Lab Exercises for Computer Networking Courses

HARVEY MUDD COLLEGE

Michael Erlinger, mike@cs.hmc.edu

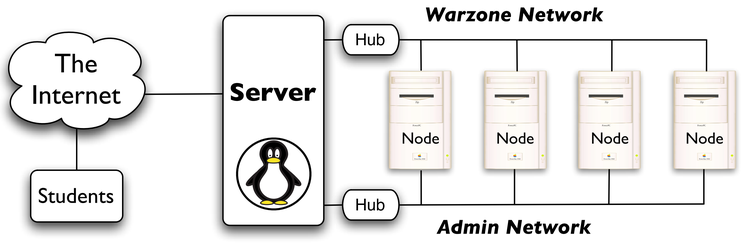
ITiCSE 2006

TinkerNet

TinkerNet was developed as a low-cost platform for teaching bottom-up, hands-on networking at the undergraduate level. Using "throw away" PCs, cheap components, and free software, TinkerNet enables students to build their own networking stack from Ethernet up to TCP or UDP, and to have their packets actually transmitted on the wire. Since nothing is emulated, standard networking tools such as packet sniffers may be used to test student generated traffic from a host located on the TinkerNet network.

System Overview

At its core, TinkerNet (Figure 1) is a system for easily letting students insert code for processing, generating, and responding to network packets into an OS kernel and booting it on a real PC. The system is designed to work with very limited hardware resources, and can likely be assembled with parts that can be found unused in a decent sized institution.



When using TinkerNet, students are provided with a skeleton source tree containing the function prototypes they must implement, as well as a GNU Makefile pre-configured (to build the student's source, to link the student object code to the existing object code for handling the admin network, and to prepare the image to be sent to a node). Using tools on the server, students can have their kernel remotely booted on one of the nodes and view output from that kernel. At no time does the student have to be aware of the existence of the admin network or the infrastructure in place to support it. Finally, when the student is done testing a particular build of their kernel, they can simply push a button on the interface (tinkerboot) and have their node reboot and rejoin the pool of ready nodes.

Laboratory Experiments

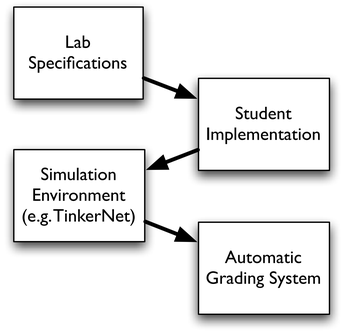
We have created a semester-long set of laboratory experiments focused on student development of a fully functional network protocol stack. In this set of experiments each new experiment builds on previous experiments. We begin with an experiment to review some issues around programming in C, and then work our way up from raw Ethernet packets to fully functional IP and then UDP. The final two experiments have students create their protocol and implementation for machine discovery and implement Blast , a micro-protocol which fragments and reassembles large messages. We believe that there are many other experiments which could be created, but that a full implementation of TCP would require much more time than is available in a semester. We envision TinkerNet being used in advanced courses to implement application protocols and/or network devices, such as a router.

Current Set of Laboratory Experiments

- Lab 1: The goal of this assignment is to gain proficiency with C programming, and to (begin to) learn the differences between C and C++. There are also a few exercises that address networking in general, focusing on concepts like byte ordering and struct usage.
- Lab 2: The goals of this assignment are to gain familiarity with the lab environment, to successfully compile a TinkerNet kernel, and to implement functions that send and receive Ethernet packets.
- Lab 3: The goals of this assignment are to gain more familiarity with the lab environment, to successfully compile a TinkerNet kernel, and to implement functions that send and receive ARP packets.
- Lab 4: The goals of this assignment are to implement an end-host version of the Internet Protocol. The implementation must be able to recognize IP packets addressed to a specific IP address and ignore those addressed to other IP addresses.
- Lab 5: The goals of this assignment are to implement the sending and receiving of UDP datagrams, as well as a simple service to test this functionality.
- Lab 6: The goals of this assignment are to design, to create, and to implement a peer-to-peer protocol that will be used to locate other hosts on the network running the same protocol. Using this protocol, two machines will simultaneously boot on TinkerNet, locate each other, and then transmit data between themselves.
- Lab 7: The goals of this assignment are to implement the micro-protocol Blast. Blast fragments and reassembles large messages and attempts to recover from dropped fragments by retransmitting them.

Future Work

While TinkerNet has been fun to build and fulfills our requirements for a networking laboratory, we recognized that the computer science education community is more interested in a set of laboratory experiments for students and an automatic grading tool that can evaluation student work.



It is our intention to recreate our student laboratory experiments in such a way that each experiment can produce a controlled output that can be evaluated. For example, a student's Ethernet implementation could be required to output certain frames based on frames received. An auto-grader could then send packets and evaluate student responses. We will be seeking support to develop such a system and invite other interested parties to contact us.

Acknowledgments

Titus Winters, Mark Kegel, Daniel Turner, Ryan Ausanka-Crues, Erik Shimshock