ELEC 331 Assignment 3

Wireshark IP

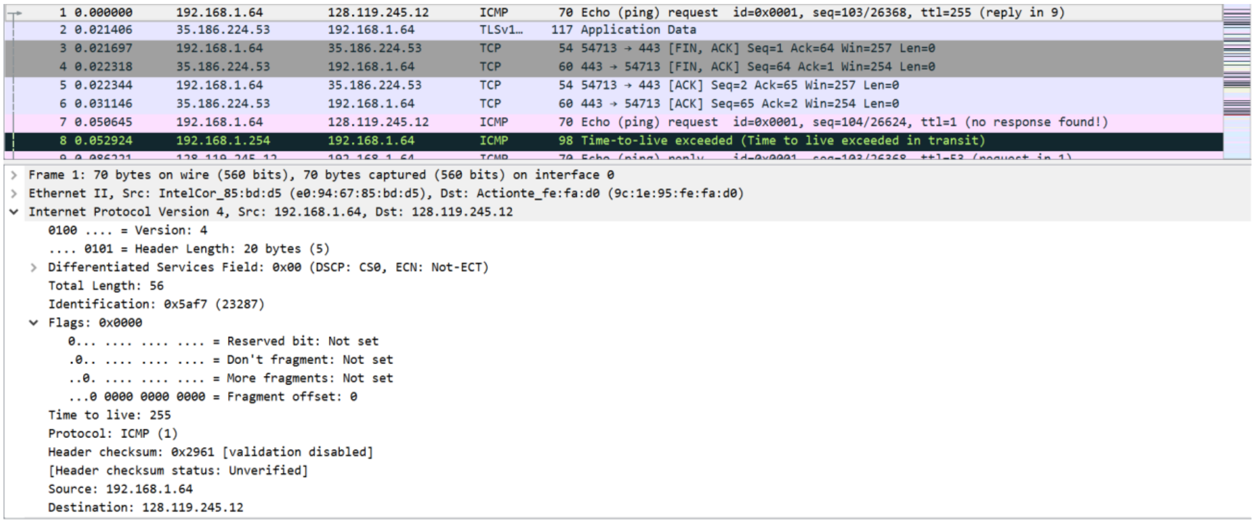


Figure #1: First ICMP Echo Request

1. The IP address of my computer is 192.168.1.64
2. The value of the upper layer protocol is ICMP (0x01)
3. The header has 20 bytes as indicated by the Header Length field. The total packet length is 56 bytes as indicated by the Total Length field, leaving 36 bytes for the payload.
4. No. This IP datagram has not been fragmented as indicated by the More Fragments bit.

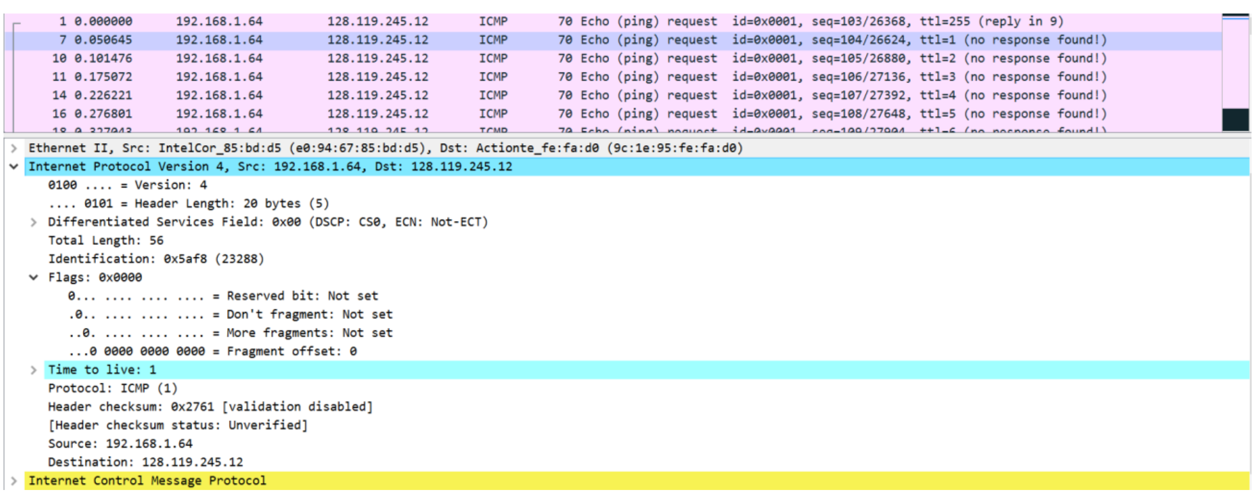


Figure #2: Second ICMP Echo Request

5. The fields that always change are:
   a. ID

   b. TTL
   c. Header checksum

6.
   a. The fields that stay constant are:
      i. Version – Same version of IP (IPv4)
      ii. Header Length – All packets are ICMP, so header length does not change
      iii. Differentiated Services – All packets are ICMP, so same service class
      iv. Source and Destination IP – Same host and same destination
      v. Upper Layer Protocol – All packets are ICMP packets
   b. The fields that must stay constant are:
      i. Same as a).
   c. The fields that must change are:
      i. ID – IP packets must have different ID
      ii. TTL – This is incremented at every router
      iii. Header checksum – Since TTL changes, the checksum must also change

7. The pattern I observe is that the value of the ID field increments after each ICMP request.

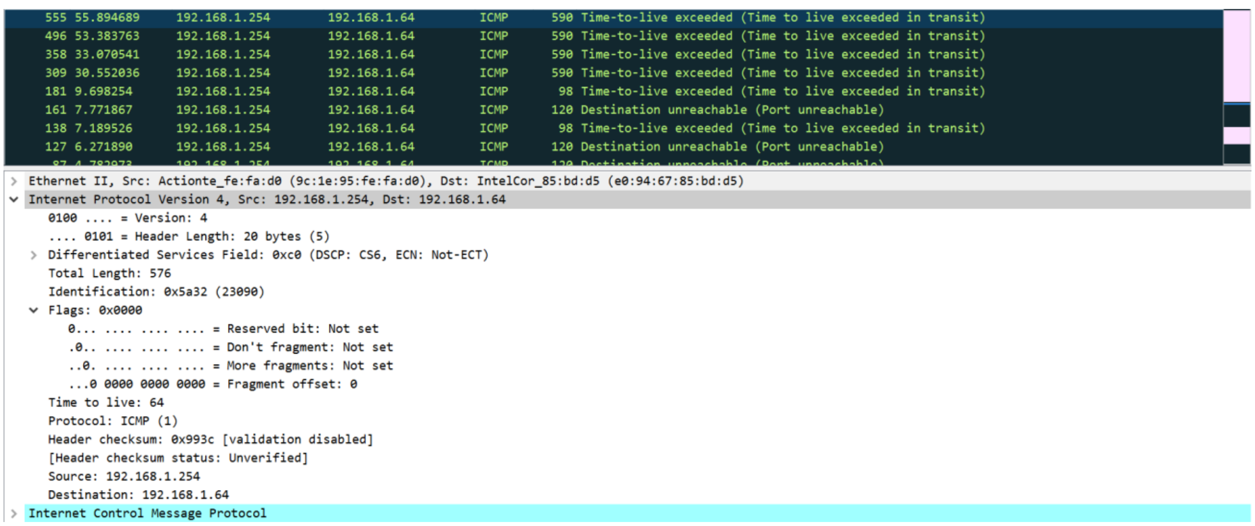


Figure #3: First ICMP Echo Response – TTL Expired

8. The value of the ID field is 23090 and the value of the TTL field is 64.
9. The value of the ID field changes as it gets incremented. The value of the TTL field does not change however because this is all for the "first-hop router."

ELEC 331 Assignment 3

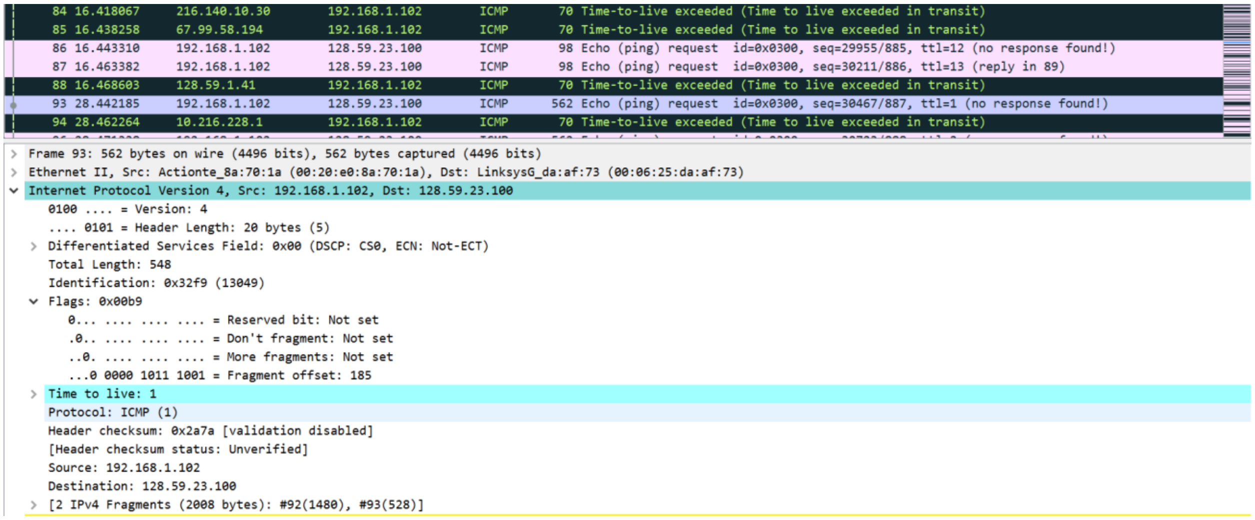


Figure #4: Fragmented ICMP Request

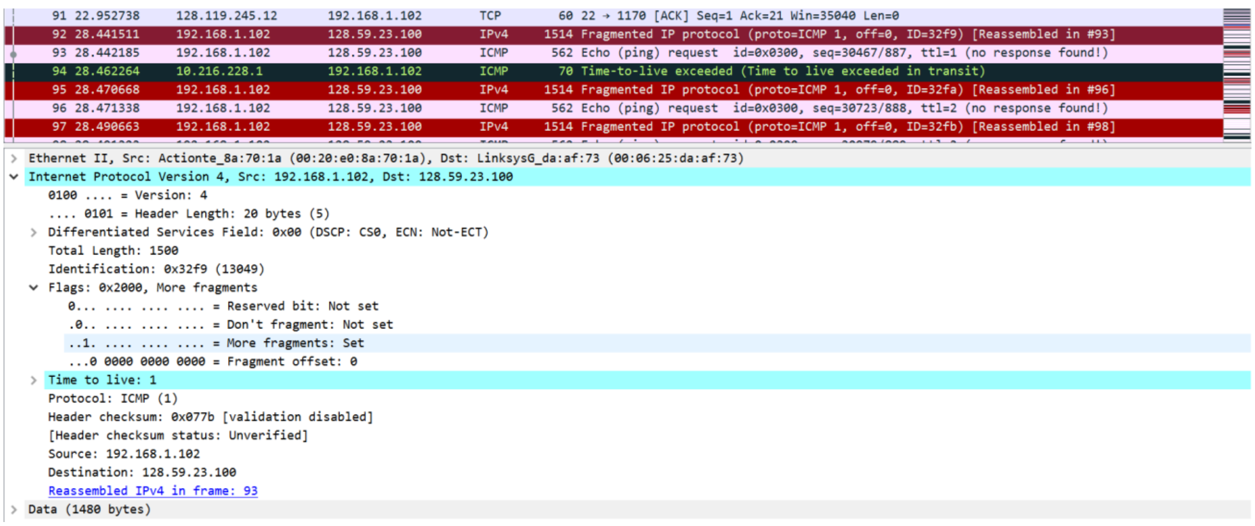


Figure #5: First fragment of the Fragmented Datagram (2000 Bytes)

10. I will be using the provided trace for these questions. Here, the packet was fragmented into 2 IPv4 fragments totalling up to 2000 bytes. (Fig 4)
11. The header has the "More fragments" flag set to 1, meaning it has been fragmented. Looking at the fragment offset field, we see that it is 0, meaning that it is the first fragment. (Fig 5). The unfragmented IP datagram is 2000 bytes but is 2020 bytes long due to headers. The first fragment is 1500 bytes (1480 payload and 20 header).
12. The "More fragments" flag was not set and we see the offset has been correctly set. Not only that, it mentions that the first fragment is #92 (Fig 5) and second fragment is #93 (Fig 4).
13. Between the two fragments, the total length, flags (more fragments and fragment offset), and checksum have changed.

ELEC 331 Assignment 3

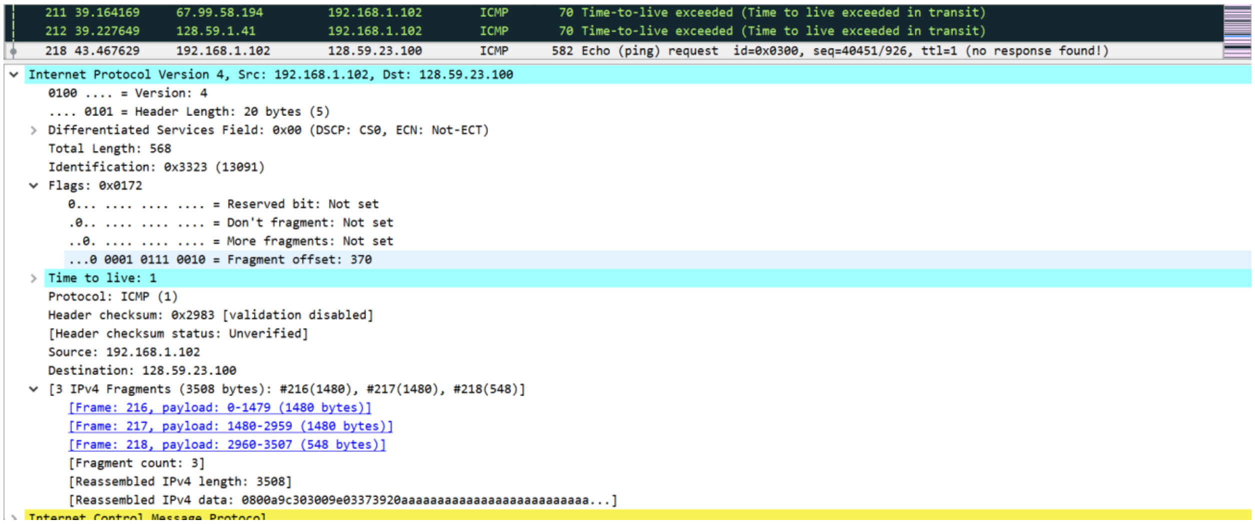


Figure #6: First fragment of the Fragmented Datagram (3500 Bytes)

14. 3 fragments were created.
15. The fields that change among the fragments are:
    a. More fragments
    b. Fragment offset
    c. Checksum
    d. Length (last packet only has 568 while the other have 1500)

ELEC 331 Assignment 3

Wireshark ICMP

```
C:\WINDOWS\system32>ping -n 10 www.ust.hk

Pinging www.ust.hk.w.kunlunsl.com [64.71.142.56] with 32 bytes of data:
Reply from 64.71.142.56: bytes=32 time=9ms TTL=116
Reply from 64.71.142.56: bytes=32 time=9ms TTL=116
Reply from 64.71.142.56: bytes=32 time=11ms TTL=116
Reply from 64.71.142.56: bytes=32 time=9ms TTL=116
Reply from 64.71.142.56: bytes=32 time=23ms TTL=116
Reply from 64.71.142.56: bytes=32 time=10ms TTL=116
Reply from 64.71.142.56: bytes=32 time=12ms TTL=116
Reply from 64.71.142.56: bytes=32 time=9ms TTL=116
Reply from 64.71.142.56: bytes=32 time=11ms TTL=116
Reply from 64.71.142.56: bytes=32 time=9ms TTL=116

Ping statistics for 64.71.142.56:
    Packets: Sent = 10, Received = 10, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 9ms, Maximum = 23ms, Average = 11ms

C:\WINDOWS\system32>
```

Figure #7: Ping results

```
258 33.604664   192.168.1.64    64.71.142.56    ICMP   74 Echo (ping) request  id=0x0001, seq=213/54528, ttl=128 (reply in 259)
259 33.614149   64.71.142.56    192.168.1.64    ICMP   74 Echo (ping) reply    id=0x0001, seq=213/54528, ttl=116 (request in 258)
268 34.608973   192.168.1.64    64.71.142.56    ICMP   74 Echo (ping) request  id=0x0001, seq=214/54784, ttl=128 (reply in 269)

  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 60
    Identification: 0x5b03 (23299)
  v Flags: 0x0000
      0... .... .... .... = Reserved bit: Not set
      .0.. .... .... .... = Don't fragment: Not set
      ..0. .... .... .... = More fragments: Not set
      ...0 0000 0000 0000 = Fragment offset: 0
    Time to live: 128
    Protocol: ICMP (1)
    Header checksum: 0x4f56 [validation disabled]
    [Header checksum status: Unverified]
    Source: 192.168.1.64
    Destination: 64.71.142.56
v Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0x4c86 [correct]
    [Checksum Status: Good]
    Identifier (BE): 1 (0x0001)
    Identifier (LE): 256 (0x0100)
    Sequence number (BE): 213 (0x00d5)
    Sequence number (LE): 54528 (0xd500)
    [Response frame: 259]
  > Data (32 bytes)
```

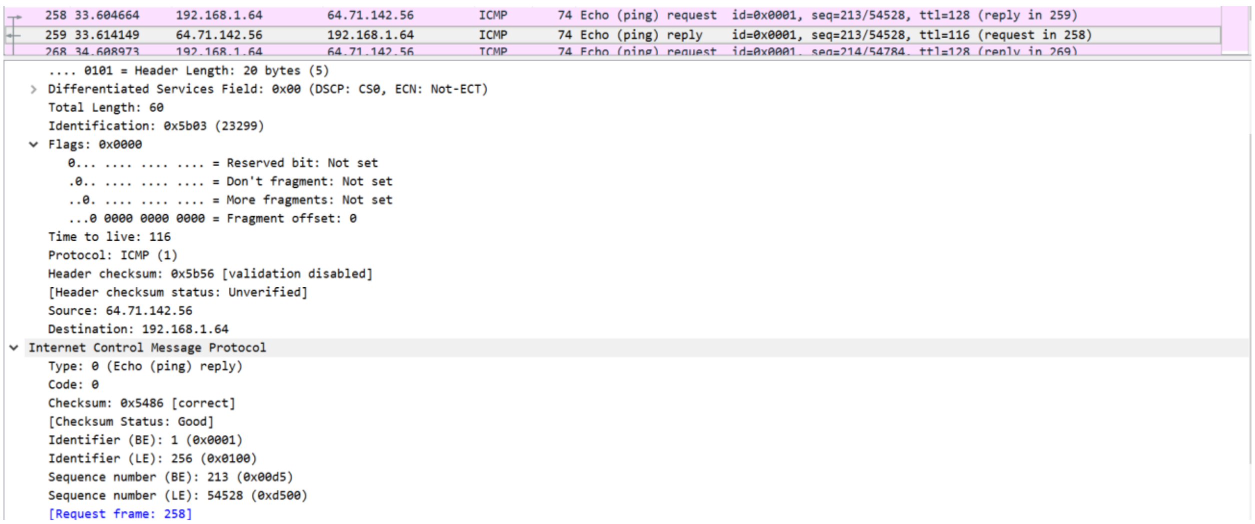
Figure #8: ICMP PING request

1. As before, the IP address of my host is 192.168.1.64 and the IP address of the destination host is 64.71.142.56.
2. An ICMP packet does not have source and destination ports because ICMP is part of the network layer with IP instead of the transport layer with TCP / UDP. As it is an integral part of the network layer, it was designed to communicate network layer information between hosts and routers and not between application layer processes.

ELEC 331 Assignment 3

3. Referring to Fig 8, we see that the Type is 8 for Echo (ping) request and Code is 0. It has a few other fields, namely checksum, identifier, sequence number, and data fields. There are two versions of both sequence number and identifier, for little and big endian.

   The checksum, sequence number, and identifier fields are all 2 bytes each.



```
258 33.604664   192.168.1.64   64.71.142.56   ICMP   74 Echo (ping) request  id=0x0001, seq=213/54528, ttl=128 (reply in 259)
259 33.614149   64.71.142.56   192.168.1.64   ICMP   74 Echo (ping) reply    id=0x0001, seq=213/54528, ttl=116 (request in 258)
268 34.608973   192.168.1.64   64.71.142.56   ICMP   74 Echo (ping) request  id=0x0001, seq=214/54784, ttl=128 (reply in 269)
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 60
    Identification: 0x5b03 (23299)
  v Flags: 0x0000
      0... .... .... .... = Reserved bit: Not set
      .0.. .... .... .... = Don't fragment: Not set
      ..0. .... .... .... = More fragments: Not set
      ...0 0000 0000 0000 = Fragment offset: 0
    Time to live: 116
    Protocol: ICMP (1)
    Header checksum: 0x5b56 [validation disabled]
    [Header checksum status: Unverified]
    Source: 64.71.142.56
    Destination: 192.168.1.64
v Internet Control Message Protocol
    Type: 0 (Echo (ping) reply)
    Code: 0
    Checksum: 0x5486 [correct]
    [Checksum Status: Good]
    Identifier (BE): 1 (0x0001)
    Identifier (LE): 256 (0x0100)
    Sequence number (BE): 213 (0x00d5)
    Sequence number (LE): 54528 (0xd500)
    [Request frame: 258]
```

Figure #9: ICMP PING reply

4. For the reply, Type is 0 for Echo (ping) reply and Code is 0. It has the same fields as the request above, namely checksum, identifier, sequence number, and data. In addition, the checksum, sequence number, and identifier fields are all 2 bytes each.

```
C:\WINDOWS\system32>tracert www.inria.fr

Tracing route to ezp3.inria.fr [128.93.162.84]
over a maximum of 30 hops:

  1     1 ms     1 ms     3 ms  192.168.1.254
  2    14 ms     6 ms     7 ms  10.27.146.1
  3    10 ms     8 ms     7 ms  154.11.12.201
  4     6 ms     7 ms     5 ms  sea-b2-link.telia.net [213.248.74.220]
  5     6 ms     5 ms     6 ms  gtt-ic-328413-sea-b2.c.telia.net [62.115.145.71]
  6   155 ms   144 ms   149 ms  xe-2-0-0.cr0-par7.ip4.gtt.net [213.254.230.2]
  7   149 ms   180 ms   149 ms  renater-gw-ix1.gtt.net [77.67.123.206]
  8   159 ms   154 ms   156 ms  te1-1-inria-rtr-021.noc.renater.fr [193.51.177.107]
  9   158 ms   157 ms   155 ms  inria-rocquencourt-te1-4-inria-rtr-021.noc.renater.fr [193.51.184.177]
 10   174 ms   150 ms   152 ms  unit240-reth1-vfw-ext-dc1.inria.fr [192.93.122.19]
 11   152 ms   155 ms   154 ms  ezp3.inria.fr [128.93.162.84]

Trace complete.
```

Figure #10: Tracert results

ELEC 331 Assignment 3

5. As before, the IP address of my host is 192.168.1.64 and the IP address of the destination host is 128.93.162.84.
6. No, if ICMP sent UDP packets instead, the protocol field in the IP header should be 0x11 for UDP (17).

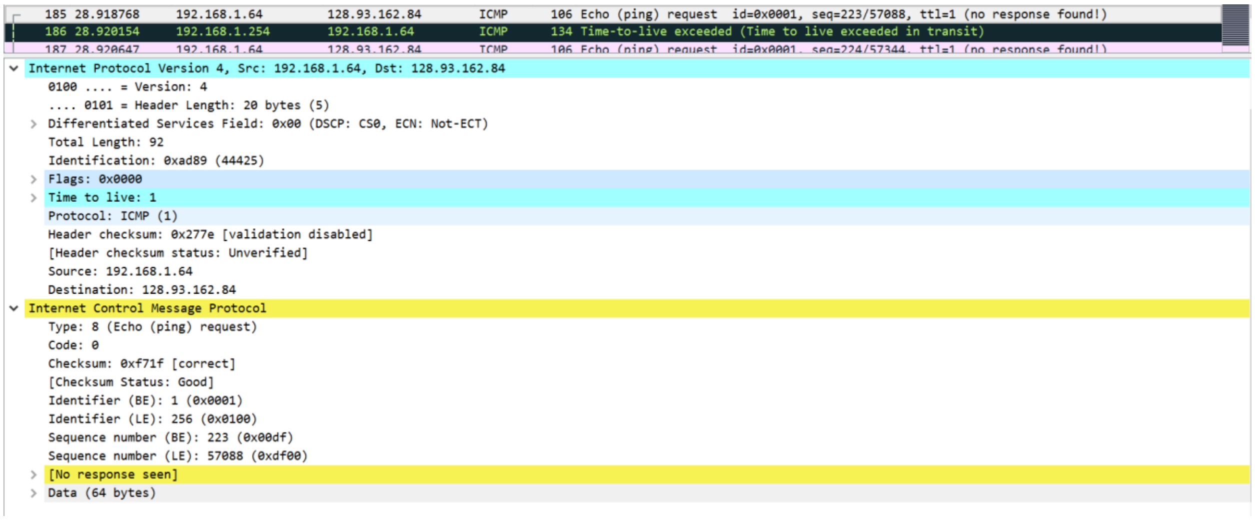


Figure #11: Tracert Request

7. The ICMP echo packet seems to have the same fields as the ping query packets from above.

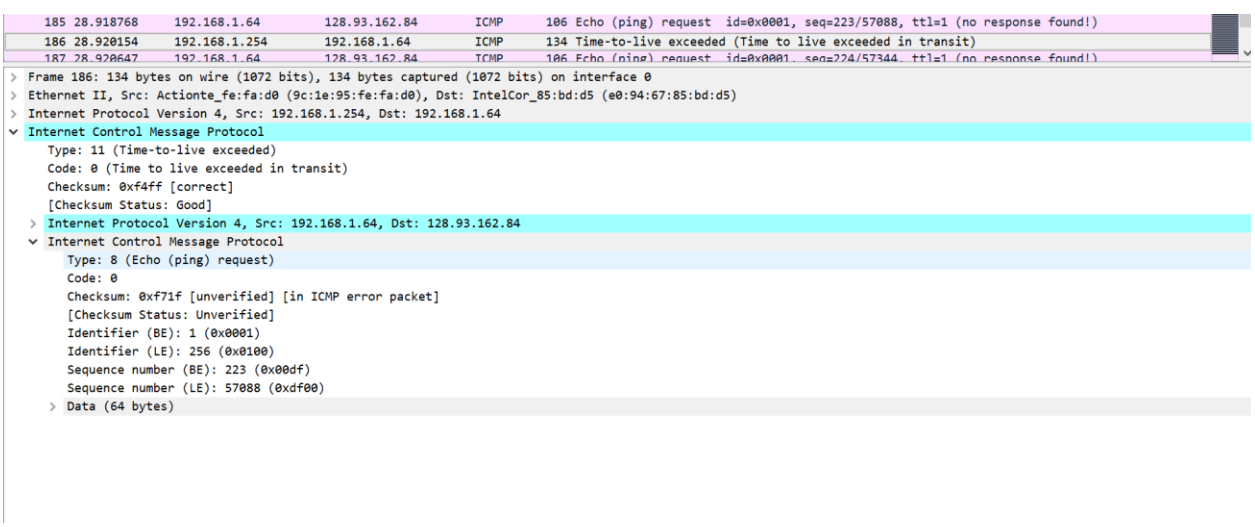


Figure #12: Tracert Error (TTL expired)

8. There are a few extra fields in the ICMP error packet – it includes the IP header and the first 8 bytes of the ICMP request the error packet corresponds to.

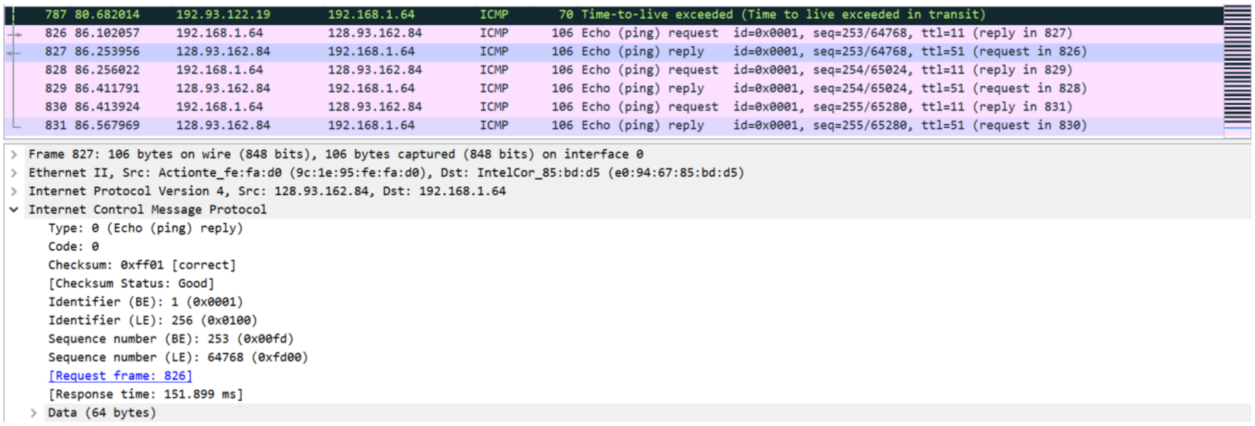


Figure #13: Last three ICMP packets

9. The last three ICMP packets are different as they have Type 0 for Echo (ping) reply instead of 11 for TTL expiration. They are different from the ICMP error packets because these datagrams have reached the destination before the TTL expired.
10. Yes. Referring to Fig 10, we see that between steps 6 and 7, there is a sudden increase in RTT. Based on the router name, I would assume that par7 is referring to Paris as the website is a French website (.fr).

ELEC 331 Assignment 3

Wireshark NAT

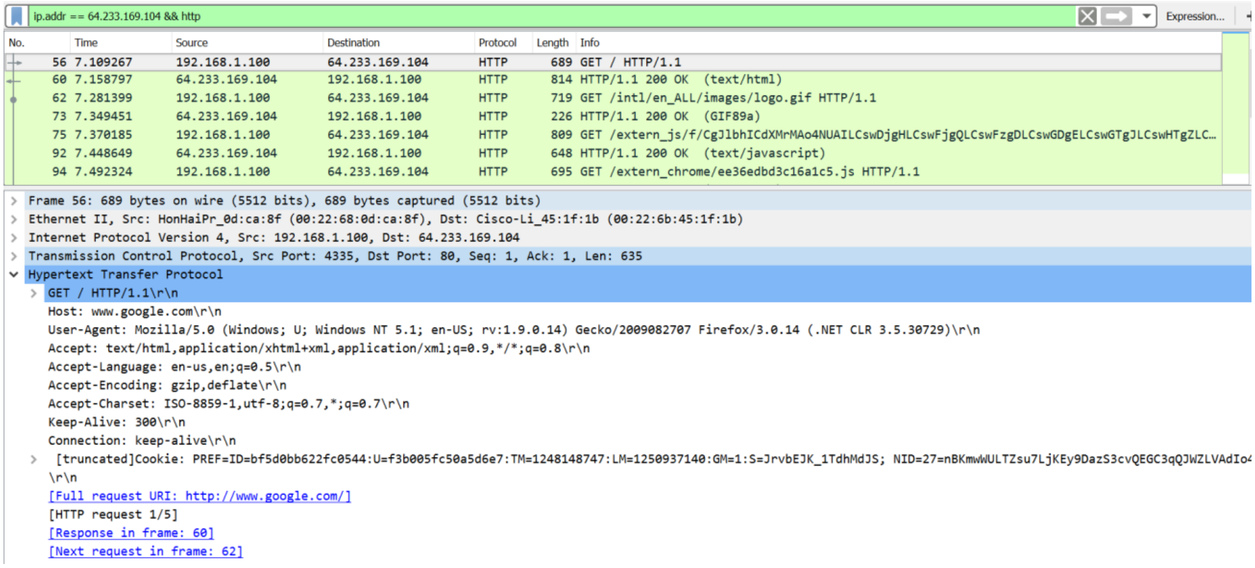


Figure #14: HTTP Request and Filter

1. The IP address of the client is 192.168.1.100
2. Filtering above (Fig 14)
3. Source: 192.168.1.100, 4335 and Destination: 64.233.169.104, 80 (A.B.C.D, Port Number)

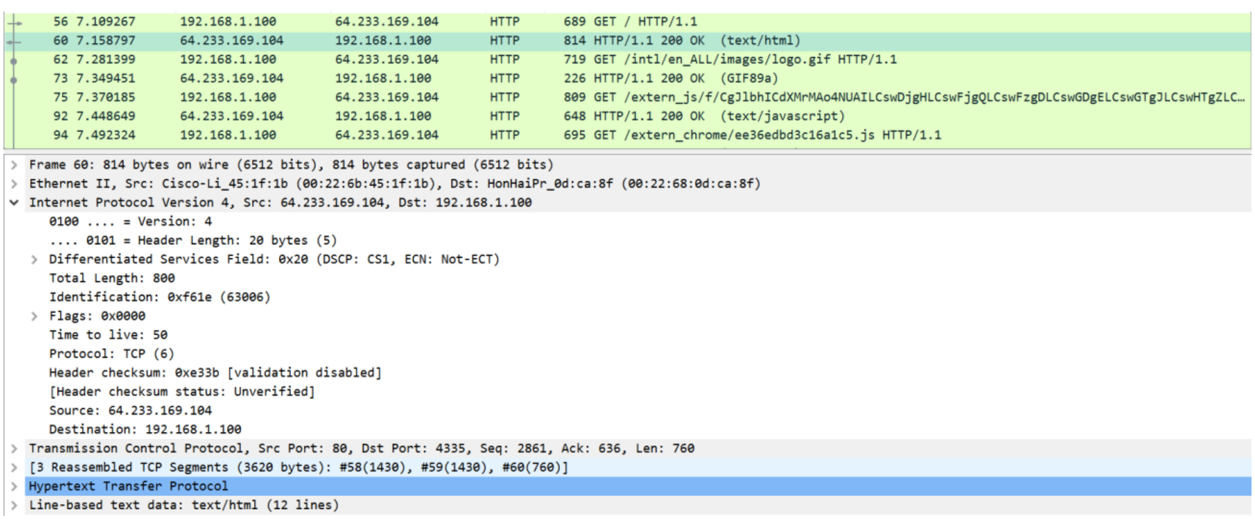


Figure #15: HTTP 200 Response

4. The 200 OK HTTP message was received at time t = 7.158797.
   Source: 64.233.169.104, 80 and Destination: 192.168.1.100, 4335 -> reverse of 3.

ELEC 331 Assignment 3

```
53 7.075657  192.168.1.100   64.233.169.104  TCP  66 4335 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=4 SACK_PERM=1
54 7.108986  64.233.169.104  192.168.1.100   TCP  66 80 → 4335 [SYN, ACK] Seq=0 Ack=1 Win=5720 Len=0 MSS=1430 SACK_PERM=1 WS=64
55 7.109053  192.168.1.100   64.233.169.104  TCP  54 4335 → 80 [ACK] Seq=1 Ack=1 Win=260176 Len=0
```

Figure #16: TCP SYN/ACK segments

5. The TCP SYN segment was sent at t = 7.075657. The TCP ACK was received at t = 7.108986.
    a. TCP SYN segment:
        i. Source: 192.168.1.100, 4335 and Destination: 64.233.169.104, 80
    b. TCP ACK response:
        i. Source: 64.233.169.104, 80 and Destination: 192.168.1.100, 4335

```
 85 6.069168  71.192.34.104   64.233.169.104  HTTP  689 GET / HTTP/1.1
 90 6.117570  64.233.169.104  71.192.34.104   HTTP  814 HTTP/1.1 200 OK  (text/html)
 93 6.241357  71.192.34.104   64.233.169.104  HTTP  719 GET /intl/en_ALL/images/logo.gif HTTP/1.1
103 6.308118  64.233.169.104  71.192.34.104   HTTP  226 HTTP/1.1 200 OK  (GIF89a)
106 6.330131  71.192.34.104   64.233.169.104  HTTP  809 GET /extern_js/f/CgJlbhICdXMrMAo4NUAILCswDjgHLCswFjgQLCswFzgDLCswGDgELCswGTgJLCswHTgZLC…
121 6.407366  64.233.169.104  71.192.34.104   HTTP  648 HTTP/1.1 200 OK  (text/javascript)
125 6.452270  71.192.34.104   64.233.169.104  HTTP  695 GET /extern_chrome/ee36edbd3c16a1c5.js HTTP/1.1

> Frame 85: 689 bytes on wire (5512 bits), 689 bytes captured (5512 bits)
> Ethernet II, Src: Dell_4f:36:23 (00:08:74:4f:36:23), Dst: Cisco_bf:6c:01 (00:0e:d6:bf:6c:01)
> Internet Protocol Version 4, Src: 71.192.34.104, Dst: 64.233.169.104
> Transmission Control Protocol, Src Port: 4335, Dst Port: 80, Seq: 1, Ack: 1, Len: 635
v Hypertext Transfer Protocol
  > GET / HTTP/1.1\r\n
    Host: www.google.com\r\n
    User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.0.14) Gecko/2009082707 Firefox/3.0.14 (.NET CLR 3.5.30729)\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
    Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7\r\n
    Keep-Alive: 300\r\n
    Connection: keep-alive\r\n
  > [truncated]Cookie: PREF=ID=bf5d0bb622fc0544:U=f3b005fc50a5d6e7:TM=1248148747:LM=1250937140:GM=1:S=JrvbEJK_1TdhMdJS; NID=27=nBKmwWULTZsu7LjKEy9DazS3cvQEGC3qQJWZLVA
    \r\n
    [Full request URI: http://www.google.com/]
    [HTTP request 1/5]
    [Response in frame: 90]
    [Next request in frame: 93]
```

Figure #17: NAT ISP Side of the GET request

6. The request appears on the ISP side at t = 6.069168. For this request, the destination IP and Port and source Port are the same, only the source IP has changed.
    a. Source: 71.192.34.104, 4335 and Destination: 64.233.169.104, 80

ELEC 331 Assignment 3

No.	Time	Source	Destination	Protocol	Length	Info
56	7.109267	192.168.1.100	64.233.169.104	HTTP	689	GET / HTTP/1.1
60	7.158797	64.233.169.104	192.168.1.100	HTTP	814	HTTP/1.1 200 OK (text/html)
62	7.281399	192.168.1.100	64.233.169.104	HTTP	719	GET /intl/en_ALL/images/logo.gif HTTP/1.1
73	7.349451	64.233.169.104	192.168.1.100	HTTP	226	HTTP/1.1 200 OK (GIF89a)
75	7.370185	192.168.1.100	64.233.169.104	HTTP	809	GET /extern_js/f/CgJlbhICdXMrMAo4NUAILCswDjgHLCswFjgQLCswFzgDLCswGDgELCswGTgJLCswHTgZLC…
92	7.448649	64.233.169.104	192.168.1.100	HTTP	648	HTTP/1.1 200 OK (text/javascript)
94	7.492324	192.168.1.100	64.233.169.104	HTTP	695	GET /extern_chrome/ee36edbd3c16a1c5.js HTTP/1.1

```
> Frame 56: 689 bytes on wire (5512 bits), 689 bytes captured (5512 bits)
> Ethernet II, Src: HonHaiPr_0d:ca:8f (00:22:68:0d:ca:8f), Dst: Cisco-Li_45:1f:1b (00:22:6b:45:1f:1b)
v Internet Protocol Version 4, Src: 192.168.1.100, Dst: 64.233.169.104
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 675
    Identification: 0xa2ac (41644)
  > Flags: 0x4000, Don't fragment
    Time to live: 128
    Protocol: TCP (6)
    Header checksum: 0xa94a [validation disabled]
    [Header checksum status: Unverified]
    Source: 192.168.1.100
    Destination: 64.233.169.104
```

Figure #18: NAT Home GET Request IP Datagram

No.	Time	Source	Destination	Protocol	Length	Info
85	6.069168	71.192.34.104	64.233.169.104	HTTP	689	GET / HTTP/1.1
90	6.117570	64.233.169.104	71.192.34.104	HTTP	814	HTTP/1.1 200 OK (text/html)
93	6.241357	71.192.34.104	64.233.169.104	HTTP	719	GET /intl/en_ALL/images/logo.gif HTTP/1.1
103	6.308118	64.233.169.104	71.192.34.104	HTTP	226	HTTP/1.1 200 OK (GIF89a)
106	6.330131	71.192.34.104	64.233.169.104	HTTP	809	GET /extern_js/f/CgJlbhICdXMrMAo4NUAILCswDjgHLCswFjgQLCswFzgDLCswGDgELCswGTgJLCswHTgZLC…
121	6.407366	64.233.169.104	71.192.34.104	HTTP	648	HTTP/1.1 200 OK (text/javascript)
125	6.452270	71.192.34.104	64.233.169.104	HTTP	695	GET /extern_chrome/ee36edbd3c16a1c5.js HTTP/1.1

```
> Frame 85: 689 bytes on wire (5512 bits), 689 bytes captured (5512 bits)
> Ethernet II, Src: Dell_4f:36:23 (00:08:74:4f:36:23), Dst: Cisco_bf:6c:01 (00:0e:d6:bf:6c:01)
v Internet Protocol Version 4, Src: 71.192.34.104, Dst: 64.233.169.104
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 675
    Identification: 0xa2ac (41644)
  > Flags: 0x4000, Don't fragment
    Time to live: 127
    Protocol: TCP (6)
    Header checksum: 0x022f [validation disabled]
    [Header checksum status: Unverified]
    Source: 71.192.34.104
    Destination: 64.233.169.104
```

Figure #19: NAT ISP GET Request IP Datagram

7. No, nothing in the HTTP GET message was changed. This can be confirmed by referring to Figures 14 and 17. Regarding the IP datagram carrying the HTTP GET, the Version, Header Length, and Flags were not changed. The only thing that was changed was checksum as both IP address and TTL changed. These changes come from a NAT table lookup.

ELEC 331 Assignment 3

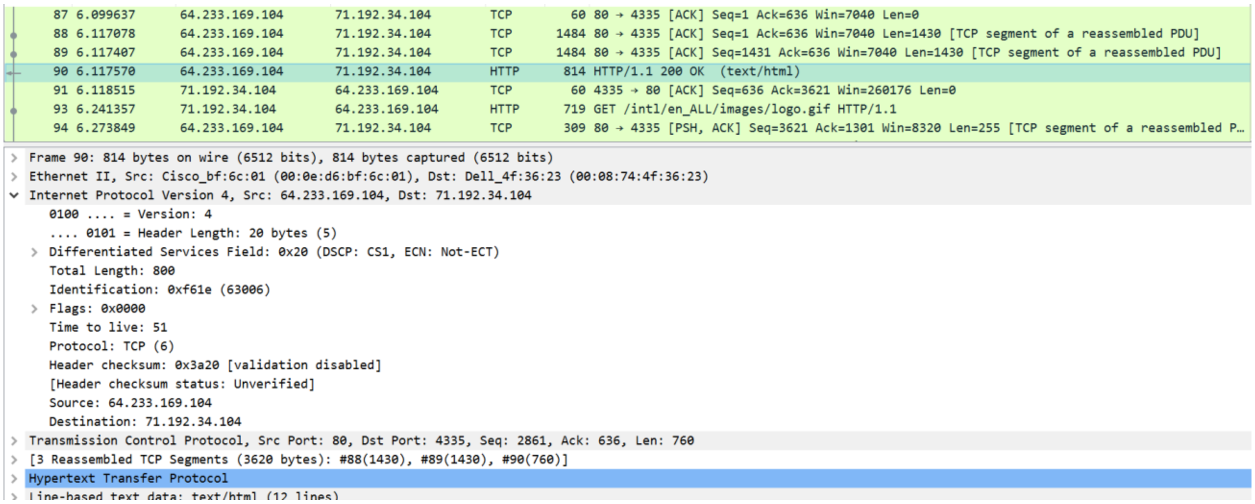


Figure #20: NAT ISP 200 OK IP Datagram

8. The first 200 OK HTTP message came arrived at t = 6.117570. Comparing Figures 15 and 20, it looks like the only things that are different are the Destination IP address, TTL, and checksum.
    a. Source: 64.233.169.104, 80 and Destination: 71.192.34.104, 4335

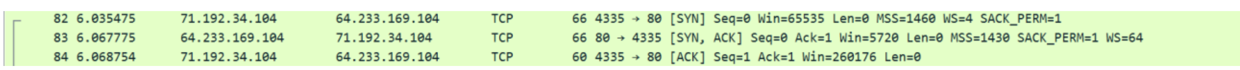


Figure #21: TCP SYN/ACK Segments

9. The TCP SYN segment was captured at t = 6.035475 and the TCP ACK segment was captured at t = 6.067775.
    a. TCP SYN:
        i. Source IP changed
    b. TCP ACK:
        i. Destination IP changed
10. NAT Table

NAT translation table	
WAN side addr	LAN side addr
71.192.34.104, 4335	192.168.1.100, 4335

ELEC 331 Assignment 3

Wireshark Ethernet and ARP

Note: I will be using the provided ethereal trace for this part of the assignment.

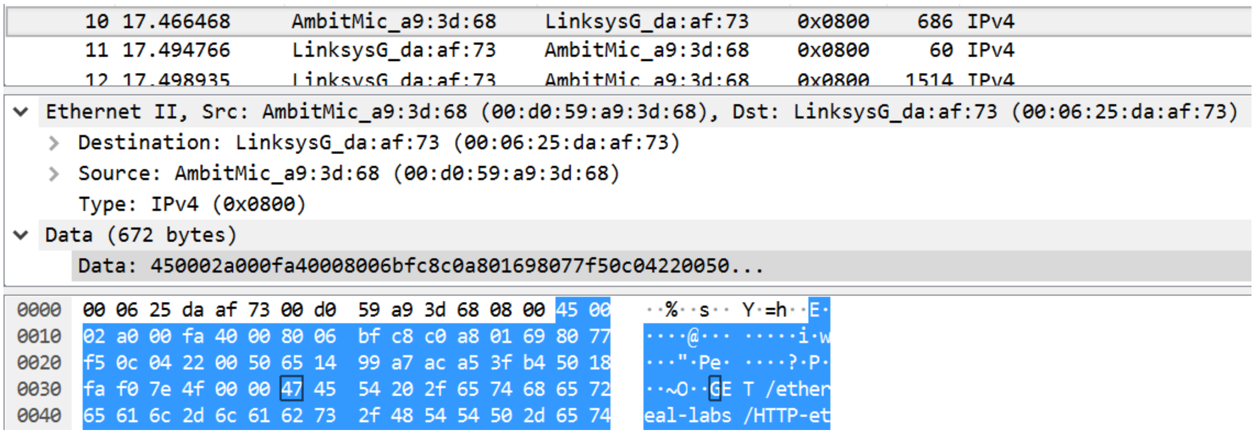


Figure #22: Ethernet Frame for HTTP GET Request

1. The 48-bit Ethernet address of my computer is 00:d0:59:a9:3d:68.
2. The 48-bit destination Ethernet address is 00:06:25:da:af:73. This corresponds to the Linksys router that is used to get off the local subnet.
3. The hexadecimal value for the Frame Type is 0x0800, corresponding to the IPv4 protocol.
4. As the ASCII G is the first thing in the payload, it would appear 54 bytes from the very start of the Ethernet frame.

   This is because the HTTP GET message is carried inside of a TCP segment, which is carried inside an IP datagram, which is finally carried inside the Ethernet frame. Thus, we must consider 20 bytes of header from the TCP segment, 20 bytes from the IP datagram, and finally 14 bytes for Ethernet frame's type and source and destination address. Looking at figure 22, we see that the G in GET appears at 0x37 (55), which has 54 bytes in front of it.

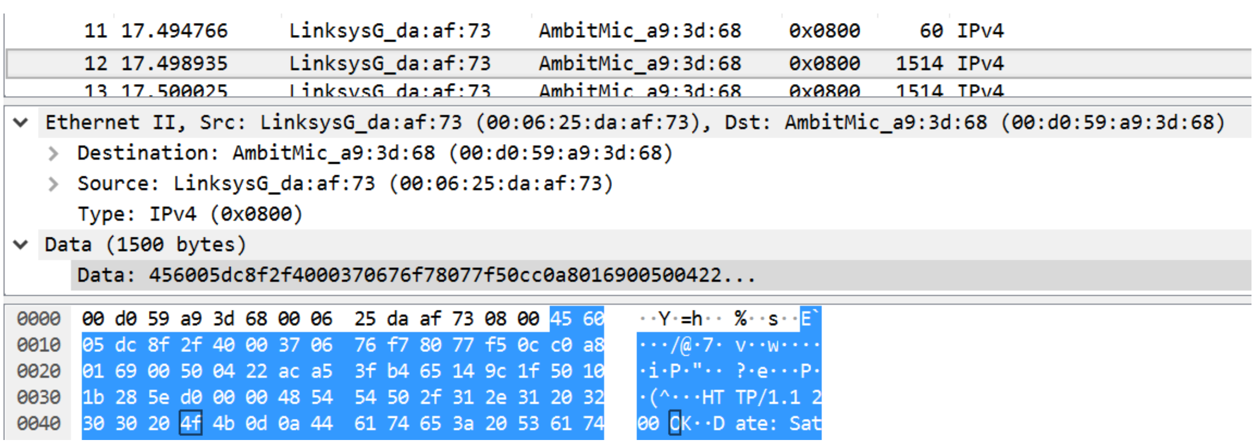


Figure #23: Ethernet Frame for HTTP 200 OK Message

5. The Ethernet source address is 00:06:25:da:af:73, which is the Ethernet address of the Linksys router as mentioned in Question 2.
6. The Ethernet destination address is 00:d0:59:a9:3d:68, the address of my computer.
7. The hexadecimal value for the Frame Type is 0x0800, corresponding to the IPv4 protocol.
8. Similar to Question 4, as the O in OK is the first character. However, we see that it appears at 0x44 (67), which has 63 bytes in front of it. Thus, there must be additional headers or something of that sort.

```
C:\WINDOWS\system32>arp -a

Interface: 192.168.124.1 --- 0x3
  Internet Address      Physical Address      Type
  192.168.124.254       00-50-56-e3-05-c8     dynamic
  192.168.124.255       ff-ff-ff-ff-ff-ff     static
  224.0.0.2             01-00-5e-00-00-02     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.3.22          01-00-5e-7f-03-16     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static

Interface: 192.168.1.64 --- 0x7
  Internet Address      Physical Address      Type
  192.168.1.65          4c-8b-30-9e-cc-40     dynamic
  192.168.1.69          38-8b-59-87-c1-6a     dynamic
  192.168.1.254         9c-1e-95-fe-fa-d0     dynamic
  192.168.1.255         ff-ff-ff-ff-ff-ff     static
  224.0.0.2             01-00-5e-00-00-02     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.3.22          01-00-5e-7f-03-16     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static

Interface: 192.168.30.1 --- 0x14
  Internet Address      Physical Address      Type
  192.168.30.254        00-50-56-ef-48-98     dynamic
  192.168.30.255        ff-ff-ff-ff-ff-ff     static
  224.0.0.2             01-00-5e-00-00-02     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.3.22          01-00-5e-7f-03-16     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static

C:\WINDOWS\system32>
```

Figure #24: ARP Table

ELEC 331 Assignment 3

9.

Internet Address	IP Address
Physical Address	MAC Address
Type	Protocol Type

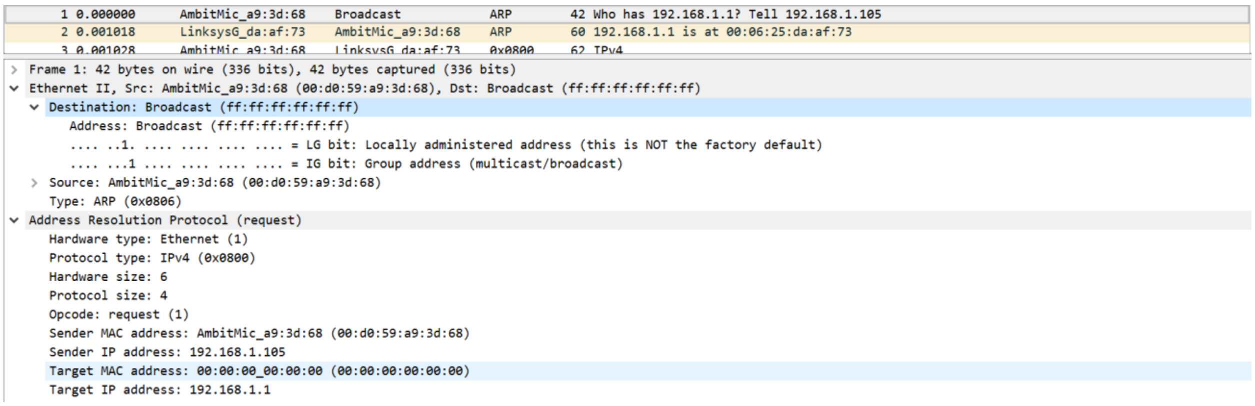


Figure #25: ARP Request Message

10.

a. Source: 00:d0:59:a9:3d:68.
b. Destination: ff:ff:ff:ff:ff:ff, for broadcast.

11. The hexadecimal value for the Frame Type is 0x806, corresponding to the ARP protocol.

12.

a. The ARP opcode field begins 20 bytes from the very beginning of the Ethernet frame.
b. Referring to fig 25, we see that the opcode is 1 (0x0001) for request.
c. Yes, the ARP message contains the “Sender IP address” field.
d. In the ARP message, the Target IP address field corresponds to the IP address being queried, and thus the Target MAC address field is set to 00:00:00:00:00:00 to “question” the machine whose IP is the value of the Target IP address field.

ELEC 331 Assignment 3

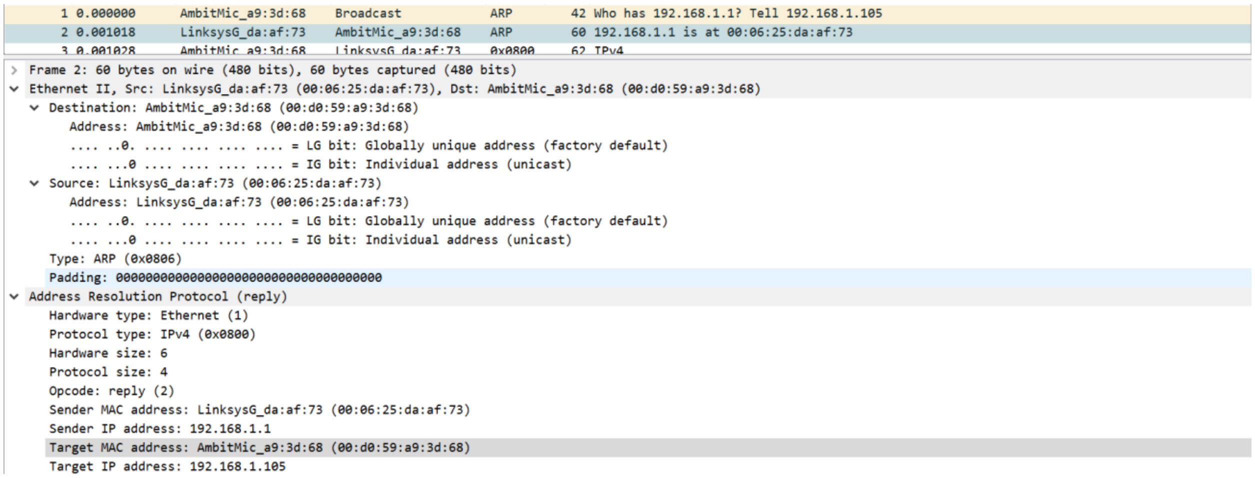


Figure #26: ARP Response Message

13.
  a. Same answer as question 12 a), being 20 bytes from the very beginning.
  b. Referring to fig 25, we see that the opcode is 2 (0x0002) for reply
  c. The “answer” to the previous query appears in the “Sender MAC address” field.

14.
  a. Source: 00:06:25:da:af:73
  b. Destination: 00:d0:59:a9:3d:68

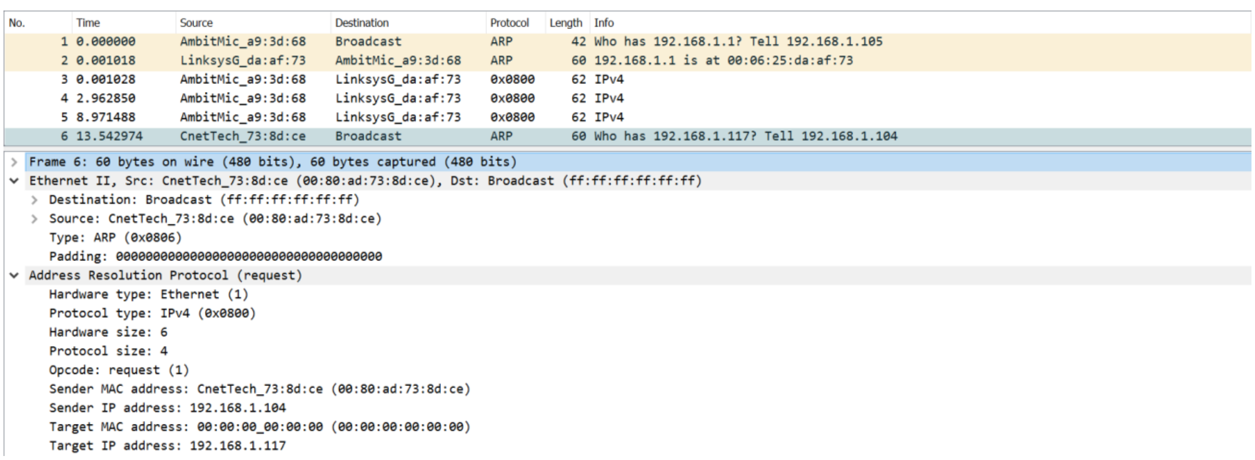


Figure #26: Unreplied ARP Request Message

15. There is no reply on our trace because while the ARP request is similar to the DHCP request in that they are both broadcasts, the ARP reply is a unicast reply. As the ARP request was not

made by us (different source IP and MAC address), we are unable to capture the ARP reply packet.