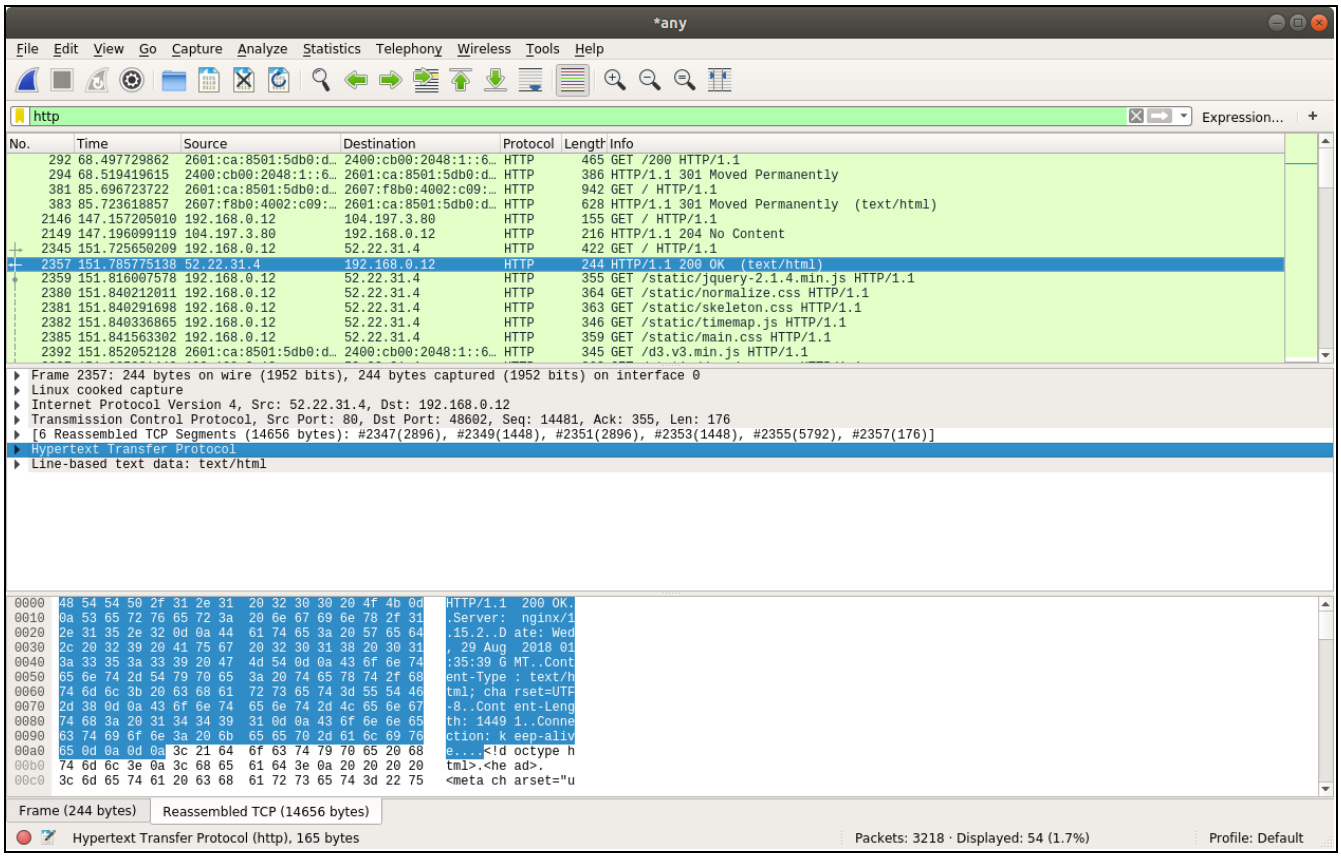
Csc 4220/6220 Fall 2018 Assignment#1

1) Capture the packets with Wireshark when you access a website and see how many protocols are involved in the packet-transfer and list them by making screenshot like above with filters for each.

2) Find your own IP address in the screenshot that you take and provide a screenshot of it too.

My IP address is 192.168.0.12. There are 3 protocols (not including the link layer of the router and computer) being TCP, IPv4, and HTTP for a single TCP frame connected to oldweb.today



3) Try to get the screenshots of your http messages with both GET and OK for one of the requested services and measure the time difference between those messages. (To display the Time field in time-of-day format, select the Wireshark View pull down menu, then select Time Display Format, then select Time-of-day.

It took less than .06 milliseconds for the messages to transfer.

```
2345 2018-08-28 21:35:39.688789913 192.168.0.12   52.22.31.4     HTTP   422 GET / HTTP/1.1
2357 2018-08-28 21:35:39.748914842 52.22.31.4     192.168.0.12   HTTP   244 HTTP/1.1 200 OK  (text/html)
```

GET request packet details.

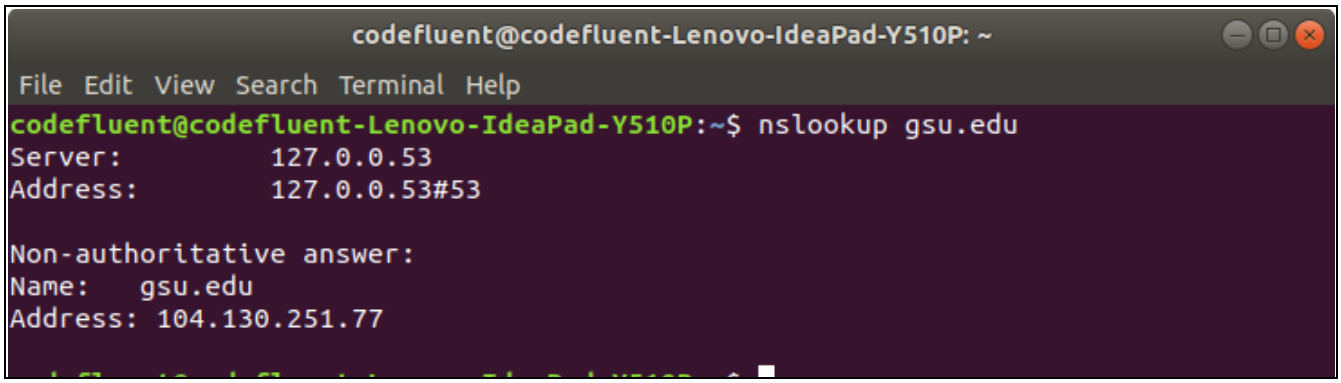
```
▸ Frame 2345: 422 bytes on wire (3376 bits), 422 bytes captured (3376 bits) on interface 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 192.168.0.12, Dst: 52.22.31.4
▸ Transmission Control Protocol, Src Port: 48602, Dst Port: 80, Seq: 1, Ack: 1, Len: 354
▾ Hypertext Transfer Protocol
  ▾ GET / HTTP/1.1\r\n
    ▾ [Expert Info (Chat/Sequence): GET / HTTP/1.1\r\n]
        [GET / HTTP/1.1\r\n]
        [Severity level: Chat]
        [Group: Sequence]
      Request Method: GET
      Request URI: /
      Request Version: HTTP/1.1
    Host: oldweb.today\r\n
    User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:59.0) Gecko/20100101 Firefox/59.0\r\n
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\n
    Accept-Language: en-US,en;q=0.5\r\n
    Accept-Encoding: gzip, deflate\r\n
    Referer: https://www.google.com/\r\n
    Connection: keep-alive\r\n
    Upgrade-Insecure-Requests: 1\r\n
    \r\n
    [Full request URI: http://oldweb.today/]
    [HTTP request 1/4]
    [Response in frame: 2357]
    [Next request in frame: 2359]
```

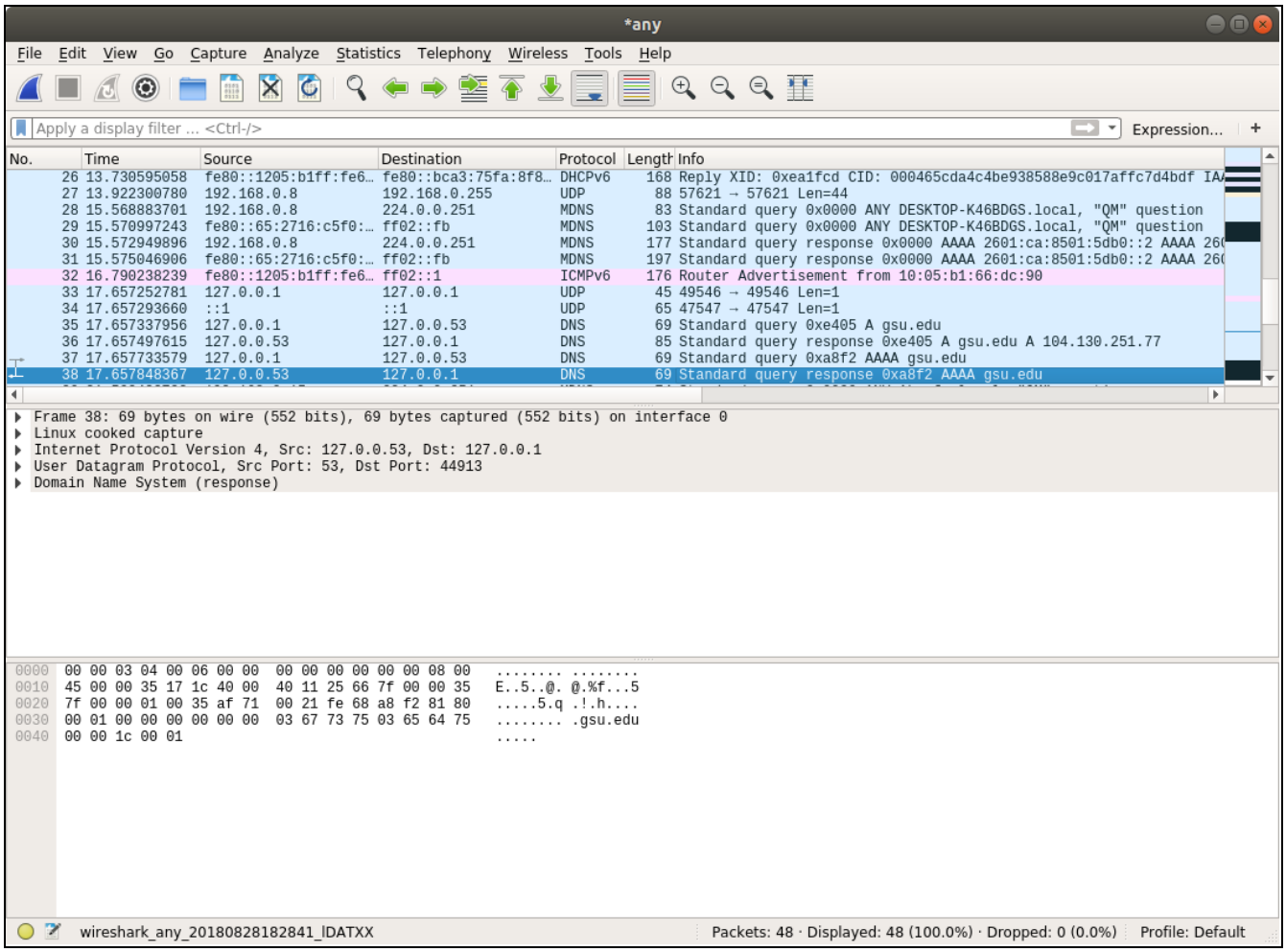
OK response packet details and HTML payload.

Wireshark · Packet 2357 · wireshark_any_20180828213307_iPvW97

```
▸ Internet Protocol Version 4, Src: 52.22.31.4, Dst: 192.168.0.12
▸ Transmission Control Protocol, Src Port: 80, Dst Port: 48602, Seq: 14481, Ack: 355, Len: 176
▸ [6 Reassembled TCP Segments (14656 bytes): #2347(2896), #2349(1448), #2351(2896), #2353(1448), #2355(5792), #2357(176)]
▾ Hypertext Transfer Protocol
  ▾ HTTP/1.1 200 OK\r\n
    ▾ [Expert Info (Chat/Sequence): HTTP/1.1 200 OK\r\n]
        [HTTP/1.1 200 OK\r\n]
        [Severity level: Chat]
        [Group: Sequence]
      Request Version: HTTP/1.1
      Status Code: 200
      [Status Code Description: OK]
      Response Phrase: OK
    Server: nginx/1.15.2\r\n
    Date: Wed, 29 Aug 2018 01:35:39 GMT\r\n
    Content-Type: text/html; charset=UTF-8\r\n
  ▸ Content-Length: 14491\r\n
    Connection: keep-alive\r\n
    \r\n
    [HTTP response 1/4]
    [Time since request: 0.060124929 seconds]
    [Request in frame: 2345]
    [Next request in frame: 2359]
    [Next response in frame: 2449]
    File Data: 14491 bytes
▾ Line-based text data: text/html
    <!doctype html>\n
    <head>\n
        <meta charset="utf-8">\n
        \n
        <title>oldweb.today</title>\n
        \n
            <script>\n
          (function(i,s,o,g,r,a,m){i['GoogleAnalyticsObject']=r;i[r]=i[r]||function(){\n
          (i[r].q=i[r].q||[]).push(arguments)},i[r].l=1*new Date();a=s.createElement(o),\n
          m=s.getElementsByTagName(o)[0];a.async=1;a.src=g;m.parentNode.insertBefore(a,m)\n
          })(window,document,'script','//www.google-analytics.com/analytics.js','ga');\n
    \n
          \n
          \n
          ga('create', 'UA-768502-10', 'auto');\n
          ga('send', 'pageview');\n
          \n
```

4) Do packet capture in Wireshark and make a DNS query with nslookup in command-prompt like the one that has been mentioned above and provide the packet transfer screenshot for wireshark and DNS query in the command-prompt.



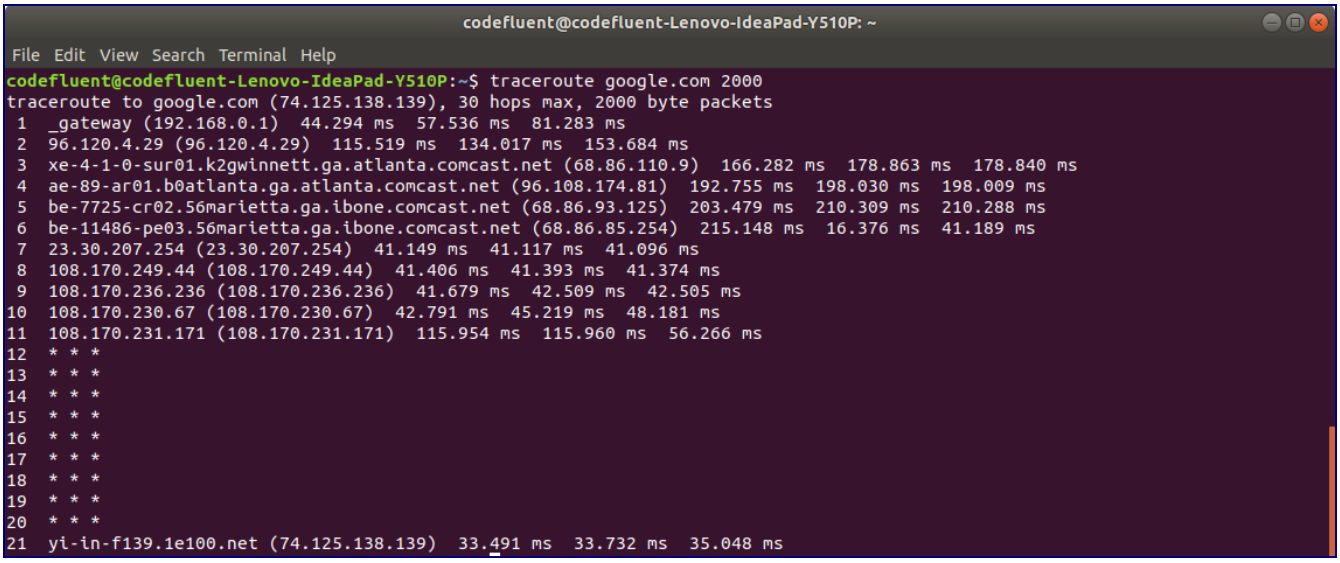


(Answers 5,6,7,8)

9 pm (5a)

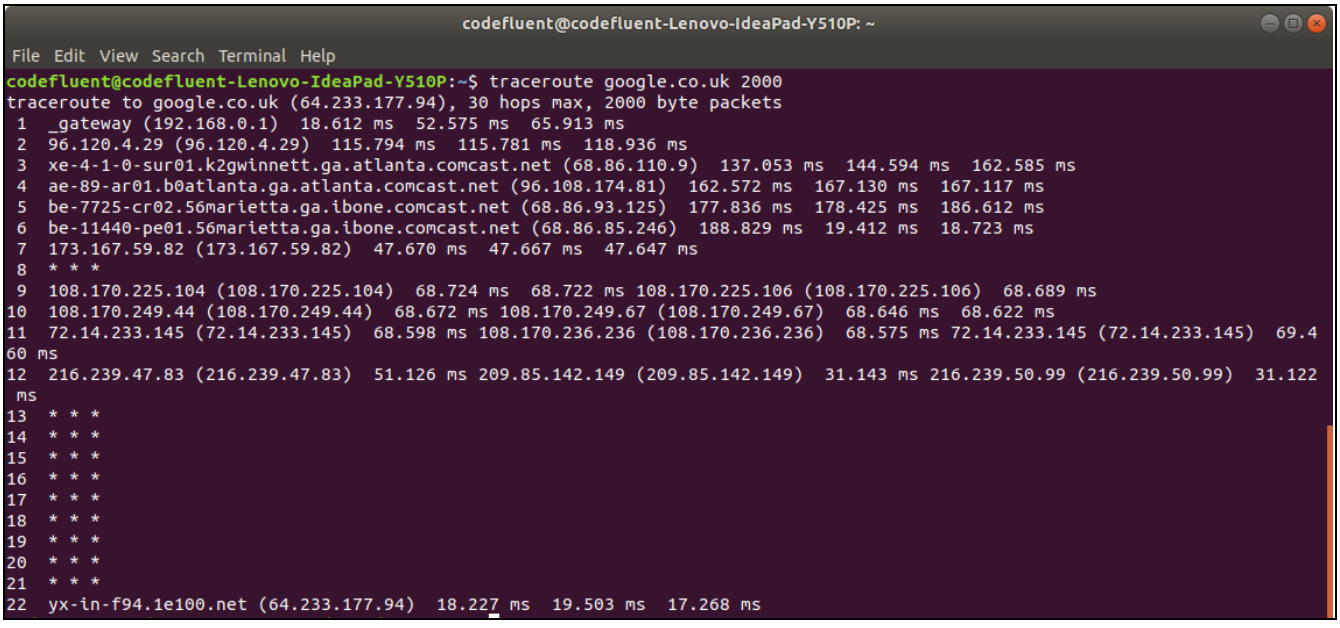
US – google.com

21 routers (6a)
The 5th router required 210.288ms round trip time as the largest delay. (7a



UK – google.co.uk

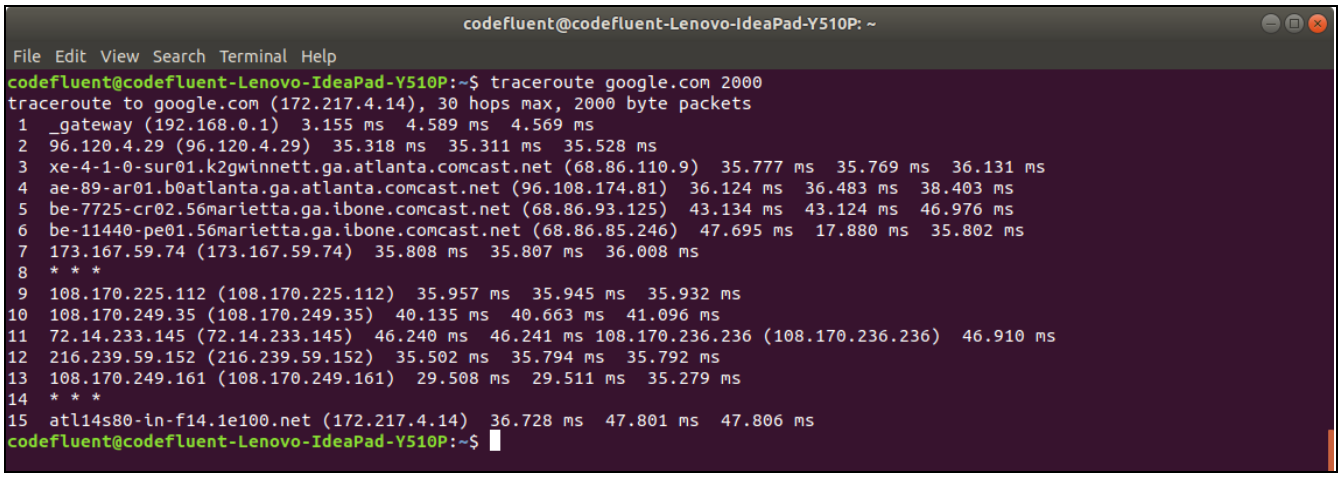
22 routers (6a)
5th router had a RTT of 186.612ms (7a)



10pm (5b)

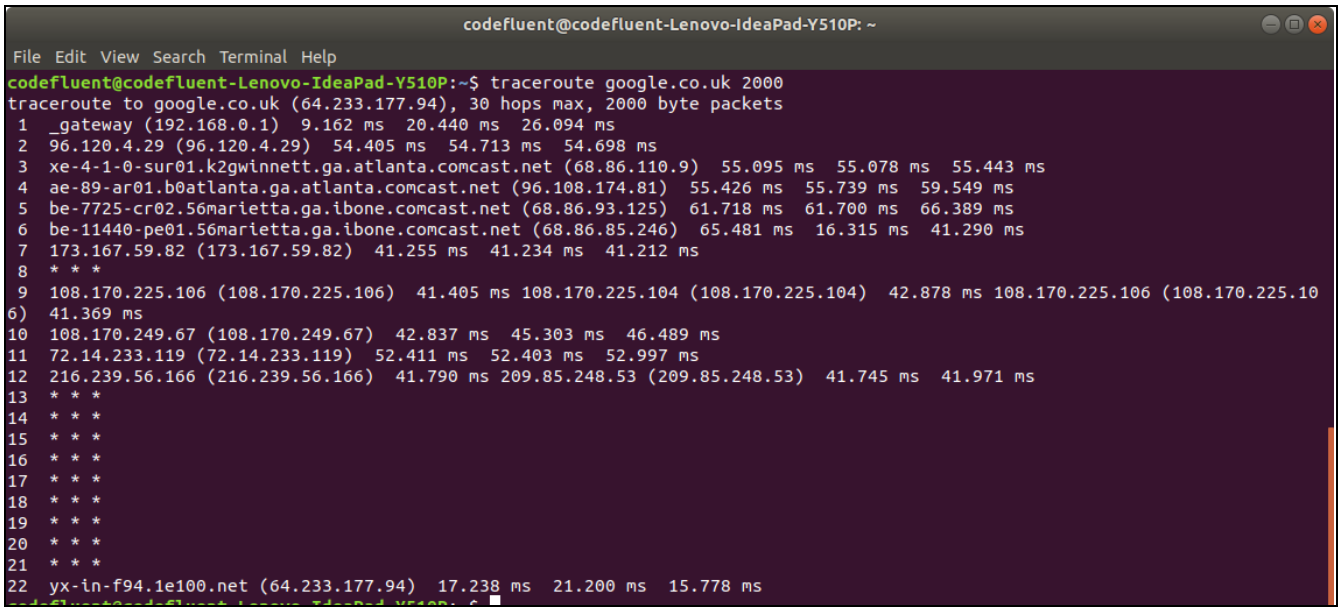
US – google.com

15 routers (6b)
Again the 5th router had a time of 46.976ms RTT for delay. (7b)



UK – google.co.uk

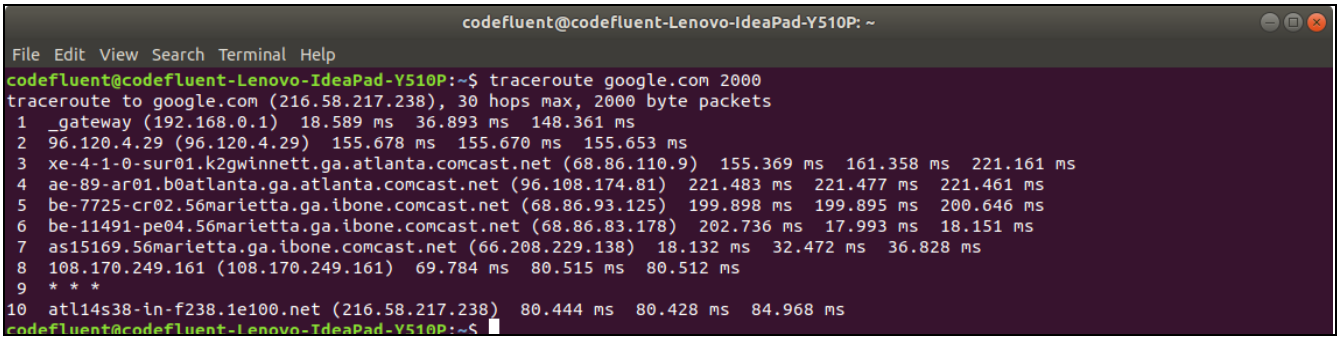
22 routers (6b)
The 5th router had a time of 66.389ms RTT for delay. (7b)



11pm (5c)

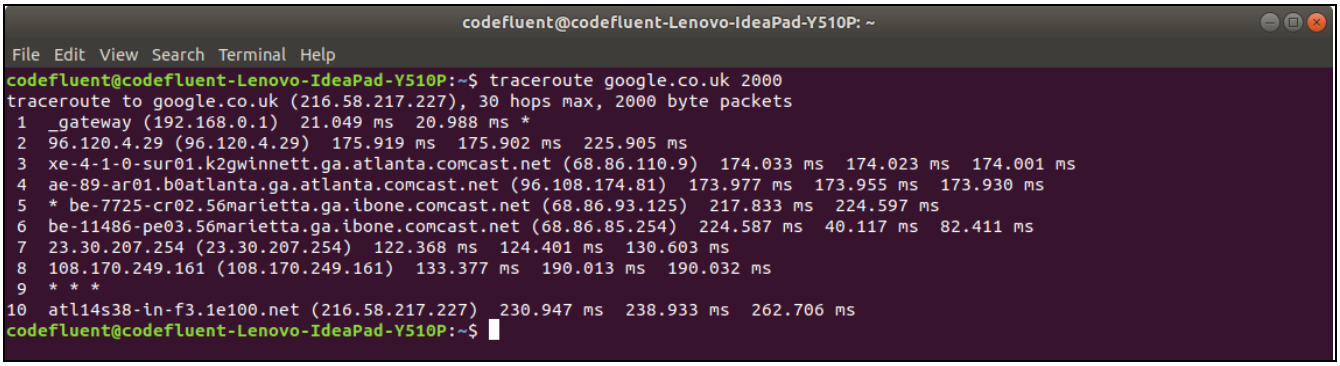
US – google.com

10 routers (6c)
The 4th router with Comcast ISP took the longest round trip time with 221.461 ms. (7c)

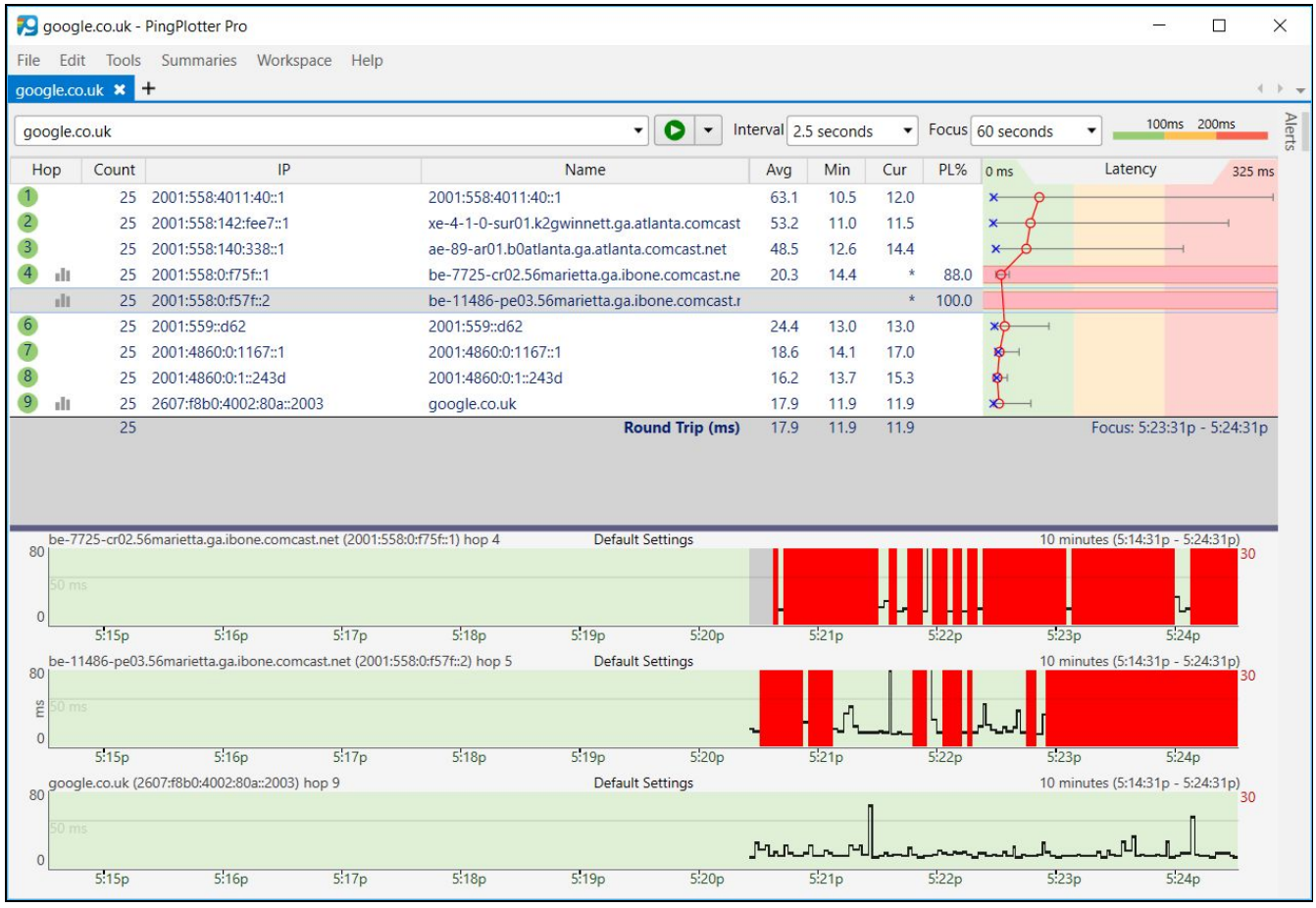


UK – google.co.uk

10 routers (6c) **I did the command a few times, probably why its taking so little hops.
1E100.net is a google domain that took the longest round trip time with 262.706 ms. (7c)



9) Observe any packet-loss with any router in between source and destination and screenshot them like the one in the above in information section.



10) Briefly give an example for each of the delay in the network that have been mentioned in the class.

queuing delay: a packet for a video is being streamed onto the network, but the router has yet to process the first few frames of the video. the next packets containing the next frames of the video must wait till the previous ones are processed.

transmission delay: a packet is being sent out by a router, but the packet has a very large HTML payload so it takes some time for the router to send it out.

nodal processing delay: the router is processing some packets sees that the destination address is not within its routing table, so it must delay sending the packets until it can locate the destination via a broadcast.

propagation delay: a packet is being sent out by a router, but an IT person used a bad wire for the link so it'll take longer than usual to send the packet due to electronic interference.