CSE 473 – Introduction to Computer Networks

Studio 1

(*Adapted from Jon Turner's Studios*)

General instructions for studios. The purpose of the studio sessions is to help you get a better understanding of the material, and to help you prepare for the labs. Use this opportunity to learn as much as you can. Talk to the others within your group. Ask the TAs for help whenever you have questions. Studios are not graded and there is nothing to turn in, but the more you learn during studio, the better prepared you will be to tackle the labs.

When doing the studios in one of the Urbauer labs, be sure to display the computer used by the "current driver" on the large monitor so that everyone can follow along, and so that the TAs can observe what you're doing. There are two ways you can do the studio. One way is for a single person to do all the typing, while the others observe and participate in the discussion. The other is for everyone to do the exercises together. You may find this gives you a better understanding of the material. If you choose this second approach, try to stay "in sync" with the others in your group and talk to each other about what you're doing. Either way you do it, be sure to change drivers several times during the studio. Using the first approach, you'll find it simplest if you just get up and switch seats from time-to-time. With the second approach, you can just switch the big display from one computer to another.

The main objective for this session is to familiarize you with the *Wireshark* packet sniffer, which is already installed on the Urbauer lab computers. You can find a link to the introductory *Wireshark* lab from Kurose and Ross on the class wiki in the section entitled "Material from Kurose & Ross/Student Resources/Intro to Wireshark labs".

Next, we will be using *Wireshark* to observe traffic flowing between a client and server pair that you will run. In your svn repository, you will find two Java programs *UdpEchoServer.java* and *UdpEchoClient.java* in the *studio1* folder. Review the source code and make sure you all understand how it works.

On one computer run *UdpEchoServer* by typing the following in a command prompt window in the folder that contains the *UdpEchoServer* program.

java UdpEchoServer

and on the other, run *UdpEchoClient* by typing

java UdpEchoClient serverName 30123 "echo me now"

where you should type the name of the computer running the server in place of *serverName* (to get the name of a computer, type *hostname* in a shell or command window). This will send a string to the server. Check the response you get back. Now, repeat this, using *Wireshark* to observe the packets going between your two computers. Use *Wireshark's* filtering mechanism so that you capture only udp packets going between the two computers (type the filter string "udp.port==30123" in the filter text box). Observe the packets on both sides and compare what you are seeing in the two cases.

To stop the server, type CTRL-Break in the window where the server is running (for Unix and Linux, use CTRL-C). Now, re-run the server using a different port number and run the client using this new port number for the server.

Now, modify the server program so that it capitalizes all characters in the received string before sending it back to the client (use Java's *String.toUpperCase*() method to capitalize all characters in a string; you'll need to convert the byte array to a string before you can use *toUpperCase*). Run this version and observe the results using *Wireshark*.

Using *Wireshark*, examine one of the packets in detail. How many bytes does the packet contain? How many of these are accounted for by the IP header? The UDP header? The actual string sent from the client to the server? Notice how when you click on items in the center sub-window, the actual bytes are highlighted in the bottom sub-window. Check a few of the values to see how they correspond.

Finally, modify the client, so that it encodes the strings it sends using UTF-16, instead of US-ASCII. Test this using the original version of the server. Examine the packets using *Wireshark*. Look at the first few bytes in the payload and compare them to the string that was sent. Consult the UTF-16 encoding table that you can find at

www.fileformat.info/info/charset/UTF-16/list.htm

Are the encoded bytes consistent with what you expect?