ECE 537 Spring'14 Programming Assignment 1

University of Wisconsin-Madison

Instruction:

Please complete the programming assignment and submit the answers in pdf to Moodle, by 6pm Mar. 24, 2014. The assignment should be completed individually, but you can discuss the basic ideas with other people.

Tasks:

Task 1: In this assignment, you will develop a simple Web server in Java that is capable of processing only one request. Specifically, your Web server will (i) create a connection socket when contacted by a client (browser); (ii) receive the HTTP request from this connection; (iii) parse the request to determine the specific file being requested (note: the file should be a plain .txt file); (iv) get the requested file from the server's file system; (v) create an HTTP response message consisting of the requested file preceded by header lines; and (vi) send the response over the TCP connection to the requesting browser. If a browser requests a file that is not present in your server, your server should return a
"404 Not Found" error message.

Skeleton code for the server is provided in the Appendix I. Your task is to complete the code, run your server, and then test your server by sending requests from browsers running on different hosts. Here are some suggestions:

(1) You can test your code on localhost if you do not have access to multiple computers;

(2) You need to configure your browser options (e.g., for Firefox: Tools->options->Advanced->network->connections) such that all its requests are directed to your own server;

(3) If you run your server on a host that already has a Web server running on it, then you should use a different port than port 80 for your Web server.

Task 2: In this assignment, you will investigate the traffic patterns of different application-layer protocols. You can capture protocol-specific packets using Wireshark, and then parse the captured packets to examine their properties. The following are specific steps you should take.

(1) Run Wireshark to capture packets generated by four different protocols: HTTP, FTP, VoIP, and BitTorrent. Each capture should last for at least 1 minute. During the capture, try to stop all other irrelevant applications that may access the Internet. Save the captured packets in *.pcap* files (the default file format in Wireshark).

- HTTP: capture the traffic when you visit a website, e.g., news.yahoo.com, and its sub-pages. Try to mimic the click behavior when you normally browse that website.
- FTP: run an FTP file-downloading session for 1 minute. Below is an example FTP-accessible file:

ftp://ubuntu.osuosl.org/pub/ubuntu-releases/precise/ubuntu-12.04.4-desktop-amd64.iso

Alternative FTP website can be found here: https://launchpad.net/ubuntu/+cdmirrors

- VoIP: start a VoIP call, e.g., using Skype, with some remote host, and capture the traffic. Again, make it a real VoIP call, in which speech and silent periods interleave.
- BitTorrent: run a BitTorrent session (you'll need to install a BitTorrent client software first). Download files that are NOT copyrighted, e.g.,

  http://www.ubuntu.com/download/alternative-downloads

(2) Extend the Java pcap file parser library provided in Appendix II, so that the payload size and arrival time of all packets can be parsed based on the .pcap file you saved. The *PcapParser.java* file provides all the utility functions you need to analyze the packet statistics. Visit the following website if you want to understand the format of the packet captured by Wireshark.
http://www.tcpdump.org/pcap.html

(Note: Relevant information starts from "The pcap_pkthdr structure is defined in…")

(3) For each application, plot:

- The cumulative distribution function (CDF) and probability density function (PDF) of the captured payload sizes. Matlab is a convenient tool for plotting CDF and PDF.
- The cumulative distribution function (CDF) and probability density function (PDF) of the packet inter-arrival time, i.e., temporal separation between two consecutive packets.

Answer the following questions:

- How does the payload size distribution of different applications differ from each other? Briefly explain what causes the difference.
- Similarly, explain your observations of the inter-arrival time distribution.