Eddie C. Fox

February 19, 2017

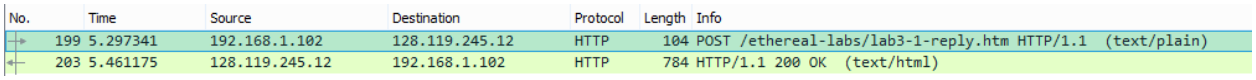
CS 373 Introduction to Networking

Lab 3

Note: When my last assignment was graded, they said to include the screenshots in the document, but they are usually too small to read, so I'll include clickable links for the ones that can't be read.

Part 2: A first look at the captured trace

1. In the trace, there is only two http packets.



No.	Time	Source	Destination	Protocol	Length	Info
199	5.297341	192.168.1.102	128.119.245.12	HTTP	104	POST /ethereal-labs/lab3-1-reply.htm HTTP/1.1 (text/plain)
203	5.461175	128.119.245.12	192.168.1.102	HTTP	784	HTTP/1.1 200 OK (text/html)

We can tell that 192.168.1.102 is the address of the client sending the file to the server because it is sending the post file.

2. We can tell the address of gaia host is 128.119.245.12 because the gaia host is sending the ok after receiving everything.

http://puu.sh/uaPnr/0c7ddadca4.png

When the client is sending a tcp packet, the source port is 1161 and the destination port is 80. When gaia is sending a packet, the source port is 80 and the destination is 1161.

3. My IP Address is 192.168.1.10, with a source port of 13400.

http://puu.sh/uaPG4/f2836699d1.png

Part 3: TCP Basics

4. Sequence number is 0. It can be identified as a SYN segment because it has flag 0x002 (SYN)

http://puu.sh/uaPYR/92bd548b69.png

5. Sequence number is 0. Acknowledgement number is 1. The host determined that value because the acknowledgment number is equal to the value of the next expected byte, which is 1 because it is looking for first one, as no data from the file has been transmitted

yet. It can be determined to be the SYSACK segment because it has flag 0x012 (SYN, ACK).

http://puu.sh/uaQ58/79af63ade6.png

6. Sequence number is 1.

http://puu.sh/uaQCp/14de7154b7.png

7. Sequence numbers: 1, 649, 1909, 3169, 4429, 5689

http://puu.sh/uaRYT/599b32852f.png

Times derived by subtracting ack packet time from sent packet time.

Round trip time of sequence 1: 135.246 ms

EstimtedRTT at this point is 135.246 ms, since it's the only only value.

Round trip time of sequence 649: 135.031 ms

Estimated RTT = (135.246 * 0.875) + (135.031 *.125) = 118.34025 + 16.878875 = 135.219125 ms

Round trip time of sequence 1909: 137.502 ms

Estimated RTT = (135.219125 * .875) + (137.502 *.125) = 118.32 + 17.19 = 135.51 ms

I started to round at this point because the decimal places were getting out of control

There was no ack for 3169 so I couldn't calculate RTT there.

Round trip time of sequence 4429: 136.808 ms

Estimated RTT = (135.51 *.875) + (136.808 *.125) = 118.57125 + 17.101 = 135.67 ms

Round trip time for sequence 5689: 135.614 ms

Estimated RTT = (135.67 *.875) + (135.614 * .125) = 118.71125 + 16.95175 = 135.663 ms

8. Lengths:

Length of sequence #1: 648

Length of sequence #649: 1260

Length of sequence #1909: 1260

Length of sequence #3169: 1260W

Length of sequence # 4429: 1260

Length of sequence #5689: 1260

9. The minimum buffer space is equal to the window size that is shown at the first acknowledgement of the server. This is 29,200 bytes. The window size steadily grows as the file transmission goes on until 263,424. It's possible that I am looking at it wrong, and the window size value is the minimum buffer space. That would be 260 bytes.

http://puu.sh/uaUTt/2a5e5472b8.png

10. There is no re-transmissions because when using wireshark to plot a stevens graph of sequence number vs time, the sequence numbers are continually increasing. If there was a re-transmission, you would expect the sequence number of a packet to be lower than the others surrounding it at a later point in time, as the sequence number would drop to a lower value.

http://puu.sh/uaXur/a8fbab03b6.png

11. I am assuming the ACK's typically acknowledge 1260 bytes of data, based off of the typical length of a packet.

http://puu.sh/uaWeo/7ac210097e.png

In my experience, the client was sending sequence numbers much faster than it received the acknowledgements from the host, so the acknowledgement numbers tended to lag behind. While I didn't see acknowledgements every other process, I did see large bursts of ACK packets where a bunch of ack's that were from awhile ago would suddenly come flooding in before the client would transmit more segments.

http://puu.sh/uaWjk/b2b503e98f.png

12. We can calculate the average throughput by determining the amount of data passed through the connection during the time it was up and divide by the total time of the connection.

Total amount of data can be determined by the ACK number number: 152,970. Hence, there were about 152,970 bytes during the connection.

The connection time can be determined by subtracting the time of the last acknowledgement from the first acknowledgement. 4.429217 – 3.588585 = 0.840632.

Therefore: 152.97kb / 0.840632 seconds = 181.97kb per second thoroughput.

13. Reposting the graph:

http://puu.sh/uaXur/a8fbab03b6.png

The program did not seem to be able to handle congestion very well, as the amount of packets sent before the burst of acknowledgements continued to increase.

14. When I was reading about TCP's, I imagined an ideal process where there was a sent packet, then an acknowledgement after. While I knew about pipelining, I imagined it would only be several packets at a time before acknowledgement, but in my case, the client sent numerous packets at times before acknowledgement began to come in, and this acknowledgement came in bursts.