네트워크실험 2주차
Experiments on Communication Networks

Contents

- **실험 1: TCP**
  - Capturing a bulk TCP transfer from your computer to a remote server
  - A first look at the captured trace
  - TCP Basics
  - TCP congestion control in action

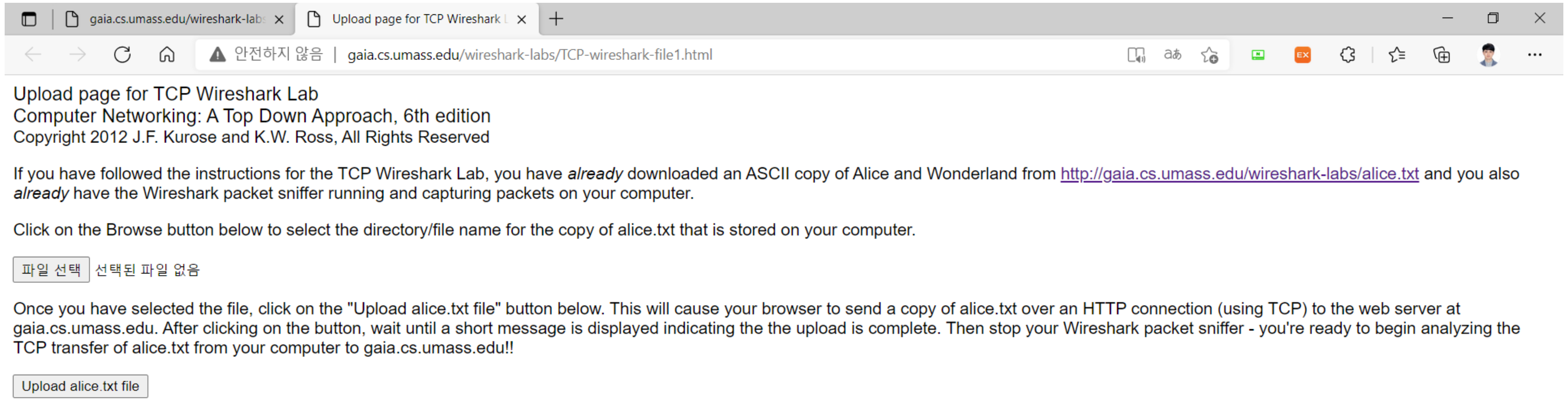
- **실험 2: UDP**
  - The Assignment

- **실험 3: IP**
  - Capturing packets from an execution of traceroute
  - Basic IPv4
  - Fragmentation

실험 1: TCP

■ Capturing a bulk TCP transfer from your computer to a remote server

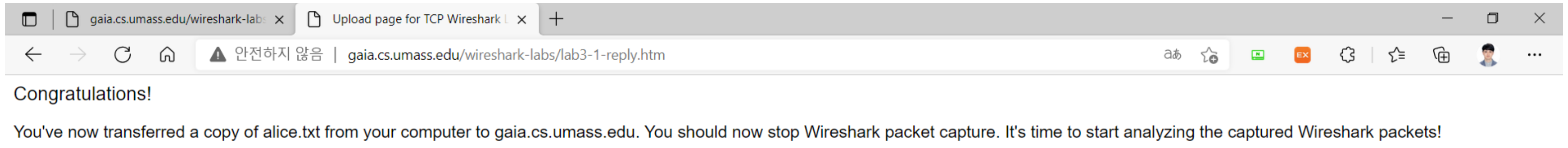
- Start up your web browser. Go the http://gaia.cs.umass.edu/wiresharklabs/alice.txt and retrieve an ASCII copy of *Alice in Wonderland*. Store this file somewhere on your computer.
- Next go to http://gaia.cs.umass.edu/wireshark-labs/TCP-wireshark-file1.html.
- You should see a screen that looks like:



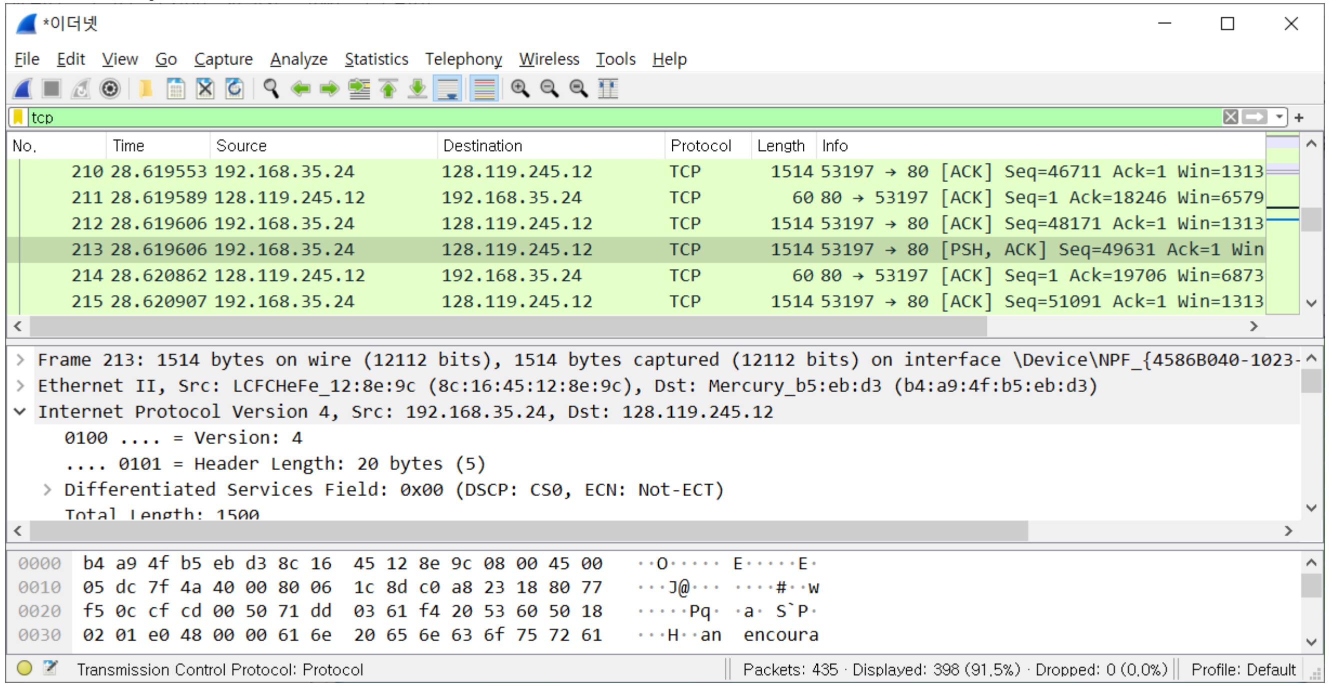
- Use the *Browse* button in this form to enter the name of the file (full path name) on your computer containing *Alice in Wonderland* (or do so manually). Don’t yet press the “*Upload alice.txt file*” button.
- Now start up Wireshark and begin packet capture (*Capture->Start*) and then press *OK* on the Wireshark Packet Capture Options screen (we’ll not need to select any options here).

실험 1: TCP

- **Capturing a bulk TCP transfer from your computer to a remote server**
  - Returning to your browser, press the "*Upload alice.txt file*" button to upload the file to the gaia.cs.umass.edu server. Once the file has been uploaded, a short congratulations message will be displayed in your browser window.



  - Stop Wireshark packet capture. Your Wireshark window should look similar to the window shown below.



실험 1: TCP

- **A first look at the captured trace**
  - Answer the following questions, by opening the Wireshark captured packet file tcp-ethereal-trace-1 in http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip.
  - 1. What is the IP address and TCP port number used by the client computer (source) that is transferring the file to gaia.cs.umass.edu? To answer this question, it's probably easiest to select an HTTP message and explore the details of the TCP packet used to carry this HTTP message, using the "details of the selected packet header window".
  - 2. What is the IP address of gaia.cs.umass.edu? On what port number is it sending and receiving TCP segments for this connection?

- **TCP Basics**
  - 3. What is the sequence number of the TCP SYN segment that is used to initiate the TCP connection between the client computer and gaia.cs.umass.edu? What is it in the segment that identifies the segment as a SYN segment?

  - 4. What is the sequence number of the SYNACK segment sent by gaia.cs.umass.edu to the client computer in reply to the SYN? What is the value of the Acknowledgement field in the SYNACK segment? How did gaia.cs.umass.edu determine that value? What is it in the segment that identifies the segment as a SYNACK segment?

  - 5. 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.

실험 1: TCP

TCP Basics

- 6. Consider the TCP segment containing the HTTP POST as the first segment in the TCP connection. 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? When was the ACK for each segment received? 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? What is the `EstimatedRTT` value after the receipt of each ACK? Assume that the value of the `EstimatedRTT` is equal to the measured RTT for the first segment, and then is computed using the `EstimatedRTT` equation below for all subsequent segments.
- *Note*: Wireshark has a nice feature that allows you to plot the RTT for each of the TCP segments sent. Select a TCP segment in the "listing of captured packets" window that is being sent from the client to the gaia.cs.umass.edu server. Then select: *Statistics->TCP Stream Graph->Round Trip Time Graph*.

```
EstimatedRTT = 0.875 * EstimatedRTT + 0.125 * SampleRTT
```

실험 1: TCP

- **TCP Basics**
  - 7. What is the length of each of the first six TCP segments?
  - 8. What is the minimum amount of available buffer space advertised at the received for the entire trace? Does the lack of receiver buffer space ever throttle the sender?
  - 9. Are there any retransmitted segments in the trace file? What did you check for (in the trace) in order to answer this question?
  - 10. How much data does the receiver typically acknowledge in an ACK? Can you identify cases where the receiver is ACKing every other received segment.
  - 11. What is the throughput (bytes transferred per unit time) for the TCP connection? Explain how you calculated this value.

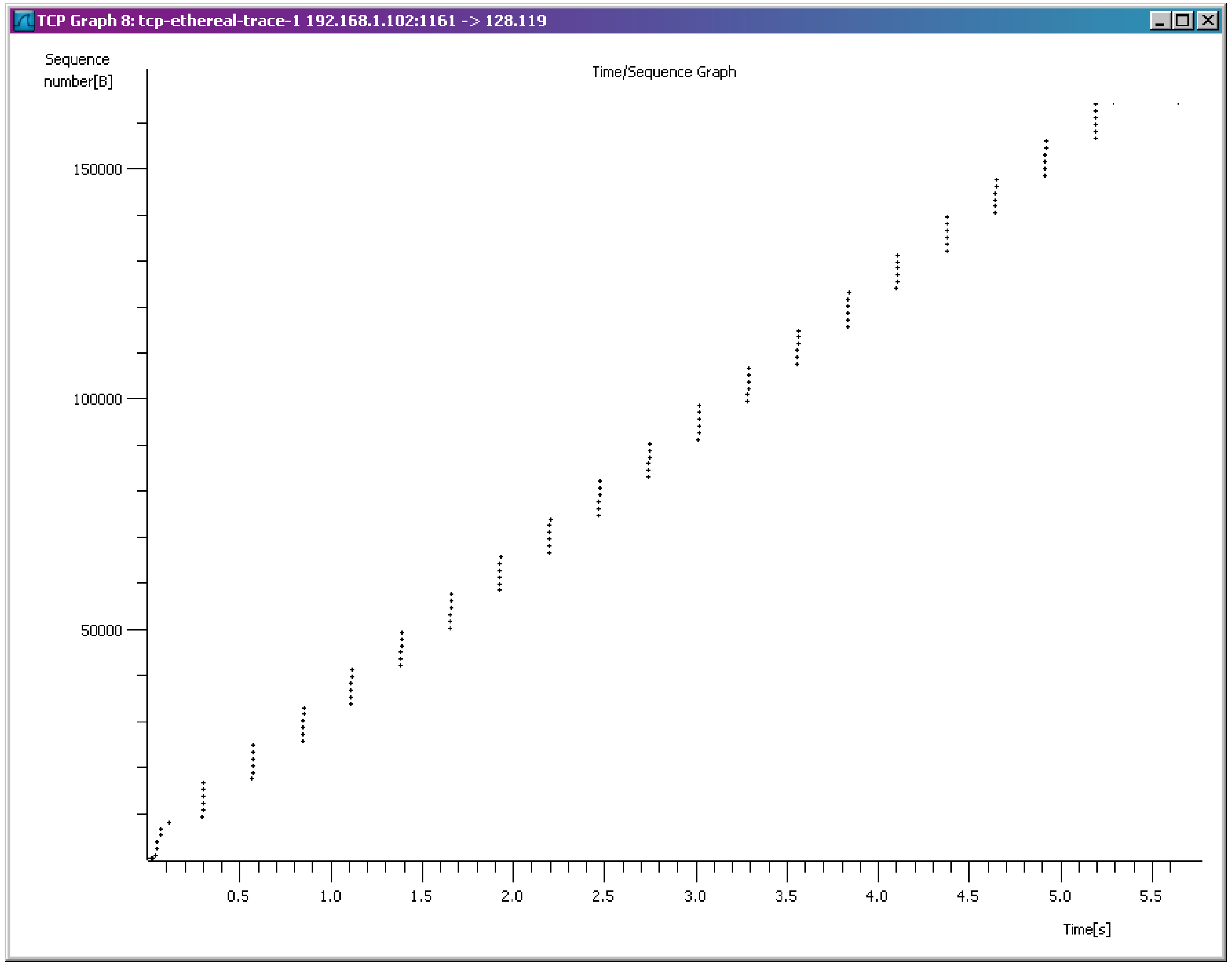
실험 1: TCP

- **TCP congestion control in action**
  - Select a TCP segment in the Wireshark's "listing of captured-packets" window. Then select the menu : Statistics->TCP Stream Graph-> Time-Sequence-Graph(Stevens). You should see a plot that looks similar to the following plot, which was created from the captured packets in the packet trace tcp-etherealtrace-1 in http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip:

  - 12. Use the *Time-Sequence-Graph(Stevens)* plotting tool to view the sequence number versus time plot of segments being sent from the client to the gaia.cs.umass.edu server. Can you identify where TCP's slowstart phase begins and ends, and where congestion avoidance takes over? Comment on ways in which the measured data differs from the idealized behavior of TCP that we've studied in the text.

실험 1: TCP

- TCP congestion control in action



실험 2: UDP

- **The Assignment**
  - Start capturing packets in Wireshark and then do something that will cause your host to send and receive several UDP packets.
  - After stopping packet capture, set your packet filter so that Wireshark only displays the UDP packets sent and received at your host.
  - Pick one of these UDP packets and expand the UDP fields in the details window.
  - If you are unable to find UDP packets or are unable to run Wireshark on a live network connection, you can download a packet trace containing some UDP packets.
  - Download the zip file http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip and extract the file http-ethereal-trace-5, which contains some UDP packets carrying SNMP messages.

실험 2: UDP

- **The Assignment**
  - 1. Select one UDP packet from your trace. From this packet, determine how many fields there are in the UDP header. Name these fields.
  - 2. By consulting the displayed information in Wireshark's packet content field for this packet, determine the length (in bytes) of each of the UDP header fields.
  - 3. The value in the Length field is the length of what? (You can consult the text for this answer). Verify your claim with your captured UDP packet.
  - 4. What is the maximum number of bytes that can be included in a UDP payload? (Hint: the answer to this question can be determined by your answer to 2. above)
  - 5. What is the largest possible source port number? (Hint: see the hint in 4.)
  - 6. 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 Protocol field of the IP datagram containing this UDP segment.

실험 3: IP

- **Capturing packets from an execution of traceroute**
  - The `tracert` program provided with Windows does not allow one to change the size of the ICMP message sent by `tracert`. So it won't be possible to use a Windows machine to generate ICMP messages that are large enough to force IP fragmentation. However, you can use `tracert` to generate small, fixed length packets to perform Part 1 of this lab. At the DOS command prompt enter:

```
>tracert gaia.cs.umass.edu
```

  - If you want to do the second part of this lab, you can download a packet trace file that was captured.
    - You can download the zip file http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces-8.1.zip and extract the trace file *ip-wireshark-trace1-1.pcapng*. This trace file can be used to answer these Wireshark lab questions without actually capturing packets on your own. Once you've downloaded a trace file, you can load it into Wireshark and view the trace using the *File* pull down menu, choosing *Open*, and then selecting the trace file name.

실험 3: IP

- **Capturing packets from an execution of traceroute**
  - Start up Wireshark and begin packet capture. (*Capture->Start* or click on the blue shark fin button in the top left of the Wireshark window).
  - Enter two `traceroute` commands, using gaia.cs.umass.edu as the destination, the first with a length of 56 bytes. Once that command has finished executing, enter a second `traceroute` command for the same destination, but with a length of 3000 bytes.
  - Stop Wireshark tracing.
  - If you're unable to run Wireshark on a live network connection, you can use the packet trace file, *ip-wireshark-trace1-1.pcapng*. You may well find it valuable to download this trace even if you've captured your own trace and use it, as well as your own trace, as you explore the questions.

실험 3: IP

■ **Capturing packets from an execution of traceroute**

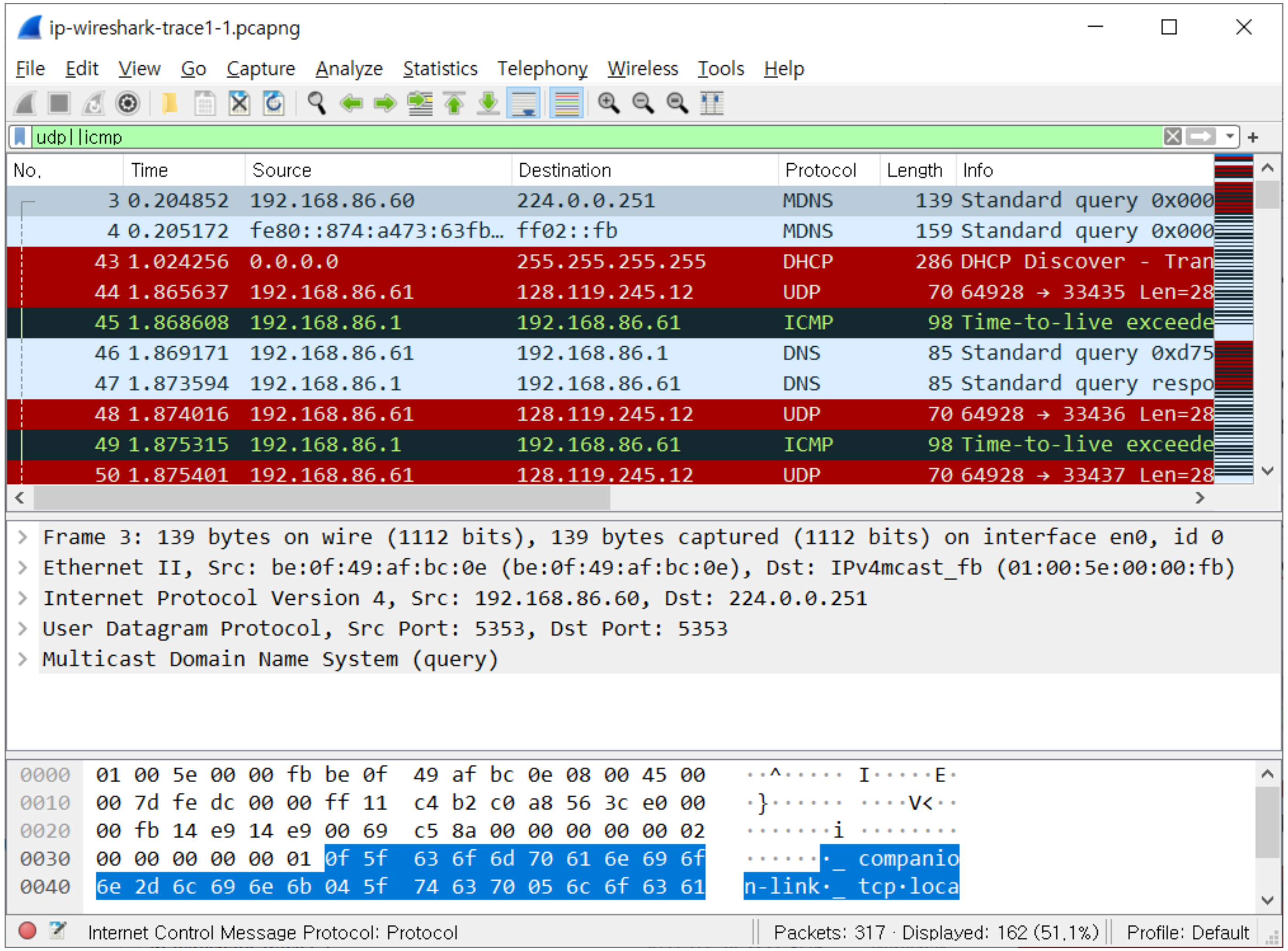
선택 C:₩WINDOWS₩system32₩cmd.exe

```
Microsoft Windows [Version 10.0.19043.1165]
(c) Microsoft Corporation. All rights reserved.

C:₩Users₩wonjo>tracert gaia.cs.umass.edu

최대 30홉 이상의
gaia.cs.umass.edu [128.119.245.12](으)로 가는 경로 추적:

  1    <1 ms    <1 ms    <1 ms  192.168.35.1
  2     *        *        *     요청 시간이 만료되었습니다.
  3     *        *        *     요청 시간이 만료되었습니다.
  4     2 ms     2 ms     1 ms  100.84.124.129
  5     1 ms     1 ms     1 ms  100.84.124.57
  6     1 ms     1 ms     2 ms  10.66.252.212
  7     3 ms     3 ms     2 ms  10.222.24.232
  8     2 ms     2 ms     2 ms  1.255.76.109
  9     5 ms     2 ms     2 ms  211.176.50.153
 10   133 ms   132 ms   132 ms  58.229.4.175
 11     *        *        *     요청 시간이 만료되었습니다.
 12   130 ms   130 ms   129 ms  be3360.ccr42.lax01.atlas.cogentco.com [154.54.25.149]
 13   146 ms   142 ms   141 ms  be2932.ccr32.phx01.atlas.cogentco.com [154.54.45.161]
 14   150 ms   151 ms   151 ms  be2930.ccr21.elp01.atlas.cogentco.com [154.54.42.78]
 15   168 ms   168 ms   167 ms  be2927.ccr41.iah01.atlas.cogentco.com [154.54.29.221]
 16   179 ms   180 ms   179 ms  be2687.ccr41.atl01.atlas.cogentco.com [154.54.28.69]
 17   197 ms   197 ms   197 ms  be2112.ccr41.dca01.atlas.cogentco.com [154.54.7.157]
 18   197 ms   197 ms   197 ms  be2806.ccr41.jfk02.atlas.cogentco.com [154.54.40.105]
 19   203 ms   202 ms   202 ms  be3471.ccr31.bos01.atlas.cogentco.com [154.54.40.153]
 20   204 ms   204 ms   204 ms  be2729.rcr51.orh01.atlas.cogentco.com [154.54.40.182]
 21   197 ms   197 ms   196 ms  38.104.218.14
 22   192 ms   192 ms   192 ms  69.16.0.8
```

실험 3: IP

- **Basic IPv4**
  - In your trace, you should be able to see the series of UDP segments (in the case of MacOS/Linux) or ICMP Echo Request messages (Windows) sent by `traceroute` on your computer, and the ICMP TTL-exceeded messages returned to your computer by the intermediate routers.
  - 1. Select the first UDP segment sent by your computer via the traceroute command to gaia.cs.umass.edu. Expand the Internet Protocol part of the packet in the packet details window. What is the IP address of your computer?
  - 2. What is the value in the upper layer protocol field in this IPv4 datagram's header? [Note: the answers for Linux/MacOS differ from Windows here].
  - 3. How many bytes are in the IP header?
  - 4. How many bytes are in the payload of the IP datagram? Explain how you determined the number of payload bytes.
  - 5. Has this IP datagram been fragmented? Explain how you determined whether or not the datagram has been fragmented.

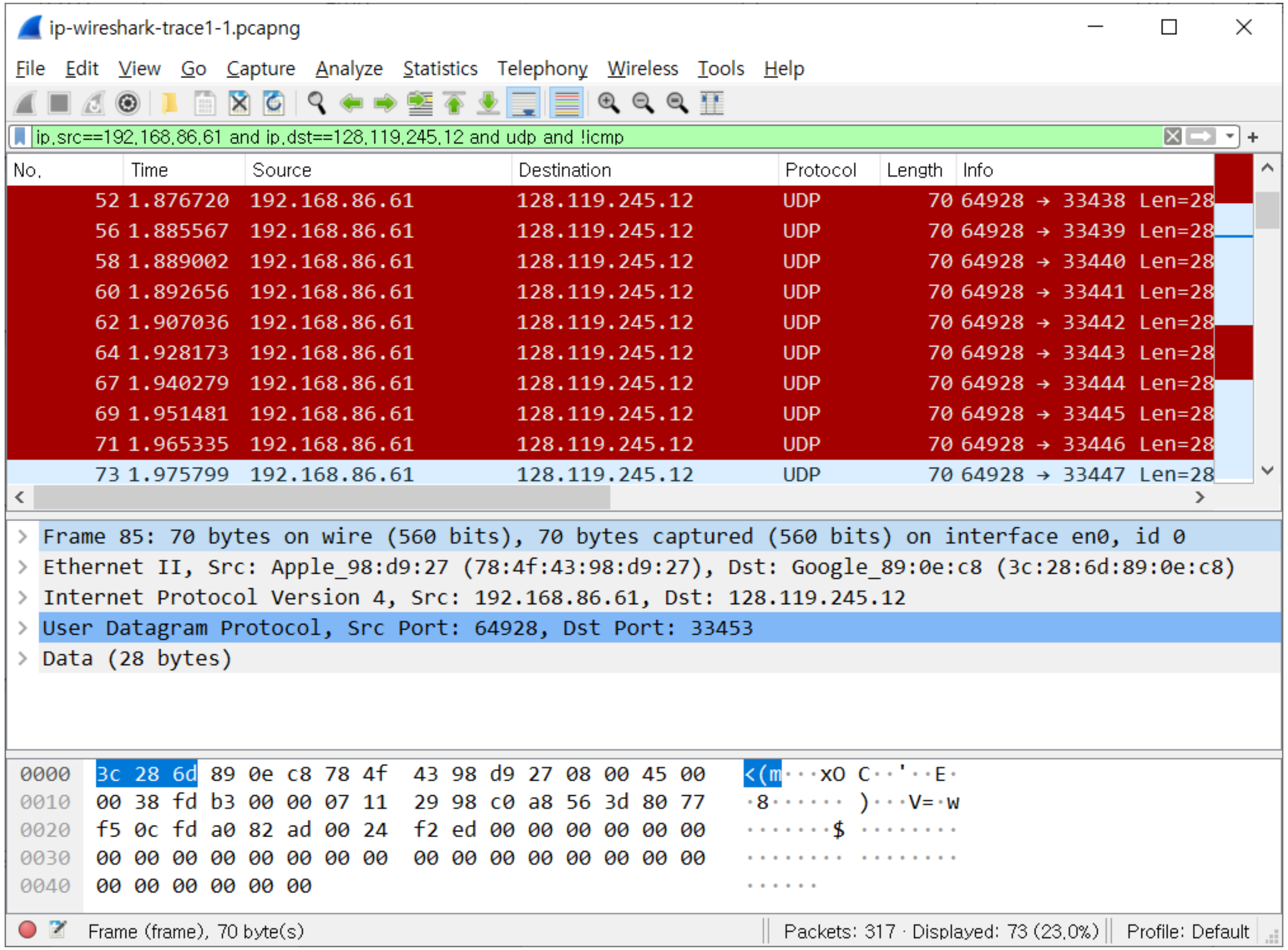
- **Basic IPv4**


ip-wireshark-trace1-1.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
udp||icmp
No. Time Source Destination Protocol Length Info
3 0.204852 192.168.86.60 224.0.0.251 MDNS 139 Standard query 0x000
4 0.205172 fe80::874:a473:63fb… ff02::fb MDNS 159 Standard query 0x000
43 1.024256 0.0.0.0 255.255.255.255 DHCP 286 DHCP Discover - Tran
44 1.865637 192.168.86.61 128.119.245.12 UDP 70 64928 → 33435 Len=28
45 1.868608 192.168.86.1 192.168.86.61 ICMP 98 Time-to-live exceede
46 1.869171 192.168.86.61 192.168.86.1 DNS 85 Standard query 0xd75
47 1.873594 192.168.86.1 192.168.86.61 DNS 85 Standard query respo
48 1.874016 192.168.86.61 128.119.245.12 UDP 70 64928 → 33436 Len=28
49 1.875315 192.168.86.1 192.168.86.61 ICMP 98 Time-to-live exceede
50 1.875401 192.168.86.61 128.119.245.12 UDP 70 64928 → 33437 Len=28
Frame 3: 139 bytes on wire (1112 bits), 139 bytes captured (1112 bits) on interface en0, id 0
Ethernet II, Src: be:0f:49:af:bc:0e (be:0f:49:af:bc:0e), Dst: IPv4mcast_fb (01:00:5e:00:00:fb)
Internet Protocol Version 4, Src: 192.168.86.60, Dst: 224.0.0.251
User Datagram Protocol, Src Port: 5353, Dst Port: 5353
Multicast Domain Name System (query)
0000 01 00 5e 00 00 fb be 0f 49 af bc 0e 08 00 45 00
0010 00 7d fe dc 00 00 ff 11 c4 b2 c0 a8 56 3c e0 00
0020 00 fb 14 e9 14 e9 00 69 c5 8a 00 00 00 00 00 02
0030 00 00 00 00 00 01 0f 5f 63 6f 6d 70 61 6e 69 6f
0040 6e 2d 6c 69 6e 6b 04 5f 74 63 70 05 6c 6f 63 61
Internet Control Message Protocol: Protocol
Packets: 317 · Displayed: 162 (51.1%)
Profile: Default

실험 3: IP

- **Basic IPv4**
  - Next, let's look at the *sequence* of UDP segments being sent from your computer via `traceroute`, destined to 128.119.245.12. The display filter that you can enter to do this is "ip.src==192.168.86.61 and ip.dst==128.119.245.12 and udp and !icmp". This will allow you to easily move sequentially through just the datagrams containing just these segments.
  - 6. Which fields in the IP datagram *always* change from one datagram to the next within this series of UDP segments sent by your computer destined to 128.119.245.12, via `traceroute`? Why?
  - 7. Which fields in this sequence of IP datagrams (containing UDP segments) stay constant? Why?
  - 8. Describe the pattern you see in the values in the Identification field of the IP datagrams being sent by your computer.

■ **Basic IPv4**


ip-wireshark-trace1-1.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
ip.src==192.168.86.61 and ip.dst==128.119.245.12 and udp and !icmp
No. Time Source Destination Protocol Length Info
52 1.876720 192.168.86.61 128.119.245.12 UDP 70 64928 → 33438 Len=28
56 1.885567 192.168.86.61 128.119.245.12 UDP 70 64928 → 33439 Len=28
58 1.889002 192.168.86.61 128.119.245.12 UDP 70 64928 → 33440 Len=28
60 1.892656 192.168.86.61 128.119.245.12 UDP 70 64928 → 33441 Len=28
62 1.907036 192.168.86.61 128.119.245.12 UDP 70 64928 → 33442 Len=28
64 1.928173 192.168.86.61 128.119.245.12 UDP 70 64928 → 33443 Len=28
67 1.940279 192.168.86.61 128.119.245.12 UDP 70 64928 → 33444 Len=28
69 1.951481 192.168.86.61 128.119.245.12 UDP 70 64928 → 33445 Len=28
71 1.965335 192.168.86.61 128.119.245.12 UDP 70 64928 → 33446 Len=28
73 1.975799 192.168.86.61 128.119.245.12 UDP 70 64928 → 33447 Len=28
Frame 85: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface en0, id 0
Ethernet II, Src: Apple_98:d9:27 (78:4f:43:98:d9:27), Dst: Google_89:0e:c8 (3c:28:6d:89:0e:c8)
Internet Protocol Version 4, Src: 192.168.86.61, Dst: 128.119.245.12
User Datagram Protocol, Src Port: 64928, Dst Port: 33453
Data (28 bytes)
0000 3c 28 6d 89 0e c8 78 4f 43 98 d9 27 08 00 45 00
0010 00 38 fd b3 00 00 07 11 29 98 c0 a8 56 3d 80 77
0020 f5 0c fd a0 82 ad 00 24 f2 ed 00 00 00 00 00 00
0030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0040 00 00 00 00 00 00
Frame (frame), 70 byte(s)
Packets: 317 · Displayed: 73 (23.0%)
Profile: Default

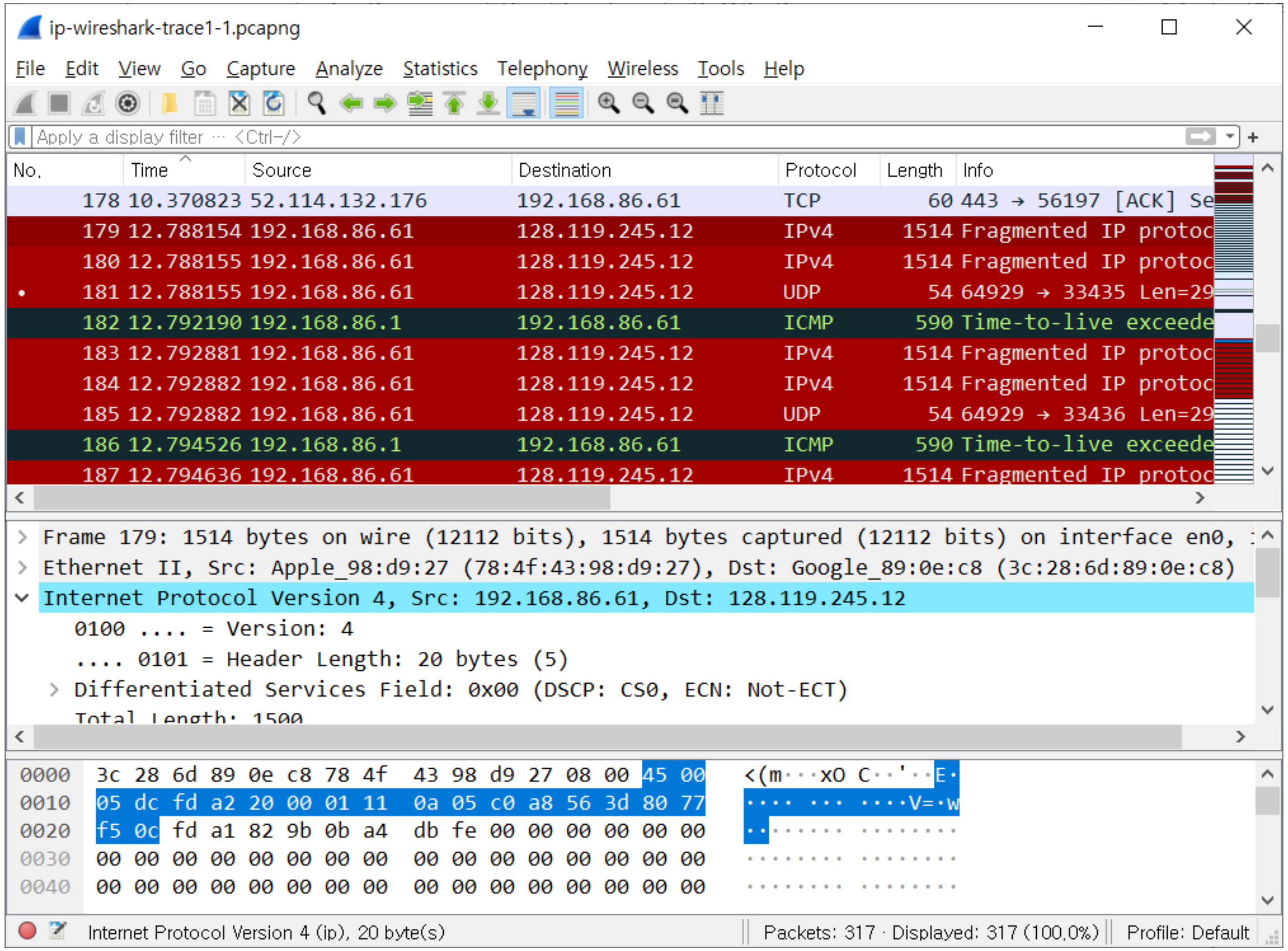
실험 3: IP

- **Fragmentation**
  - In this section, we'll look at a large (3000-byte) UDP segment sent by the `traceroute` program that is fragmented into multiple IP datagrams.
  - Sort the packet *listing* from Part 1, with any display filters cleared, according to time, by clicking on the *Time* column.

  - 9. Find the first IP datagram containing the first part of the segment sent to 128.119.245.12 sent by your computer via the traceroute command to gaia.cs.umass.edu, after you specified that the traceroute packet length should be 3000. Has that segment been fragmented across more than one IP datagram?
  - 10. What information in the IP header indicates that this datagram been fragmented?
  - 11. What information in the IP header for this packet indicates whether this is the first fragment versus a latter fragment?
  - 12. How many bytes are there in is this IP datagram (header plus payload)?
  - 13. Now inspect the datagram containing the second fragment of the fragmented UDP segment. What information in the IP header indicates that this is not the first datagram fragment?
  - 14. What fields change in the IP header between the first and second fragment?

실험 3: IP

■ **Fragmentation**



결과 보고서

- **실험 1: TCP**
  – 질문에 답하기 (5, 6, 7, 8, 9 슬라이드)

- **실험 2: UDP**
  – 질문에 답하기 (12 슬라이드)

- **실험 3: IP**
  – 질문에 답하기 (16, 18, 20 슬라이드)

  – 모든 답변은 Wireshark를 통해 확인한 packet에서 근거를 찾을 것. (사진 첨부)

예비 보고서

- Citizens Broadband Radio Service (CBRS) 대역의 주파수 공유 시스템에 대해 간단히 조사하고 설명하시오.
- IQ signal에 대해 설명하고 이를 시각화하는 방법에 대해 서술하시오.
- Python과 Pytorch를 설치하십시오.