Massachusetts Institute of Technology
Department of Electrical Engineering and Computer Science

6.033 Lab, Spring 1998 — Handout 3 — A TCP Proxy Server — February 13

Due date: February 20

1 Introduction

The first programming project in 6.033 is meant to prepare you for the next two projects by acquainting you with UNIX programming, especially network programming. Unlike the other labs, we will start you off with some code for setting up network connections, since the code for doing this is very mechanical. From this point, you will build a fully functional TCP proxy.

2 Running a simple client/server program

We'll be using a trivial client/server program as our stepping stone. The C sources for the program can be found in the 6033 locker in the `lab/src/one` directory. You should copy the contents of the directory to a directory under your control.

Now, build the client and server. Start the server by typing `./server`. In another window, type `./client` *name*, where name is the name of the machine on which the server is running. You can find out the name of a machine you are using into by typing `hostname`. Every time you run the client, the server will print the notorious "hello, world!" message.

3 Understanding the client and the server

Once you get the programs running, you should study the code and try to understand it. You might find it helpful to read chapter 10 ("Communication in distributed systems") of the Tanenbaum book to obtain a general understanding of client/server programming. The specifics of each UNIX networking call can be found in the manual pages (e.g., type `man socket`). If you cannot make sense out of the program using the sources, Tanenbaum, and the man pages, you might want to borrow "UNIX network programming," written by Richard Stevens; or, get in touch with your TA.

4 TCP Proxy Server

A TCP proxy server is a server which acts as an intermediary between a client and another server, called the destination server. Clients establish connections to the TCP proxy server, which then establishes a connection to the destination server. The proxy server sends data received from the client to the destination server and forwards data received from the destination server to the client.

Interestingly, the TCP proxy server is actually both a server and a client. It is a server to its client and a client to its destination server.

A TCP proxy server can be useful to get around services which restrict connections based on the network addresses. For example, there are many servers at MIT which will only serve data to addresses within the MIT network. By running a proxy server on the MIT network, clients from outside MIT can use those servers by connecting through the proxy server. The MIT servers will think they are serving data to a machine on the MIT network (namely the proxy server machine) and they'll be right. However, the proxy is forwarding the data out of the MIT subnet, thus subverting the protection mechanism. This is a violation of the rules of use on most of those server, so we do not encourage you to do this.

The proxy server you will build for this lab will be invoked at the command line as follows:

`% ./proxy` *destination-host destination-port listen-port*

For example, to redirect all connections to port 3000 on your local machine to the MIT web server, do:

```
% ./proxy web.mit.edu 80 3000 &
```

The proxy server will accept connections from multiple clients and forward them using multiple connections to the server. No client or server should be able to hang the proxy server by refusing to read or write data on its connection.

5 How/What to hand in

When you're ready to hand in the lab, copy your project to the `/mit/6.033/lab/handins/p1/`*username* directory, where *username* is your Athena user name.

The project should include a valid Makefile and the source file or files for the simple TCP proxy. The Makefile should create a proxy server named `proxy`.

6 References

Stevens, W. Richard. *Advanced Programming in the UNIX Environment.* Addison-Wesley, 1992.

Stevens, W. Richard. *Unix Network Programming.* Addison-Wesley, 1990.