

## Lab 2

1. The packet that corresponds to the initial HTTP request made by this computer used the method GET to make the request. The URI that my computer requested was <http://www.example.com> which was the same the link typed in. This is because the initial request is made to access the site and since the URL refers to the subset of URIs, it shows it makes a request to the corresponding URI.

The image displays a Wireshark packet capture of an initial HTTP request. The top pane shows a list of captured packets, with packet 849 selected. The middle pane shows the details of this packet, and the bottom pane shows the raw packet data in hexadecimal and ASCII.

**Packet List:**

No.	Time	Source	Destination	Protocol	Length	Info
849	19.01895800	10.0.2.15	93.184.216.34	HTTP	453	GET / HTTP/1.1
851	19.03708300	93.184.216.34	10.0.2.15	HTTP	1006	HTTP/1.1 200 OK (text/html)
855	19.10525500	10.0.2.15	93.184.216.34	HTTP	397	GET /favicon.ico HTTP/1.1
857	19.12047600	93.184.216.34	10.0.2.15	HTTP	1014	HTTP/1.1 404 Not Found (text/html)

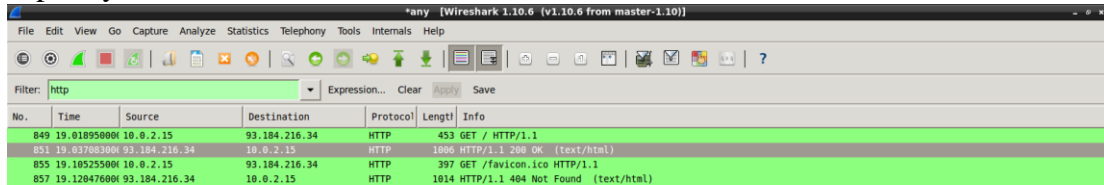
**Packet Details:**

- Frame 849: 453 bytes on wire (3624 bits), 453 bytes captured (3624 bits) on interface 0
- Linux cooked capture
- Internet Protocol Version 4, Src: 10.0.2.15 (10.0.2.15), Dst: 93.184.216.34 (93.184.216.34)
- Transmission Control Protocol, Src Port: 55338 (55338), Dst Port: http (80), Seq: 1, Ack: 1, Len: 397
  - Source port: 55338 (55338)
  - Destination port: http (80)
  - [Stream index: 48]
  - Sequence number: 1 (relative sequence number)
  - [Next sequence number: 398 (relative sequence number)]
  - Acknowledgment number: 1 (relative ack number)
  - Header length: 20 bytes
  - Flags: 0x018 (PSH, ACK)
  - Window size value: 29200
  - [Calculated window size: 29200]
  - [Window size scaling factor: -2 (no window scaling used)]
  - Checksum: 0x4391 [validation disabled]
  - [SEQ/ACK analysis]
- Hypertext Transfer Protocol
  - GET / HTTP/1.1\r\n
  - Host: www.example.com\r\n
  - Connection: keep-alive\r\n
  - Upgrade-Insecure-Requests: 1\r\n
  - User-Agent: Mozilla/5.0 (X11; Linux i686) AppleWebKit/537.36 (KHTML, like Gecko) Ubuntu Chromium/53.0.2785.143 Chrome/53.0.2785.143 Safari/537.36\r\n
  - Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,\*/\*;q=0.8\r\n
  - Accept-Encoding: gzip, deflate, sdch\r\n
  - Accept-Language: en-US,en;q=0.8\r\n
  - \r\n
  - [Full request URI: http://www.example.com/]
  - [HTTP request 1/2]
  - [Response in frame: 851]
  - [Next request in frame: 855]

**Raw Data:**

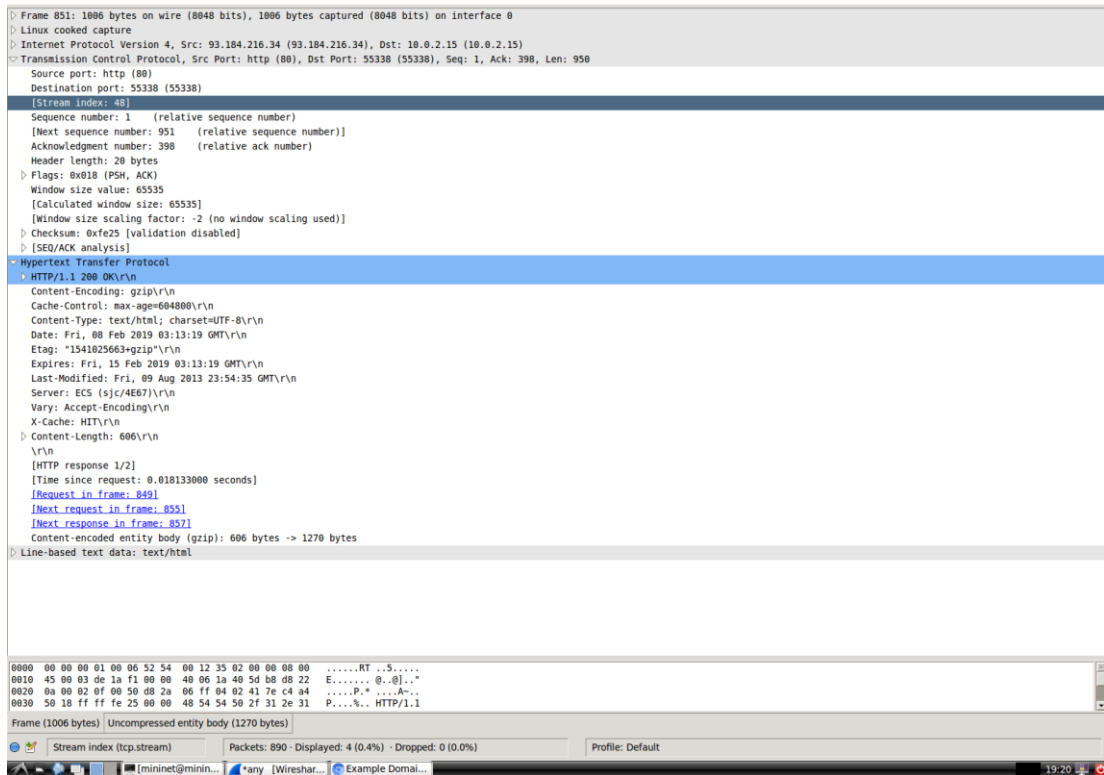
```
0000 00 04 00 01 00 06 00 00 27 27 c5 3a 00 00 00 00 .....':.....
0010 45 00 01 b5 cd b8 40 00 40 06 29 a1 0a 00 02 0f E....@. ....
0020 5d b8 d8 22 d8 2a 00 50 41 7e c3 17 06 ff 04 02 ]...".P A.....
0030 50 18 72 10 43 91 00 00 47 45 54 20 2f 20 48 54 P.r.C... GET / HT
0040 54 50 2f 31 2e 31 0d 0a 48 6f 73 74 3a 20 77 77 TP/1.1, Host: ww
0050 77 2e 65 70 61 6d 70 6c 65 2e 63 6f 6d 0a 43 w.examp e.com.-C
```

2. You will notice in the picture below that the packet corresponding to the initial HTTP response the server issued in response to my request returned a status code of 200 OK. The content type is labeled as text/html, which makes sense considering that the website is purely text based.



The image shows a Wireshark packet capture window with the filter set to 'http'. The packet list shows four packets. Packet 851 is the HTTP 200 OK response from 93.184.216.34 to 10.0.2.15.

No.	Time	Source	Destination	Protocol	Length	Info
849	19.01895000	10.0.2.15	93.184.216.34	HTTP	453	GET / HTTP/1.1
851	19.03708300	93.184.216.34	10.0.2.15	HTTP	1806	HTTP/1.1 200 OK (text/html)
855	19.10525500	10.0.2.15	93.184.216.34	HTTP	397	GET /favicon.ico HTTP/1.1
857	19.12047600	93.184.216.34	10.0.2.15	HTTP	1814	HTTP/1.1 404 Not Found (text/html)



The image shows the packet details pane for packet 851. It displays the raw data, the Ethernet II header, the Internet Protocol Version 4 header, the Transmission Control Protocol header, and the Hypertext Transfer Protocol header. The HTTP response status is 200 OK, and the content type is text/html.

```
> Frame 851: 1806 bytes on wire (8048 bits), 1806 bytes captured (8048 bits) on Interface 0
> Linux cooked capture
> Internet Protocol Version 4, Src: 93.184.216.34 (93.184.216.34), Dst: 10.0.2.15 (10.0.2.15)
> Transmission Control Protocol, Src Port: http (80), Dst Port: 55338 (55338), Seq: 1, Ack: 398, Len: 950
  Source port: http (80)
  Destination port: 55338 (55338)
[Stream index: 48]
  Sequence number: 1 (relative sequence number)
  [Next sequence number: 951 (relative sequence number)]
  Acknowledgment number: 398 (relative ack number)
  Header length: 20 bytes
  > Flags: 0x018 (PSH, ACK)
  Window size value: 65535
  [Calculated window size: 65535]
  [Window size scaling factor: -2 (no window scaling used)]
  > Checksum: 0xfe29 [validation disabled]
  > [SEQ/ACK analysis]
  > Hypertext Transfer Protocol
    > HTTP/1.1 200 OK\r\n
    Content-Encoding: gzip\r\n
    Cache-Control: max-age=604800\r\n
    Content-Type: text/html; charset=UTF-8\r\n
    Date: Fri, 08 Feb 2019 03:13:19 GMT\r\n
    Etag: "1541025663+gzip"\r\n
    Expires: Fri, 15 Feb 2019 03:13:19 GMT\r\n
    Last-Modified: Fri, 09 Aug 2013 23:54:35 GMT\r\n
    Server: ECS (sjc/4667)\r\n
    Vary: Accept-Encoding\r\n
    X-Cache: HIT\r\n
  > Content-Length: 606\r\n
  \r\n
  [HTTP response 1/2]
  [Time since request: 0.018133000 seconds]
  [Request in frame: 849]
  [Next request in frame: 855]
  [Next response in frame: 857]
  Content-encoded entity body (gzip): 606 bytes -> 1270 bytes
  > Line-based text data: text/html

0000  00 00 00 01 00 00 52 54  00 12 35 02 00 00 08 00  .....RT..5....
0010  45 00 03 06 1a f1 09 00  40 06 1a 40 5d b0 d8 22  E.....@..*
0020  0a 00 02 0f 00 50 d8 2a  06 ff 04 02 41 7e c4 a4  ....P.*...A...
0030  50 18 ff ff fe 25 00 00  48 54 54 50 2f 31 2e 31  P....%.. HTTP/1.1

Frame 1006 bytes Uncompressed entity body (1270 bytes)
Stream index (tcp.stream) Packets: 890 - Displayed: 4 (0.4%) - Dropped: 0 (0.0%) Profile: Default
```

3. The packets corresponding to the initial request made by my computer are identified down below. You will notice that this time the status code returned is different from the one previously, because when you navigate to <http://www.soe.ucsc.edu> it redirects you to a secure site with https as the protocol. Thus, incurring the 301 status code which means the site was moved permanently.

The image shows a Wireshark packet capture window. The top toolbar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Tools, Internals, and Help. The filter bar shows 'http'. The packet list pane displays two packets:

No.	Time	Source	Destination	Protocol	Length	Info
951	73.00331708	10.0.2.15	128.114.47.25	HTTP	454	GET / HTTP/1.1
955	73.01733808	128.114.47.25	10.0.2.15	HTTP	748	HTTP/1.1 301 Moved Permanently (text/html)

The packet details pane for the selected packet (Frame 951) shows the following structure:

- Linux cooked capture
- Internet Protocol Version 4, Src: 10.0.2.15 (10.0.2.15), Dst: 128.114.47.25 (128.114.47.25)
- Transmission Control Protocol, Src Port: 54211 (54211), Dst Port: http (80), Seq: 1, Ack: 1, Len: 398
  - Source port: 54211 (54211)
  - Destination port: http (80)
  - [Stream index: 155]
  - Sequence number: 1 (relative sequence number)
  - [Next sequence number: 399 (relative sequence number)]
  - Acknowledgment number: 1 (relative ack number)
  - Header length: 20 bytes
  - Flags: 0x018 (PSH, ACK)
  - Window size value: 29200
  - [Calculated window size: 29200]
  - [Window size scaling factor: -2 (no window scaling used)]
  - Checksum: 0xbd42 [validation disabled]
  - [SEQ/ACK analysis]
- Hypertext Transfer Protocol
  - GET / HTTP/1.1
  - Host: www.soe.ucsc.edu
  - Connection: keep-alive
  - Upgrade-Insecure-Requests: 1
  - User-Agent: Mozilla/5.0 (X11; Linux i686) AppleWebKit/537.36 (KHTML, like Gecko) Ubuntu Chromium/53.0.2785.143 Chrome/53.0.2785.143 Safari/537.36
  - Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,\*/\*;q=0.8
  - Accept-Encoding: gzip, deflate, sdch
  - Accept-Language: en-US,en;q=0.8
  - [Full request URI: http://www.soe.ucsc.edu/]
  - [HTTP request 1/1]
  - [Response in frame: 955]

The packet bytes pane shows the raw data of the packet, with a hex view on the left and a ASCII view on the right. The ASCII view shows the beginning of the HTTP request: 'GET / HTTP/1.1 Host: www.soe.ucsc.edu'.

4. The screenshot below shows a packet received through the method HEAD which was used in the Linux terminal to retrieve. This returns the header of the website, which in this case was used for [www.example.com](http://www.example.com).

The screenshot displays the Wireshark network protocol analyzer interface. The top menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Tools, Internals, and Help. The toolbar contains various icons for file operations, capture control, and analysis. The filter bar at the top shows 'http' as the active filter. The packet list pane on the left shows a list of captured packets, with packet 18438 selected. The packet details pane on the right shows the structure of the selected packet, including Ethernet II, Internet Protocol Version 4, Transmission Control Protocol, and Hypertext Transfer Protocol. The packet bytes pane at the bottom shows the raw data of the selected packet in hexadecimal and ASCII.

No.	Time	Source	Destination	Protocol	Length	Info
6372	146.7198408	10.0.2.15	210.36.194.174	HTTP	462	GET /pewdiepie HTTP/1.1
6574	146.8069870	216.58.194.174	10.0.2.15	HTTP	398	HTTP/1.1 301 Moved Permanently
6576	146.8182440	10.0.2.15	216.58.194.174	HTTP	467	GET /user/PewDiePie HTTP/1.1
6578	146.8632210	216.58.194.174	10.0.2.15	HTTP	700	HTTP/1.1 301 Moved Permanently
10844	382.3864330	10.0.2.15	172.217.6.46	HTTP	686	GET /hello.htm HTTP/1.1
10850	382.4095830	172.217.6.46	10.0.2.15	HTTP	351	HTTP/1.1 404 Not Found (text/html)
10853	382.4537170	10.0.2.15	172.217.6.46	HTTP	625	GET /favicon.ico HTTP/1.1
10856	382.4709220	172.217.6.46	10.0.2.15	HTTP	638	HTTP/1.1 301 Moved Permanently (text/html)
10937	387.1558990	10.0.2.15	172.217.6.46	HTTP	686	GET /hello.htm HTTP/1.1
10945	387.1768980	172.217.6.46	10.0.2.15	HTTP	351	HTTP/1.1 404 Not Found (text/html)
18243	1206.436257	10.0.2.15	93.184.216.34	HTTP	169	GET / HTTP/1.1
18245	1206.455940	93.184.216.34	10.0.2.15	HTTP	1653	HTTP/1.1 200 OK (text/html)
18438	1249.012163	10.0.2.15	93.184.216.34	HTTP	192	HEAD / HTTP/1.1
25469	6646.424899	10.0.2.15	128.114.47.25	HTTP	170	GET / HTTP/1.1
25471	6646.441018	128.114.47.25	10.0.2.15	HTTP	748	HTTP/1.1 301 Moved Permanently (text/html)

Frame 18438: 192 bytes on wire (1536 bits), 192 bytes captured (1536 bits) on interface 0

- Linux cooked capture
- Internet Protocol Version 4, Src: 10.0.2.15 (10.0.2.15), Dst: 93.184.216.34 (93.184.216.34)
- Transmission Control Protocol, Src Port: 60207 (60207), Dst Port: http (80), Seq: 1, Ack: 1, Len: 136
  - Source port: 60207 (60207)
  - Destination port: http (80)
  - [Stream index: 2606]
  - Sequence number: 1 (relative sequence number)
  - [Next sequence number: 137 (relative sequence number)]
  - Acknowledgment number: 1 (relative ack number)
  - Header length: 20 bytes
  - Flags: 0x018 (PSH, ACK)
  - Window size value: 29200
  - [Calculated window size: 29200]
  - [Window size scaling factor: -2 (no window scaling used)]
  - Checksum: 0x428c [validation disabled]
  - [SEQ/ACK analysis]
- Hypertext Transfer Protocol
  - HEAD / HTTP/1.1\r\n
  - TE: deflate,gzip;q=0.3\r\n
  - Connection: TE, close\r\n
  - Host: www.example.com\r\n
  - User-Agent: lwp-request/6.03 libwww-perl/6.05\r\n
  - \r\n
  - [Full request URI: http://www.example.com/]
  - [HTTP request 1/1]

0000 00 04 00 01 00 06 08 00 27 27 c6 3a 00 00 00 00 ..... '' :....  
0010 45 00 00 b0 64 1e 40 00 40 06 94 40 0a 00 02 0f E...d.@. @..@....  
0020 5d b0 d8 22 eb 2f 00 50 ee 58 76 91 18 59 66 02 j...\*/P..Xv..Yf.  
0030 50 18 72 10 42 8c 00 00 48 45 41 44 20 2f 20 48 P.r.B... HEAD / H  
0040 54 54 50 2f 31 2e 31 0d 0a 54 45 3a 20 64 65 66 TTP/1.1. .TE: def  
0050 6c 61 74 65 2c 67 7a 69 70 3b 71 3d 2e 33 0d late,gz1 p;q=0.3.

5. You can see the steps taken by my computer before the web page was loaded. First, it established a connection with the server using DNS protocols and then established a 3-way handshake using TCP protocols, where you can see [SYN], [SYN, ACK], and [ACK]. Finally, results in the HTTP protocol which uses the method GET to retrieve the web page. This makes sense because we are given web page site URL, which needs to fetch information from the server before displaying the web page.

The image shows a Wireshark packet capture window titled "Capturing from any [Wireshark 1.10.6 (v1.10.6 from master-1.10)]". The packet list on the left shows 186 packets. The selected packet is packet 181, an HTTP GET request. The packet details pane on the right shows the structure of the packet, including the Ethernet II header, Internet Protocol Version 4 header, Transmission Control Protocol header, and Hypertext Transfer Protocol header. The packet bytes pane at the bottom shows the raw data in hexadecimal and ASCII.

No.	Time	Source	Destination	Protocol	Length	Info
172	14.394075000	126.36.125.08	10.0.2.15	TCP	62	https > 43989 [ACK] Seq=12685 Ack=3808 Win=65535 Len=0
173	14.889645000	10.0.2.15	192.168.1.1	DNS	77	Standard query 0x3377 A www.example.com
174	14.909250000	192.168.1.1	10.0.2.15	DNS	93	Standard query response 0x3377 A 93.184.216.34
175	14.909600000	10.0.2.15	93.184.216.34	TCP	76	36515 > http [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=1804228 TSecr=0 WS=128
176	14.909721000	10.0.2.15	93.184.216.34	TCP	76	36516 > http [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=1804228 TSecr=0 WS=128
177	14.930623000	93.184.216.34	10.0.2.15	TCP	62	http > 36515 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
178	14.930663000	10.0.2.15	93.184.216.34	TCP	56	36515 > http [ACK] Seq=1 Ack=1 Win=29200 Len=0
179	14.932798000	93.184.216.34	10.0.2.15	TCP	62	http > 36516 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
180	14.932834000	10.0.2.15	93.184.216.34	TCP	56	36516 > http [ACK] Seq=1 Ack=1 Win=29200 Len=0
181	14.958998000	10.0.2.15	93.184.216.34	HTTP	453	GET / HTTP/1.1
182	14.959657000	93.184.216.34	10.0.2.15	TCP	62	http > 36515 [ACK] Seq=1 Ack=398 Win=65535 Len=0
183	14.974655000	93.184.216.34	10.0.2.15	HTTP	1023	HTTP/1.1 200 OK (text/html)
184	14.974687000	10.0.2.15	93.184.216.34	TCP	56	36515 > http [ACK] Seq=398 Ack=968 Win=30944 Len=0
185	14.990040000	127.0.0.1	127.0.0.1	TCP	76	39474 > 6633 [SYN] Seq=0 Win=43690 Len=0 MSS=65495 SACK_PERM=1 TSval=1804250 TSecr=0 WS=128
186	14.998012000	127.0.0.1	127.0.0.1	TCP	56	6633 > 39474 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0

Frame 181: 453 bytes on wire (3624 bits), 453 bytes captured (3624 bits) on interface 0  
Linux cooked capture  
Internet Protocol Version 4, Src: 10.0.2.15 (10.0.2.15), Dst: 93.184.216.34 (93.184.216.34)  
Transmission Control Protocol, Src Port: 36515 (36515), Dst Port: http (80), Seq: 1, Ack: 1, Len: 397  
Source port: 36515 (36515)  
Destination port: http (80)  
[Stream index: 31]  
Sequence number: 1 (relative sequence number)  
[Next sequence number: 398 (relative sequence number)]  
Acknowledgment number: 1 (relative ack number)  
Header length: 20 bytes  
Flags: 0x018 (PSH, ACK)  
Window size value: 29200  
[Calculated window size: 29200]  
[Window size scaling factor: -2 (no window scaling used)]  
Checksum: 0x4391 [validation disabled]  
[SEQ/ACK analysis]  
Hypertext Transfer Protocol  
GET / HTTP/1.1\r\nHost: www.example.com\r\nConnection: keep-alive\r\nUpgrade-Insecure-Requests: 1\r\nUser-Agent: Mozilla/5.0 (X11; Linux i686) AppleWebKit/537.36 (KHTML, like Gecko) Ubuntu Chromium/53.0.2785.143 Chrome/53.0.2785.143 Safari/537.36\r\nAccept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,\*/\*;q=0.8\r\nAccept-Encoding: gzip, deflate, sdch\r\nAccept-Language: en-US,en;q=0.8\r\n\r\n[Full request URI: http://www.example.com/]  
[HTTP request 1/2]  
[Response in frame: 183]  
[Next request in frame: 190]

0000 00 04 00 01 00 06 00 27 27 c6 3a 02 0f 08 00 ..... ''.....  
0010 45 00 01 b5 92 f5 40 00 40 06 64 64 0a 00 02 0f E.....@.dd....  
0020 5d b8 d8 22 8e a3 00 50 9d 5e 15 38 29 2e f8 02 ].\*...P.^.)...  
0030 50 18 72 10 43 91 00 00 47 45 54 20 2f 20 48 54 P.r.C... GET / HT  
0040 54 50 2f 31 2e 31 0d 0a 48 6f 73 74 3a 20 77 77 TP/1.1.. Host: ww  
0050 77 2e 65 78 61 6d 70 6c 65 2e 63 6f 6d 0d 0a 43 w.examl e.com..C

- 
- The screenshot displays the Wireshark 1.10.6 interface. The top menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Tools, Internals, and Help. The toolbar contains various icons for file operations, capture control, and analysis. The main window is divided into three panes:
- Packet List:** Shows a list of captured packets. The selected packet is 58, which is a TCP ACK packet (Seq=161, Ack=734, Win=65535, Len=0) from 10.0.2.15 to 216.58.195.68.
  - Packet Details:** Shows the hierarchical structure of the selected packet. The layers are: Ethernet II, Internet Protocol Version 4, Transmission Control Protocol, and Hypertext Transfer Protocol. The TCP layer shows the sequence number 161, acknowledgment number 734, and window size 65535. The HTTP layer shows the status code 200 (OK).
  - Packet Bytes:** Shows the raw data of the selected packet in hexadecimal and ASCII. The data is a valid HTTP response, starting with the status line "HTTP/1.1 200 OK".
- The status bar at the bottom indicates that the capture is in progress, with 8952 packets displayed (100.0%). The profile is set to Default.

7. The IP address I was given for [www.google.com](http://www.google.com) is 216.58.195.68, which is slightly different from that of the IP address we were given above.

The image shows a Wireshark packet capture of a network session. The top pane displays a list of packets. Packet 19 is a TCP SYN packet from 10.0.2.15 to 216.58.195.68 on port 443. Packet 20 is the corresponding SYN-ACK response from 216.58.195.68 to 10.0.2.15. The middle pane shows the details of the selected packet (19), including the TCP header fields: Source port: 54929, Destination port: https (443), Sequence number: 0, and Window size: 29200. The bottom pane shows the raw packet data in hexadecimal and ASCII format.

No.	Time	Source	Destination	Protocol	Length	Info
14	2.999465000	127.0.0.1	127.0.0.1	TCP	56	6633 > 48422 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
15	2.999090000	127.0.0.1	127.0.0.1	TCP	76	48423 > 6633 [SYN] Seq=0 Win=43690 Len=0 MSS=65495 SACK_PERM=1 TSval=2940500 TSecr=0 WS=128
16	2.999943000	127.0.0.1	127.0.0.1	TCP	56	6633 > 48423 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
17	3.062416000	10.0.2.15	192.168.1.1	DNS	76	Standard query 0xb67 A www.google.com
18	3.080832000	192.168.1.1	10.0.2.15	DNS	92	Standard query response 0xb67 A 216.58.195.68
19	3.080232000	10.0.2.15	216.58.195.68	TCP	76	54929 > https [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=2940520 TSecr=0 WS=128
20	3.093810000	216.58.195.68	10.0.2.15	TCP	62	https > 54929 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
21	3.093854000	10.0.2.15	216.58.195.68	TCP	56	54929 > https [ACK] Seq=1 Ack=1 Win=29200 Len=0
22	3.094322000	10.0.2.15	216.58.195.68	TLSv1.2	259	Client Hello
23	3.094732000	216.58.195.68	10.0.2.15	TCP	62	https > 54929 [ACK] Seq=1 Ack=204 Win=65535 Len=0
24	3.128022000	216.58.195.68	10.0.2.15	TLSv1.2	1486	Server Hello
25	3.128048000	10.0.2.15	216.58.195.68	TCP	56	54929 > https [ACK] Seq=204 Ack=1431 Win=31240 Len=0
26	3.128120000	216.58.195.68	10.0.2.15	TLSv1.2	1188	Certificate
27	3.128128000	10.0.2.15	216.58.195.68	TCP	56	54929 > https [ACK] Seq=204 Ack=2563 Win=34080 Len=0

Frame 19: 76 bytes on wire (608 bits), 76 bytes captured (608 bits) on interface 0  
Linux cooked capture  
Internet Protocol Version 4, Src: 10.0.2.15 (10.0.2.15), Dst: 216.58.195.68 (216.58.195.68)  
Transmission Control Protocol, Src Port: 54929 (54929), Dst Port: https (443), Seq: 0, Len: 0  
Source port: 54929 (54929)  
Destination port: https (443)  
[Stream index: 0]  
Sequence number: 0 (relative sequence number)  
Header length: 40 bytes  
Flags: 0x002 (SYN)  
Window size value: 29200  
[Calculated window size: 29200]  
Checksum: 0xa7bc [validation disabled]  
Options: (20 bytes), Maximum segment size, SACK permitted, Timestamps, No-Operation (NOP), Window scale

0000 00 04 00 01 00 06 08 00 27 27 c6 3a 00 00 08 00 .....':....  
0010 45 00 00 3c 45 4c 40 00 40 06 4d e2 0a 00 02 0f E...<ELQ. @M....  
0020 d8 3a c3 44 d6 91 01 bb e6 44 9b 24 00 00 00 00 .:D....:D\$.  
0030 a0 02 72 10 a7 bc 00 00 02 04 05 b4 04 02 08 0a .:f....  
0040 00 2c de 68 00 00 00 00 01 03 03 07 ..h....

8. My computer did want to complete the request recursively because you can see in the picture below that my computer still uses the DNS protocol to do a query search and if you look at the flags, it says recursion is desired so it does it recursively.

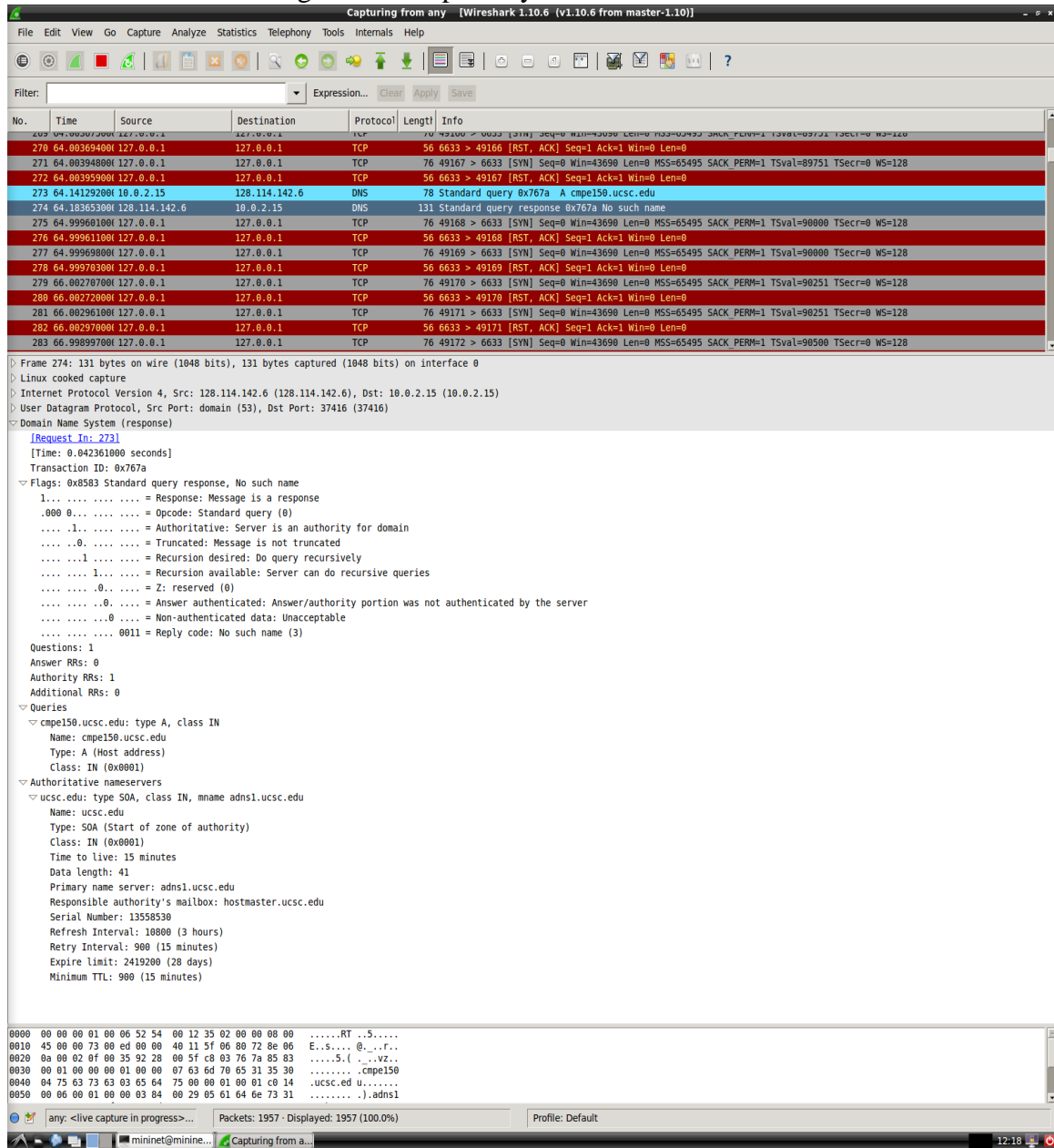
The image shows a Wireshark packet capture window titled "Capturing from any [Wireshark 1.10.6 (v1.10.6 from master-1.10)]". The packet list shows a series of TCP and DNS packets. Packet 152 is a DNS Standard query response from 10.0.2.15 to 128.114.142.6. The packet details pane shows the following information:

- Transaction ID: 0x62b7
- Flags: 0x0100 Standard query
- 0... .. = Response: Message is a query
- .000 0... .. = Opcode: Standard query (0)
- ... 0. .... = Truncated: Message is not truncated
- ... 1. .... = Recursion desired: Do query recursively
- ... ..0... .. = Z: reserved (0)
- ... ..0... .. = Non-authenticated data: Unacceptable
- Questions: 1
- Answer RRs: 0
- Authority RRs: 0
- Additional RRs: 0
- Queries
  - www.google.com: type A, class IN

The packet bytes pane shows the raw data of the DNS response, including the transaction ID 0x62b7 and the flags 0x0100.



9. There was no IP address given to me for cmpe150.ucsc.edu because it was not authenticated by the server therefore it never resolved by giving an IP address.
10. The authoritative name server for the ucsc.edu is adns1.ucsc.edu if you look at the picture below. You will see it is given in the primary name server.



11. The initial window size that my computer advertised to the server was 29200, whereas the window size that the server advertised was 65535, as shown in the picture below. You will notice that the 3-way handshake was initiated first before the receiving an HTTP code of 200 OK.

The image shows a Wireshark 1.10.6 packet capture window. The title bar reads "Capturing from any [Wireshark 1.10.6 (v1.10.6 from master-1.10)]". The menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Tools, Internals, and Help. The toolbar contains various icons for file operations, capture control, and analysis. Below the toolbar is a filter bar with a text input field and buttons for "Expression...", "Clear", "Apply", and "Save".

The main packet list table has columns: No., Time, Source, Destination, Protocol, Length, and Info. It displays several packets, with packets 120 through 129 highlighted in green, indicating they are part of the selected filter. Packet 120 is a TCP SYN packet from 10.0.2.15 to 80.249.99.148. Packet 121 is a TCP SYN-ACK packet from 80.249.99.148 to 10.0.2.15. Packet 122 is a TCP ACK packet from 10.0.2.15 to 80.249.99.148. Packet 123 is an HTTP GET packet from 10.0.2.15 to 80.249.99.148. Packet 124 is a TCP ACK packet from 10.0.2.15 to 80.249.99.148. Packet 125 is a TCP SYN packet from 127.0.0.1 to 127.0.0.1. Packet 126 is a TCP RST, ACK packet from 127.0.0.1 to 127.0.0.1. Packet 127 is a TCP SYN packet from 127.0.0.1 to 127.0.0.1. Packet 128 is a TCP RST, ACK packet from 127.0.0.1 to 127.0.0.1. Packet 129 is a TCP segment of a reassembled PDU from 80.249.99.148 to 10.0.2.15.

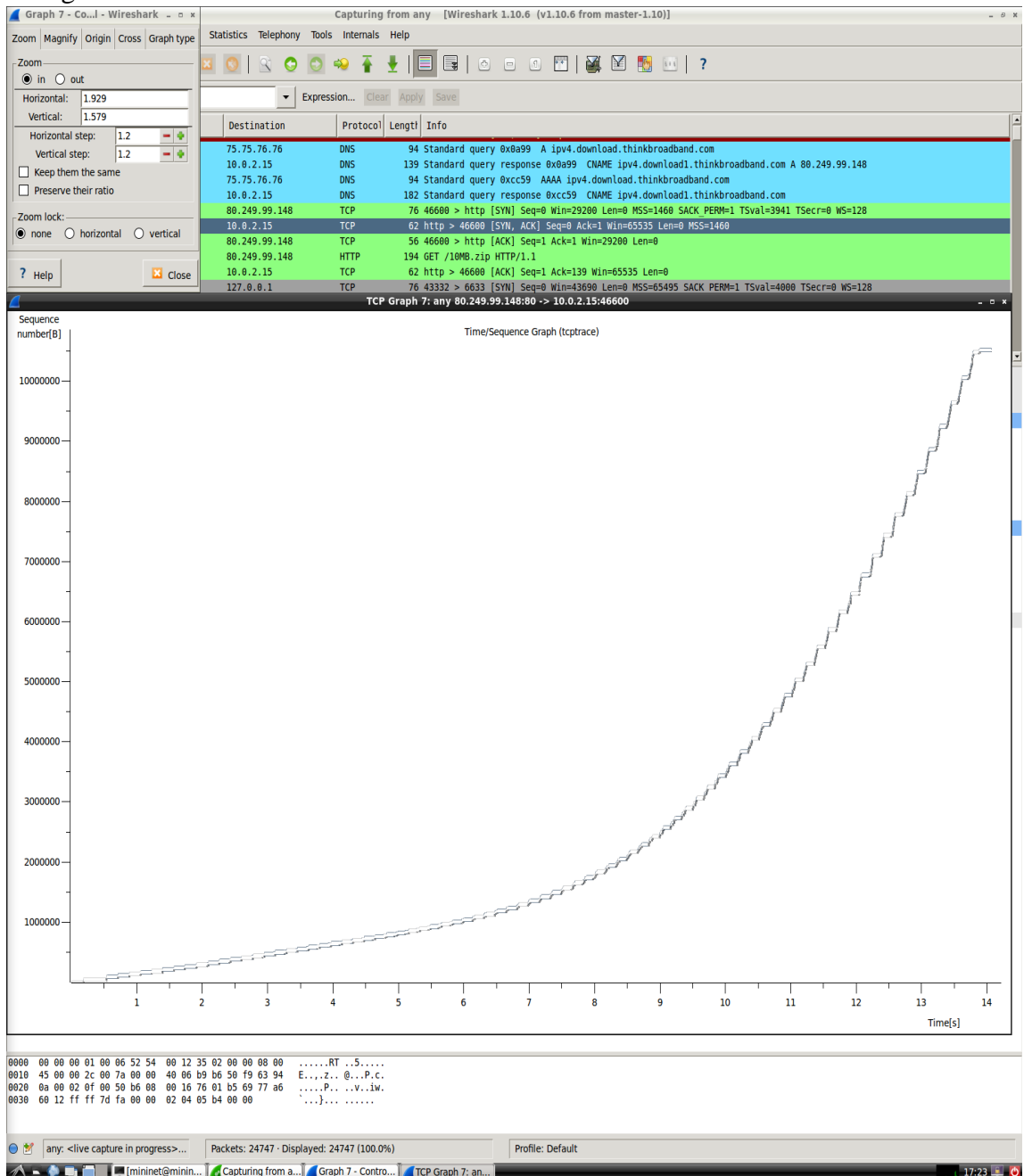
The packet details pane shows the selected packet (120) with the following details:

- Frame 120: 76 bytes on wire (608 bits), 76 bytes captured (608 bits) on interface 0
- Linux cooked capture
- Internet Protocol Version 4, Src: 10.0.2.15 (10.0.2.15), Dst: 80.249.99.148 (80.249.99.148)
- Transmission Control Protocol, Src Port: 46600 (46600), Dst Port: http (80), Seq: 0, Len: 0

The packet bytes pane shows the raw data of the selected packet, with a hex dump and ASCII representation.

The status bar at the bottom shows "any: <live capture in progress>...", "Packets: 13521 · Displayed: 13521 (100.0%)", and "Profile: Default". The bottom right corner shows the time "16:37" and a red status icon.

12. In the graph depicted below, you can see that this tcptrace graph has the source address as the server's address and the destination address as my computer's address. This corresponds to the [SYN, ACK] packet which responds to the computer's request. Doing a tcptrace on the TCP protocol returns a graph, where it shows the number of packets sent, corresponding to the sequence number, over the total time, which was 14 seconds. You will notice that the sequence number represents the size of the download which was 10Mb, which is the equivalent to 10,000,000 bits. The top line represents the receive window which is the server window, whereas the bottom line are the ACKs. In between those lines is the TCP segment and between the TCP segment and the ACKs are the bytes in flight.



13. The picture below depicts the tcptrace graph of the packets sent and received, where there are sections where the download experiences 0% and 100% loss. 0% loss being circled along with where the slow-start occurs. The parts that flatline correspond to the 100% loss, whereas everything else corresponds to the 0% loss. You can see the slow start close to the beginning of each section when we change packet loss, which starts off slow and increases exponentially.

