Internet pricing, self-configuring networks), integration of wireless networks (architectures and protocols, and robustness, ubiquitous and ad-hoc systems), high-capacity network systems. The research resources include extensive wide-area high-speed networking facilities (IP routers and ATM switches), a testbed array of networked computers for distributed systems and simulation research as well as state-of-the-art network software design tools.

Telecommunication systems engineering: The telecommunication systems engineering area focuses on an overall systems viewpoint combining communications theory, digital signal processing, communications networks and principles of optical communications systems. Engineers and computer scientist involved with next generation telecommunication systems will require a wide range of knowledge. Thus the emphasis of this area is on providing a breadth of academic experience. By exposing the students to courses in random signal theory, digital communications, communications networks and digital signal processing the students will learn the fundamentals of each of these areas. The principal area of concentration for the faculty is on telecommunication systems. The research resources include extensive digital communications, optical and network laboratory facilities.

Innovative teaching techniques are being used in our networking program. For example, courses including our fundamental network protocols and systems course and our Internet routing protocols course are now offered through a mixture for web pages, streaming audio/video lectures, and interactive laboratories and problem sessions. The objective has been to enrich the student experience by moving interactive hours with the instructor from the lecture hall to the laboratory. This offers the students' greater breadth and hands-on experience while keeping the students' time commitments feasible. The reaction to these courses has been highly favorable and, based on improved student knowledge, highly effective.

White Paper: Undergraduate Curriculum in Computer Networking

Michael B. Greenwald
Department of Computer and Information Science
University of Pennsylvania

August 20, 2002

We, as a community, exercise much thought and energy on a question posed by this workshop: "What are the "core" topics that should be covered in a first [undergraduate] course? …Which undergraduate multi-course sequences are possible? …". There appears to be general consensus, regardless of the set of topics one believes in, that it is impossible to fit all these necessary topics into a single introductory course. Therefore we content ourselves with asking whether there are a "small set of approaches …(e.g. a more quantitative EE-style versus a more software/algorithms CS-style, hands-on versus in-class lectures; bottom-up versus top-down approaches)" that we can apply to introductory courses.

I contend that by framing the question in terms of choosing the best topics to be taught in *the* introductory networking course, we do a disservice to most of our students. The needs and desires of students enrolled in a so-called "introductory" computer networking class vary so widely that it is either impossible to choose a set of topics appropriate to even half the class or else the material is dealt with so superficially that no one benefits.

We tend to design courses for people who are most like us (teachers of networking): a good portion of the introductory networking courses are aimed at (CS?) students who want to go on and do graduate work/research in networking, with emphasis on protocol design and network algorithms. Another significant fraction are aimed at (EE?) students who want to go on and do graduate work/research in networking, with emphasis on the physical layer and/or queuing theory.

But this does not represent the reality of students that I have encountered (or, at least, this "significant fraction" is surely at most 10 %of the students). In fact, the students I have taught are likely to have come from one of the following constituencies:

- Students who (think they) want to go out and implement network protocols. They want enough of an introductory course to get their foot in the door. They will probably follow up with a course or two.

- Students who (think they) want to go out and design network interfaces.

- Students who (think they) want to learn how to write programs that use existing protocols.
- Students who (think they) want to go out and administer/manage networks.
- Business school students who want to "understand the technology" well enough to be a manager in a networking startup.
- Business school students who want to "understand the technology" well enough to knowledgeably invest in a networking startup.
- Students who want a "liberal arts" exposure to networking: History of networking, an overview of the "deep" or "fundamental" questions posed in the field. It is not just students with no technical background who may be interested in such a course, but may potentially be technically sophisticated students with an interest in becoming science/technical writers.

And this doesn't even include the miscellaneous other students such as the theory folks who think of networks as graphs and most want to hear about multi-commodity flow and min-cut.

What does the fact that students have such diverse needs and goals mean to us? Fundamentally, that we may be better able to address the needs of each of these communities separately better than addressing them all together. Separate curricula may be appropriate for each of these groups. Offering such a large number of independent introductory courses will require a large number of teachers. Matching the course to the constituency may also allow us to match the teaching slot to the appropriate department; it may be best for the business school to hire a lecturer for *their* introductory course(s) on computer networking. Treating each of these groups independently also allows us to more accurately calculate the probability that students will take more than one undergraduate networking course. If we expect, with high probability, that followup courses will be taken, then we don't need to cram everything into the introductory course (and some of the questions posed here become moot).

An advanced networking course on QoS

Sanjay Jha
School of Computer Science and Engineering
The University of New South Wales
Sydney 2052, Australia
e-mail: sjha@cse.unsw.edu.au

Most advanced networking courses had been offered either based on queuing/performance analysis or network/distributed systems programming. The volume and pace of Quality of service research and development in the Internet demanded a course focussed on fundamental concepts that are required to build a class based Internet [1]. At UNSW, we developed/taught an advanced networking subject to provide sufficient depth for major QoS concepts and architectures taking systems approach [2]. The subject teaches the fundamentals and practical solutions to quality of service (QoS) based networks, with an emphasis on the next generation Internet architectures and protocols. Topics include scheduling policies (fair queuing, priority queuing etc.), congestion avoidance/control schemes (RED, RIO etc.), admission control, multimedia protocols (RTP, RTCP etc). This subject covered recent QoS related developments by IETF/IEEE such as: Intserv, Diffserv, RSVP, LAN QoS, Policy-based QoS management and QoS in Mobile/Wireless Network.

The student experience consisted of lectures (2-3hrs approx) as well practical sessions in the Advanced Systems Laboratory. The assessment of the subject included substantial hands on project on building a network system in Linux environment. In order to provide breadth of knowledge, students were required to produce a report and make presentations (we call it mini-conference) on a topic related to recent advances in the area of QoS.

One of the challenges in teaching this subject has been to set-up laboratory session using non-proprietary equipment. Although we initiated/developed/re-used some of laboratory sessions based on Linux, I would like to discuss the idea of a more unified approach in developing/sharing hands-on laboratory sessions with other educators.

References:

1. Jha and Hassan, “Engineering Internet QoS”, Artech House, 2002, http://www.cse.unsw.edu.au/~sjha/qosbook/
2. COMP9333 Advanced Computer Networks http://www.cse.unsw.edu.au/~cs9333/

Experimental and Simulation Techniques for Computer Networking (Experimental Networking)

RPI ECSE 4963 Senior Level Lab and Design Course in Networking

Shivkumar Kalyanaraman
shivkuma@ecse.rpi.edu

At RPI we sensed a real need for both our networking research programs to become efficient in experimentation and for our student population to learn the tools and techniques to become real developers or experimentalists. In response to this need, and with generous support from CAIDA, Intel IXA program and internal RPI-ECSE department TA funds, we developed a full-fledged experimental networking class for senior students, and plan to move our first full-length networking class to the junior year. This way we hope to attract more undergraduates into networking research.

The aim of the class is to teach students (who have taken a first networking class) simulation, design, implementation and experimental analysis techniques so that they can start coming up with new ideas and effectively evaluating them by using these tools and techniques. We have also made it a requirement for all research groups to implement and validate their work in Linux and Emulab. So we expect both graduate and undergraduate students to participate in this class. Each "tool" (eg: simulation, linux/Click/IXA platforms) is first presented in a "black box" fashion, i.e., students learn how to be users of the tool and set up meaningful experiments quickly. This is following by a second stage, where they get "into" the tool and develop their ideas (eg: AQM schemes, TCP tweaks, routing protocols). Experiment design techniques from the performance analysis literature are also introduced. The course has a 4-week final project where students have to take a "major" theme/idea that is roughly specified, and go through the entire design process (using simulation, implementation, iterative refinement) and validate their ideas through a variety of experiments and write a paper. This procedure mimicks the real publication process in networking research.

This course will be offered in Fall with a limited enrolment of about 30 students; and will be expanded next year to 50 students. The department has made available a large 75 seat studio classroom which will be leveraged for this purpose.

Course Description:

This proposed senior-level laboratory course will focus on tools, techniques used in the design and analysis of computer networks and protocols. Tools developed in the course include simulation, animation, visualization, experiment design, trace collection/analysis, experimentation with a combination of Linux and Cisco routers, protocol development on Linux, and modular router development platforms (Linux-based "Click" router platform and the Intel IXA platform). Each tool will be developed in a networking protocol context. Example contexts include TCP reliable transport, TCP congestion control, routing protocols (RIP, OSPF, BGP), traffic management, network management and 802.11 wireless networks. The course will also have a term project that involves a networking "theme" which has to be designed and analyzed through an assortment of tools learnt in class. Prerequisite: ECSE-4670, C programming skills.

This course will top off the networking curriculum for undergrads (with ECSE-2660, ECSE-4670) and will allow the creation of a new undergraduate concentration in computer networking.

Course Outline:

1. **Simulation and animation tools**: help students understand complex networking concepts by viewing the networking system as a black box, varying external knobs ("parameters"), or limiting the views of the protocol.

2. **Simulation development:** involves students actually developing the networking protocol code, albeit in a controlled environment, the simulator. Then they can “run” simulations and vary external knobs (parameters) to incrementally refine their design.

3. **Experiment design:** involves design of a large set of simulation experiments, and fit regression or other functional models to correlate parameter knobs to observed metrics. This is a valuable tool for incremental design and performance analysis because it helps students understand the nature of protocols.

4. **Linux-based protocol development:** Students will develop variants of protocols on real OS platforms such as Linux, and set up experiments to instrument, measure and visualize protocol/system behavior.

5. **Modeling and Analysis using Archived Measurements:** Students will learn how to develop measurement archives and how to analyze protocol behavior based upon pre-collected archives.
6. **Experimentation with a combination of Linux and Cisco routers:** Students will learn how to create experimental scenarios with a combination of customizable and off-the-shelf networking equipment.

7. **Development on modular platforms (Click router and Intel IXA):** Recent developments allow modular code development inside the OS kernel. Students will learn to use these platforms to rapidly prototype and test new networking protocols.

Week-by-Week Plan:

Week 1 Aug 29
Lab 1, Networking commands and socket programming

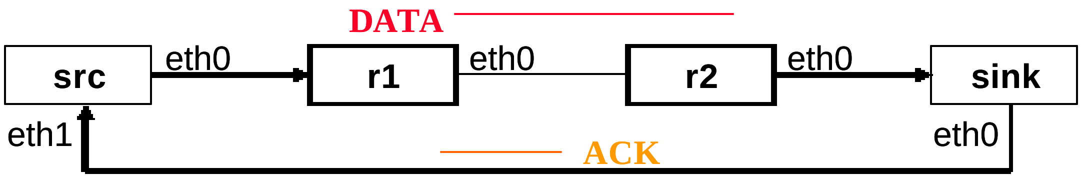
Hardware	1 PC with Internet connection to run networking commands;
Software	networking command (ping, traceroute, tcpdump, iproute2, ...); user space socket programming environment;

Week 2-5 Sept 5,12,19,26
Lab 2, Network simulator NS2 (and NAM)
Lab 3, TCP Tahoe, Reno, and SACK comparisons
Lab 4, Experiment design
Lab 5, Active queue management, part I (RED)

Hardware	1 PC with Internet connection to access NS2 documents;
Software	NS2 version ns-2.1b8a-allinone (or the most recent version) which includes TCP and RED simulation code and scripts; Xgraph, and the sample files (expdtemplate.tcl and sack.tcl);

Week 6 Oct 3
Lab 6, TCP traffic experiment, part I

Hardware	1 Internet connection to access reference papers; 4 PCs connected as follows. Machines r1 and r2 are also PCs but configured as routers and connected to each other with a 10Mbps Ethernet hub to physically create a bottleneck.
Software	tcpdump, tcptrace, xplot, traceroute, route, netstat, ping, iproute2, netperf, sudo, (route pre-configed); NISTNET, and script “wan” as used in testbed-lab; Part II, plus RED, delay, TBF queues.



src/sink/r1/r2 : each with 2 Ethernet 100/10Mbps Cards
Cable: 3 cross-over, 2 regular (1 to connect to the Internet)
Hub: 1 Ethernet 10Mbps hub

Week 7-8 Oct 10, 17
Lab 7, Routing protocols lecture (RIP, OSPF, BGP, EIGRP etc.)
Lab 8, BGP routing table analysis

Hardware	1 PC
Software	MRT, R-Toolkit for graph fitting, awk, Perl, etc. (SSFNet for a later simulation lab)

Week 9-10 Oct 24, 31
Lab 9, Linux kernel programming and Click modular router

Hardware	2(or 3) PCs connected by 1(or 3) crossover cables (for a later lab)
Software	Click router v1.2.2 (or most recent version from www.pdos.lcs.mit.edu/click);

Advanced Labs: Research or Engineering

Week 11-13 Nov 7, 14, 21
Lab 10/1, Active queue management, part II (ARED, REM, AVQ) (see Lab 2)
Lab 10/2, TCP traffic experiment, part II (RED, ARED or BLUE) (see Lab 6)
Lab 10/3, Intel IXA project (see a separate doc)
Lab 10/4, Cisco routing labs (RIP, OSPF, EIGRP) (see below)
Lab 10/5, SSFNet and BGP routing simulations (see Lab 7/8)

Lab 10/6, Multimedia Streaming

Hardware	3 PCs connected using 2 crossover cables;
	1 Haupauge WinTV card, 1 VCR;

Software	q_tbf packet shaper, netperf packet snooper, fame-v0.0.8, cisco mp4live, gtv

Elective Lab 1, Ethernet CSMA/CD multi-access simulations (pending)
Elective Lab 2, 802.11 CSMA/CA wireless access simulations (see Lab 2)
Elective Lab 3, A simple OSPF lab using Click module router (see Lab 9)

Week 14-15 Nov 28, Dec 5 : **Presentations: Idea Sharing**

What should we teach in a first grad-level networking course?

Jim Kurose
Department of Computer Science
University of Massachusetts
Amherst MA 01003

Ten years ago, the first graduate-level Computer Networks course at many schools was an introductory course in computer networking (e.g., taught from Tanenbaum's, Comer's, or Schwartz's text). This was not surprising, as networking was relatively young as an academic discipline, and the wave of interest in networking and the Internet had not yet hit. Many undergraduate CS, CSE, and EE programs did not offer an undergraduate course in computer networking, or when offered, was offered jointly to advanced undergraduates and first/second-year graduate students.

More recently, it is becoming increasingly common for incoming graduate students to have already had an introductory networking course – a trend that will accelerate as an undergraduate networking course becomes a standard piece of the CS, CSE, and EE curriculum. When graduate students arrive having already taken an introductory undergraduate networking course, the question becomes: **what should we teach in the first graduate course in networking?** I've seen this question answered in practice in several ways:

- *An accelerated (re)introduction to networking*. In this type of course, networking fundamentals are taught again, perhaps at an accelerated pace (these are graduate students after all!), and perhaps with readings of "classic" papers in the field.
- *One of a number of specialized courses:* high-speed networks, multimedia networks, network security, peer-peer networking, wireless networks, Internet protocol implementation, network performance modeling, etc. These "hot topic" courses are often based on material from recent conference and journal papers, are often aligned with the instructor's research interests, and often have a seminar flavor. Looking back on my own teaching of such courses, I wonder about the "currency" (say, ten years from now) of what the students have learned.

I am interested in developing a curriculum for a first networking course for graduate students that (like the introductory course) would teach *fundamental material* that would be of lasting value for a systems-oriented graduate student, particularly one working in the networking field. While any definition of what is meant by "fundamental" would undoubtedly provoke a fight, I'd suggest that there are some defining traits to look for in such material – it is likely to be useful to the student ten or more years down the road; it is likely to be broadly applicable in different areas of networking and other systems-oriented fields; it is likely to draw on experience, expertise, and knowledge developed over a number of years. Such a course would ideally provide a solid foundation for, and a gateway to, other advanced networking courses and concepts.

Listed below are initial thoughts on what might be offered in such a course. (Thanks are due here to Don Towsley for several interesting discussions on this topic):

- **Protocols: Mechanisms.** What protocol mechanisms/techniques are found in common in a number of protocols? Hard-state versus soft-state, use of randomization, exploiting redundancy, fault tolerance, announce/listen service location, timer-based protocols.
- **Protocols: Implementation principles.** George Varghese has very nice material on 15 protocol implementation *principles*.

- **Protocols: Specification/verification techniques.** Finite state machines, Petri nets, temporal logic, and elementary proof techniques.

- **Protocols: Network algorithmics.** Self stabilization (routing examples), broadcast/controlled flooding (link state broadcast, ad hoc routing), Kelly's optimization framework (congestion control example), control theory viewpoint of closed loop control (TCP)
- **Network architecture : the big picture.** Overlays (techniques; IP-over-ATM; MPLS; VPN, application-level overlays); lessons from the Internet (and other networks: ATM, telephony); circuit switching versus packet switching revisited; policy, flexibility, and optimized performance.
- **Simulation.** Principles of discrete event simulation, analysis of simulation output, simulation pitfalls, handling scale.
- **Performance analysis.** Intro to queueing: M/M/1; closed loop system models; packet versus fluid models; bounding techniques (e.g., Chernoff bound); normal distributions (equivalent bandwidth)
- **Measurement.** Workload models; traffic and topology characterization, analysis (LRD, heavy tails)

Current Approaches in Teaching Computer Networks

I currently follow the classical bottom-up approach. Although there is a trend on doing a top-down approach, I still think the classical approach is better, especially for undergraduate students. I think that explaining the details bottom up is more logical; students have a complete understanding of how networks work and how they transfer information before programming the application. Students first understand how information is transferred from one node to an adjacent one; reliable point-to-point communication is introduced. Then we increase the network and add more nodes and the routing function is explained. Finally the end-to-end communication is introduced to explain how to cope with the unreliability of the network service; flow and congestion control are introduced. It is harder to see (perhaps for me to teach) the logical sequence in the opposite direction. The bottom-up approach can go from the simple to the complex while the top-down has to go from the complex to the simple. Students can use the socket abstraction earlier in the course to code their applications but they have little idea about the underlying infrastructure and mechanisms that make that information transfer possible.

The top-down approach is excellent in those cases where the class project is related to socket programming. Students get an abstraction of the network and are given the tools they need to transfer the information from A to B early in the course. After that, they are ready to start doing their projects, having plenty of time to complete them. However, they still have no idea of what is behind the scenes. They will find that out later as the course progresses. If the course doesn’t have a socket programming project associated with it (usually my case) I don’t see any advantage of the top-down approach over the bottom-up. In addition, most students do their projects in the very last two weeks of classes anyway!

I also think that the approach to take has much to do with the type of project to be assigned for the class. In my particular case, I am biased toward simulation experiments in both undergraduate and graduate course, although with different objectives. In the undergraduate course, I use OPNET Modeler to let students simulate a “real” scenario using one or more of the networking technologies seen in class. Usually, I ask them to perform a comparison between a scenario that is performing poorly and another one showing the “fixes”. I give one lecture on simulation basics and provide a simple OPNET tutorial so while the course advances they can get acquainted with the tool. OPNET has the advantage that all the students need to do is to drag and drop icons, links, etc, and set up parameters. I have found that the simulation experiments have given students a good real life experience, they go out of the class with basic simulation experience, knowing how to use a good tool, and with a better knowledge of the networking technologies, in particular those they include in their simulations.

For the graduate class, I use ns-2 as the simulation tool. The course emphasis is more on research work. I also give a lecture on simulation basics and provide a tutorial on ns-2 along with a complete simulation example. First, I ask the students to find research papers and reproduce the paper results. Then, I ask them to make a hypothesis about some aspect of the paper the author(s) didn't try. Finally, they have to perform the simulations they need in order to verify their hypothesis. In some cases research papers can be written with a little bit of more effort after the course. The approach taken and content covered in the class is similar to the undergraduate class but with a little bit of more depth in the material and different type of assignments. For instance, students are required to read, summarize and give their opinions on several key research papers related to the topics currently seen in class.

Simulations have another advantage. I have found that graphical animation capabilities built in current simulation tools are a very good way to enhance the lectures. I have used the ns-2 educational examples to show how TCP and data link layer ARQ mechanisms work. The visual demonstration has a big impact on students; they learn the mechanisms easier and retain them longer.

We don't have a networking lab for students to do lab work. This is something I have been thinking about but two main things have precluded me to pursue them as of yet. First, is the time and resources; second the cost of the lab and its maintenance. I am currently looking for virtual labs that students can do seated from their PCs at home or from the computer labs.

We are in the process of designing two graduate courses on networking to be taught after the first introductory-level one. One of those will focus on network performance and the other one in specific network technologies that because of lack of time are not covered in the first course, like ad hoc networks, MPLS, optical networking, etc.

10 Thoughts on Networking Labs

Jörg Liebeherr
Department of Computer Science
University of Virginia
Charlottesville, VA 22904
Email: jorg@cs.virginia.edu

Here are some thoughts on my objectives for a lab course in computer networking.

Objective 1: Try to make education in computer networking more concrete.

Traditional courses in networking do not provide hands-on access to Internet equipment and software. In fact, courses that provide exposure to actual network environments are still mostly absent in an undergraduate curriculum. As a result, even after an introductory networking course, key concepts such as the dynamics of routing algorithms and other networking protocols are viewed by students at an abstract algorithmic level. Lab courses try to remedy this situation and provide students with hands-on experience of networking concepts.

Objective 2: Don't teach a vendor-specific course on router configuration

Teaching a system or network administration course should be a non-goal for a lab course at a college or university. We are not interested in teaching students details of router configuration commands. Certification courses and training programs that cover details of router configuration and troubleshooting already exist and need not replicated at universities.

Objective 3: Use science labs as model

Lab courses in the sciences can serve as a model for a lab course in networking. The labs can be organized so that guided observations and measurements by students lead to insight and understanding of the subject. The object of study are network traffic and network protocols. Instead of scales and voltage meters, students work with traffic measurement tools, e.g., *tcpdump and ethereal*.

Objective 4: Build on prior knowledge

Lab courses are ideal as a second course in computer networking. After an introductory networking course, students understand the notions of flow control or routing algorithms in computer networks, but have never observed these algorithms running in a real network. In a lab course, students add to their knowledge of computer networking by experimenting with protocols in an actual network.

Objective 5: De-emphasize skill – emphasize learning

The primary purpose of a lab course is to study networking protocols in operation. A lab course should not try to make students experts in configuring or troubleshooting Cisco routers. The knowledge of router configuration provided in the labs should be just so students can complete the lab exercises. However, students should take away from a lab course an appreciation for the complexity of IP router configuration.

Objective 7: Students should feel in control of the equipment

Labs should be organized so that students can completely understand the configuration of each lab experiment, and can do the lab setup themselves.

Objective 8: "Keep it real"

Don't use simulators, emulators, and web-based configuration tools, as they present layers of abstractions between students and the equipment they operate. Don't use complicated configuration tools, since they direct focus at configuration equipment and away from the study of protocols.

Objective 8: Organize a lab in 3 phases: Prelab, lab session, lab report

To maximally utilize the time that students work on the equipment, before each lab session, students should complete preliminary exercises (prelab exercises) and read background material. During a lab session, students are asked to collect data. The data is analyzed in reports to be completed after a lab session.

Objective 9: Leverage time investment

Designing, writing, and testing a single lab are a substantial investment of time. Therefore, a lab course should stay relevant for several years. Lab experiments that require expensive hardware or software present a risk, as it may not be feasible to maintain the equipment or upgrade the software.

Objective 10: Control the need for supervision

"Closed labs" where students conduct their experiments under supervision by an instructor or teaching assistant, create an excellent environment for learning, but require a substantial amount of time by teaching assistants or instructors. An "open lab" approach, where the equipment is located a public area and where students can do their lab experiments without any supervision, can significantly reduce the need for supervision.

Computer Networking Curriculum: In Search of How much of What

Deep Medhi
University of Missouri–Kansas City
dmedhi@umkc.edu
August 11, 2002

Computer Networking is a dynamic field. When I first started teaching it a decade ago, having just a course is all that was pretty much needed. Over time, with the growth in knowledge base of Computer Networking, it dawned on me that we needed to create and teach multiple courses. Not only that the course content needed to be dynamic as well! Dynamism of course content also meant that deciding what to take out and what new topics to include on a timely basis.

Fortunately, with our graduate program emphasis in networking/telecommunications, it has allowed us the platform to have multiple courses. Over the course of the past ten years, I personally created and taught ten different graduate level courses including special topics courses[1]. Here's the list (in reverse chronological order of first offering):

- CS690Z: Special Topics: Access Networking
- CS590P: Special Topics: IP Telephony
- CS526: Network Routing
- CS521: Network Architecture - II
- CS590C: Topics in Network Management
- CS590T: ATM Networks and Internetworking
- CS520: Network Architecture - I
- CS522: Computer Networks: Design and Analysis

Having taught these many different courses have also given me a perspective on how much of what is needed which is itself somewhat dynamic (over time). It is important to point out that talking to my counterparts in industry has allowed me to see what new topics or theoretical underpinnings I should emphasize on. The overall experience has also allowed me to develop a new undergraduate hands-on course recently; obviously, given that I was trying to cover many topics in this undergraduate course also turned out to be more of a challenge than I initially envisioned.

If given the opportunity (at the workshop), I'd like to discuss some of the things I've learned about developing different courses, an how emphasis can be put on different issues. Hope this can benefit others, and at the same time, I would like to hear others' experience which will certainly benefit me.

[1] While this can be taxing, it has been a great challenge and experience!

Northeastern University, Boston
MS degree in Telecommunication Systems Management

Starting in fall 2002, Northeastern University is offering a MS degree in Telecommunication Systems Management. This is a multi-disciplinary degree program involving the Colleges of Engineering, Computer Science and Business Administration. Students need to take a minimum of 40 quarter hour credits (will be 30 semester hours starting in fall 2003). There are four new required core courses and electives are chosen from a predetermined set of existing courses in the three participating colleges.

The four core courses are

Fundamentals of Communication Systems,
Telecommunications Architecture & Systems,
Data Networking, and
Telecommunications Public Policy and Business Management

Fundamentals of Communication Systems offers a comprehensive understanding of the underlying physical layer technologies used in the telecommunications industry. The breadth ranges from topics such as signals, channel characteristics, transmission principles, and modulation techniques to multiplexing, radio transmission and photonic communications.

Telecommunications Architecture and Systems provides complete coverage of the telecom network today, focusing on the architecture of the network, network systems, and overlays. The scope ranges from topics such as network synchronization and xDSL to signaling, Intelligent Networks and service management. The course covers cellular/PCS networks, from analog to 3rd generation, and addresses the evolution of the network to packet technologies.

Data Networking teaches the basics of data networking protocols and architectures in a relatively non-quantitative manner. The course covers protocols from the data link to the application layer focusing on IP and ATM networks, and includes topics such as routing algorithms, congestion control, Ethernet, security, QoS, network management, naming & addressing.

Telecommunications Public Policy and Business Management introduces business management issues such as basic accounting, finance, marketing and operations in the telecommunications field, as well as topics such as the time value of money and decision-making. The course also provides an understanding of the regulatory environment of the telecom industry including global trends in market reform.

Electives

Students need to choose at least *three* electives from one of *three* areas of concentration: *Telecom Networking*, *Telecom System Development*, and *Telecom Business Management*. At least one course is required in Telecom Business Management and one in either of the remaining two concentrations.

Telecom Networking focuses on network and communications technology. Electives include topics such as digital communications, spread spectrum communications, wireless communications, broadband networks, mobile networks, network security, internetworking, and performance analysis.

Telecom System Development covers the development of software systems and applications. Electives include topics such as software engineering, object oriented design, database management, software testing and validation, and distributed systems.

Telecom Business Management addresses engineering management and marketing. Electives include topics such as project management, financial management, engineering economy, managing professionals, managerial communications, marketing management, and new product development.

Program Director: Peter O'Reilly, poreilly@coe.neu.edu
Full details are at http://www.coe.neu.edu/COE/grad_school/telecom

Using Ns in the Classroom and Lab[*]

Christos Papadopoulos and John Heidemann[†]

1 Introduction

The ns-2 network simulator is widely used in research to evaluate new networking protocols (see `http://www.isi.edu/nsnam/ns/ns-research.html`). Even though ns has been used by researchers, it has seen relatively little use in the classroom and laboratory. Yet network simulation is a good fit for classroom and laboratory use because simulation allows capturing and dissecting all aspects of protocol operation in a much simpler way that is possible with real code and experiments. On the other hand, ns is a large piece of software, portions of it can be complex, and start-up overhead makes it difficult to bring to bear by a new student or busy professor.

This white paper describes how we are using ns in networking education. We seek to apply simulation to two different areas: the classroom and the laboratory. For *classroom* use, we seek to augment lectures with animations that show specific protocol behavior. In *laboratory* use, students modify simulation scenarios to explore protocol design choices. Here Ns is used directly by students, who submit their work for grading. We give a brief summary of how we have used ns in these roles and changes we have made to ns to make it more amenable to such purposes.

2 Classroom Use

Ns' companion tool nam provides packet-level animations of ns simulations. Animations have been used before to show algorithms such as sorting. We believe that animations are also useful to illustrate network protocols, by visualizing packet exchanges and state distributed in different nodes.

We are building a library of animations that illustrate several networking protocols, including transport-level issues (stop and wait, the effect of various back-off strategies, and TCP-specific issues such as slow-start and fast retransmit), router queueing policies (drop-tail, RED, etc.), multicast routing (flood and prune, PIM shared- and source-specific-trees), and reliable multicast (SRM, PGM).

Experiences: Our experience with animations in the classroom have been very positive. Since they play-out at some fraction of real-time, animations are particularly good at demonstrating time-dependent concepts such as the delay in TCP's reaction to congestion or loss. Animations are also good at representing distributed state since one animation shows several nodes at the same time. For example, nodes are annotated with SRM timer values to illustrate the benefits RTT-biased and randomized delay intervals, and colors are used to distinguish nodes waiting to send a repair request from those whose requests have been suppressed.

We are also gaining experience authoring animations. Important lessons learned include the need to have separate animations, each focused on a specific concept, rather than tackling multiple concepts in a single animation. It's also important that the animation be completely self-contained. Initially we anticipated animations to be augmented with web pages (particularly for self-guided animations) In practice, however, it has proven difficult for an observer to know when to focus on the web page and when on the animation. Finally, as a practical matter, it should be very easy to get started using animations. It should be easy to find animations, understand the context of the work, and put together the necessary pieces quickly. The ability download a binary version of nam and animation scripts are important simplifications (as opposed to compiling the software and running simulations) if they are to be integrated into a busy semester.

Approaches: Our initial experiences with classroom annotation have prompted several developments. First, we established a web-based repository of educational ns scripts at `http://www.isi.edu/nsnam/repository/`. This database stores scripts in a uniform format and allows anyone to contribute new scripts via a simple electronic form. As of this writing, the database contains about two dozen modules contributed from four different institutions.

Second, we have refactored some of or early animations into smaller, better focused modules to clarify the concepts.

Finally, we plan to gradually improve nam's annotation capabilities. Although nam currently provides packet-level animations with a fair amount of control over node labeling, color, shape, and packet color, more work is needed to add text annotations for packets (for example, to label a packet "3rd dup-ack" for TCP fast-retransmit), and make these capabilities easier to use from ns scripts.

[*]This work was supported as part of the CONSER project, funded by NSF as grant number ANI-9986208.

[†]Christos Papadopoulos and John Heidemann are with the USC Comptuer Science Department and the USC Information Sciences Institutes. E-mail: {christos,johnh}@isi.edu.

3 Laboratory Use

An important complement to classroom lectures are laboratory experiments. In networking, this often implies programming, protocol design, experiments and measurement. We believe that simulation has an important role here, since it allows students to examine problems with much less work and of much larger scope than are possible with experiments on real hardaware. Simulation can be easier than experimentation because simulators do not need to reproduce all the details of the real world and they can be easily instrumented. In addition, simulations of dozens or hundreds of nodes are easy on limited hardware, many more than is affordable if physical hardware was required.

We have used ns to do several types of laboratory experiments. The simplest are of the form "run this script" and examine the trace output or nam animation, asking students to identify TCP behavior. The next step up is to have them modify the simulation script in simple ways that require some or little understanding of the script details. Examples include "change the router queueing policy from drop-tail to RED", "vary the link propagation delay and observe the results", or "observe this scenario and describe what to change to improve throughput". We have also assigned simple protocol implementations in the context of a message passing framework (described below) or modifying an existing C++ implementation.

Experiences: Our experiences with experiments as homework problems have also been positive, but clearly such problems must be designed with care. If more complicated assignments are to be assigned (such as those requiring new coding), it's best to introduce the simulator gradually.

One observation that initially surprised us is that many students were not familiar with the concepts behind discrete event simulation. Confusion between real-time and simulation-time, and multithreading and event-driven programming can be prevented with a brief review of the concepts (typically a half-hour to hour of lecture time).

We were pleased to discover that students adapted quite quickly to using either Tcl to specify new scenarios, or C++ to changing existing modules, and many were able to use Tcl as an scripting language to specify the scenario. Efforts that require them to work in both languages simultaneously are probably best reserved for more advanced classes. We have been hesitant, however, to give students a blank slate. The framework of an existing script is necessary to avoid stumbling over initialization details that are irrelevant to protocol design.

Approaches: Our experience has suggested several helpful approaches. First, we are developing a graphical editor based on nam that allows strictly GUI-based creation of simple scenarios. With the editor, topology and traffic design can be done by point, drag and click operations. Configuration of parameters is done with dialog boxes, and the simulation can be launched directly from the editor window. The editor hides irrelevant details such as initialization and scripting, allowing undergraduates to do simple experiments from scratch. While, we do not believe a GUI editor can encompass the whole range of simulations possible in ns (there are simply too many options to make that feasible), the editor exposed a subset of the ns functionality without any traditional programming. (Our experiences with lab use of the editor so far are limited to one semester.)

If ns is to be used for coding complete new protocols, the amount of background students require must be minimized. We are developing a *message passing* module in ns to allow simple protocols to be developed within a subset of the simulator. This includes simple ways to add headers and process messages at each node, with alternative implementations either completely in Tcl and completely in C++. Early experience in one semester has been promising: as an example homework, students successfully simulated scenarios showing the synchronization of periodic routing messages, as described by Floyd and Jacobson. As a side benefit, the message passing model may also be useful for researchers who want to quickly prototype a new protocol.

Finally, as a practical matter, an ns installation can sometimes be difficult and by default it consumes a large amount of disk space. We recommend installing ns on personal machines using the "allinone" package, which provides a simple download and installation process. Variations in Windows development environments have encouraged us to provide a pre-compiled binary for that platform.

To mitigate the size of an ns installation, in systems where many accounts have a shared file system we use a shared installation of ns. For assignments where students need to modify and recompile ns, we have a procedure where students create symbolic links to the source. Students then remove the symbolic links and make copies of the specific files they would like to change.

4 Conclusions

We have been happy with our use of ns in the class and lab, although we plan to continue to refine the tools and lessons. Perhaps ns and nam are now able to serve not just as tools for researchers, but also as tools for education.

Security: Should it be central to a first course on networking?

Keith W. Ross
Institut Eurecom

Network security has become, of course, one of the most important networking issues in the “real world” over the past few years. Network security is a vast subject, which includes secure communications (such as secure online commerce and secure e-mail) as well as network attacks and their prevention. I believe that network security will remain near the top of network practitioners’ concerns for many years (decades?) to come.

In most first courses on computer networking today, network security is briefly taught at the end of the course if taught at all. The question I would like to raise is whether security should become more prominent in a first-course on networking. Specifically:

> *In a modern undergraduate course, should instructors introduce basic security principles (encryption, public key infrastructure, digital signatures, etc.) in the first few weeks of the course, and then continuously discuss how these principles can be applied in the various layers as the course progresses?*

For many networking instructors, network performance has been, and continues to be, one of the central issues in an introductory course. Many instructors introduce network delay concepts (for example, transmission delay and propagation delay) early in a course, and then leverage these concepts while discussing networking protocols as the course progresses. One of the salient arguments for continuously stressing performance concepts throughout the course has been: if students think about performance while learning networking protocols, they will continue to think about performance (as they should) when they design new networking protocols and technologies.

I continue to support this argument, as I believe performance issues remain central to networking. But it is clear that modern practitioners and researchers should also think about security when they design new networking protocols and technologies. Thus, as has been the case with performance and delay, it can be argued that students should be continuously thinking about security while learning networking protocols.

One interesting pedagogic question: Will undergraduate students be better trained if they learn about security at the end of the course (the series approach) or if they learn about security throughout the course (the parallel approach). Because contemporary network practitioners and researchers should think about security at just about every step of the design process, one might advocate the parallel approach. On the other hand, perhaps many students will be overwhelmed trying to learn network principles, protocols, performance, *and* security all in parallel?

How might a parallel course be designed? Here are some off-the-cuff thoughts:

- Weeks 1-2: Introduction to computer networking, including the network core, the network edge, delay and loss concepts, protocol concepts.
- Weeks 3-4: Introduction to security, including overview of symmetric- and public-key encryption, authentication, integrity, and key distribution.
- Weeks 5-6: Application layer, including secure e-mail and security issues in P2P file sharing.
- Weeks 7-8: Transport layer, including secure sockets layer (SSL) and SYN flooding attacks.
- Weeks 9-11: Network layer, including firewalls and IPsec
- Weeks 12-13: Link layer, including sniffing attacks and security for wireless LANs

(Thanks to Jim Kurose to several interesting thoughts on this general topic.)

Informal Notes on Some Curricular Issues in Computer Networking

Harish Sethu
Assistant Professor
Department of ECE, Drexel University
3141 Chestnut Street, Philadelphia, PA 19104
E-mail: sethu@ece.drexel.edu

***Abstract*— In these informal notes, we focus on two curricular issues in computer networking. In Section I, we discuss student learning in networking courses and raise several points of discussion but reach no particular conclusions. In Section II, we attempt to answer the question: Is there a set of advanced, foundational material that applies broadly across the field at the graduate level?**

I. On Student Learning

The College of Engineering at Drexel University uses a web-based course evaluation system that seeks to measure student learning in courses. Students rate themselves on their level of expertise (on a scale of 1 to 5) *before* and *after* the course in topics corresponding to up to six different learning objectives defined for each course. To the author's relief, students have never rated their expertise level lower after going through one of his courses!

Two of the many examples of these learning objectives used in the course evaluation surveys are as follows:

- A good understanding of the transport layer protocols in the Internet, primarily TCP, and the associated issues of flow control, congestion control and reliable data transfer.
- A good understanding of security issues in computer networking, including fundamental principles of cryptography, public-key encryption, authentication, data integrity, key distribution and certification, PGP, Internet commerce, and IP Security.

These learning objectives are defined specifically for each course based on the syllabus actually used for the course.

During the last 3 years, the author has taught three different introductory courses in networking using three different textbooks [1–3] for which self-reported learning data were recorded. [1] and [2] were used for a first-year graduate course while [3] was used for a senior-year undergraduate course. The graduate courses were taught at an accelerated pace in comparison to the undergraduate courses. Self-reported learning data were also recorded for specialized courses taught by the author using two other textbooks: an undergraduate course on network programming using [4] and a graduate course on queueing theory using [5]. The core aspects of the author's teaching style and related details—the author's presentation style, the class size (close to 60), the dreaded EE-style quantitative exams, and even the author's accent—did not change during this period. Fig. 1(a) shows the average difference in the self-reported expertise level of the students before and after the course using each of the textbooks discussed above.

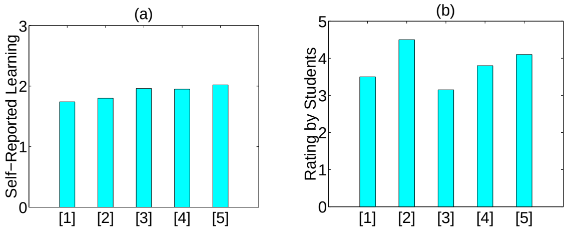


Fig. 1. Results from student evaluations of networking courses: (a) Difference between the self-reported expertise level of a student before and after a course, plotted against the textbook used in the course; (b) The rating of the textbook by the students in the course.

The following are some inconclusive observations and unsubstantiated opinions:

- The survey represents a self-selected sample with an approximately 60% response rate. Analysis of this data needs some help from sampling theorists: is it possible to draw any conclusions from such data or is a self-selected sample altogether worthless?
- The learning measured is self-reported; this measures *perceived* learning and not necessarily *actual* learning.
- Graduate students tend to report slightly lower learning scores than undergraduates (is it "the more you know, the less you think you know?" or "the more you already know, the less you feel you learn from an introductory course" or "the more accelerated the course, the less you feel you learn?")
- The learning reported in more advanced courses is slightly higher (such as in a course on queueing theory). This is to be expected since fewer students begin the course with some knowledge of the material.
- Of all courses taught within the College of Engineering at Drexel University, the learning reported by students varies greatly from as little as 0.7 for some courses to as high as 2.3 for some others. However, the differences between courses taught by the same instructor, as shown in Fig. 1(a), tend to be minor even with differences in content, textbook and how advanced the course is. It appears that perceived learning depends more on the instructor (teach-

ing style, enthusiasm, motivational skills, etc.) than on anything else. It is the opinion of this author that while *perceived* learning may depend primarily on the instructor, *actual* learning depends on the instructor as well as on a variety of additional course details such as the textbook and its approach (top-down vs. bottom-up) and whether classes are held on Friday evenings.

- As can be seen from Fig. 1(a) and Fig. 1(b), there exists little or no correlation between how much students like a textbook and how much they feel they have learned from the course. Is it possible that a highly rated textbook liked by students improves actual learning significantly but does not quite improve perceived learning? ("A great teacher makes the subject seem so easy that students feel they didn't really learn much, even though they actually did?")
- It is the author's opinion, unfortunately based merely on anecdotal and experiential evidence, that a top-down approach to teaching an introductory networking course renders the course significantly more interesting and also increases *actual* learning in comparison to a bottom-up or other approaches.

II. On Foundational Courses

In this section, we make an attempt to answer the question: Is there a set of advanced, foundational material that applies broadly across the field at the graduate level? For lack of a better methodological means of arriving at an answer, we assume that such a set of foundational courses should significantly improve a graduate student's ability to understand and appreciate the mathematical and other tools used in research papers in networking. (Sincere apologies to those who go to graduate school for some purpose other than to learn to read research papers.)

In this study, we create three categories of mathematical or other tools typically used in networking papers, and we also add a miscellaneous category for tools that do not fit under any of these three categories. An advanced graduate course can cover each of the first three categories described below:

- *Course A (The Art of Measurement, Simulation and Analysis):* Object-oriented modeling techniques for building discrete-event simulation models; process interaction and other approaches; exploiting multi-threading; memory management and disk scheduling strategies; pseudo-random number generation; network measurement tools and techniques; Data analysis and presentation techniques.
- *Course B (Probability Theory and Queueing Systems)*: An advanced introduction to probability theory; random variables; random processes; The exponential distribution and the Poisson process; Markov chains; queueing systems and their applications in networking; Brownian motion and stationary processes; self-similarity.
- *Course C (An Advanced Introduction to Algorithms):* Advanced data structures; sorting and searching algorithms; graph algorithms; data analysis; dynamic programming; matrix operations; complexity theory.

Many other graduate-level foundational material used in typical networking papers could not be categorized into one of these three course categories and were placed in the miscellaneous category. A vast number of topic areas fall under this category; they include, among others, control theory, game theory, algebraic number theory, information theory and signal processing algorithms.

This author examined a total of 279 papers published in IEEE INFOCOM 2002, ACM SIGCOMM 2002 and during the last one year in IEEE/ACM Transactions on Networking (yes, this author did indeed read through them all ... kind of). The tools used in each paper were examined and each paper categorized under zero, one, two, three or all four categories. The following rules were observed in the process of categorization:

- If a paper used undergraduate-level material, such as simple calculus or simple differential equations, or well-known results in discrete mathematics, it was not placed in any category since graduate-level foundational material was judged unnecessary to understand this paper.
- If a paper used very simple simulation scenarios, it was not necessarily categorized under Course A. Only if the author judged the simulation environment to be complex enough, was the paper placed under the Course A category.
- If a paper used a very advanced technique or some very esoteric result that would normally not be covered in a graduate-level course, and, in addition, did not use any other graduate-level material, then also the paper was NOT placed in any category.

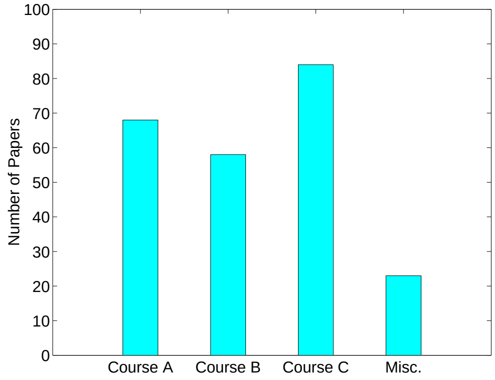


Fig. 2. The number of recent papers that use mathematical and other tools covered by specific course materials.

Fig. 2 shows the number of papers that fall under each of the four categories. This suggests that a set of three advanced foundational courses—one on probability theory and queueing systems, one on simulation and measurement techniques, and one on algorithms—can make the vast majority of research papers more readable to graduate students and prepare them for independent research.

References

[1] S. Keshav, *An Engineering Approach to Computer Networking: ATM Network, the Internet, and the Telephone Network,* Addison-Wesley, 1997.

[2] J. F. Kurose and K. W. Ross, *Computer Networking: A Top-Down Approach Featuring the Internet,* Addison-Wesley, 2001.

[3] L. L. Peterson and B. S. Davie, *Computer Networks: A Systems Approach,* Morgan-Kaufmann, 2000.

[4] W. R. Stevens, *UNIX Network Programming: Volume 1,* Prentice Hall, 2nd edition, 1998.

[5] D. Bertsekas and R. Gallager, *Data Networks,* Prentice Hall, 2nd edition, 1992.

"Network Design and Evaluation"
A Project Course based on Network Processors

Peter Steenkiste
Department of Computer Science and
Department of Electrical and Computer Engineering
Carnegie Mellon University

"Network Design and Evaluation" is a capstone course in the ECE department at Carnegie Mellon University. Small teams of 2-4 students develop a router extension, using an IXP 1200-based router as a starting point. The students have to go through a design, implementation, and evaluation phase, and at each stage they have to write documents and give presentations describing their work. The course is a 500 level course, so it is taken by both seniors and graduate students (typically MS students). The prerequisites for the course are: completion of at least one course in computer networking, and some project experience. This course has been taught in Fall 2000 and 2001, and will be taught again in Fall 2002. This abstract gives a brief overview of the course structure. A more detailed course description can be found at http://www.cs.cmu.edu/~prs/cap/SylCap.html

Course Structure

A hardware platform based on a network processor combined with a suitable software environment is an attractive platform for teaching hands-on course on network internals. It is more realistic than more traditional platforms (software emulation environments or PC-based routers). The reason is that it has a more realistic system architecture, e.g. separate processors for control and data plane processing and different types of memory for different types of data. While commercial routers would be even more realistic, they are closed systems that are too complex to be used for teaching purpose. Using a network processor platform is good compromise between complexity and the degree of realism.

The "Network Design and Evaluation" course has three components:

- The first component is a set of lectures, lab sessions, and programming assignments that is designed to bring the students up to speed on the IXP 1200 platform. Since the system is complex, coming up to speed quickly is challenging. The goal is that all students become familiar with the overall systems architecture, but when learning about the details of the system, students in each team specialize. Note that dealing with a large amount of information (such as the documentation of a complex system) is an important skill that is typically not taught or required in regular courses.
- The second component is a set of lectures on router architecture and network processors that teaches students about router design, beyond the platform used in the course. These lectures are independent from the project and can be taught at any time during the semester.
- The main component is the course project, which is the focus of the remainder of this abstract.

The Capstone Project

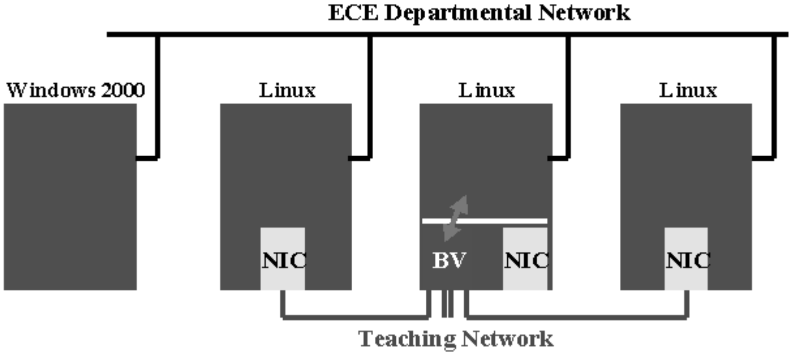
There are two ways of organizing the projects. The first option is to use predefined projects, i.e. all teams do the same project or they choose from a small number of predefined projects. The advantage of this approach is that it is easier to prepare and it is less likely that there will be surprises later in the semester since the teaching assistants can try out the projects. The drawback is that the project will have the feel of a large programming assignment. However, when using IXP-based projects in a short amount of time (e.g. as part of an introductory course), this option is definitely the way to do. The option we use is to let students define their own projects. This is more work for the instructors and there is a higher risk for "failed" projects, but it has the big advantage that the students feel like they own their project. As a result, they are more likely to take initiative. It turns out that there are a small number of topics that are natural candidates for course projects. They include: a firewall, a NAT, RED and ECN support, and various types of router support for QoS. However, students sometimes propose less obvious topics, e.g. source-based routing and porting RSVP.

The project proposed by each team of students must cover three functions. A first component is a data plane component, i.e. a network element that is involved in processing packets that are forwarded by the router. This component will be implemented on the network processor. Examples include a packet scheduler that supports quality of service, or packet filtering for a security firewall. The second component is a control plane component that manages and controls the data forwarding function. Examples could be a signaling protocol that sets up the QoS packet scheduler, or a management interface for a firewall. A third component consists of one or more applications that stress the new network feature. For example, a video streaming application that uses a connection with a bandwidth guarantee, or an application that tries to break into a network protected by a firewall. The balance between these three components can differ significantly across projects.

The goal of the course is not just to implement a project, but also to gain experience in writing documents and giving project presentations. Each team typically makes four presentations (project proposal, two design reviews, and a final presentation) and writes four documents (project proposal, design document, a status update, and a final report). Besides technical issues, the reports and presentations must also cover project management (milestones, task assignments for project members, risk, ..). Students get written feedback on both the documents and the presentations. Besides class meetings, we also schedule periodic (roughly once every 3 weeks) individual meeting with each team to go over the project status.

The specific platform used in the course is the Bridalveil platform. It is an evaluation platform for the Intel IXP 1200 chip and consists of a PCI card that has an IXP chip, static and dynamic memory, and the necessary buses. The IXP architecture has a set of microengines that are used for fast-path data forwarding, and a StrongArm processor that is used for the control plane and for slow-path processing. The card has four 10/100 Ethernet interfaces, and using the Intel-provided L3 Forwarder project, it functions as a small 4x4 router. Both the StrongArm and the PC host run Linux, and the IXP device is managed by a PCI driver (e.g. download code, monitor card, ..).

Each team in the course has its own experimental station. The initial setup consists of a Bridalveil card hosted by a Linux PC (see figure below). Two other Linux PCs have a second network interface card (NIC) that connect them to the Bridalveil card, so they function as hosts on the teaching network. Each team also has access to a Windows PC that runs the IXP software development environment. This set up is sufficient for most development and testing. Later in the semester, more NICs and Bridalveil cards can be added to test projects over larger networks, and teams can combine their hardware.



Experience

Our experience with the course has overall been very positive. The students like working with the platform and enjoy seeing their router functions being used by real applications running on real hardware. In the two semesters that we have offered the course so far, all teams were able to run their project on the hardware. Informal monitoring also shows that students are learning relevant skills: the quality of presentations and document improves throughout the semester, and the students' understanding of networking clearly improves also. The steep learning curve for the platform is a challenge, but more so for the teaching assistant than for the students. Students can focus on specific part of the system (e.g. the forwarding path) while the teaching assistant may need to help with all parts of the system.

The ISI Summer Graduate Research Experience Program (SGREP)[1]

Joseph D. Touch, Amy S. Hughes, Yu-Shun Wang, Lars Eggert
USC/Information Sciences Institute

August 10, 2002

Abstract— **SGREP is a summer research program targeting CS, CE, and EE MS students. The program, currently in its fourth year, unifies the often competing goals of teaching and research. It provides new opportunities for students to work on real research projects and for research faculty to teach.**

1. Introduction

SGREP is a summer internship program for masters-level graduate students, designed to integrate the goals and limitations of funded research with academic credit. Students get an opportunity to participate in real research projects and gain exposure to project planning, management, and administration, in exchange for directed-research class credit. Research faculty get an opportunity to participate in teaching, and to evaluate students for future paid positions on projects. SGREP is specifically designed to address the funding constraints of university research while enabling research faculty to teach and train students in ways that may otherwise present conflicts.

The SGREP program began in the summer of 1999, as an experiment in combining academic project experience for students with real project experience. The objective is to serve the mutual goals of academic graduate students and ISI's project leaders. It is offered to motivated graduate students in CS, CE, and EE at USC, and is completing its fourth year.

SGREP students benefit from participation in small groups with full-time researchers, working on a real project, and helping to plan and coordinate their contributions. ISI's projects benefit by having tangential issues or components implemented, while (we believe) maintaining the focus of the project leaders on their projects.

2. Integrating Teaching and Research

Both research faculty and students are challenged by the competing goals and requirements of funded research and academics. Research faculty often cannot allocate the funds or time to teach, and students, especially the more transient ones (Masters), often do not get the opportunity to participate in long-term funded research projects. Conventional research assistantships are difficult for starting graduate students to acquire, and teaching traditional core curriculum classes would divert project leaders from their research projects.

Conventionally, full-time faculty are required to teach some number of classes each semester, and can “buy out” of some of them using research funding. Research faculty require a “reverse buy-out,” so that teaching funds pay to release a fraction of their allocation to a research contract.

Research faculty are often presented with a dilemma – they like to teach, but are often constrained by their funding. If they teach a core class, funds are available for the “reverse buy-out”, but faculty often cannot free sufficient time to participate. If they teach an elective (*e.g.*, topics) class, some (lesser) "reverse buy-out" is still needed, but funds for such classes are often limited. Research faculty thus often need to find a way to integrate teaching with their project goals.

Conversely, students require an incentive to participate. Paid participation is prohibitive, because summer work typically costs a project more than the equivalent academic-year support. A full-time summer student can cost as much as a nine-month half-time research student (even assuming no summer tuition vs. 12 credit-hours per semester), and will likely contribute much less, *e.g.,* due to fixed learning-curve delays.

Alternately, students can receive academic credit, either for a core or elective course, or for directed research, in exchange for their

[1] Effort partially sponsored by the Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory, Air Force Materiel Command, USAF, under agreements number F30602-98-1-0200 entitled “X-Bone” and number F30602-01-2-0529 entitled “DynaBone”. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Defense Advanced Research Projects Agency (DARPA), the Air Force Research Laboratory, or the U.S. Government.

participation. For the latter, students need further incentive to participate full-time, vs. the substantially lower cost of taking a summer class.

3. Solution

SGREP's solution is to integrate directed research with project goals. Students focus on non-critical portions of existing projects sponsored by DARPA, the NSF, and other funding agencies. They participate both as students, in twice-weekly seminars, and as regular researchers in project meetings. The seminars cover a variety of topics, including project management and research methodology. Below is a brief list of topics.

- ***working in a lab***
  - *keeping a lab notebook*
  - *coordinating shared resources*
  - *establishing lab procedures*
  - *system administration and security*
- ***project planning***
  - *establishing achievable goals*
  - *handling change and roadblocks*
  - *personnel skills*
  - *budgets and estimation*
- ***communication skills***
  - *how to read, write, and review*
  - *interactive presentations and demos*
- ***conducting research***
  - *experimental method*
  - *data analysis methods*

SGREP is held during the summer because this is the most productive time for many ISI researchers, who, unlike their purely academic counterparts, do not break for the summer. Summer is also the time when students are free from competing academic pursuits and obligations, and can take advantage of intensive full-time experience.

Management of the SGREP program involves extensive review of student applicants, as well as detailed procedures for potential project advisors. The program is managed by research faculty, but includes non-faculty staff as well as senior graduate students as advisors, providing the latter a unique opportunity to participate in project management and advising.

In addition to the seminars, students are expected to attend weekly SGREP meetings as well as project meetings, and to work at ISI at least four days a week. Their programming skills are closely reviewed, because the compressed summer schedule limits the practical learning curve. Project components are selected which have immediate returns, but are not in the critical path. More successful projects allow improvisation and redirection, subject to the student's interest.

In past years, SGREP projects have included adding optional features, examining research issues, and porting code to different systems. In some cases, code developed was included in public releases. In other cases, SGREP students were invited to continue during the academic year as paid students.

4. Related Programs

There are a number of industrial programs on which SGREP is based. The primary of these is GTE Labs' Industrial Undergraduate Research Participation program (IURP), which one of the authors was a participant for several years. This program interviewed applicants from an open call, and placed them in projects which were specifically developed for summer work. The environment provided an opportunity where undergraduates could participate in industrial research, focusing on tangential or highly compartmentalized components of real projects.

Similar programs have run at IBM, Xerox PARC, AT&T, and Bell Communications Research (now Telcordia), as well as at various universities, and a few research institutes (*e.g.,* Cold Spring Harbor). Each program has (or had) a slightly different nature than SGREP, such as focusing on undergraduates, unspecified assignment to a general project, or lacking a comprehensive seminar program. Many of these programs focus on industrial participation or university training for industrial employment. These programs, including IURP, were/are also larger than SGREP, which provides opportunities for only a handful of students at this time.

5. References

Information on SGREP is available at http://www.isi.edu/touch/sgrep

Virtual Internets for Lab and Class Experiments[1]

Joseph D. Touch, Yu-Shun Wang, Lars Eggert
USC/Information Sciences Institute
August 10, 2002

Abstract— **The X-Bone is a system for deploying and managing virtual Internets (VIs). VIs, sometimes referred to as overlay networks or VPNs, are used for testbeds, demos, and lab experiments, to provide a virtual topology on which to examine routing protocols and distributed applications. Current use of shared lab and testbed resources often requires explicit reservation in fixed time slots; the X-Bone automates resource reservation, and also supports concurrent shared resource use.**

1. Introduction

Virtual Internets enable concurrent shared use of network resources for class and lab experiments [10]. The X-Bone system is designed to support such Virtual Internets, providing virtual IP-based networks on FreeBSD and Linux OSes, and is currently available [9][12].

Most current network testbeds, whether restricted to a single lab or distributed across departments or organizations, are used for experiments using out-of-band reservation systems that hearken back to 1960's OS batch scheduling. Specific machines are reserved for fixed blocks of time, and users are often expected to restore "safe" configurations. This type of sharing is inefficient and costly, in terms of equipment, lab space, and user efficiency.

Such dedicated reservation of resources is required where experiments require OS modification. More recent techniques, such as divert sockets in FreeBSD, and loadable kernel modules, allow application-layer implementation of network experiments. Other uses require deployed network topologies, without deploying new protocols – such as testing routing configurations, or developing distributed applications.

The X-Bone is a system for automatic deployment and management of Virtual Internets - also known as IP overlays. It requires no new protocols, and works with existing applications and operating systems. USC/ISI is currently developing the X-Bone research prototype, already available as a FreeBSD port (/usr/ports/net/xbone) and Linux RPM, into a tool for education and research. It can be used to deploy networks, and also to coordinate and deploy applications on those networks, which can be useful for distributed system experiments [9].

2. Concurrent shared use

The X-Bone deploys and manages configurations of FreeBSD and Linux hosts and host-based routers. These overlays use IP encapsulation, and are achieved by the careful configuration of virtual interfaces and routing table entries. The result supports concurrent shared experiments and applications.

In conventional VPNs, a host is a member of only one VPN at a time. The VPN connects that host to a preexisting secure network. By contrast, the X-Bone supports multiple, concurrent overlay networks. Each network is deployed as a whole, and both hosts and routers can be members of multiple overlays at once.

This version of concurrent overlays is a network equivalent of Virtual Memory, we call Virtual Internets (VIs) [10]. Like their VM counterparts, VIs provide protection and abstraction. Protection prevents traffic from one overlay from being seen on other overlays, and is achieved by per-hop IPsec encryption, as well as partitioned forwarding at routers. Abstraction allows users and applications to view the network as a simple, convenient topology (*e.g.,* a ring), regardless of the actual connectivity.

Protection enables tests of new protocols without affecting the rest of the Internet, or other VIs. It also provides VPN-like privacy, notably securing network management (*e.g.,* routing protocols, monitoring).

[1] Effort sponsored by the Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory, Air Force Materiel Command, USAF, under agreements number F30602-98-1-0200 entitled "X-Bone" and number F30602-01-2-0529 entitled "DynaBone". The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Defense Advanced Research Projects Agency (DARPA), the Air Force Research Laboratory, or the U.S. Government.

Abstraction allows applications to use the network in a "do what I mean" fashion. Current distributed applications often embed network awareness, *e.g.,* neighbor discovery and organization into trees or rings. VIs allow this capability to be offloaded, much as VM offloads page management from programmers. In one recent example, this abstraction was used to support geographic network overlays in ways that would be impossible in the base network [6].

3. Virtual Testbed and Lab Infrastructure

VIs support distributed virtual testbeds, as well as increased shared use of lab facilities. Because the X-Bone supports recursive VIs, a virtual testbed network can be created and individual experiments deployed in that testbed.

A variety of distributed testbeds have been developed or are currently being developed. One of the earliest was DartNet, a "testbed you can break."[2], which evolved into CAIRN [3]. DartNet and CAIRN were composed of dedicated links connected to dedicated routers, which could be arbitrarily reprogrammed. This infrastructure was useful, but very costly – the links consuming the majority of the expense.

One of the more significant used of DartNet was to develop multicast IP [5]. Multicast IP was deployed there by modifying all the routers, but this limited its reach to only a set of contiguous (connected) routers. To overcome this constraint, and enable more dispersed incremental deployment, first source routing, then later IP encapsulation tunnels were used. This is the first example of an overlay network.

Similar overlays have been used to deploy other network protocols, *e.g.*, IPv6 and Active Nets [1][2]. Application-layer tunnels (UDP) have been used to deploy peer networks in a similar fashion. VIs are a generalization of this architectural extension, which enables more widespread experiments in protocol and network architecture, without (contrasted to peer nets) recapitulating network capabilities not under test. [10] Other examples of emerging overlay infrastructure include the Grid and PlanetLab; both build on the VI capability of the X-Bone, providing resource location and process management [7][8]. Emulab provides similar capabilities in a directly-connected environment [4].

The X-Bone's VIs can also be used to support shared use of lab equipment. Experiments that do not require exclusive use, such as router configuration experiments and tests of routing protocols, can utilize the partitioning of VIs to allow concurrent experiments.

Where exclusive use is required, the X-Bone's access control capability can limit the number of concurrent users to 1, effectively isolating performance-based experiments. In this latter case, exclusive bandwidth interconnectivity is provided by a local Ethernet switch. The result limits each component to a single user at a time, but abstracts the user from explicit resource allocation and management.

The X-Bone is currently being extended for testbed and educational lab use under a grant from the NSF. Of particular interest is an opportunity to inform the educational community of its capabilities, and to obtain feedback on how to further enhance its utility for these communities.

4. References

Information on the X-Bone is available at http://www.isi.edu/xbone

[1] 6-Bone – http://www.6bone.net

[2] A-Bone – http://www.isi.edu/abone

[3] CAIRN – http://www.cairn.net

[4] Emulab – http://www.emulab.net

[5] Eriksson, H., "MBone: The Multicast Backbone," Communications of the ACM, Aug. 1994, pp.54-60.

[6] Finn, G., Touch, J., "Network Construction and Routing in Geographic Overlays," (submitted).

[7] Grid – http://www.globus.org

[8] PlanetLab – http://www.planet-lab.org

[9] Touch, J., "Dynamic Internet Overlay Deployment and Management Using the X-Bone," Computer Networks, Jul. 2001, pp. 117-135.

[10] Touch, J., Eggert, L., Wang, Y., "Virtual Internets," (submitted), also ISI tech report ISI-TR-558, July 2002.

[11] Wang, Y., Touch, J., "Application Deployment in Virtual Networks Using the X-Bone," Proc. DARPA Active Nets Conference and Exposition (DANCE), May 2002, pp. 484-493.

[12] X-Bone URL – http://www.isi.edu/xbone

[2] Excluding the ARPAnet, as it was not intended for ongoing experiments in network or transport protocols.