Chris VanBlargen Spencer LaValle

CS305 Networks

Prof. Amir Sadovnik

Write Up

**Introduction:**

Our Network simulator attempts to fully simulate packets from the application layer all the way down through the physical layer. Each layer disregarding the transport and application layer are responsible for propagating the byte array (payload) through each layer. Only once this payload reaches the physical layer does a socket get opened, and is the data transmitted.   
**Description (HTTP Implementation)**

Our HTTP implantation includes HTTP 1.0 and 1.1. The only differences between these two is the idea of persistent connections meaning that once a connection has been established between the client and server, the connection remains open unless a timeout of inactivity occurs. The advantage of keeping the connection open is less overhead as the three-way handshake (syn, ack, synack) doesn’t need to be executed for every payload of data, rather, only a single handshake is needed to establish a connection, and the payload can be sent freely.

These messages start at the client with a prefix of either: “COMMAND:”, “IP:”, “PROPAGATION:”, “TRANSMISSION:”, and “HTTP:” where we “skip” over said prefix and read the next token after the colon. For example, a client could send: “COMMAND: GET \n IP:localhost, HTTP:1.1, PROPGATION: 10, TRANSMISSION: 5” and that would request a page (GET), at the IP (localhost), with HTTP protocol 1.1, with a constant propagation delay of 10ms, and a transmission delay per byte of 5ms. While it isn’t the cleanest, it’s functional. To parse these, we have a switch statement responsible for parsing the original prefix, and sending the “suffix” of a command through to be interpreted.

The expected result of these messages is a payload with the code (200/ 304, 404) and the data appended to the end (assuming it’s not a 404).

**Description (Markup Implementation)**

<WORK ON THIS>

Description (Code Design & Data Structures)

Our project uses byte arrays as the primary data structure for message passing. If we were to recreate this project from scratch, rather than using the archaic byte array with its encoding, we’d much prefer passing an Object with member functions to be called through the layers, rather than relying on string to byte array conversions whenever the payload data needs to be verified/ modified/ cached.

Regarding Data structures, the only ones used were strings, byte arrays, Booleans, and text-files that function as the web-page to be sent from server to client when the page is requested.

Correctness Results

Results Analysis

Conclusion

References

Team Contribution