University Methodology for Internetworking Principles and Design Projects

Randal T. Abler, *Member, IEEE*, Henry L. Owen, *Member, IEEE*, and George F. Riley, *Member, IEEE*

***Abstract*—An undergraduate engineering internetworking learning environment that presents both internetworking principles and laboratory experimentation is described. The learning environment uses the source code availability of the Linux operating system as a case study of the implementation issues and ramifications of internet networking infrastructures. Laboratory use of experimentation with internetworking equipment and software allows interaction with internetworking principles and fundamentals as well as implementation and performance issues. The objectives of this environment include providing a comprehensive mechanism whereby students are exposed to fundamentals and principles that may readily be applied to experimental-based internetwork research and internetwork product development. A follow-on capstone design environment is also briefly described.**

***Index Terms*—Capstone design class, design projects, internet infrastructure experimentation, internetworking laboratory, undergraduate networking.**

I. Introduction

DURING a recent industry-sponsored research project, the authors were required to implement a prototype router with new nonstandard functionality. Application of internetworking principles in the new Linux-based router prototype introduced many interesting design issues. The authors found it challenging to find students with both a theoretical and an experimental background in internetworking. It was relatively easy to find students with theoretical knowledge of networking protocols and routing functions. It was virtually impossible to find students with the implementation skills necessary to experimentally implement new ideas and then test them out. As a result, the authors discovered that there was a great need for an educational environment for teaching fundamental internetworking so that the end result was not only an understanding of internetworking theory, but also an ability to be able to implement new experimental-based internetworking infrastructures.

This paper presents an approach that may be used to complement students' theoretical networking knowledge with an experimental internetworking environment. The approach puts internetworking fundamental concepts and the application of this material in the same learning environment. The main enabler of this approach is the availability of source code from the Linux operating system so that it may be understood well enough to use as a platform for implementing new and different internetworking functionality. This methodology has enabled a follow-on senior capstone design experience where participants are able to apply their internetworking knowledge in creative internet-related design projects.

Manuscript received February 26, 2002; revised June 24, 2002.

The authors are with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0250 USA (e-mail: henry.owen@ece.gatech.edu).

Digital Object Identifier 10.1109/TE.2002.808239

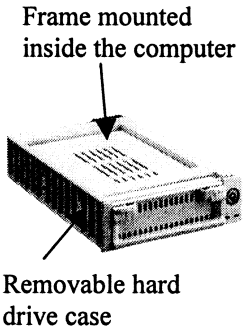

Fig. 1. Removable hard drive frame and hard drive case.

The specific objectives of this class include:

1) to develop further a student's understanding of computer communications principles;
2) to develop a student's ability to propose and evaluate alternative approaches to meeting specific communication requirements;
3) to give the student knowledge of how internet protocols are implemented in Internet infrastructure, including computer-operating systems and dedicated embedded hardware;
4) to provide the student knowledge of computer communications standards in terms of the standards current status and future direction;
5) to develop a student's ability to formulate a computer communications problem, analyze it, and then communicate the results of his/her work in written and graphical form.

These objectives included the ability to demonstrate through experimental implementations, internetworking principles, and practices including sockets programming, networking hardware and software configuration, protocols and performance issues, and operating system implementations of internetworking.

II. Laboratory Environment

The key component to this approach is an internetworking laboratory in which each participant is assigned his or her own computer hard drive. Each of the laboratory computers has a removable hard drive frame as shown in Fig. 1. Participants insert their assigned hard drive into any available laboratory set up that

brings them back to exactly where they were the last time they were in the laboratory.

A. Laboratory Equipment Required

The laboratory enables experimentation with the principles covered in the classroom instructional component of this learning environment. This laboratory equipment consists of a supply of hard drives and removable hard drive frames, personal computers (600 MHz Pentium III-based systems with 128 MB of RAM), a numerous supply of various Ethernet networking interface cards, such as the Intel EEPRO100, Ethernet hubs, Ethernet switches, and several routers (such as Cisco1700, Cisco 2600, Cisco 3000, Cisco 4000, Enterasys SmartSwitch router), a Spirent Smartbits 2000 hardware network traffic generator, a Domino Plus DA-360 Internetwork Analyzer from Wandel and Goltermann, and a Corporate Systems Center Pro Drive workstation hardware-based hard drive copier. The hardware failure rate associated with issuing hard drives to each student and having him/her transport them has been extremely low. Experience proves that the fragility of the hard drives is not an issue. The drives are issued to the students at the beginning of a semester and remain with them. A better system would be to have a securable storage area in the laboratory accessible only by the individual users of each of the hard drives. Participants work individually on most laboratory activities; however, because of equipment limitations, groups are established for some activities.

The operating system used in the laboratory is Linux. The reason for this choice is that the source code for Linux, in general, and the networking protocol stack, in particular, are readily available. This system allows detailed examination of one implementation of internetworking. In this case the authors have chosen the Red Hat distribution [1]. The selection of Linux as opposed to other operating systems, such as FreeBSD, was a result of having used both Linux and FreeBSD in various research projects where experimental protocol extensions were implemented. The Linux community appeared to be moving at a much faster pace than the FreeBSD community. Additionally, commercial bookstores have whole shelves full of Linux books, although those same stores have only a few books on other operating systems such as FreeBSD. Since the large number of readily available reference sources for Linux is an advantage for students, Linux remained as the operating system of choice. There is no reason why a similar class could not use FreeBSD or, for that matter, other distributions of Linux other than Red Hat.

The contents of an entire hard drive including the Linux operating system may be copied from one hard drive to another using a hardware-based hard-drive copier. Software-based copiers are not able to copy the later versions of Linux operating systems because of the Linux file systems.

B. Initial Laboratory Activities

Laboratory activities start with the individual installation of the operating system and configuration of the networking components, including internet protocol (IP) addresses, default networking gateway, domain name server, broadcast address, network address, etc. Each participant is assigned a static private IP address. Static IP addresses are used so that in later networking experiments, the IP addresses of the various machines are readily known.

C. Sockets Programming of Traffic Generators

After installation and configuration of the Linux operating system, the C programming language is used to write programs that use the "sockets application programming interface" to implement first a transmission-control protocol (TCP) and then later a user-datagram protocol (UDP) software traffic generator and consumer. C is chosen as the programming language because it is, by far, the most common language used to implement Internet infrastructure. Additionally, the Linux operating system is implemented in the C programming language. The software traffic programs are written with the server as the traffic consumer and the client as the traffic generator. The programs allow control of the data rate, mean and variance of the packet size, and control of the interval between packets, destination IP address, and destination port. The traffic generator and sink keep statistics on both the transmitted and received traffic. These traffic generators are used throughout later laboratory activities. These programs are used instead of the similar public domain program "ttcp" [2] because the student insight obtained from writing the C programs to accomplish traffic generation and measurement has been found to be very valuable. The resulting network "protocol states," as well as the individual protocol headers and data generated for each line of C code, is examined. This examination results in a detailed understanding of exactly what is going on in the network as the result of the student's traffic programs.

Packet sniffers are used to examine the traffic between computers. This configuration allows examination of the traffic created by the TCP and UDP generators and examination of address resolution protocol (ARP) and ping activity. Additionally, it is instructive to examine the clear text nature of passwords when using applications such as telnet and file transfer protocol (FTP). After also using secure shell (ssh) and secure copy (scp), the encrypted nature of those passwords may be examined. An example of a Linux-available, software-packet sniffer is ethereal. An example of a hardware-based packet-sniffer is the Wandel and Goltermann Domino Plus DA-360 Internetwork analyzer. Doing full packet decodes with sniffers provides a level of understanding about Internet traffic that is difficult to obtain without hands-on experimentation [3].

D. Laboratory Experimentation With Router Performance

Placing multiple Ethernet network interface cards into a Linux-based machine allows one to configure the machine to act as a router. This addition allows student experimentation with routing concepts. Static configuration of the routing table as a part of a laboratory exercise allows insight into the routing process. When IP forwarding concepts have been examined, the performance of this PC-based router may be examined by using a traffic generation and measurement scenario. With the availability of a hardware traffic generator, such as a Spirent Smart Bits 2000, one may, in an automated fashion, generate known traffic volumes at various rates and measure the performance of the PC's routing and forwarding capability. An example laboratory configuration that may be used for this

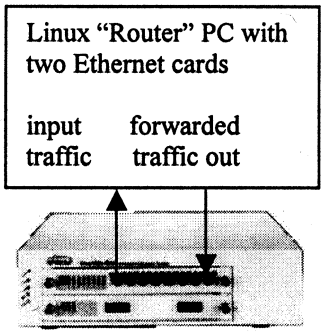

Fig. 2. PC-based linux router performance evaluations.

purpose is shown in Fig. 2. Varying packet size to demonstrate that the limiting factor is the interrupt rate and not the data rate is very instructional. Motivating how much work in terms of how many software instructions that must be executed prior to the arrival of the next packet leads into the examination of how many CPU cycles are involved in the implementation of a real Internet protocol stack. Examination of PC router performance may be done initially by using Ethernet network interface cards that are ISA bus-based so as to show the limiting factor of the slow ISA bus to around 8 Mb/s (using for example NE2000-based, 10 Mb/s network interface cards). Recognition that it is not possible to obtain the rated speed of the network interface card because of PC bus architecture is enlightening to some students. Network Interface cards in a PCI-based machine demonstrate the higher bus speed capabilities. Experiments with 100 Mb/s (using for example Intel EEPRO100 network interface cards) and gigabit network interface cards (such as the Intel Pro/1000F optical network interface cards) are used to show gigabit performance limitations. The concept of reducing interrupts through interrupt mitigation may be examined by using Linux gigabit network-interface card drivers, such as the one that may be found as a part of the click modular router project [4].

E. Laboratory Examination of Routing Protocols

Examination of how routers exchange routing table information is carried out by experimenting with routing information protocol (RIP) and then the routing protocol open shortest path first (OSPF) in the network topologies shown in Figs. 3 and 4.[1] By administratively bringing down a link and then later bringing it back up, one may examine the verbose output from the routing protocols (or use a packet sniffer) to watch these protocols in action as they distribute the routing information throughout the network. Observation and analysis of the routing protocols in action yield a level of understanding that is difficult to obtain through theoretical examination alone. The routing protocol information and packet formats may be observed and correlated to the theory of how the routing protocol works.

It is not necessary to use a diverse set of routers in terms of model numbers and manufacturers. In fact, there exist Linux implementations of RIP and OSPF routing protocols so that in theory one does not even need a real router. The use of this diverse group of routers is motivated by two main reasons. First, since this is a "hands-on" oriented class, the instructors feel it is beneficial to expose students to more, as opposed to fewer, types of actual internet, infrastructure equipment. Since the instructors provide the instructions to students on what commands to use on which routers and since this is not a command-oriented class but instead a principles and practices class, the instructors find no problems with using several different makes and models of routers simultaneously. The second reason for using this diverse set of routers is that this equipment is what was available. Recently, the class has been taught using ten routers of identical make and model (Cisco 1700). However, the instructors do

[1]The routers used in these experiments are Cisco 1700, Cisco 3000, Cisco 4000, Cisco7000, and Enterasys SmartSwitch routers.

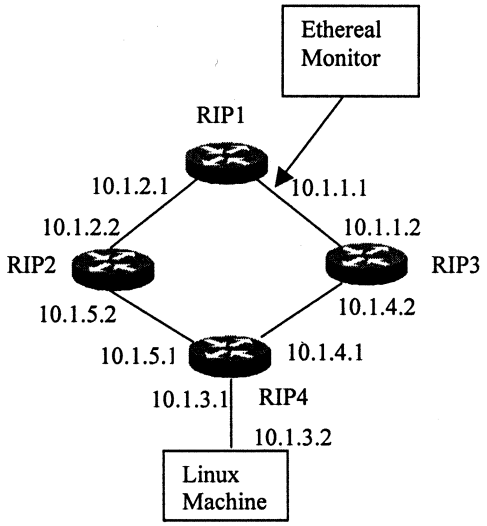

Fig. 3. RIP network configuration.

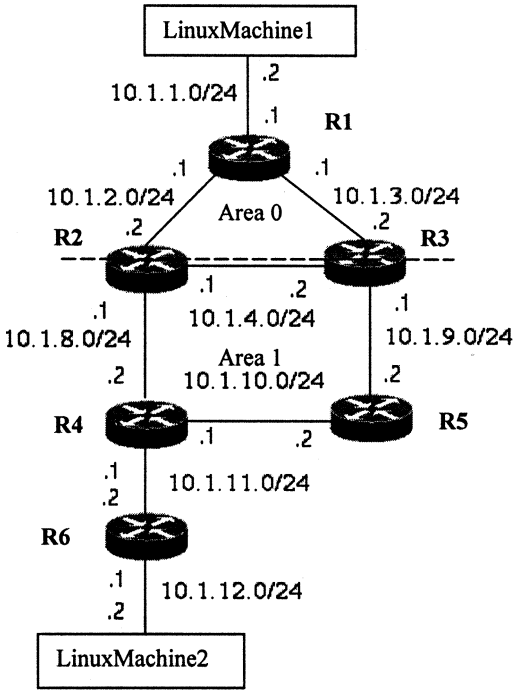

Fig. 4. OSPF network configuration.

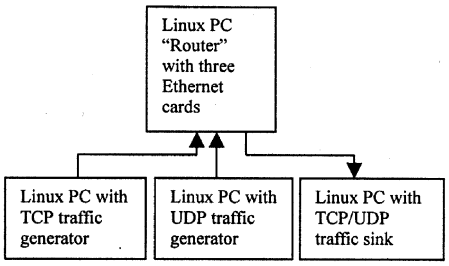

Fig. 5. Modified protocol stack performance measurement configuration.

not believe there is an advantage except from the maintenance and teaching-assistant "training overhead" reduction. In summary, the actual make and model of the routing equipment is not significant except from the standpoint of students feeling more confident from having worked with several different makes and models of routers and having seen multiple vendor equipment internetworking correctly. The key characteristic of the routers is that they have an operating system (for example, Cisco IOS) that allows configuration and troubleshooting. A pure Linux machine environment running RIP and then OSPF routing protocols would be sufficient if actual routers were not available.

F. Experimentation With Changing the Networking Implementation

One portion of the laboratory component of this learning environment involves modifying and recompiling the network protocol stack so as to change the behavior of the Internet implementation. One good experimental application of internetworking principles is to change the priority of TCP data with respect to UDP data. With TCP flow control, the Internet characteristically reduces the amount of TCP data injected into the network when there is congestion. UDP on the other hand, continues transmission at its data rate oblivious to congestion in the Internet. It is instructive to characterize the effects of TCP and UDP in the same network by using the TCP and UDP traffic generators written earlier. Fig. 5 shows an example configuration where routing and the interaction of these protocols may be observed. After running TCP and UDP traffic at various simultaneous injection rates, it is possible to observe that UDP can in effect choke off TCP traffic at a bottleneck, such as a Linux PC-based router. To attempt to solve this problem, one can go into the Linux network protocol software and change its operation so that TCP data is given first priority. Then any time there is UDP data, the UDP data will be forwarded only if there is no TCP data desiring output bandwidth. The performance of the network is completely changed. After making this modification, one can use the software traffic generators written earlier in this learning environment to characterize this new behavior. One can observe that TCP flows now have the ability to completely choke off UDP flows. This situation is, of course, equally undesirable. As a final experimental examination, one can again modify the Linux network protocol source code to include a class-based queuing methodology to demonstrate how "fairness" may finally be achieved in an internetworking environment that has more than the basic functionality typically found in the present Internet infrastructure. Class-based queuing capability is included in the distribution of the Linux kernel. Table I shows example laboratory activities for a typical semester long environment. Examples of laboratory assignments may be found in [5].

TABLE I
EXAMPLE LABORATORY ACTIVITIES

Lab 1	Linux Operating System Installation and Network Configuration
Lab 2	Write a TCP traffic server and client C program
Lab 3	Write a UDP traffic server and client C program
Lab 4	Use a packet sniffer to examine passwords, file transfers, network traffic
Lab 5	PC based Linux router traffic forwarding performance characterization
Lab 6	Configuration of a Linux Router and TCP/UDP traffic interaction
Lab 7	Routing Information Protocol Examination
Lab 8	Open Shortest Path First Routing Protocol examination
Lab 9	Modify Linux Operating System to always give priority to TCP traffic
Lab 10	Modify Linux Kernel to use Class Based Queuing

III. CLASSROOM INSTRUCTIONAL ENVIRONMENT

The prerequisites for this class are a C programming class and sometimes a previous networking survey class at the level of, for example [6]. However, students who have had no previous exposure to an introductory network survey class do just as well. This fact may be the result of the high level of self-taught networking competency found in highly motivated, networking-oriented students.

The classroom instructional aspect of this environment requires several different instructional themes. The first theme is C programming using the sockets application-programming interface. Allocation of three weeks to sockets-level programming has proven sufficient to yield sufficient "sockets programming" sophistication for students with some previous C programming experience. The course lecture topics in this component of the class include clients and servers, example "sockets programs," elementary TCP sockets, client server examples, elementary UDP sockets, and discussions on how to generate both TCP and UDP traffic for evaluation of network performance.

The second theme required in the classroom instructional component is that of the details of internetworking and TCP/IP. These topics are typically covered in traditional university networking classes, thus the level of detail required here is dependent upon other classes that are used as prerequisites to this environment. The subject areas and the order in which they are presented are totally driven by the laboratory activities listed in Table I. There are numerous excellent traditional university-oriented networking textbooks available that cover these fundamentals and principles. One classic example is [6]. A typical set of lecture topics for this component would include TCP/IP protocol architecture, file transfer protocol details, TCP bulk-data flow, bandwidth-delay product, slow start, timeout and retransmission, congestion avoidance, fast retransmission, examination of tcpdump traces, address classes, physical addresses, IP routing, ethernet, subnet addressing, subnet masks,

subnets with variable length masks, supernetting, classless interdomain routing, IP forwarding, routing versus switching, routing protocols routing information protocol, open shortest path first, and border gateway protocol.

The third theme in the classroom instruction is how internet functionality is implemented in an operating system. Using Linux as an example implementation, tracing an internet protocol datagram through the machine allows one to see exactly what is involved in the implementation of an internetwork. The goal of this theme includes the ability to understand how to change the C code implementation for the last two laboratory exercises, shown in Table I. This component of the class involves the case study of tracing the life of a packet through the Linux operating system. The lectures provide both an overview of this process and a detailed study of the data structures and line-by-line examination of the source code involved.

There is no single textbook that can be used for this entire environment. For the firstclassroom, instructional component [7] is used; for the second, component [8] is used; and then for the third, component class notes are used. There are several books appearing that are beginning to address the third instructional component [9], [10], but none of them yield sufficient detail into the networking aspects of the Linux kernel. Thus, this material has to be derived from a detailed study of the source code.

The students are assessed throughout the class by three written exams, a final written exam, and ten laboratory assignments. Written reports are required for some of the laboratory assignments, particularly in labs where interpretation of collected performance results or trouble shooting of the routing protocol is required as a part of the lab assignment. An example weighting of the student assessment components is 20% for each of the written exams, and the remaining 20% for the laboratory assignments and written reports.

IV. Capstone Design Experience

When the previous semester-long class has been completed, an additional semester-long senior capstone design experience is available. This follow-on course is one option that satisfies the major design project requirement for electrical engineering (EE) and computer engineering (CmpE) majors. "Working in teams, students complete a semester-long project requiring specification, design, implementation, and testing. Formal written project proposals and final reports are required and all students participate in oral presentations. Projects incorporate engineering standards and realistic constraints that include most of the following considerations: economic, environmental, sustainability, manufacturability, ethical, health and safety, social, and political. Projects for this course are based upon student's prior coursework in electrical and computer engineering" [12]. There is no lecture component in this capstone design experience. "The EE and CmpE design experiences are intended to provide a 'capstone' or major design experience that culminates the students' undergraduate engineering program, integrating their accumulated technical knowledge with practice-oriented aspects of design" [12]. The experience consists of a required preparatory course, project engineering and professional practice [13], plus a capstone design environment. This experience is the primary mechanism for satisfying the following portion of ABET general engineering criterion 4, professional component [14]: "Students must be prepared for engineering practice through the curriculum culminating in a major design experience based on the knowledge and skills acquired in earlier coursework and incorporating engineering standards and realistic constraints that include most of the following considerations: economic; environmental; sustainability; manufacturability; ethical; health and safety; social; and political. Additionally, this experience is one of the elements for demonstrating that graduates possess the attributes required by ABET general engineering criterion 3, program outcomes and assessment" [14]. "Assessment of the effectiveness of the design experiences in achieving these objectives must be documented, primarily through appropriate written project reports. Use of additional methods, such as project reviews by industry partners, is also appropriate and desirable" [14].

Typical examples of resulting projects are an internetwork packet sniffer with a graphical interface to control and display the collection of Internet packets with decoding of the packet contents; a mobile internet telephone implemented with a battery-powered, single-board computer using voice over IP; a network security monitor that alerts a network manager automatically by email when access violations or unusual activity occurs; a wireless Linux protocol sniffer using wireless ethernet; an internet vulnerability scanner; a network performance monitor that shows bottlenecks and throughput; a "Bluetooth" wireless file transfer utility; and a personal digital-assistant-controlled X10 remote electronic equipment controller.

Performance measurement metrics of the students include a written and oral presentation for project proposal, critical design review, and final project review as shown in Figs. 6–8. Figs. 6–8 are included only to give an example of the types of evaluation criterion that have been applied to evaluate the project results. Weekly email status reports are required and a peer review process involving all participants assigning other participants grades at each of the three course milestones is also considered in the final grade assignments. Group meetings with the instructor each week are required.

The laboratory equipment required in the follow-on class includes the equipment available for the previous class, as described in detail previously, and additional equipment such as single-board, battery-capable computers, wireless 802.11 cards, Bluetooth cards, etc. The single-board computers are available for those networking implementations that are required to be mobile. The single-board computers run the Linux operating system and are capable of using wireless 802.11 cards, for example. Projects that require mobility and global positioning capability may use this type of platform.

V. Conclusion

The learning environment presented in this paper was created out of a need to prepare undergraduate students for both industrial and research orientations. A common interview technique for industrial positions involves asking technical networking questions. Asking technical questions is a typical

Proposal Grading Criterion
_____/10 What is the product that is planned to be designed and prototyped?
_____/10 What competitor's products exist, and how will this product differ?
_____/10 What are the specific building blocks (tasks) of the project and the time line (schedule) for completing these building blocks or milestones? What is the order in which the tasks will be completed?
_____/10 How are the tasks divided and what are the tasks assignments to each individual?
_____/10 If there is a Graphical User Interface, what will the screens look like and what information will they display?
_____/10 What are the challenges and anticipated problems of the project in general and the tasks specifically? What is the degree of difficulty and the risk involved in each task?
_____/10 What will be demonstrated at the end of the semester, and what will this demo show?
_____/10 What equipment will be needed to accomplish the project and the demo? What parts, components, number of computers, configuration, etc.?
_____/10 What parts of existing products or code will be used, and what will be designed and implemented?
_____/10 Testing Plan: How will product testing be accomplished?
_____/10 Economics Analysis: How much are other products selling for, and how much will it cost to develop this product?
_____/10 Report Format, Clarity, Style
_____/10 Bibliography and web links related to project
_____/20 Presentation in class
_____/150 Total

Fig. 6. Example proposal evaluation criteria.

Critical Design Review Criterion
_____/30 Report Format, Clarity, Style
_____/20 Presentation In Class
_____/20 How stable is the solution? Has the method of how to complete the project stabilized, or are changes in direction and initial software still occurring?
_____/20 How much progress has been made on the product? What is working at present?
_____/10 What can be demonstrated at present?
_____/10 How well was the original schedule followed? Were milestones met?
_____/10 Were all tradeoffs examined, and were the reasons for choosing the solution reasonable?
_____/10 Testing Plan: How is product testing being accomplished?
_____/10 Documentation: Is documentation being completed in parallel, or is the product documentation appearing to be delayed until the last minute?
_____/10 Bibliography and web links related to your project
_____/150 Total

Fig. 7. Example critical design review criteria.

interview technique to determine if the individual is more focused on the creating of network diagrams and the configuring routers process of networking or on the foundations of how it really works. With the plethora of commercial networking certificates available, it is common to find two-week wonders who know how to configure a router but have no idea how routing protocols really work. Individuals that have participated in the environment described in this paper are also prepared for a research environment, particularly when that environment involves experimentation and prototyping of new network techniques. This environment was created to provide a complement to pure theoretical networking classes, enabling networking-oriented students to gain enough experimental and implementation skills to be able to succeed in the near term in an internetworking design, capstone design class.

One of the major challenges encountered with this environment is the difficulty associated with the ever-changing networking environment. Revisions to the Red Hat Linux distribution are released at a rate greater than or equal to the number of times this class is offered. Consequently, changes and revisions are required to both the laboratory exercises themselves (that often do not work with the new operating system versions because of networking and library revisions in the source code) and to the source code instructional materials. As an example, the migration from the Red Hat 6.x releases to the Red Hat 7.x release had a major impact on this class. It is challenging to keep up with the changes and optimizations made by the Linux networking community. An additional challenge is acquisition and maintenance of the networking equipment used in the laboratory component. Having obtained a single piece of certain types of equipment (for example, the hardware traffic generator used to perform router throughput performance measurements), the authors find that bottlenecks with those pieces of equipment limit participants throughput. Although the energy and effort associated with this learning environment is significantly larger than with more traditional classroom instruction, the additional benefit in terms of resulting networking competence is well worth the additional costs.

One could argue that when the laboratory is functioning, it is not necessary to upgrade Linux versions any faster than once per year. Certainly this is a valid argument since this laboratory network is separated from the remainder of the campus network. The authors have found that there are at least two reasons for staying "somewhat" current in the laboratory. The first reason is that many retailers sell Red Hat distributions but only keep the latest distribution in stock. Although it is possible to download the distribution, many students prefer to purchase their own copies for home use and related projects. When the laboratory software version is not up-to-date, it is not possible for students to purchase these older distributions from these retailers. The second reason to stay somewhat current is that newer kernel versions often have new functionality that was not available in the older kernel versions. While possible to find patches and modules that add these types of functionality, it is much better from a logistics standpoint to have these functionalities already integrated. For example, the ethereal tool capability was manually added to older distributions. In the later ones, it is already present.

Laboratory equipment upgrades do not all have to be all encompassing. It is possible to upgrade individual computers. However, when upgrading individual computers, throughput performance tests will not work consistently over all equipment. Running a TCP traffic generator and sink over a 600-MHz

Final Project Criterion
_____**/20 General: Written Report Format, Clarity, Style, Neatness:** (1) Table of contents in the report? (2) Report logically organized? (3) Technical explanation of how the code works? Flow charts and diagrams included? (4) Graphical user Interface? If so, appearance of screens? Information displayed? (5) Details adequate to reinstall, recreate, and modify by others? (6) References, bibliography, and/or web links included? (7) Clear explanation of other approaches attempted? Why not used in the end product?
_____**/20 Project Characteristics:** (1) Level of complexity of the project? "Coolness, gee wow" factor? (2) Level of success in delivering the product? How much guidance necessary from instructor?
_____**/10 Marketing:** (1) Estimate of selling cost in dollars? (2) Marketing section of the report to sell product to customers/instructor? Cost estimate for installation/use of product? (3) Comparison to other companies' products? List of competitors' products and how they differ? (4) Economics section? Price list of other products? Cost of developing this product?
_____**/10 Logistics:** (1) Credit given to those who assisted group? (2) Contribution of each project member listed? (3) Inclusion of projected versus actual schedule? Explanation of why the projected schedule was not met? (4) Specific project building blocks (tasks) enumerated? Time line (schedule) provided for completing building blocks and milestones? Order for tasks to be accomplished? (5) Division of tasks? List of assigned and accomplished tasks? (6) Explanation of project challenges and problems? Degree of difficulty in each task outlined? Total accumulated number of "person hours" required to complete project?
_____/10 Testing: (1) Test plan used? Testing accomplished how? (2) No bugs in the product release determined how?
_____**10 Conclusions:** (1) Suggestions provided for changes/enhancements in product? Why?
_____**/30 Appendix and WEB Page Creation with Electronic Report File, Final Presentation, and Code:** (1) Appendix includes commented source code printout? (2) List of all supporting software and operating systems used? (3) Existing code used? Statement of what code is new? (4) "Make files" included in printouts and electronic copy of the code? (5) Instructions included for loading and installing all components? (6) User document with "demo" explaining how to run code? (7) Instructor obtained source code and presentation materials from the group to place on class web page?
_____**/20 Project Class Presentation:** (1) Evaluation of group oral presentation. (2) Evaluation of group oral summary of information in written report.
_____**/20 Project Demo:** (1) Clear diagram of test bed presented? (2) Written explanation of "demo" and what one should see? (3) Good oral description of "demo"? (4) "Demo" clear, successful? Adequate presentation of project capabilities?
_____**/150 Total**

Fig. 8. Example final project review criteria.

Linux router versus a 1.1-GHz router will yield different results. Having hardware that is as consistent as possible throughout the lab makes it it possible to use diverse equipment types and still be able to implement this laboratory. Equipment upgrades are equally important as software upgrades. However, in this environment, the authors have upgraded software releases four times and have upgraded the hardware only once. This difference is mostly a result of the difficulty in finding funding to upgrade and enhance the hardware, while software upgrades are easier to obtain.

The instructors of the class believe that the hands-on laboratory-oriented environment produces an enhanced educational result as opposed to the previous method of only theoretically presenting the material. Student feedback indicates that the hands-on-laboratory environment allows a level of understanding that is difficult to obtain from only a theoretical

TABLE II
STUDENT EVALUATION OF CLASS AVERAGE VALUES OF: 5 = STRONGLY AGREE, 4 = AGREE, 3 = PARTLY AGREE AND DISAGREE, 2 = DISAGREE, AND 1 = STRONGLY DISAGREE. * = INSTITUTE SURVEY CHANGED AND NO LONGER ASKED THESE QUESTIONS

	Spring 2000	Fall 2000	Spring 2001	Fall 2001
Presented/Explained Objectives at Beginning	4.3	4.6	4.8	*
Held Student Interest	4.2	4.2	4.7	*
Course Well Planned and Organized	4.0	4.1	4.8	4.7
Covered Course Objectives	4.3	4.4	4.8	4.6
Explained Complex Material	3.8	3.7	4.3	4.2
Exams/Evaluation Procedures Fair	4.0	4.2	4.7	4.4
Exams Covered Course Content	4.1	4.4	4.8	4.8
Exams of Appropriate Difficulty	4.1	4.3	4.6	4.4
Number of Students Responding to Survey	10	54	53	17

examination of the principles. Additional student feedback on the class is shown in Table II. The goal of graduating students who are ready for experimental internetworking environments has been met. Graduates of this class are ready and able to make changes to the networking protocol stack and thus change the behavior of existing internet implementations. There are many situations where simulation and analytical analysis provide all of the required results. However, when prototyping is required to verify predicted results or enable experimental results, students from this class are able to accomplish those tasks. Feedback from student participants indicate that the greater effort required to set up and maintain this type of class is well worth the extra effort compared to a traditional networking class, where networking equipment and a hands-on networking environment are not involved.

ACKNOWLEDGMENT

The authors would like to thank J. Grimminger and J. Sokol, Siemens ZT IK2 Corporate Technology Division, Munich, Germany, who suggested some of the laboratory assignments. They would also like to thank the numerous graduate teaching assistants who have made excellent improvements to the laboratory assignments. Finally, they would like to thank equipment donations and academic purchase programs from Cisco, Enterasys, Intel, Spirent, and Wandel and Goltermann.

REFERENCES

[1] *Red Hat Home Page*, http://www.redhat.com,, Mar. 25, 2002.
[2] M. Muss and Army Research Lab, Baltimore, MD, "The story of the TTCP program,", http://ftp.arl.mil/~mike/ttcp.html, Mar. 26, 2002.
[3] E. Hall, *Internet Core Protocols: The Definitive Guide*. Sebastopol, CA: O'Reilly, 2000, pp. 1–449.
[4] *The Click Modular Router Project Home Page*, http://www.pdos.lcs.mit.edu/click/, Mar. 25, 2002.
[5] *ECE4110 Internetwork Programming Home Page*, http://users.ece.gatech.edu/owen/academic.htm, March 26, 2002.
[6] W. Stallings, *Data and Computer Communications*. Upper Saddle River, NJ: Prentice–Hall, 2000, pp. 1–810.
[7] R. Stevens, *Unix Network Programming*, 2d ed. Upper Saddle River, NJ: Prentice–Hall, 1998, vol. 1, pp. 1–1009.
[8] ——, *TCP/IP Illustrated*. Reading, MA: Addison–Wesley, 1994, vol. 1, pp. 1–576.
[9] J. Crowcroft and I. Phillips, *TCP/IP & Linux Protocol Implementation: Systems Code for the Linux Internet*. New York: Wiley, 2001, pp. 1–925.
[10] S. Maxwell, *Linux IP Stacks Commentary*. Scottsdale, AZ: Coriolis Open Press, 2000, pp. 1–591.
[11] M. Beck, H. Bohme, M. Dziadzka, U. Kunitz, R. Magnus, and D. Verworner, *Linux Kernel Internals*, 2d ed. Reading, MA: Addison–Wesley, 1996, pp. 1–438.
[12] *Georgia Inst. Technol. School of Elect. Comp. Eng. Undergraduate Programs home page*, http://www.ece.gatech.edu/academics/undergrad/index.htm, Mar. 25, 2002.
[13] J. L. A. Hughes, "Incorporating project engineering and professional practice into the major design experience," *Proc. 31st ASEE/IEEE Frontiers in Education Conf.*, pp. 16–21, 2001.
[14] J. L. A. Hughes, "2000–2001 criteria for accrediting engineering programs," Accreditation Board Eng. Technol., Inc., Eng. Accreditation Comm., Baltimore, MD.

Randal T. Abler (M'00) received the B.E.E., M.E.E., and Ph.D. degrees from the Georgia Institute of Technology, Atlanta, in 1986, 1992, and 2000, respectively.

He is currently an Assistant Professor with the Georgia Tech Regional Engineering Program, Savannah. His research interests include the session initiation protocol, quality of service implementations, and internet technology in support of distributed education.

Henry L. Owen (M'79) received the B.S.E.E., M.S.E.E., and Ph.D. degrees from the Georgia Institute of Technology, Atlanta, in 1979, 1983, and 1989, respectively.

He is currently an Associate Professor with the School of Electrical and Computer Engineering, Georgia Institute of Technology. His research interests include network security, network protocol stack implementations, traffic engineering in multiprotocol label switching-enabled wireless IP networks, and internetworking.

George F. Riley (M'00) received the B.S.E.E. degree from the University of Alabama, Birmingham, in 1972, the M.S.C.S. degree from the Florida Institute of Techology, Melbourne, in 1996, and the Ph.D. degree in computer science in from the Georgia Institute of Technology, Atlanta, in 2001.

He is currently an Assistant Professor with the School of Electrical and Computer Engineering, Georgia Institute of Technology. His research interests include large-scale simulation methods, in particular, as they apply to computer networks.