CS 3342 Laboratory Problem Set 2

Due November 6, 2000

These problems are to be done on an individual basis following the Trinity University Academic Integrity Policy. Laboratory problems should be submitted electronically (e-mail to cs3342@ariel.cs.trinity.edu) on or before the due date and should contain a problem write-up, source code to any programs and data sets used in solving the problem. The submitted files should be ASCII text files having Unix end-of-line characters (please convert all Windows and Mac text files to Unix format–I have found that Emacs seems to do a reasonable job of such conversions). If several files need to be submitted, put them in a directory having name *your-last-name-problem-set-number* and create a tar archive of this file system and attach it to your e-mail problem submission.

An Internet `talk` Program

In this problem you are to design a simple `talk` program that will allow two users who are running on the same internet to interactively communicate with each other.

There are several design issues with which you must deal.

- You have the choice of using datagram or stream protocols.
- Some protocols require the use of a port number. Your program may allow the host which is running the server to dynamically request a port number. This port must then be communicated (a hard design issue) to the client. An alternate approach is to assign a fixed port number so that both the server and client know the port number in advance.
- Another issue is that of designing an appropriate sequence for starting two copies of the same program (or two different student programs) on two machines so that they can successfully begin communicating.
- Yet another issue is how one handles the display of messages. Essentially, your program should allow messages typed on one workstation to be displayed at another station and receive messages which are typed at the second station. The problem is to devise some approach do differientate the local echo of outgoing messages from received messages. A simple

method might be to buffer incoming message text while a line is being prepared output. When the buffered text is displayed, it might be prefixed with an appropriate prompt to separate it from the local echo of outgoing text. You should keep the user interface portion of the design as simple as possible, remembering that this is a course in networking, not user-interface design.

Problem Set 2 Solution [HTML] [PS] [PDF]