Lucas Reicher
CSE150 WIN22
Professor Parsa
1 March, 2022

Questions

1. How did you decide which type of socket to use? Why?

Since our application is supposed to send http requests and receive responses, I chose to create a socket using IPv4 (socket.AF_INET) as the address family and TCP (socket.SOCK_STREAM) as the socket type. Since HTTP relies on TCP to conduct communications between server and client, it seemed right to use SOCK_STREAM as the socket type. I also assumed (from the addresses given in the project pdf) that only IPv4 addresses would be given, so AF_INET would be the correct address family for the socket.

2. How did you choose the destination ports?

Firstly, if no destination port was given, I defaulted to port 80. In order to parse the port from user input, I performed the following steps:

Remove the "http://" scheme from the full URL, leaving me with URL:port/path

If there is a "/" in the resulting string, split on it, leaving me with URL:port and saving the path in a variable for later use.

If there is a ":" in the resulting URL:port string, split on it, giving me the URL, port, and path in individual variables.

Additionally, I checked for cases where the port was placed after the path, and raised an exception if such an input error was encountered. If the port was non-numeric, I would also throw an exception.

3. What error handling cases did you implement?

I implemented several error handling cases that would result in program termination:

Invalid # of command-line arguments (either argc != 1 or 2, or argc = 1 and the URL was in IP format).

Invalid IPv4 address (IP must have 3 '.'s, and the values on either side of the '.'s are >= 0 and <= 255)

Invalid schemes (either https:// or any other scheme that is not http://)

Non-numeric port or invalid placement of port in the URL.

Socket Errors, using socket.error to catch exceptions during socket creation, connection to the server, sending of http requests, or receiving of http responses

Chunked Encoding, by parsing the header I checked if chunked encoding was present and aborted the download if it was.

Non-numeric Content-Length. Although this exception case seemed unlikely, I would throw an exception and abort the download if the content-length value was not an integer.

Empty reply from server. In order to avoid infinitely looping on socket.recv(), I would check if the returned data from socket.recv() had a length of 0. If this happens, I print an error message and log the unsuccessful download in Log.csv

The only times I would log the exception is if the socket had been created. It did not seem necessary to log errors when they simply had to do with user input (like a non-numeric port) since there was so little relevant information to log. If an exception occured after the socket was created and connected to the server, I would include the client+server IPs/ports in the log entry.

4. How does your program terminate? What happens to the TCP connection?

In all cases, the program terminates by first closing the TCP connection (with socket.close()) and then gracefully exiting the python program (with sys.exit()). There are no cases where the program will unexpectedly crash without closing the socket. Concerning chunked encoding, the download is terminated resulting in a lack of FIN TCP messages. However, the TCP connection is still closed.

5. For the unsuccessful URLs, why were they unsuccessful?

Either there was a socket error (meaning the URL did not map to a valid server, the connection timed out, the port was not being listened to by the server or it was but the server refused the connection i.e. port 443), the response was chunk encoded, the reply from the server was empty, or the server status code was not 200.

6. What happens if you try to access a site using HTTPS?

The program will not send requests if the scheme is specified as "https://". If however, a user uses the "http://" scheme but specifies destination port 443 (signifying an https connection), the program will attempt to connect and send a request to the server. In this case, a number of things may happen. Either the server will close the connection, the server will respond with status code 403 (Forbidden), or the server's response will be empty. In all cases where a request/connection is attempted, the exception will be logged.