An Undergraduate Networked Systems Laboratory

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

Ralph Droms
Cisco Systems

Abstract

This paper 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 two departments have been pioneers in developing laboratory exercises where students discover, design, explore and experiment with the latest concepts in their fields on state-of-the-art equipment. The laboratory facilities accommodate study of several computer networking hardware and software technologies, computer systems and organization.

Introduction

The Networked Systems Laboratory (NSL) provides new opportunities for undergraduate instruction and supports faculty and undergraduate research in parallel computing [1,2], distributed computing [3], and computer network systems [4,5]. 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.

For example, in systems-oriented courses such as Operating Systems (CSCI 315), students should have the opportunity to modify the system software and/or the hardware. In particular, in an operating systems course, the faculty member may ask the students to design an I/O driver and compile the new driver with the kernel. In order not to interfere with the rest of the campus, any machines used for such under-the-hood activities should be isolated from the rest of the campus networks, for example, by a firewall router.

The Networked Systems Laboratory will be used in the Computer Science Department for teaching the Operating Systems course (CSCI 315), Networks course (CSCI 363), Parallel Computation (CSCI 366), Distributed Computing (CSCI 355) as well as the Senior Design course (CSCI 475). The Electrical Engineering Department will use the facility in the Senior Design course, Computer Communication Networks (ELEC 475/675) and Digital and Analog Communication Systems (ELEC 470/670).

"Proceedings of the 2002 American Society for Engineering Education Annual Conference & Exposition

In addition, faculty from all engineering departments will have the opportunity to utilize the Networked Systems Laboratory. For example, faculty members teaching the Computer Architecture course (CSCI 320) might use the lab students to give students the experience of measuring the performance of a system with modified hardware, such as less cache memory.

Networking Systems Laboratory Equipment

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. The 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

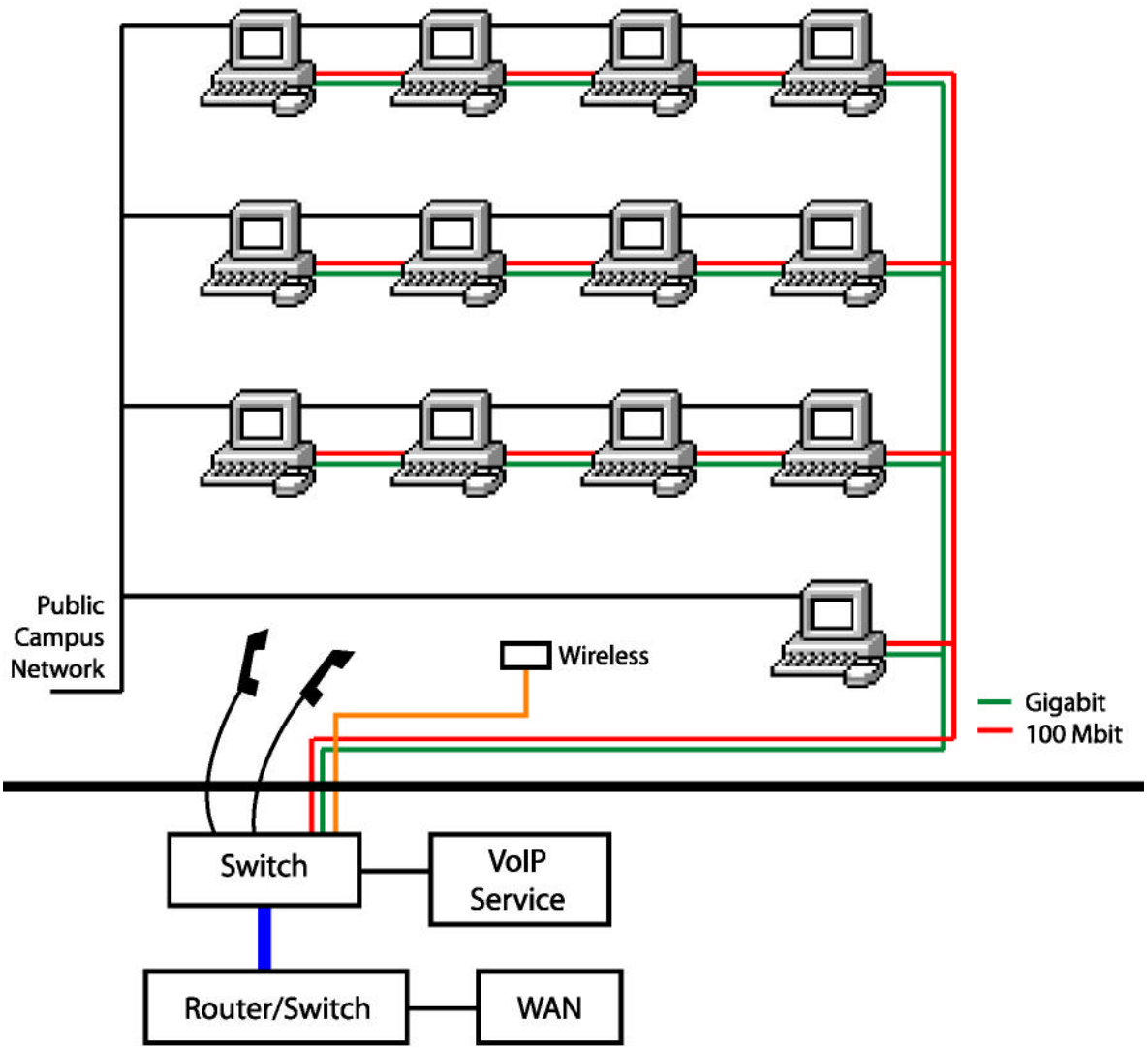

Figure 1: Network System Laboratory Configuration.

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.

The Present

•Computer Networking: We have been developing a user's guide for the laboratory as well as some demonstrations. Appendix A includes a sample list of laboratory exercises that have been used in our networking course and adapted to the NSL.

•IP Telephony [6,7]: We demonstrated the use of Voice over IP (VoIP) with the equipment in the NSL by using the existing Internet service (IP) to provide regular telephone service. With appropriate software installed, a computer on the Internet can act as a back-end for a telephone. The only extra hardware equipment needed is an IP phone. When it is connected to a PC running IP telephony software, the IP phone can act as a regular telephone. The peer can be another IP phone or a regular telephone. We have obtained good quality communication in the lab. We could not tell the difference between the regular telephone service and the service provided by IP phone. VoIP allows students:

–to test Quality of Service
–to test jitter
–to understand multicast technologies
–to combine video, phone, and data to a single device
–to become familiar with state of the art network technologies

Parallel Programming

•Mandelbrot Plots: Many problems require the compute power of more than one machine. One such problem is the computing of Mandlebrot plots. As a demonstration, we used Message Passing Interface (MPI) [8] middleware to coordinate and communicate between C programs on each of the machines to solve Mandelbrot plots. The results from each machine were passed to a single machine to display the plot.

–compute intensive problem
–utilizes the compute power of many machines
–uses network to communicate
–uses MPI middleware

The Future

CSCI 315 (Operating Systems): Students of CSCI 315 will use the NSL facility to modify OS kernel such as the Red Hat Linux 7.2 that is already in the lab. They will also study the performance of the operating systems by modifying the code and then collecting and comparing the statistics such as queue length of CPU and turn-around time of the jobs. They can also study the impact of memory management policy on the performance. Without an isolated lab environment such as NSL, students can only study these issues in a simulated environment because if they wouldn't be able to change the code of a production operating system. They need to worry about the impact and the potential side-effect on the people who will use the same system for other purposes. Thus most OS courses offer students a simulated environment where they can experiment with the system on top of a piece of software that simulates the behavior of a true operating system. Though useful, a simulated environment is always a compromise from the actual environment such as the one provided by NSL.

CSCI 363 (Computer Networks): Students of CSCI 363 will benefit greatly from the NSL. They will be able to do many different projects with NSL that they can't otherwise do in regular environment. For example, they can actually monitor the network traffic using software such as *snoop*. In a typical shared network environment, students can't have access to system software such as *snoop* because of the security concern. Snoop can trace and interpret the network traffic at the packet level, enabling the user of the snoop to watch live traffic on the network and potentially figure out exactly what is going on in the network traffic. In an open, shared lab such capability poses serious privacy and security concern. Thus almost all system administrators turn this function off from ordinary users. With an isolated lab facility such as NSL this type of project becomes feasible. Students can benefit greatly from being able to actually implement such projects. They can learn insight into low level network characteristics that otherwise is impossible to learn.

Students in CSCI 363 can also easily compare different features and performance of different network protocols. NSL is equipped with three different types of network equipment, a wireless LAN, a gigabit Ethernet, and a 100 M bps Fast Ethernet. Students can easily conduct performance studies over the three different types of network, thus giving them invaluable experiences with these modern network protocols.

Students in ELEC 475 and CSCI 363 will investigate the following networking issues and computing issues using the network technologies and the multiple, identical computers in the NSL:

1. User identity and location management: The problem of determining who is "logged in" and location will be more complex in the future as people use mobile computers, net phones, wireless personal digital assistants (PDAs), net devices, and walk-up computing within an

organization. Not all of these devices will have existing "user login" mechanisms like Windows operating systems.

2. Quality of Service (QoS) /policy networking: Networked applications require different delivery services for different types of data. Quality of Service is highly dependent upon the information that is being transmitted by the network. Certain information will require exact transmission, but not speed, e. g., bank account balances. Some transmissions can survive small omissions of information, but need to be sent without delay, e. g., audio or video messages. Still other transmissions require both speed and completeness, or neither. As a result, there needs to be a policy in place to recognize and accommodate each situation in a different way (differentiated service). Quality of Service describes the types of delivery service and the ways in which the underlying network hardware provides those services. It is managed by network policies, which control the way individual network components forward network data and meet requirements for reliability or delay.

3. Information model/directory integration: As applications and services in networks become more tightly integrated, they require an underlying framework for exchange of information. For example, in a situation where a user may access the network from multiple locations, the user's current login information must be coordinated with the network's policies for QoS and service access. Information is often exchanged within a network through a directory, as provided by Lightweight Directory Access Protocol (LDAP) or Active Directory. Methods and standards for information organization and integration of directory services into network applications need to be developed.

4. Internet Protocol version 6 (IPv6) integration: Internet Protocol version 4 is a way of communicating between computers and is approaching the end of its run due to a lack of addressing capability and inflexibility of controlling quality of service. A new protocol, version 6, will step up to fill the gap caused by the exponential growth of the Internet. For computer networks to work with old and new machines, version 6 needs to be integrated into existing networks. Ipv6 also will address security, and include QoS capabilities.

5. Wireless LANs and networking: This technology for linking computers is useful in limited geographical areas, such as homes, buildings or campuses. Algorithms for access to shared resources and the speed of communication are burgeoning fields with many possibilities. Measures of performance and testing the capabilities of standard (IEEE 802.11) wireless network interface cards (NICs) are of interest. Wireless networks are an important research area to facilitate ubiquitous computing and access to the Internet.

6. Benchmarking cluster and parallel computers: Cluster computers are a network of multiple low-end computers (PCs). They provide an efficient and economical way to solve large-scale scientific problems compared to high-end super computers. Another approach to increasing the speed of computing is the use of parallel computers that are composed of multiple processors connected in various ways. Benchmarking measures the execution speed of application programs and determines which connection strategy and networking protocols are best suited for a class of problems. Benchmarking computer networks is a related topic

since the speed of the Internet is also dependent on the availability of fast computers or parallel computing.

7. Experimental Web search and E-Commerce applications: Information on the Web is abundant. How to effectively find the information the user wants is a challenging task. A machine learning approach can be used to aid the Web search, where the user acts as the teacher and intelligent Web search engine is a learner. After the user issues a search request, the intelligent Web search engine finds an initial collection of Web pages for the user. The user can then interactively refine the search query with the intelligent Web search engine until a satisfying collection of documents is found. To conduct research in building and studying performance of this type of search engines, an isolated network of computers is highly desirable. The NSL will provide an excellent testbed for such projects.

All of the above areas must focus on meeting the users' demand for critical characteristics of speed, reliability, and security of communications.

Summary

This paper has described a new undergraduate network systems laboratory to enable students to experiment "under the hood" of computer network systems. The laboratory hardware and software allows students to experiment with and explore emerging networking issues.

Acknowledgement

The Network Systems Laboratory was made possible through funding from The Verizon Foundation, The Air Products Foundation and The Gladys Brooks Foundation. Additional funding was provided by J. Randall and Kathleen MacDonald.

References

1. Barry Wilkinson and Michael Allen, *Parallel Programming -- Techniques and Applications Using Networked Workstations and Parallel Computers*, Prentice Hall, 1999.
2. Ian Foster, *Design and Building Parallel Programs*, Addison Wesley Publishing, 1995.
3. Andrew S. Tanenbaum and Maarten van Steen, *Distributed Systems: Principles and Paradigms*, Prentice Hall, 2002.
4. Douglas E. Comer, *Computer Networks and Internets*, Second Edition, by CD-ROM by Ralph Droms, Prentice-Hall, 1999
5. Alberto Leon-Garcia and Indra Widjaja**,** *Communication Networks***,** McGraw-Hill, 2000
6. David J. Wright, *Voice Over Packet Networks* John Wiley & Sons, 2001.
7. Jonathan Davidson, Jim Peters, *Voice Over IP Fundamentals*, Macmillan, 1999.
8. Peter Pacheco, *Parallel Programming with MPI*, Morgan Kaufman Publishers, 1996.

Biographical

MAURICE F. ABURDENE is the T. Jefferson Miers Professor of Electrical Engineering and Professor of Computer Science at Bucknell University. He has taught at Swarthmore College, the State University of New York at Oswego, and the University of Connecticut. His research areas include, parallel algorithms, simulation of dynamic systems, distributed algorithms, computer communication networks, control systems, computer-assisted laboratories, and signal processing.

XIANNONG MENG is an Associate Professor in the Department of Computer Science at Bucknell University in Lewisburg, Pennsylvania, U.S.A. His research interests include distributed computing, data mining, intelligent Web search, operating systems and computer networks. He received his Ph.D. in computer science from Worcester Polytechnic Institute in Worcester, Massachusetts, U.S.A.

DANIEL C. HYDE is an Associate Professor in the Department of Computer Science at Bucknell University, Lewisburg, Pennsylvania, USA. His research interests include parallel and distributed computing. Dan received the BSEE degree from Northeastern University and the PhD degree in computer science at the University of Illinois, Urbana-Champaign.

RALPH DROMS is a Technical Leader with Cisco Systems in Chelmsford, MA. He designs internet protocols and applications for network device configuration and management. Ralph is chair of the Dynamic Host Configuration working group of the IETF and author of the DHCP RFCs. Prior to joining Cisco in 2000, Ralph was a faculty member of the Department of Computer Science at Bucknell University. He received the PhD degree in computer science from Purdue University.

Appendix A

A Sample of Current Networking Laboratory Exercises

Lab 1 Introduction to LINUX System Calls: In this lab we introduce the concept of system calls in LINUX to the students. Although technically a part of the operating system concepts, students will use LINUX system extensively in network programming. It is good to review these concepts and practices at the beginning.

Lab 2 Simple Client/Server Programming: In this lab students work from a given set of client/server programs from the textbook (a simple Web server and its client), create a set of their own client/server programs. Students will write a *whois* server and *whois* client. The client sends a user ID to the server, the server responds with the user's full name if the user is a registered LINUX user.

Lab 3 Thread Programming: In this lab students learn to use the Pthread (POSIX thread) package to create a multi-threaded program. The intention is to get students familiar with programming in thread so they can create multi-threaded server later on in the semester.

Lab 4 Ethernet Frame Decoding: Students are given a set of a previously generated data from Ethernet packet tracing through *Snoop* on UNIX and they need to write a program to analyze the trace. Students will understand better the Ethernet packet format and how to analyze the trace data. In a public laboratory, students typically are not allowed to directly monitor the network traffic because of the security concern. With a closed and private network such as the one in NSL, this type of laboratory exercises are particularly interesting.

Lab 5 Multi-threaded Server: Using the knowledge they acquire from Laboratory 3 (Threaded Programming), students are to write a multi-thread *Chat* server that can talk to multiple clients simultaneously. A basic version of *Chat* server/client is given, students need to re-write them such that the server now is capable of handling multiple clients at the same time.

Lab 6 Internet Address Manipulation: Continue to work with the Ethernet packet tracing data started from Lab 4. In this lab, students are to write a program to extract the source and destination IP addresses, their hostname, and aliases, if any, from the trace data.

Lab 7 Socket Programming -- Client: Students are to work with socket library in this lab to create a client program. The lab exercise uses UDP interface to get time of the day from a remote server.

Lab 8 Socket Programming -- Server: Students are asked to write a UDP server in this lab to record the number of visits to a particular server. The data on the server side is persistent. Every time a client contacts the server, the server increases the visit count by one and returns it to the client. The students learn how to extract peer's information from UDP communication.

Lab 9 CGI Programming: Students learn the basics of CGI (Common Gateway Interface) programming in this lab. CGI is an essential part of modern Web servers. The intention is that hopefully students will be able to use it in their semester projects.

Lab 10 Project 1 (Service Finder): The next three laboratory exercises are designed to help students to get familiar with the concepts needed for their semester project. In this lab, students will learn how a service program can register its service with a service registration program, and how a client can contact this service registration program to find what are the services available over the network.

Lab 11 Project 2 (Registration, Password Checking): This lab is also a part of the required component in the semester project. Students will learn how to encrypt a password in LINUX system, how to manipulate the output device such that the echoing can be turned off while the password is typed from the keyboard. The goal is to send encrypted password to the remote server and to have the server to check if the password is valid.

Lab 12 Directory Maintenance: Students learn how to pass a list of variable size structure over the network. They also get an understanding of the issue of marshalling parameters --- pointers in C/C++ program don't mean the same thing on different runs of the program on different computers.

Lab 13 RPC: Although sockets provide a fundamental way of communicating among computers on the Internet, RPC (Remote Procedure Calls) provides an alternative, simple mechanism to write network applications. Students learn the basics of RPC in this lab. They write a set of RPC service and client programs in this exercise