CSc361 Fall 2020 Programming Assignment (P1): Simple Web Server (SWS) Specification

Out: by Thursday, September 17, 2020

Due: by Friday, October 2, 2020, 5pm through connex.csc.uvic.ca

In this programming assignment, you will use the STREAM socket (i.e., supported by TCP) in Python to create a Simple Web Server (SWS) supporting both persistent and non-persistent HTTP connections. There are three tutorials (T2, T3 and T4) and two lab sessions (L2 and L3) during the course of this assignment. In T2 on September 18, the tutorial instructor will go through the P1 specification and answer possible questions, and will also go through Python Socket programming. In L2 on September 22 or 23, the lab instructor will go through HTTP (both persistent and non-persistent, using nginx-light and wget) packet capture and analysis, and help students form their design for P1. In T3 on September 25, the tutorial instructor will check students P1 design, provide feedback and instruction on submission through connex. In L3 on September 29 or 30, the lab instructor will check students P1 implementation, capture and analyze SWS packets, and provide help if needed. In T4 on October 2, the tutorial instructor will provide some last-minute reminders and help. Please follow our tutorial and lab schedule closely for this assignment, which ensures its success and smoothness.

SWS requirements: Only supports “GET /filename HTTP/1.0” command, and “Connection: keep-alive” and “Connection: close” request and response header when supporting persistent HTTP connection. The request header is terminated by an empty line. If unsupported commands received or in unrecognized format, SWS will respond “HTTP/1.0 400 Bad Request”. If the file indicated by filename is inaccessible, SWS will return “HTTP/1.0 404 Not Found”. For successful requests, SWS will respond “HTTP/1.0 200 OK”, followed by response header if any, an empty line indicating the end of the response header, and the content of the file.

How to run SWS: On H2 in PicoNet, “python3 sws.py ip\_address port\_number”, where ip\_address and port\_number indicate where SWS binds its socket for incoming requests. On H1 in PicoNet, “telnet sws\_ip\_address sws\_port\_number” to connect to SWS, and type “GET /sws.py HTTP/1.0” followed by “Connection: keep-alive” and an empty line to request the file sws.py from SWS (in this case, SWS shall keep the connection alive after sending back sws.py following an empty line after “Connection: keep-alive”, and wait for the next request from the same client through the same TCP connection, until the connection times out, i.e., “Connection: close”). If the client does not include “Connection: keep-alive” or does include “Connection: close” in its request, SWS will close the connection after serving the request. For each served request, even if unsuccessfully, SWS will output a log line “time: client\_ip:client\_port request; response”, e.g., “Wed Sep 16 21:44:35 PDT 2020: 192.168.1.100:54321 GET /sws.py HTTP/1.0; HTTP/1.0 200 OK”. Please note that SWS will keep waiting to serve more clients and can serve multiple concurrent clients, until interrupted by Ctrl-C.

How to test SWS: Capture and analyze the interaction between SWS and its clients (telnet, nc, wget or a regular web browser) with tcpdump and Wireshark. Your code will be evaluated at http://picohub.csc.uvic.ca

What to submit: sws.py source file, and the tcpdump files on R, showing the interaction between SWS and its clients, in both persistent (sws-persistent.cap) and non-persistent (sws-non-persistent.cap) connections. Please also copy&paste the content of sws.py and tcpdump -r the capture file to the text input box on connex.

When: by Friday, October 2, 2020, 5pm through connex -> assignments -> p1

Questions and answers: connex -> forums -> p1

Academic integrity: This is an individual assignment and your submitted work shall be done by yourself alone.