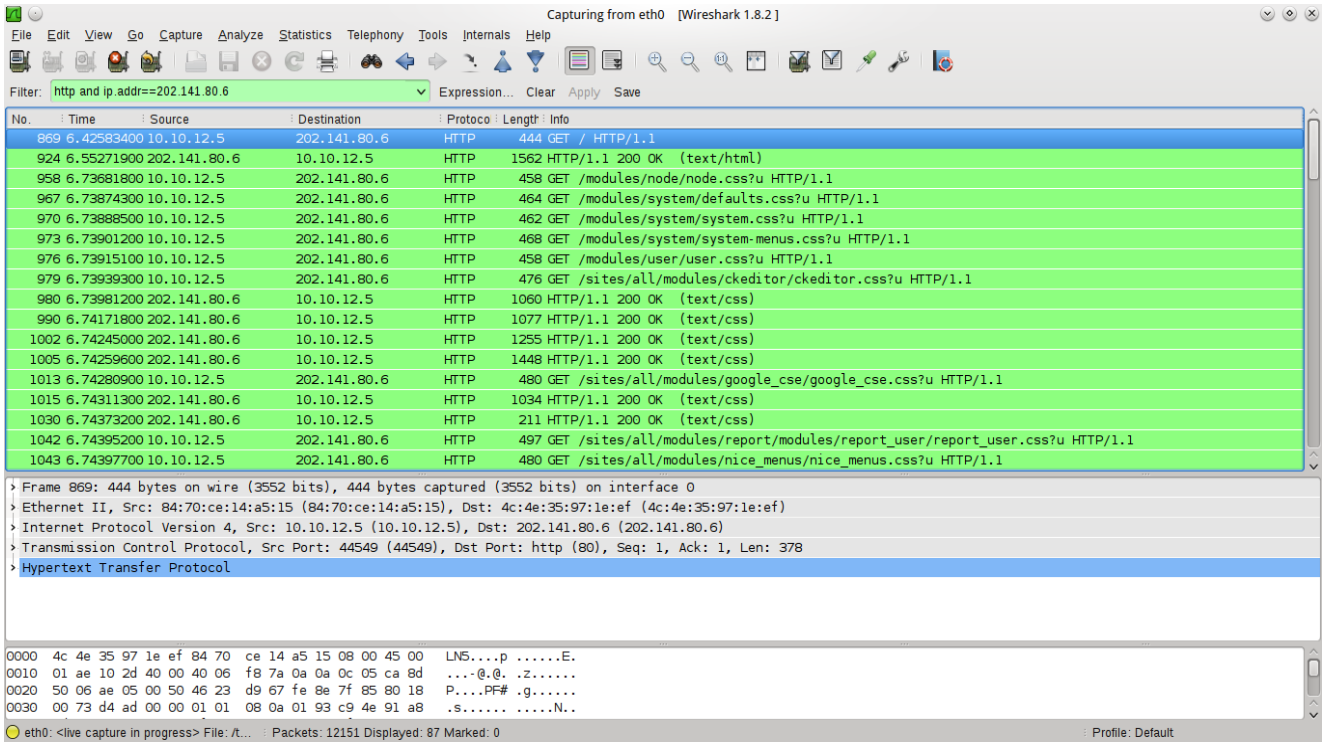
Assignment 4: Packet Capturing Using Wireshark

Simrat Singh Chhabra 11010165

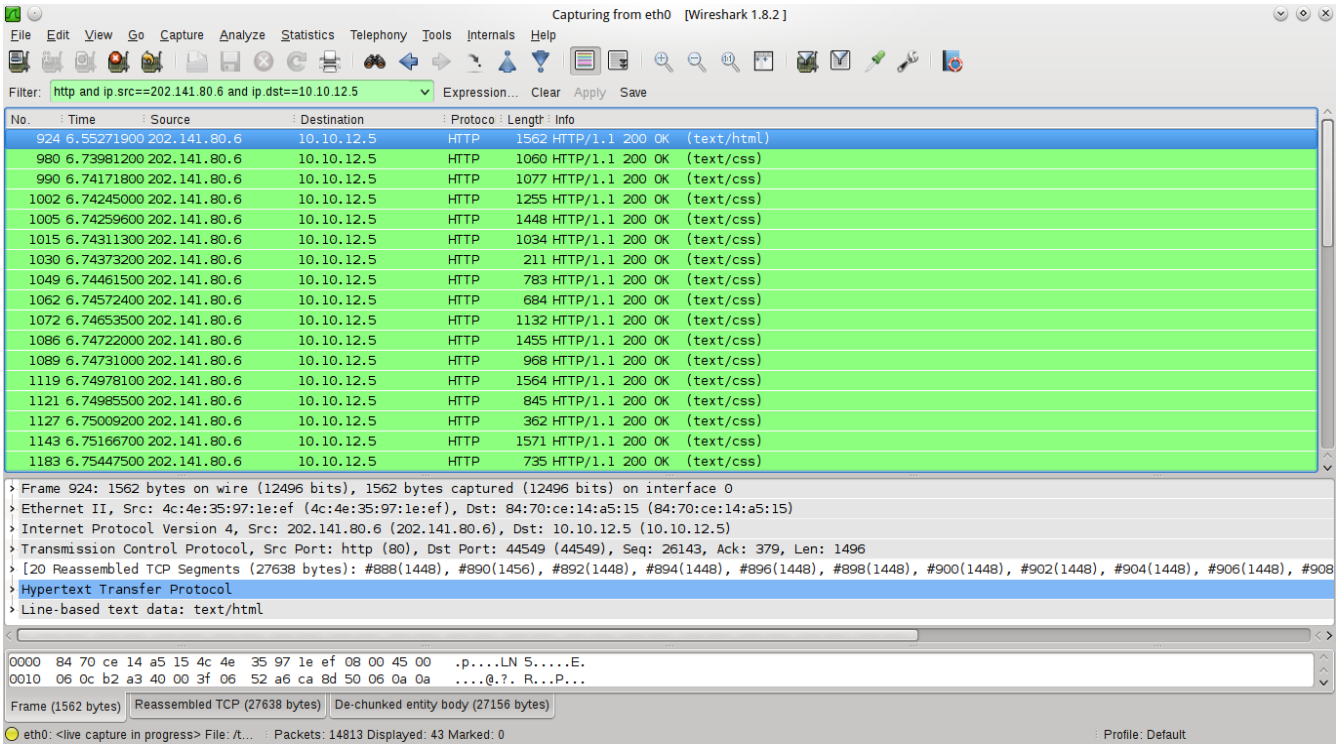
N M Harsha 11010144

Part A : Initial

7. Take a screenshot of this result. How many packets were transmitted from the IITG web server to your client in this.



Packets belonging to the http session between host and IITG web server



Packets sent from IITG web server to client

43 packets were transmitted from IITG web server to client.

Part B : HTTP

(Note : Text enclosed by [ ] denotes the source of the information in the answer)

I. The Basic HTTP GET/response Interaction

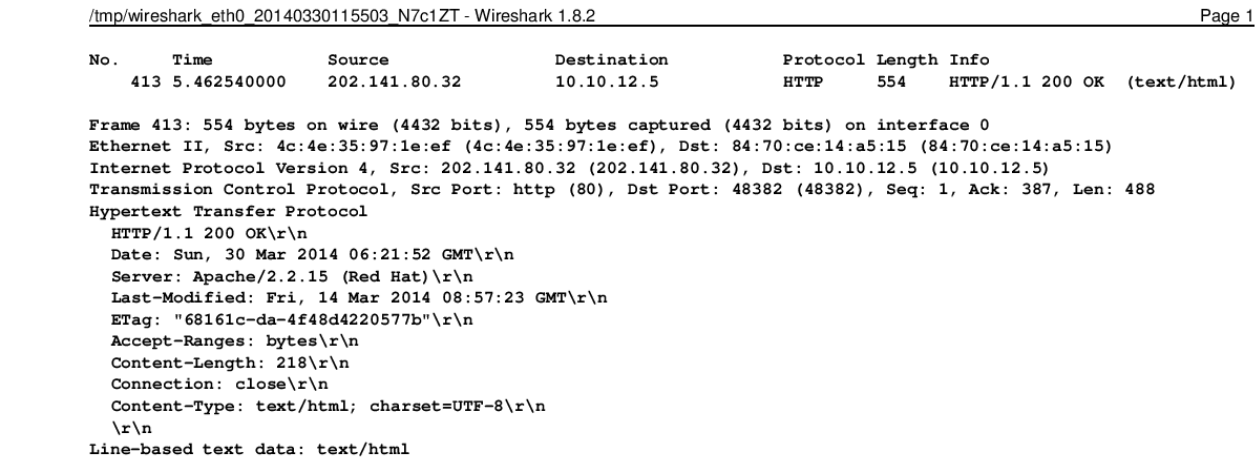
Request

```
/tmp/wireshark_eth0_20140330115503_N7c1ZT - Wireshark 1.8.2                    Page 1

No.     Time           Source                Destination           Protocol Length Info
    411 5.461417000    10.10.12.5            202.141.80.32         HTTP     452    GET /~sukumar/cs349/HTTP-wire
shark-file1.html HTTP/1.1

Frame 411: 452 bytes on wire (3616 bits), 452 bytes captured (3616 bits) on interface 0
Ethernet II, Src: 84:70:ce:14:a5:15 (84:70:ce:14:a5:15), Dst: 4c:4e:35:97:1e:ef (4c:4e:35:97:1e:ef)
Internet Protocol Version 4, Src: 10.10.12.5 (10.10.12.5), Dst: 202.141.80.32 (202.141.80.32)
Transmission Control Protocol, Src Port: 48382 (48382), Dst Port: http (80), Seq: 1, Ack: 1, Len: 386
Hypertext Transfer Protocol
  GET /~sukumar/cs349/HTTP-wireshark-file1.html HTTP/1.1\r\n
  Host: 202.141.80.32\r\n
  Connection: keep-alive\r\n
  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8\r\n
  User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/32.0.1700.77 Safari/53
  7.36\r\n
  Accept-Encoding: gzip,deflate,sdch\r\n
  Accept-Language: en-GB,en;q=0.8,en-US;q=0.6\r\n
  \r\n
  [Full request URI: http://202.141.80.32/~sukumar/cs349/HTTP-wireshark-file1.html]
```

Response



```
/tmp/wireshark_eth0_20140330115503_N7c1ZT - Wireshark 1.8.2                                    Page 1

No.     Time           Source                Destination           Protocol Length Info
    413 5.462540000    202.141.80.32         10.10.12.5            HTTP     554    HTTP/1.1 200 OK  (text/html)

Frame 413: 554 bytes on wire (4432 bits), 554 bytes captured (4432 bits) on interface 0
Ethernet II, Src: 4c:4e:35:97:1e:ef (4c:4e:35:97:1e:ef), Dst: 84:70:ce:14:a5:15 (84:70:ce:14:a5:15)
Internet Protocol Version 4, Src: 202.141.80.32 (202.141.80.32), Dst: 10.10.12.5 (10.10.12.5)
Transmission Control Protocol, Src Port: http (80), Dst Port: 48382 (48382), Seq: 1, Ack: 387, Len: 488
Hypertext Transfer Protocol
  HTTP/1.1 200 OK\r\n
  Date: Sun, 30 Mar 2014 06:21:52 GMT\r\n
  Server: Apache/2.2.15 (Red Hat)\r\n
  Last-Modified: Fri, 14 Mar 2014 08:57:23 GMT\r\n
  ETag: "68161c-da-4f48d4220577b"\r\n
  Accept-Ranges: bytes\r\n
  Content-Length: 218\r\n
  Connection: close\r\n
  Content-Type: text/html; charset=UTF-8\r\n
  \r\n
Line-based text data: text/html
```

1. Is your browser running HTTP version 1.0 or 1.1? What version of HTTP is the server running?

Ans : My browser is running HTTP version 1.1.
[GET /~sukumar/cs349/HTTP-wireshark-file1.html **HTTP/1.1**\r\n]

The server is also running HTTP 1.1.
[**HTTP/1.1** 200 OK\r\n]

2. What languages (if any) does your browser indicate that it can accept to the server?

Ans : The browser indicates that it can accept the following languages :
English (United Kingdom), English, English (United States)
[Accept-Language: **en-GB**,**en**;q=0.8,**en-US**;q=0.6\r\n]

3. What is the IP address of your computer?

Ans : The IP address of my computer is 10.10.12.5
[Internet Protocol Version 4, **Src: 10.10.12.5 (10.10.12.5),** Dst: 202.141.80.32 (202.141.80.32)]

4. What is the status code returned from the server to your browser?

Ans: The status returned from the server to my browser is 200 OK which means *'The request has succeeded.'*
[HTTP/1.1 **200 OK**\r\n]
5. When was the HTML file that you are retrieving last modified at the server?

Ans : The HTML file was last modified on Fri, 14 Mar 2014 08:57:23 GMT**.**

[Last-Modified: **Fri, 14 Mar 2014 08:57:23 GMT**\r\n]

6. How many bytes of content are being returned to your browser?

Ans : 218 bytes are returned to the browser.
[Content-Length: **218**\r\n]

7. By inspecting the raw data in the packet content window, do you see any headers within the data that are not displayed in the packet-listing window? If so, name one.

Ans : No, all of the headers within the the data are displayed in the packet-listing window.

II. The HTTP CONDITIONAL GET/response Interaction

First GET Message


```
No.     Time           Source                Destination           Protocol Length Info
    176 3.841193000    10.10.12.5            202.141.80.32         HTTP     397    GET /~sukumar/cs349/HTTP-wire
shark-file2.html HTTP/1.1

Frame 176: 397 bytes on wire (3176 bits), 397 bytes captured (3176 bits) on interface 0
Ethernet II, Src: 84:70:ce:14:a5:15 (84:70:ce:14:a5:15), Dst: 4c:4e:35:97:1e:ef (4c:4e:35:97:1e:ef)
Internet Protocol Version 4, Src: 10.10.12.5 (10.10.12.5), Dst: 202.141.80.32 (202.141.80.32)
Transmission Control Protocol, Src Port: 50260 (50260), Dst Port: http (80), Seq: 1, Ack: 1, Len: 331
Hypertext Transfer Protocol
  GET /~sukumar/cs349/HTTP-wireshark-file2.html HTTP/1.1\r\n
    [Expert Info (Chat/Sequence): GET /~sukumar/cs349/HTTP-wireshark-file2.html HTTP/1.1\r\n]
    Request Method: GET
    Request URI: /~sukumar/cs349/HTTP-wireshark-file2.html
    Request Version: HTTP/1.1
  Host: 202.141.80.32\r\n
  User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:26.0) Gecko/20100101 Firefox/26.0\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
  Connection: keep-alive\r\n
  \r\n
  [Full request URI: http://202.141.80.32/~sukumar/cs349/HTTP-wireshark-file2.html]
```

First Response


```
No.     Time           Source                Destination           Protocol Length Info
    178 3.842201000    202.141.80.32         10.10.12.5            HTTP     745    HTTP/1.1 200 OK  (text/html)

Frame 178: 745 bytes on wire (5960 bits), 745 bytes captured (5960 bits) on interface 0
Ethernet II, Src: 4c:4e:35:97:1e:ef (4c:4e:35:97:1e:ef), Dst: 84:70:ce:14:a5:15 (84:70:ce:14:a5:15)
Internet Protocol Version 4, Src: 202.141.80.32 (202.141.80.32), Dst: 10.10.12.5 (10.10.12.5)
Transmission Control Protocol, Src Port: http (80), Dst Port: 50260 (50260), Seq: 1, Ack: 332, Len: 679
Hypertext Transfer Protocol
  HTTP/1.1 200 OK\r\n
  Date: Sun, 30 Mar 2014 07:12:26 GMT\r\n
  Server: Apache/2.2.15 (Red Hat)\r\n
  Last-Modified: Fri, 14 Mar 2014 08:57:23 GMT\r\n
  ETag: "68161d-198-4f48d4220865c"\r\n
  Accept-Ranges: bytes\r\n
  Content-Length: 408\r\n
  Connection: close\r\n
  Content-Type: text/html; charset=UTF-8\r\n
  \r\n
Line-based text data: text/html
  <html><head>\n
  <meta http-equiv="content-type" content="text/html; charset=UTF-8"></head><body>Congratulations again!  Now you'
  ve downloaded the file lab2-2.html. <br>\n
   <p>\n
  If you download this multiple times on your browser, a complete copy <br>\n
  will only be sent once by the server due to the inclusion of the IN-MODIFIED-SINCE<br>\n
  field in your browser's HTTP GET request to the server.\n
  \n
  \n
  </p></body></html>
```

Second GET Message


```
No.     Time           Source                Destination           Protocol Length Info
   1581 21.400188000   10.10.12.5            202.141.80.32         HTTP     516    GET /~sukumar/cs349/HTTP-wire
shark-file2.html HTTP/1.1

Frame 1581: 516 bytes on wire (4128 bits), 516 bytes captured (4128 bits) on interface 0
Ethernet II, Src: 84:70:ce:14:a5:15 (84:70:ce:14:a5:15), Dst: 4c:4e:35:97:1e:ef (4c:4e:35:97:1e:ef)
Internet Protocol Version 4, Src: 10.10.12.5 (10.10.12.5), Dst: 202.141.80.32 (202.141.80.32)
Transmission Control Protocol, Src Port: 50262 (50262), Dst Port: http (80), Seq: 1, Ack: 1, Len: 450
Hypertext Transfer Protocol
  GET /~sukumar/cs349/HTTP-wireshark-file2.html HTTP/1.1\r\n
  Host: 202.141.80.32\r\n
  User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:26.0) Gecko/20100101 Firefox/26.0\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
  Connection: keep-alive\r\n
  If-Modified-Since: Fri, 14 Mar 2014 08:57:23 GMT\r\n
  If-None-Match: "68161d-198-4f48d4220865c"\r\n
  Cache-Control: max-age=0\r\n
  \r\n
  [Full request URI: http://202.141.80.32/~sukumar/cs349/HTTP-wireshark-file2.html]
```

Second Response


```
No.     Time           Source                Destination           Protocol Length Info
   1583 21.400992000   202.141.80.32         10.10.12.5            HTTP     218    HTTP/1.1 304 Not Modified

Frame 1583: 218 bytes on wire (1744 bits), 218 bytes captured (1744 bits) on interface 0
Ethernet II, Src: 4c:4e:35:97:1e:ef (4c:4e:35:97:1e:ef), Dst: 84:70:ce:14:a5:15 (84:70:ce:14:a5:15)
Internet Protocol Version 4, Src: 202.141.80.32 (202.141.80.32), Dst: 10.10.12.5 (10.10.12.5)
Transmission Control Protocol, Src Port: http (80), Dst Port: 50262 (50262), Seq: 1, Ack: 451, Len: 152
Hypertext Transfer Protocol
  HTTP/1.1 304 Not Modified\r\n
  Date: Sun, 30 Mar 2014 07:12:43 GMT\r\n
  Server: Apache/2.2.15 (Red Hat)\r\n
  Connection: close\r\n
  ETag: "68161d-198-4f48d4220865c"\r\n
  \r\n
```

1. Inspect the contents of the first HTTP GET request from your browser to the server. Do you see an “IF-MODIFIED-SINCE” line in the HTTP GET?

Ans : No, there is no “IF-MODIFIED-SINCE” line in the first HTTP GET request.

2. Inspect the contents of the server response. Did the server explicitly return the contents of the file? How can you tell?

Ans : Yes. The server returned the contents of the file.
We can see this here:

```
Line-based text data: text/html
  <html><head>\n
  <meta http-equiv="content-type" content="text/html; charset=UTF-8"></head><body>Congratulations again!  Now you'
  ve downloaded the file lab2-2.html. <br>\n
   <p>\n
  If you download this multiple times on your browser, a complete copy <br>\n
  will only be sent once by the server due to the inclusion of the IN-MODIFIED-SINCE<br>\n
  field in your browser's HTTP GET request to the server.\n
  \n
  \n
  </p></body></html>
```

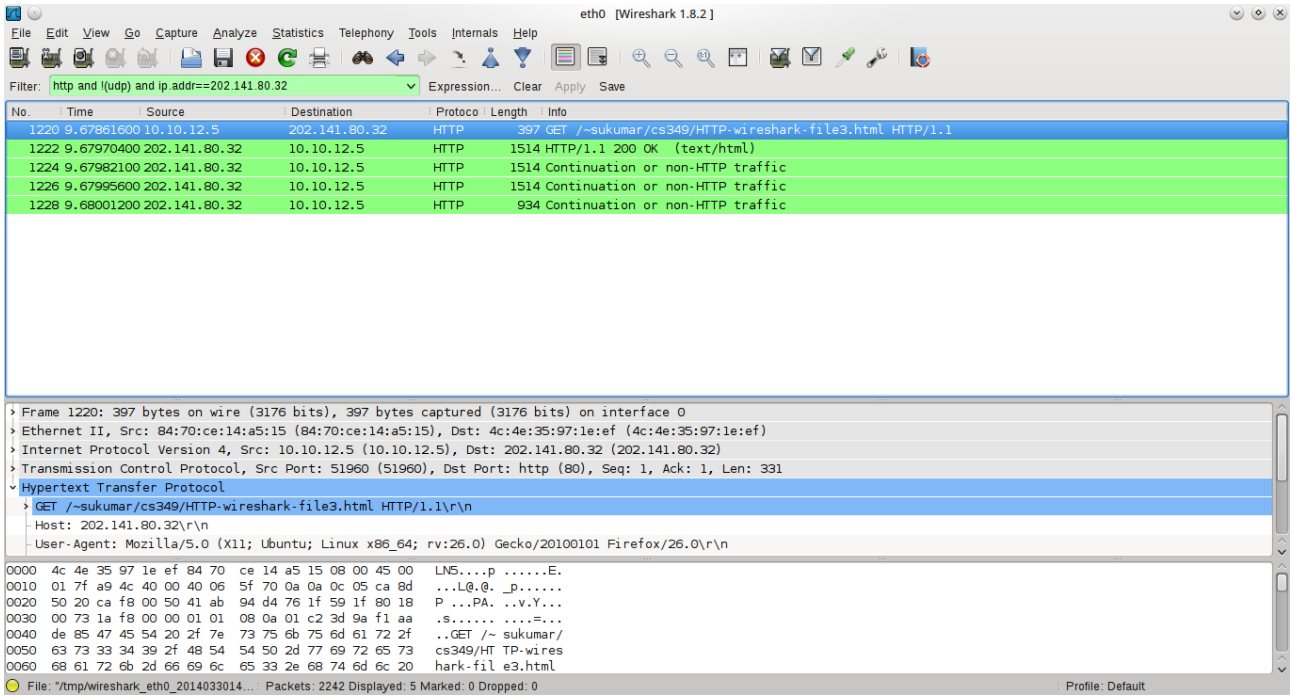
3. Now inspect the contents of the second HTTP GET request from your browser to the server. Do you see an “IF-MODIFIED-SINCE:” line in the HTTP GET? If so, what information follows the “IF-MODIFIED-SINCE:” header?

Ans : Yes, there is an “IF-MODIFIED-SINCE:” line in the second HTTP GET request.
[If-Modified-Since: Fri, 14 Mar 2014 08:57:23 GMT\r\n]
It tells the server that it should only return the contents of the file again if they have changed since Fri, 14 Mar 2014 08:57:23 GMT. (The last modified time of the file).

4. What is the HTTP status code and phrase returned from the server in response to this second HTTP GET? Did the server explicitly return the contents of the file? Explain.

Ans : The server returned the HTTP status code 304 along with the phrase Not Modified. [HTTP/1.1 304 Not Modified\r\n]
The contents of the file were not explicitly returned.
The reason for this is that the second GET request was a conditional GET request with the header If-Modified-Since. As the contents of the file on the server has not been modified after Fri, 14 Mar 2014 08:57:23 GMT, the server returned a Not Modified response to indicate to the client that the file had not been modified and hence sending the contents again was not required.

III. Retrieving Long Documents



1. How many HTTP GET request messages were sent by your browser?

Ans : **1** HTTP GET request message was sent by my browser.

2. How many data-containing TCP segments were needed to carry the single HTTP response?

Ans : **4** TCP segments were needed to carry the single HTTP response.

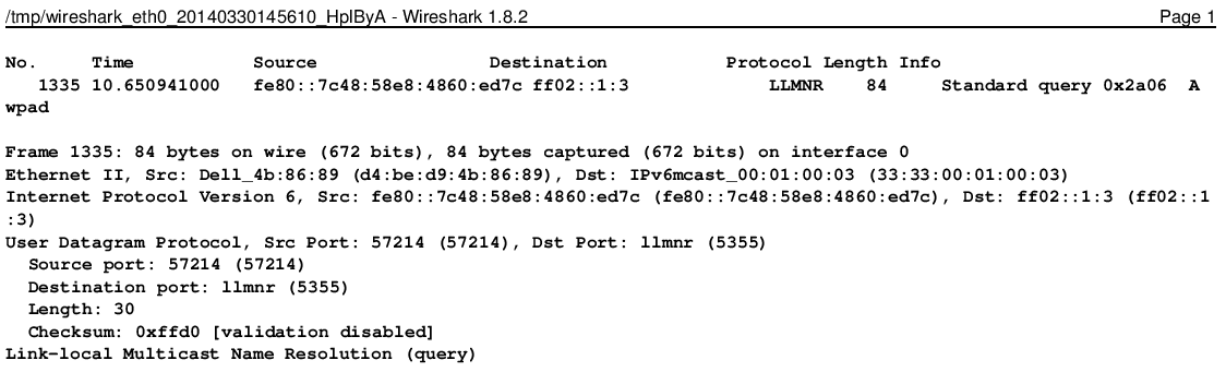
3. What is the status code and phrase associated with the response to the HTTP GET request?

Ans : The status code is 200 and the associated phrase is OK.
[HTTP/1.1 **200 OK**\r\n]

4. Are there any HTTP status lines in the transmitted data associated with a TCP-induced "Continuation"?

Ans : **No**, there is no associated HTTP status line associated with the TCP-induced "Continuation".

Part C: UDP



/tmp/wireshark_eth0_20140330145610_HplByA - Wireshark 1.8.2 Page 1

```
No.     Time           Source                Destination           Protocol Length Info
   1335 10.650941000   fe80::7c48:58e8:4860:ed7c ff02::1:3            LLMNR    84     Standard query 0x2a06  A
wpad

Frame 1335: 84 bytes on wire (672 bits), 84 bytes captured (672 bits) on interface 0
Ethernet II, Src: Dell_4b:86:89 (d4:be:d9:4b:86:89), Dst: IPv6mcast_00:01:00:03 (33:33:00:01:00:03)
Internet Protocol Version 6, Src: fe80::7c48:58e8:4860:ed7c (fe80::7c48:58e8:4860:ed7c), Dst: ff02::1:3 (ff02::1
:3)
User Datagram Protocol, Src Port: 57214 (57214), Dst Port: llmnr (5355)
  Source port: 57214 (57214)
  Destination port: llmnr (5355)
  Length: 30
  Checksum: 0xffd0 [validation disabled]
Link-local Multicast Name Resolution (query)
```

a. Select one packet. From this packet, determine how many fields there are in the UDP header. (Do not look in the textbook! Answer these questions directly from what you observe in the packet trace.) Name these fields as they are named in the Wireshark display of segment fields.

Ans: There are **4** fields in the UDP header. These are:
[1] Source Port
[2] Destination Port
[3] Length
[4] Checksum

b. What are the source and destination port numbers, in both decimal and hexadecimal format. (Hint: the hexadecimal format is given in the data in the bottommost panel in the wireshark display, and so it's easier just to read it out from there rather than converting the decimal number to hex).

Ans : Source port number -> Decimal : **57214** Hexadecimal : **df7e**
Destination port number -> Decimal : **53555** Hexadecimal : **14eb**

c. What is the value in the Length field in both decimal and hexadecimal format. What is the meaning of this value (i.e., this value is the length of what?)

Ans : Length -> Decimal : **30** Hexadecimal : **1e**

The length field specifies the length of the UDP header and the UDP data (in bytes).

d. What is the protocol number for UDP? Give your answer in both hexadecimal and decimal notation. (To answer this question, you'll need to look into the IP header.)

Ans :

```
Internet Protocol Version 6, Src: fe80::7c48:58e8:4860:ed7c (fe80::7c48:58e8:4860:ed7c), Dst: ff02::1:3 (ff02::1
:3)
  0110 .... = Version: 6
  .... 0000 0000 .... .... .... .... .... = Traffic class: 0x00000000
  .... .... .... 0000 0000 0000 0000 0000 = Flowlabel: 0x00000000
  Payload length: 30
  Next header: UDP (17)
  Hop limit: 1
  Source: fe80::7c48:58e8:4860:ed7c (fe80::7c48:58e8:4860:ed7c)
  Destination: ff02::1:3 (ff02::1:3)
  [Source GeoIP: Unknown]
  [Destination GeoIP: Unknown]
```

Protocol Number for UDP -> Decimal : **17** Hexadecimal : **11**

e. Examine a pair of UDP packets in which the first packet is sent by your host and the second packet is a reply to the first packet. Describe the relationship between the port numbers in the two packets.

Ans :

Query


```
No.     Time           Source                Destination           Protocol Length Info
   1519 13.224755000   10.10.12.5            202.141.81.2          DNS      74     Standard query 0x2ad4  A www.
google.com

Frame 1519: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
Ethernet II, Src: 84:70:ce:14:a5:15 (84:70:ce:14:a5:15), Dst: 4c:4e:35:97:1e:ef (4c:4e:35:97:1e:ef)
Internet Protocol Version 4, Src: 10.10.12.5 (10.10.12.5), Dst: 202.141.81.2 (202.141.81.2)
User Datagram Protocol, Src Port: 26492 (26492), Dst Port: domain (53)
  Source port: 26492 (26492)
  Destination port: domain (53)
  Length: 40
  Checksum: 0xacd2 [validation disabled]
Domain Name System (query)
```

Response

```
No.     Time           Source                Destination           Protocol Length Info
   1520 13.230077000   202.141.81.2          10.10.12.5            DNS      74     Standard query response 0x2ad
4 Server failure

Frame 1520: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
Ethernet II, Src: 4c:4e:35:97:1e:ef (4c:4e:35:97:1e:ef), Dst: 84:70:ce:14:a5:15 (84:70:ce:14:a5:15)
Internet Protocol Version 4, Src: 202.141.81.2 (202.141.81.2), Dst: 10.10.12.5 (10.10.12.5)
User Datagram Protocol, Src Port: domain (53), Dst Port: 26492 (26492)
  Source port: domain (53)
  Destination port: 26492 (26492)
  Length: 40
  Checksum: 0x2c50 [validation disabled]
Domain Name System (response)
```

Query == Source Port -> **26492**, Dest Port -> **53**
Response == Source Port -> **53**, Dest Port -> **26492**

Therefore, we observe that the source port of the query packet is the destination port for the response packet and the destination port of the query packet is the source port of the response packet.

Part D: TCP

(Note : Text enclosed by [ ] denotes the source of the information in the answer.)

1.Print out a captured packet and indicate where you see the information that answers the following:


```
No.     Time           Source                Destination           Protocol Length Info
    311 2.178592000    10.10.12.5            202.141.80.32         TCP      74     49481 > http [SYN] Seq=200552
7222 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=228585 TSecr=0 WS=128

Frame 311: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
Ethernet II, Src: 84:70:ce:14:a5:15 (84:70:ce:14:a5:15), Dst: 4c:4e:35:97:1e:ef (4c:4e:35:97:1e:ef)
Internet Protocol Version 4, Src: 10.10.12.5 (10.10.12.5), Dst: 202.141.80.32 (202.141.80.32)
Transmission Control Protocol, Src Port: 49481 (49481), Dst Port: http (80), Seq: 2005527222, Len: 0
  Source port: 49481 (49481)
  Destination port: http (80)
  [Stream index: 5]
  Sequence number: 2005527222
  Header length: 40 bytes
  Flags: 0x002 (SYN)
  Window size value: 14600
  [Calculated window size: 14600]
  Checksum: 0x3d75 [validation disabled]
  Options: (20 bytes), Maximum segment size, SACK permitted, Timestamps, No-Operation (NOP), Window scale
```

a) What is the IP address and TCP port number used by your client computer (source) to transfer the file to 202.141.80.32?

Ans : IP address -> 10.10.12.5
[Internet Protocol Version 4, Src: **10.10.12.5** (10.10.12.5), Dst: 202.141.80.32 (202.141.80.32)]

TCP port number -> 55382
[Source port: **55382** (55382)]

b) What is the IP address and TCP port number used by the server?

Ans: IP address -> 202.141.80.32
[Internet Protocol Version 4, Src: 10.10.12.5 (10.10.12.5), Dst: **202.141.80.32** (202.141.80.32)]
TCP port number -> 80
[Destination port: http (**80**)]
2.Print out a captured packet and indicate where you see the information that answers the following:


```
No.     Time           Source                Destination           Protocol Length Info
    311 2.178592000    10.10.12.5            202.141.80.32         TCP      74     49481 > http [SYN] Seq=200552
7222 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=228585 TSecr=0 WS=128

Frame 311: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
Ethernet II, Src: 84:70:ce:14:a5:15 (84:70:ce:14:a5:15), Dst: 4c:4e:35:97:1e:ef (4c:4e:35:97:1e:ef)
Internet Protocol Version 4, Src: 10.10.12.5 (10.10.12.5), Dst: 202.141.80.32 (202.141.80.32)
Transmission Control Protocol, Src Port: 49481 (49481), Dst Port: http (80), Seq: 2005527222, Len: 0
  Source port: 49481 (49481)
  Destination port: http (80)
  [Stream index: 5]
  Sequence number: 2005527222
  Header length: 40 bytes
  Flags: 0x002 (SYN)
  Window size value: 14600
  [Calculated window size: 14600]
  Checksum: 0x3d75 [validation disabled]
  Options: (20 bytes), Maximum segment size, SACK permitted, Timestamps, No-Operation (NOP), Window scale
```

a) What is the sequence number of the TCP SYN segment that is used to initiate the TCP connection between the client computer and 202.141.80.32?

Ans : Sequence number -> 2005527222
[Sequence number: **2005527222**]

b) What is it in the segment that identifies the segment as a SYN segment?

Ans : The Flags segment in TCP contains 9 1-bit flags. In this packet, the 2nd flag (SYN flag) has been set which indicates that the seq. no. sent is the initial sequence number.
[Flags: **0x002** (SYN)]

3.Print out a captured packet and indicate where you see the information that answers the following:


```
No.     Time           Source                Destination           Protocol Length Info
    312 2.180554000    202.141.80.32         10.10.12.5            TCP      74     http > 49481 [SYN, ACK] Seq=5
67553306 Ack=2005527223 Win=14480 Len=0 MSS=1460 SACK_PERM=1 TSval=4082042364 TSecr=228585 WS=128

Frame 312: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
Ethernet II, Src: 4c:4e:35:97:1e:ef (4c:4e:35:97:1e:ef), Dst: 84:70:ce:14:a5:15 (84:70:ce:14:a5:15)
Internet Protocol Version 4, Src: 202.141.80.32 (202.141.80.32), Dst: 10.10.12.5 (10.10.12.5)
Transmission Control Protocol, Src Port: http (80), Dst Port: 49481 (49481), Seq: 567553306, Ack: 2005527223, Le
n: 0
  Source port: http (80)
  Destination port: 49481 (49481)
  [Stream index: 5]
  Sequence number: 567553306
  Acknowledgment number: 2005527223
  Header length: 40 bytes
  Flags: 0x012 (SYN, ACK)
  Window size value: 14480
  [Calculated window size: 14480]
  Checksum: 0xf5a1 [validation disabled]
  Options: (20 bytes), Maximum segment size, SACK permitted, Timestamps, No-Operation (NOP), Window scale
  [SEQ/ACK analysis]
```

a) What is the sequence number of the SYNACK segment sent by 202.141.80.32 to the client computer in reply to the SYN?

Ans : Sequence number -> 567553306
[Sequence number: **567553306**]

b) What is the value of the ACKnowledgement field in the SYNACK segment?

Ans : Acknowledgement field -> 2005527223
[Acknowledgment number: **2005527223**]

c) How did 202.141.80.32 server determine that value?

Ans : The ACK field value is determined by adding 1 to the Seq num recd from the client. Since the Seq number sent by the client was 2005527222, the ACK number was set as 2005527222+1=2005527223.

d) What is it in the segment that identifies the segment as a SYNACK segment?

Ans : The Flags segment in TCP contains 9 1-bit flags. In this packet, the 2nd flag (SYN flag) and the 5th flag (ACK) has been set which identifies the segment as a SYNACK segment.
[Flags: **0x012** (SYN, ACK)]

4. Print out a captured packet and indicate where you see the information that answers the following: What is the sequence number of the TCP segment containing the HTTP POST command? Note that in order to find the POST command, you'll need to dig into the packet content field at the bottom of the Wireshark window, looking for a segment with a "POST" within its DATA field. Consider the TCP segment containing the HTTP POST as the first segment in the TCP connection.


```
No.     Time           Source                Destination           Protocol Length Info
    314 2.182661000    10.10.12.5            202.141.80.32         HTTP     1514   POST /~sukumar/cs349/lab3-1-r
eply.htm HTTP/1.1

Frame 314: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface 0
Ethernet II, Src: 84:70:ce:14:a5:15 (84:70:ce:14:a5:15), Dst: 4c:4e:35:97:1e:ef (4c:4e:35:97:1e:ef)
Internet Protocol Version 4, Src: 10.10.12.5 (10.10.12.5), Dst: 202.141.80.32 (202.141.80.32)
Transmission Control Protocol, Src Port: 49481 (49481), Dst Port: http (80), Seq: 2005527223, Ack: 567553307, Le
n: 1448
  Source port: 49481 (49481)
  Destination port: http (80)
  [Stream index: 5]
  Sequence number: 2005527223
  [Next sequence number: 2005528671]
  Acknowledgment number: 567553307
  Header length: 32 bytes
  Flags: 0x010 (ACK)
  Window size value: 115
  [Calculated window size: 14720]
  [Window size scaling factor: 128]
  Checksum: 0x46b8 [validation disabled]
  Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
  [SEQ/ACK analysis]
Hypertext Transfer Protocol
MIME Multipart Media Encapsulation, Type: multipart/form-data, Boundary: "---------------------------94143057820
45584668780082173"
```

What is the sequence number of the TCP segment containing the HTTP POST command?

Ans : Sequence number -> 2005527223
[Sequence number: **2005527223**]

a) What are the sequence numbers of the first six segments in the TCP connection (including the segment containing the HTTP POST)? At what time was each segment sent?

Ans : Seg 1== Seq No -> 2005527223, Time -> 0.004069000
Seg 2== Seq No -> 2005528671, Time -> 0.004079000
Seg 3== Seq No -> 2005530119, Time -> 0.004082000
Seg 4== Seq No -> 2005531567, Time -> 0.004084000
Seg 5== Seq No -> 2005533015, Time -> 0.004086000
Seg 6== Seq No -> 2005534463, Time -> 0.004090000

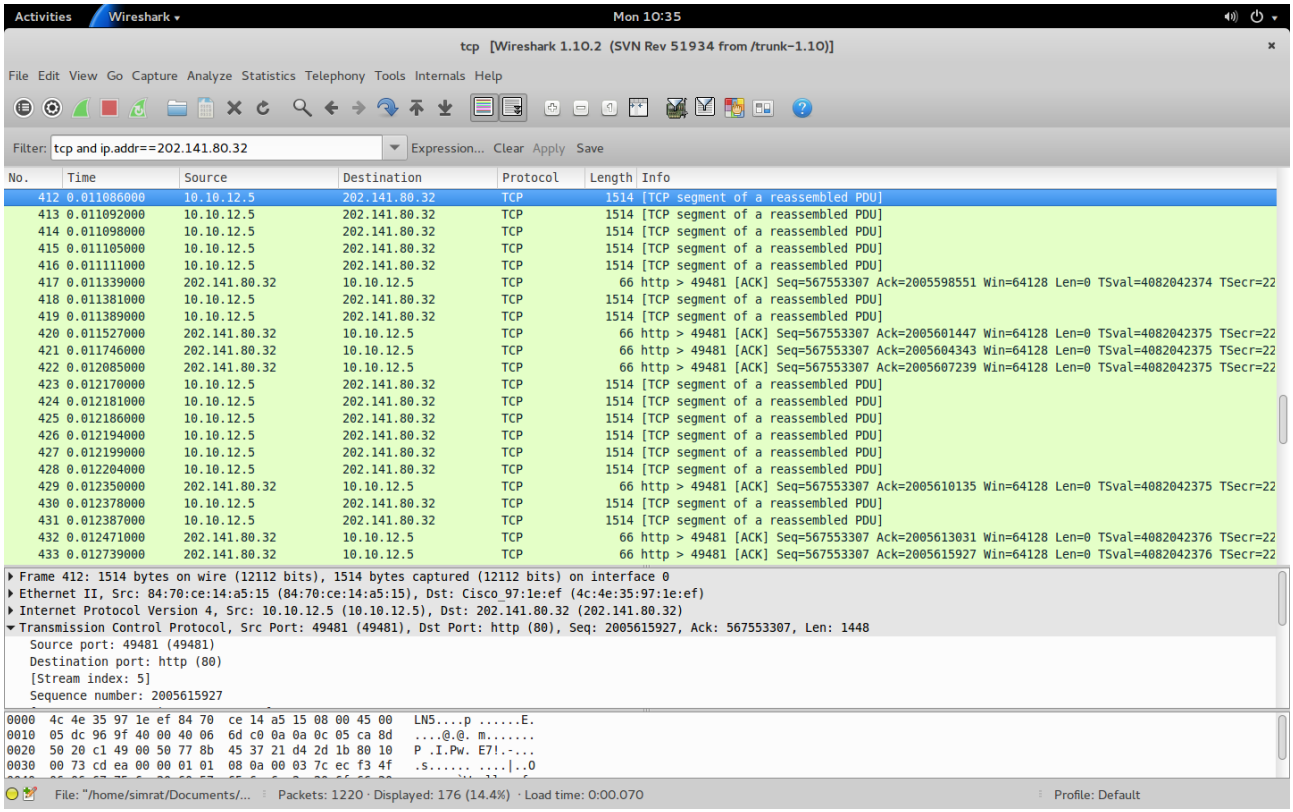
(Note : Time is written relative to the first packet sent)

b) When was the ACK for each segment received?

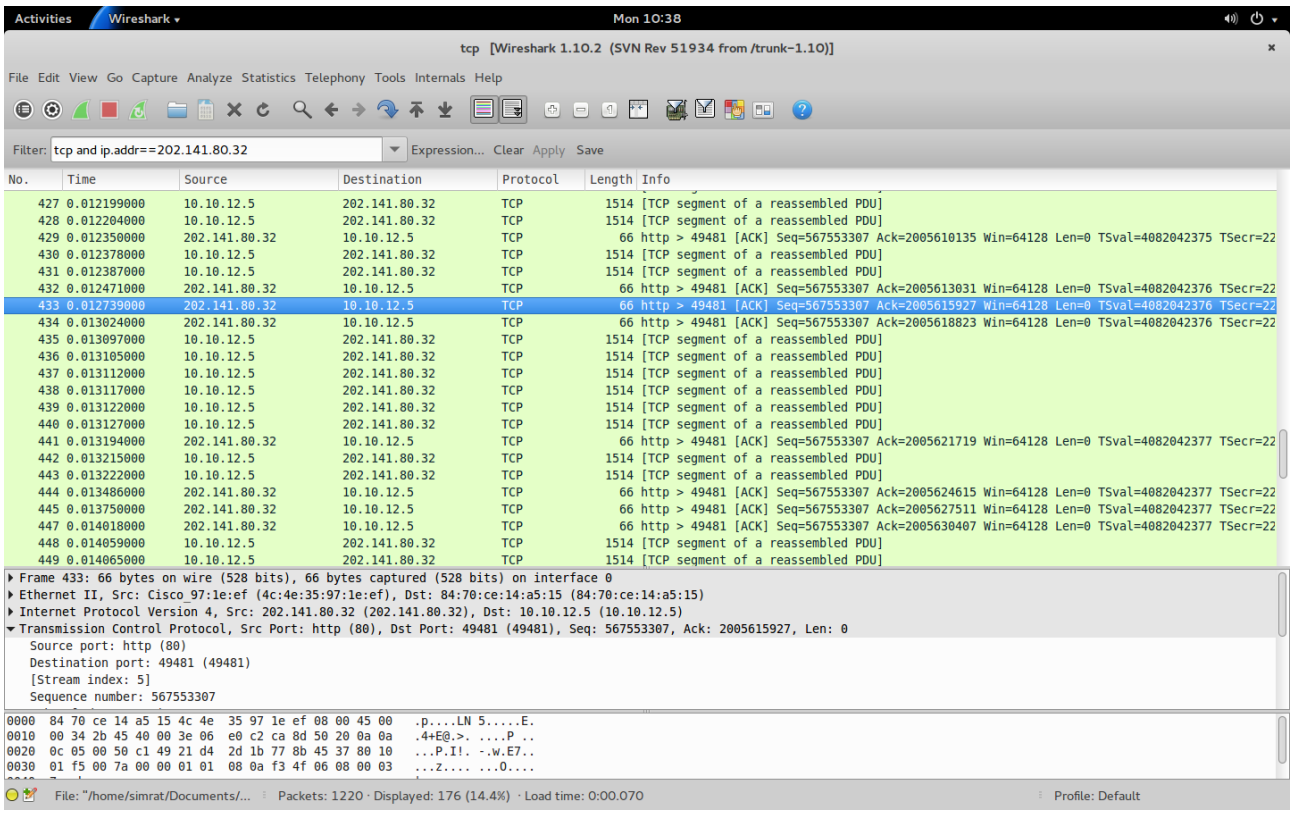
Ans : Ack for Seg 1 recd at -> 0.004722000
Ack for Seg 2 recd at -> 0.004915000
Ack for Seg 3 recd at -> 0.004967000
Ack for Seg 4 recd at -> 0.005147000
Ack for Seg 5 recd at -> 0.005184000
Ack for Seg 6 recd at -> 0.005406000

c) Do you see evidence of the use of cumulative ACKs in your trace? Explain.

Ans :



Yes, for example, in the screenshot of the trace above, segment 412 has seq no. 2005615927 and segment 413 has seq. no. 2005617375.
However, we can see in the screenshot below, there is an ACK 433 with ack. no. 2005615927 followed by an ACK 434 with ack. no. 2005618823. Hence the latter ACK is a cumulative ACK for the segments 412 and 413.



d) Given the difference between when each TCP segment was sent, and when its acknowledgement was received, what is the RTT value for each of the six segments?

Ans : RTT value = ACK recd time – Segment send time
Seg 1 -> RTT value = 0.004722000 – 0.004069000 = 0.000653 s
Seg 2 -> RTT value = 0.004915000 – 0.004079000 = 0.000836 s
Seg 3 -> RTT value = 0.004967000 – 0.004082000 = 0.000885 s
Seg 4 -> RTT value = 0.005147000 – 0.004084000 =0.001063 s
Seg 5 -> RTT value = 0.005184000 – 0.004086000 = 0.001098 s
Seg 6 -> RTT value = 0.005406000 – 0.004090000 = 0.001316 s

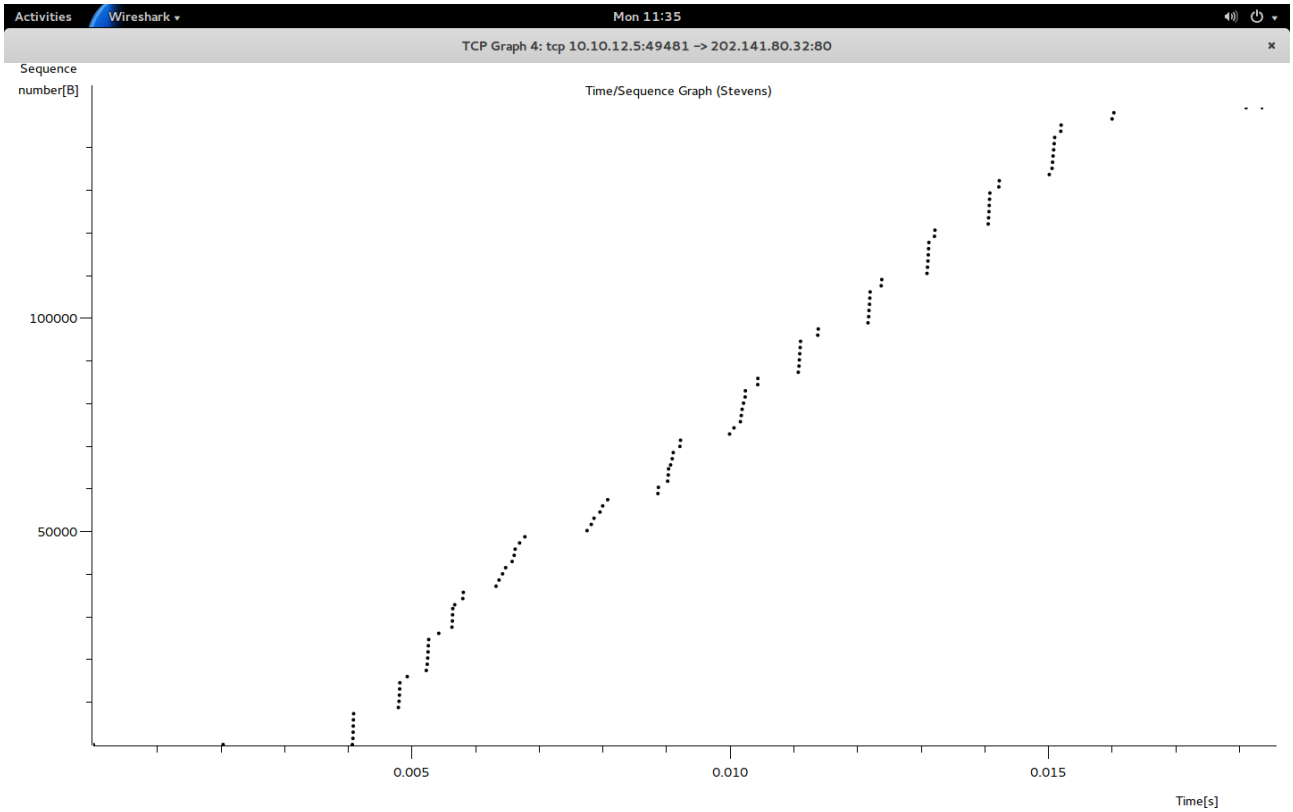
e) What is the length of each of the first six TCP segments?

Ans : Seg 1 -> Length = 1448 byes (TCP header + TCP data)
Seg 2 -> Length = 1448 bytes
Seg 3 -> Length = 1448 bytes
Seg 4 -> Length = 1448 bytes
Seg 5 -> Length = 1448 bytes
Seg 6 -> Length = 1448 bytes

f) What is the minimum amount of available buffer space advertised at the receiver for the entire trace? Does the lack of receiver buffer space ever throttle the sender?

Ans : The minimum amount of available buffer space advertised at the receiver is 14480 bytes.
[Window size value: **14480**]
No, the lack of receiver buffer space never throttles the sender.

g) Are there any retransmitted segments in the trace file? What did you check for (in the trace) in order to answer this question? How much data does the receiver typically acknowledge in an ACK? Can you identify cases where the receiver is ACKing every other received segment? Explain.



There are no retransmitted segments in the trace file. To check this, we plotted the Time/Sequence graph of the segments sent from client to server. We observed that this graph is monotonically increasing and hence no segment was retransmitted. (If there had been a retransmitted segment than there would exist atleast one such segment whose sequence number would have been less than the segment transmitted before it)

Data acknowledged = Difference between ACK seq no of 2 consecutive ACKs

Data acknowledged by ACK 4 = 2005533015 – 2005531567 = 1448 bytes
Data acknowledged by ACK 8 = 2005540255 – 2005538807 = 1448 bytes
Data acknowledged by ACK 12 = 2005546047 – 2005544599 = 1448 bytes

So, typically the receiver acknowledges in 1448 bytes in an ACK.

Data acknowledged by ACK 52 = 2005601447 – 2005598551 = 2896 bytes
Data acknowledged by ACK 53 = 2005604343 – 2005601447 = 2896 bytes

In the above ACKs, since the data acknowledged is twice the usual (1448 bytes), the receiver is acknowledging every other received segment. These ACKs are known as cumulative ACKs.