Kate Moreland

kem0149

COMP 4320: Lab 3

Computer Network Lab (3)

This lab will go over two parts:

Part1. Internet Protocol (IP)

* In this part of the lab, you will perfom a packet capture of a tracerout execution and examine the IP datagrams of the captured trace. You will be asked to answer 15 questions as you work through this section.

Part2. Network Address Translation Protocol (NAT)

* In this part of the lab, you will use packet capture files generated by Pearson to examine the operation of the NAT protocol. You will be asked to answer an additional 10 questions.

What to turn in for this lab:

1. Each part of the lab has a list of questions regarding your work in that part of the lab. You are required to give a report in PDF format including your answer to the questions given in the lab.
2. In addition to your answers, you are required to upload an unlisted video to Youtube (no more than 15 minutes) for each of the 2 parts, of you demoing the lab and demonstrating your knowledge of the material by explaining your answers to the lab questions. You need to provide the Youtube link on your report.

# Part 1: Internet Protocol

In this lab, we’ll investigate the IP protocol, focusing on the IP datagram. We’ll do so by analyzing a trace of IP datagrams sent and received by an execution of the traceroute program. We’ll investigate the various fields in the IP datagram, and study IP fragmentation in detail.

## Capturing packets from an execution of traceroute

In order to generate a trace of IP datagrams for this lab, we’ll use the traceroute program to send datagrams of different sizes towards some destination, X. Traceroute operates by first sending one or more datagrams with the time-to-live (TTL) field in the IP header set to 1; it then sends a series of one or more datagrams towards the same destination with a TTL value of 2; it then sends a series of datagrams towards the same destination with a TTL value of 3; and so on. Recall that a router must decrement the TTL in each received datagram by 1. If the TTL reaches 0, the router returns an ICMP message (type 11 – TTL-exceeded) to the sending host. As a result of this behavior, a datagram with a TTL of 1 (sent by the host executing traceroute) will cause the router one hop away from the sender to send an ICMP TTL-exceeded message back to the sender; the datagram sent with a TTL of 2 will cause the router two hops away to send an ICMP message back to the

Computer Networking: a Top-Down Approach, 8th Edition, Kurose and Ross.

sender; the datagram sent with a TTL of 3 will cause the router three hops away to send an ICMP message back to the sender; and so on. In this manner, the host executing traceroute can learn the identities of the routers between itself and destination X by looking at the source IP addresses in the datagrams containing the ICMP TTL-exceeded messages.

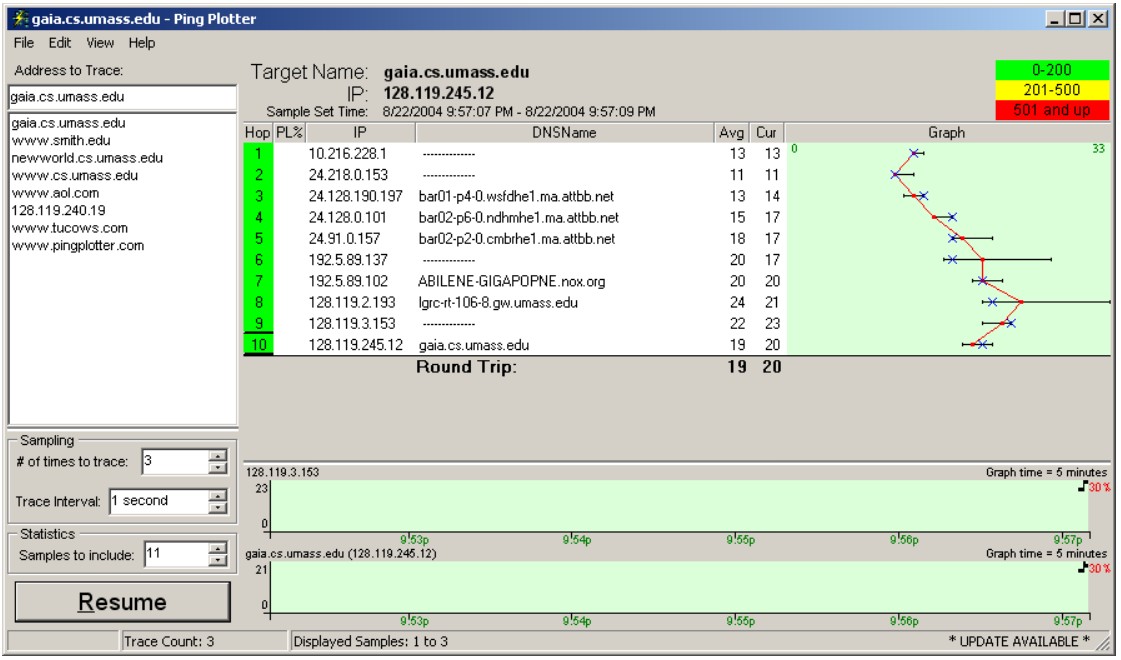
We’ll want to run traceroute and have it send datagrams of various lengths.

* Windows: The tracert program (used for our ICMP Wireshark lab) provided with Windows does not allow one to change the size of the ICMP echo request (ping) message sent by the tracert program. A nicer Windows traceroute program is pingplotter, available both in free version and shareware versions at http://www.pingplotter.com. Download and install pingplotter, and test it out by performing a few traceroutes to your favorite sites. The size of the ICMP echo request message can be explicitly set in pingplotter by selecting the menu item Edit → Options → Packet Options and then filling in the Packet Size field. The default packet size is 56 bytes. Once pingplotter has sent a series of packets with the increasing TTL values, it restarts the sending process again with a TTL of 1, after waiting Trace Interval amount of time. The value of Trace Interval and the number of intervals can be explicitly set in pingplotter.
* Linux/Unix/MacOS: With the Unix/MacOS traceroute command, the size of the UDP datagram sent towards the destination can be explicitly set by indicating the number of bytes in the datagram; this value is entered in the traceroute command line immediately after the name or address of the destination. For example, to send traceroute datagrams of 2000 bytes towards gaia.cs.umass.edu, the command would be:

traceroute gaia.cs.umass.edu 2000

Do the following:

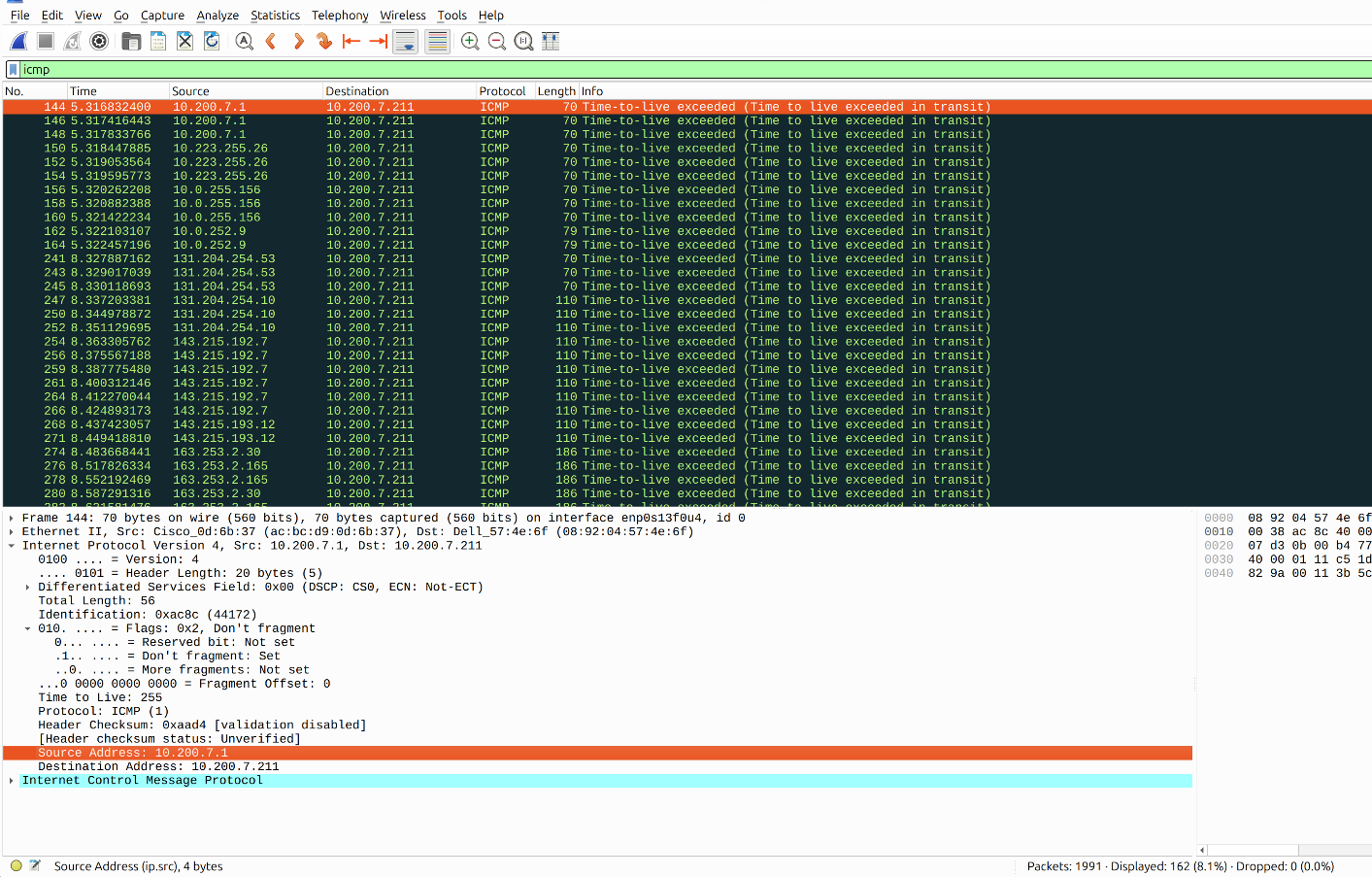
* Start up Wireshark and begin packet capture (Capture → Start) and then press OK on the Wireshark Packet Capture Options screen (we’ll not need to select any options here).
* If you are using a Windows platform, start up pingplotter and enter the name of a target destination in the “Address to Trace Window.” Enter 3 in the “# of times to Trace” field, so you don’t gather too much data. Select the menu item Edit → Advanced Options → Packet Options and enter a value of 56 in the Packet Size field and then press OK. Then press the Trace button. You should see a pingplotter window that looks something like the one in the figure below: Next, send a set of datagrams with a longer length, by selecting Edit → Advanced Options → Packet Options and enter a value of 2000 in the Packet Size field and then press OK. Then press the Resume button. Finally, send a set of datagrams with a longer length, by selecting Edit → Advanced Options → Packet Options and enter a value of 3500 in the Packet Size field and then press OK. Then press the Resume button.
* Stop Wireshark tracing.
* If you are using a Unix or Mac platform, enter three traceroute commands, one with a length of 56 bytes, one with a length of 2000 bytes, and one with a length of 3500 bytes.
* Stop Wireshark tracing



## A look at the captured trace

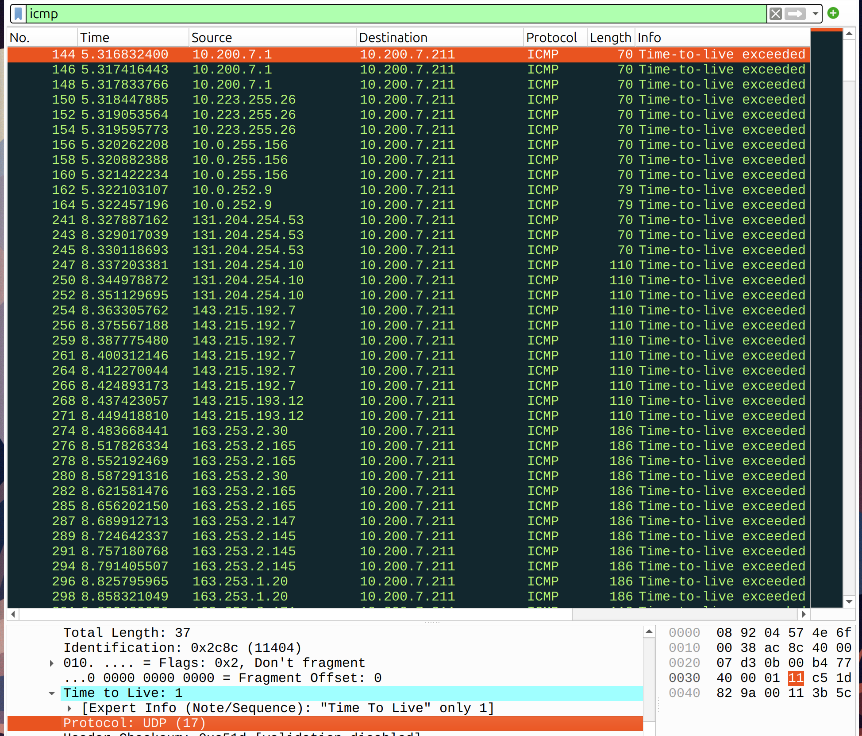
In your trace, you should be able to see the series of ICMP Echo Request (in the case of Windows machine) or the UDP segment (in the case of Unix) sent by your computer and the ICMP TTLexceeded messages returned to your computer by the intermediate routers. In the questions below, we’ll assume you are using a Windows machine; the corresponding questions for the case of a Unix machine should be clear.

1. Select the first ICMP Echo Request message sent by your computer, and expand the Internet Protocol part of the packet in the packet details window. What is the IP address of your computer?



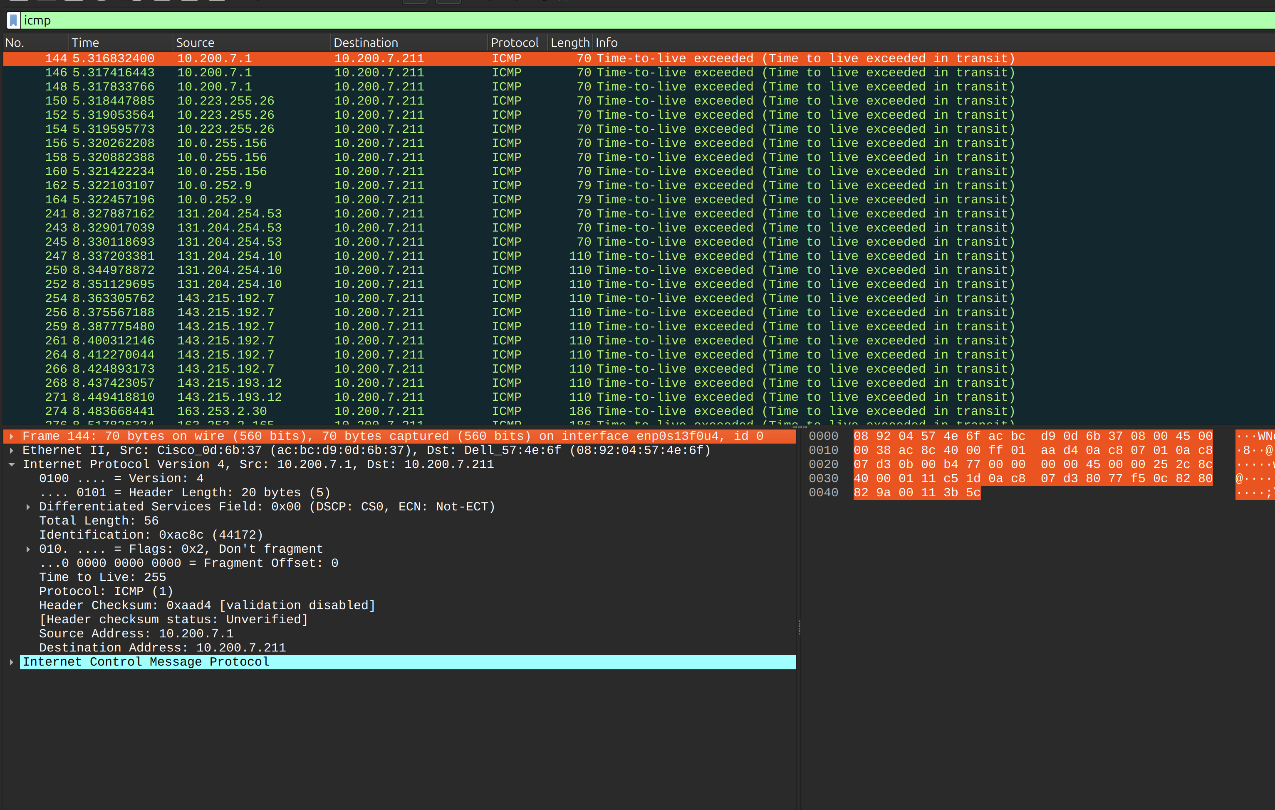
I was unable to get the ICMP Echo Request when I ran the trace with ubuntu and Wireshark. They all appeared as Time-to-Live exceeded. This is the first packet of the ICMP with my trace. The IP address of my computer is 10.200.7.1

1. Within the IP packet header, what is the value in the upper layer protocol field?



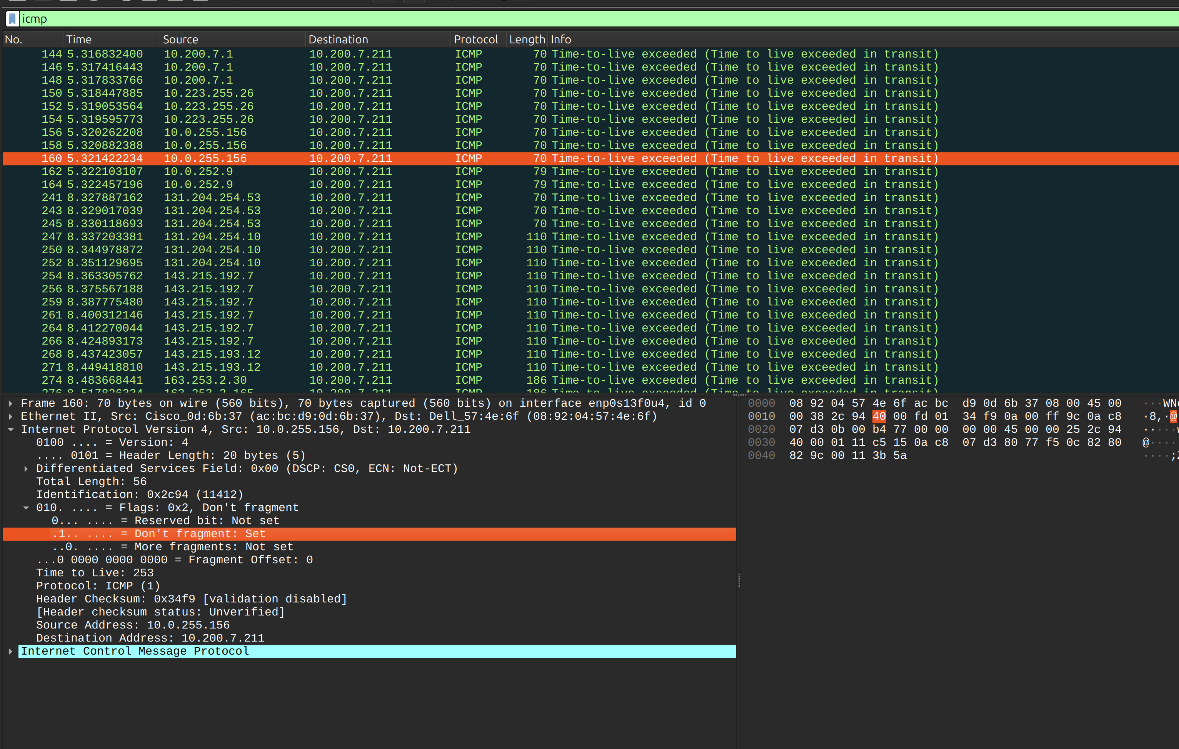
This again is the in the first time-to-live exceeded packet in my trace. The value in the upper layer protocol field is 17.

1. How many bytes are in the IP header? How many bytes are in the payload of the IP datagram? Explain how you determined the number of payload bytes.

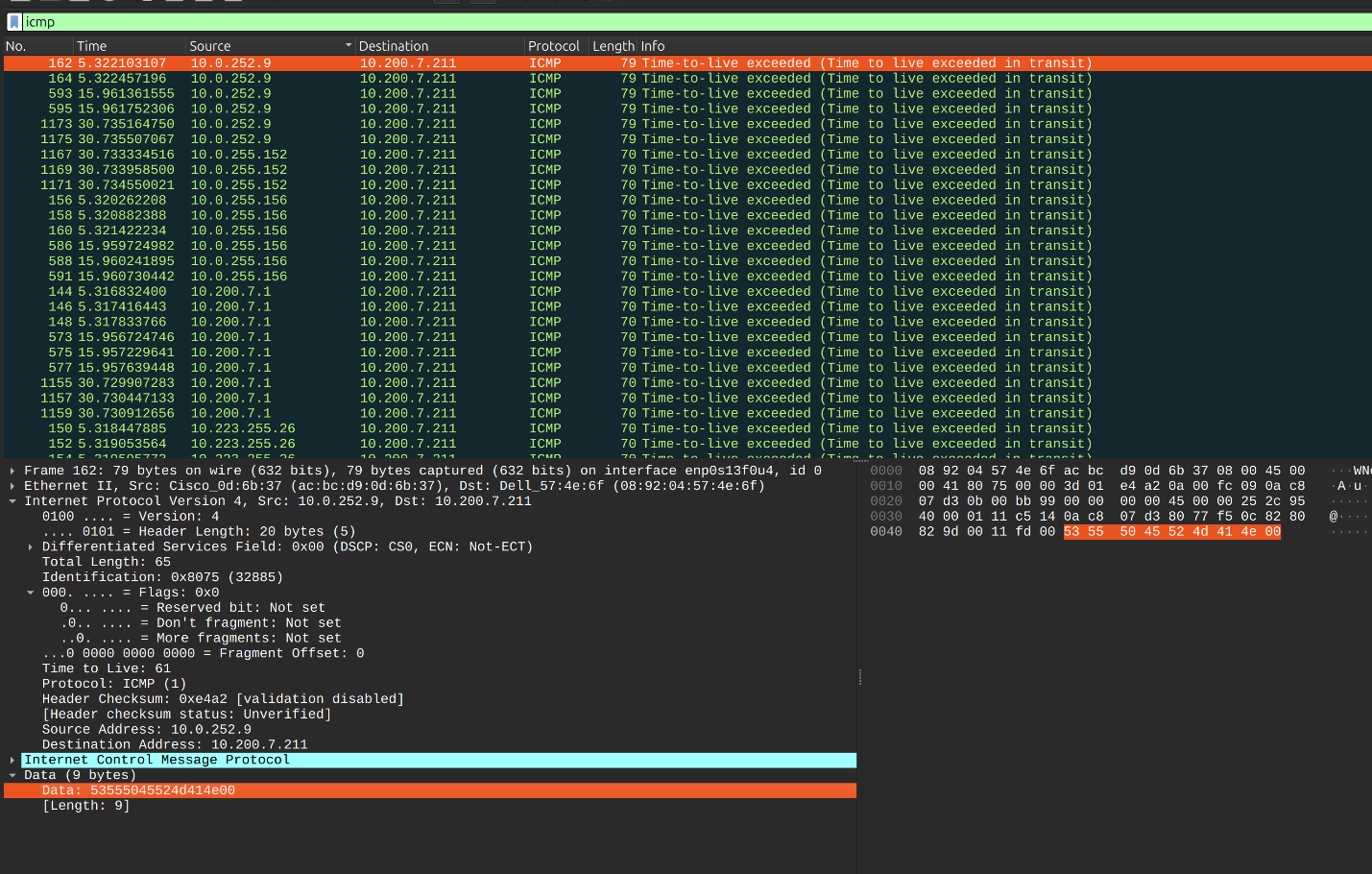


The number of bytes in the IP header for my trace is 20 bytes. Payload size = Total length – Header length. The total length is 56 bytes which is shown in the picture under the row of length. So, 56 – 20 = 36 so the payload size is 36 bytes.

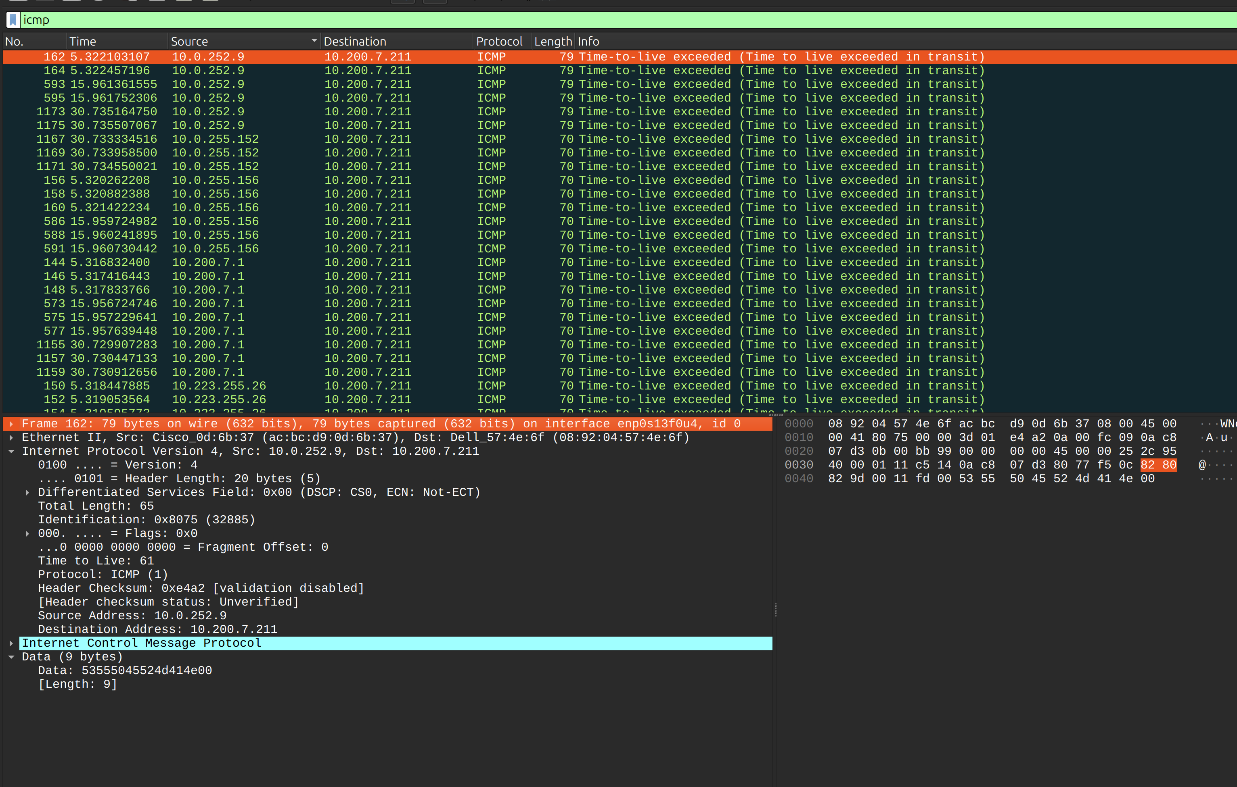
1. Has this IP datagram been fragmented? Explain how you determined whether or not the datagram has been fragmented. Next, sort the traced packets according to IP source address by clicking on the Source column header; a small downward pointing arrow should appear next to the word Source. If the arrow points up, click on the Source column header again. Select the first ICMP Echo Request message sent by your computer, and expand the Internet Protocol portion in the “details of selected packet header” window. In the “listing of captured packets” window, you should see all of the subsequent ICMP messages (perhaps with additional interspersed packets sent by other protocols running on your computer) below this first ICMP. Use the down arrow to move through the ICMP messages sent by your computer.



The IP datagram has been fragmented in my total trace. I can tell this because the flag of don’t fragment is not set in this frame shown above.

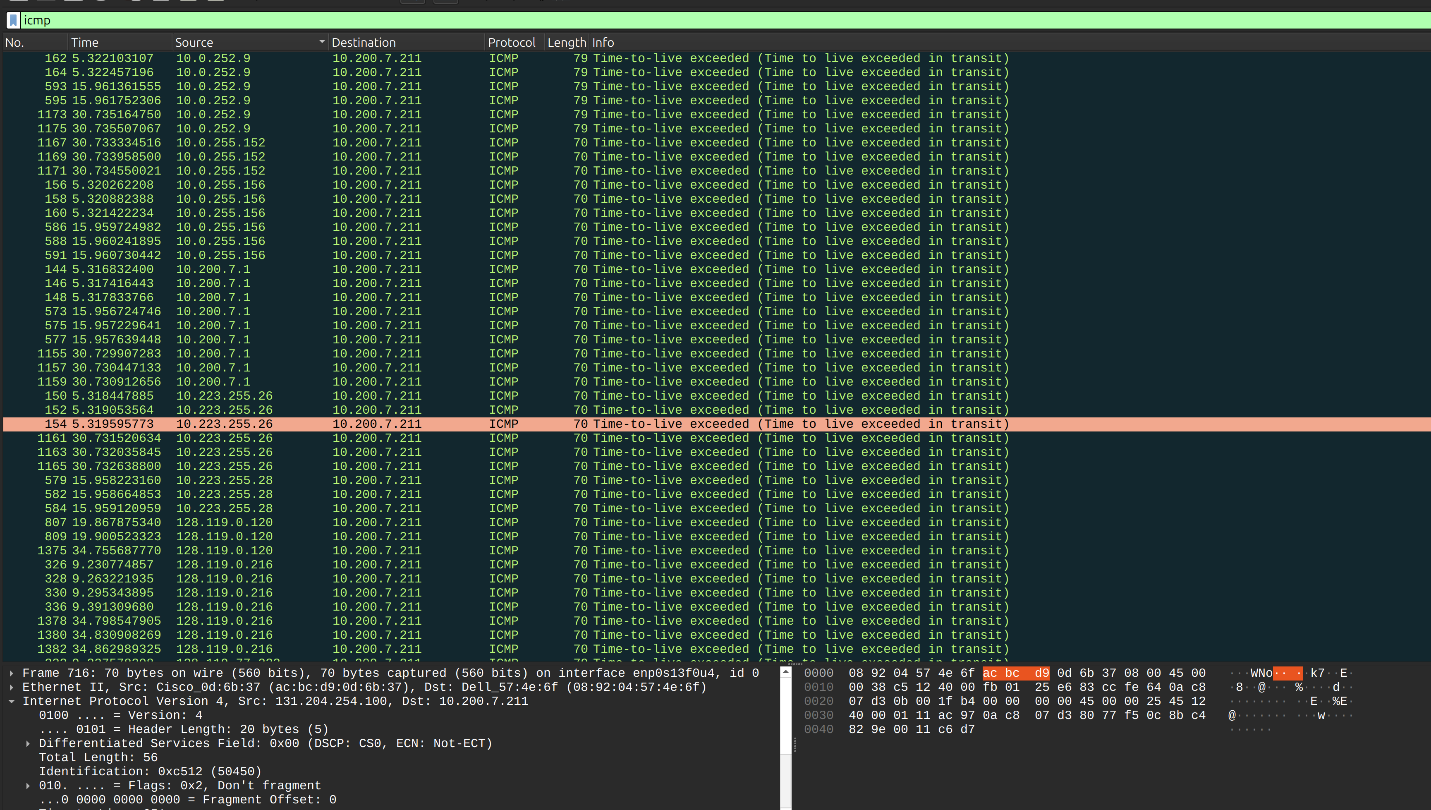


1. Which fields in the IP datagram always change form one datagram to the next within this series of ICMP messages sent by your computer? 6. Which fields stay constant? Which of the fields must stay constant? Which fields must change? Why?



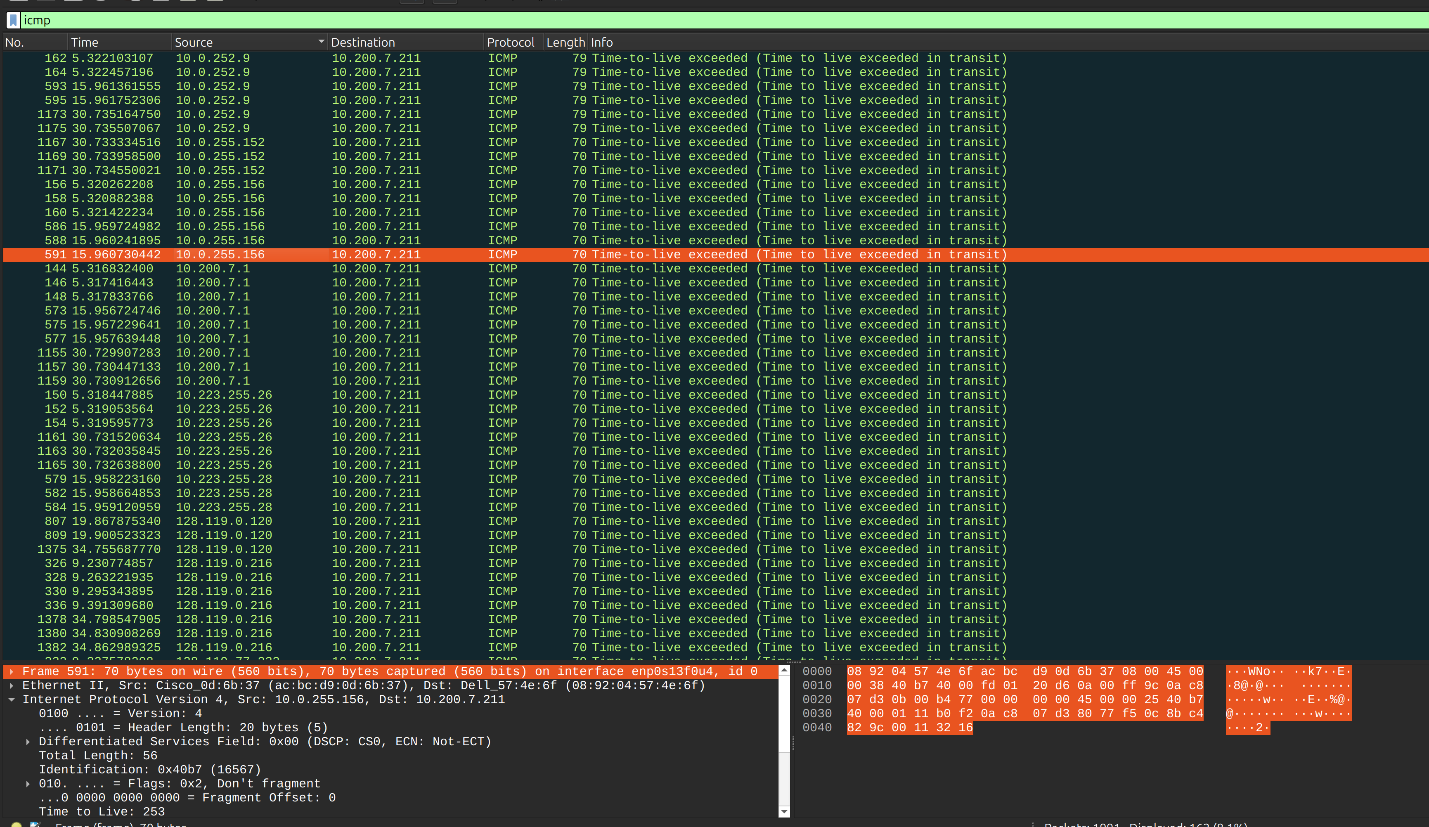
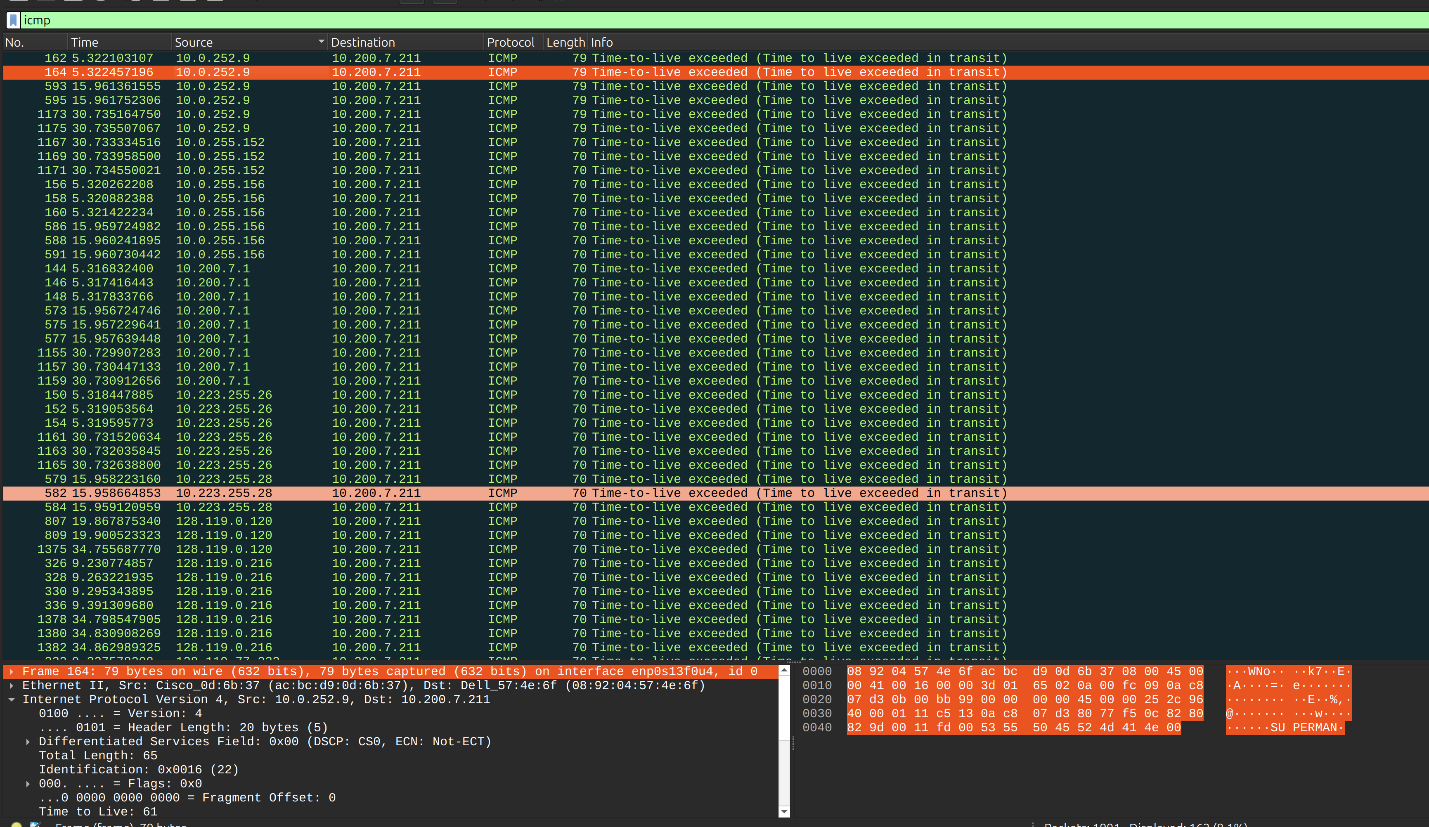
Again, I was unable to get the ICMP Echo Response message in my trace. The fields that change in the IP datagram are the identification, time to live, total length, checksum, and the source and destination address. The fields that stay constant are the version, header length, and protocol. The fields that have to change are the checksum and identification because it shows that there are different packets that have been sent.

1. Describe the pattern you see in the values in the Identification field of the IP datagram. Next (with the packets still sorted by source address) find the series of ICMP TTL-exceeded replies sent to your computer by the nearest (first hop) router.



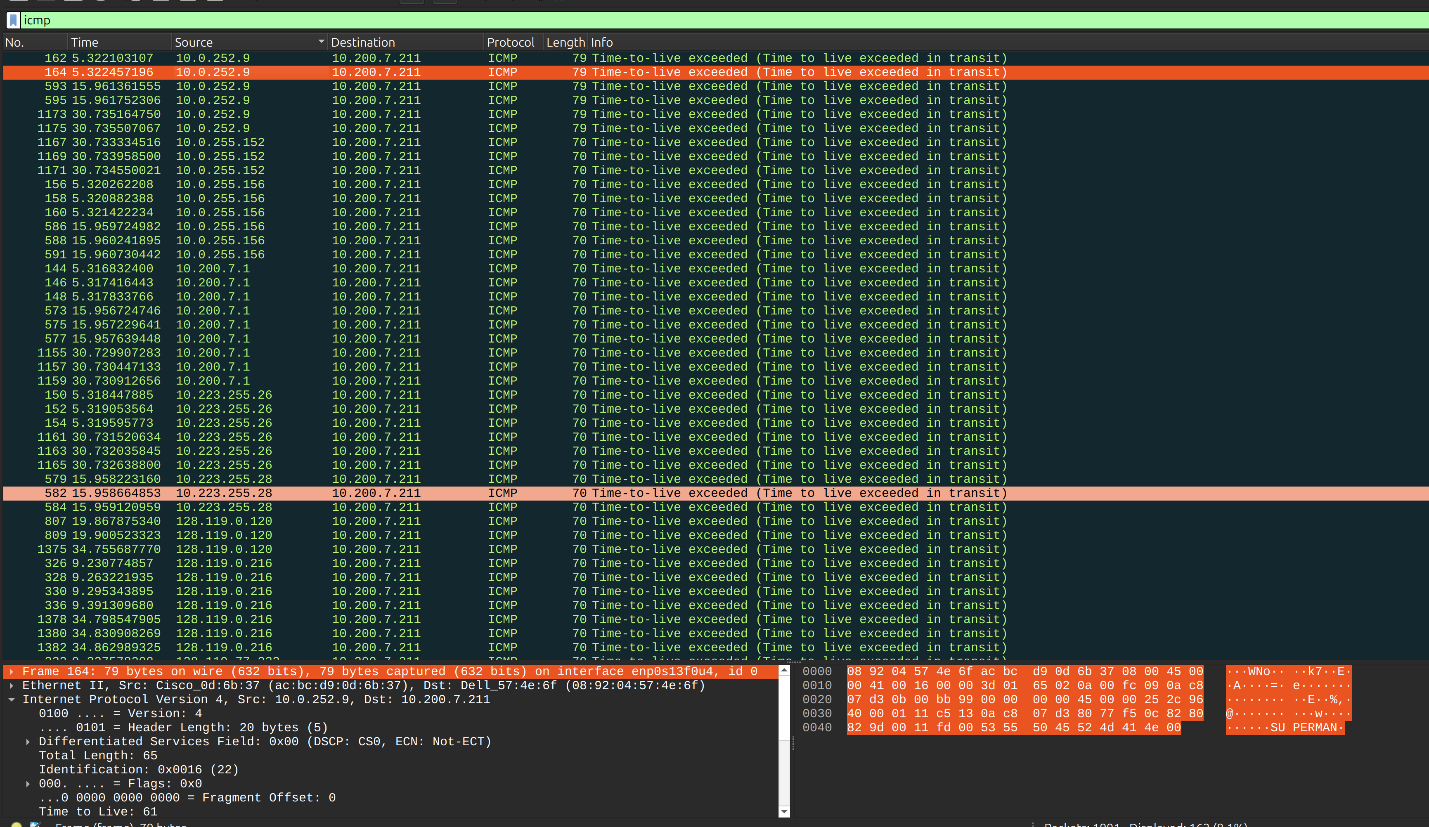
The pattern in the ICMP TTL-exceeded replies are the only messages I got in my trace. I noticed that the destination in those statements remain the same in all of the statements. The sources change about every 6 packets and attempt to connect from there.

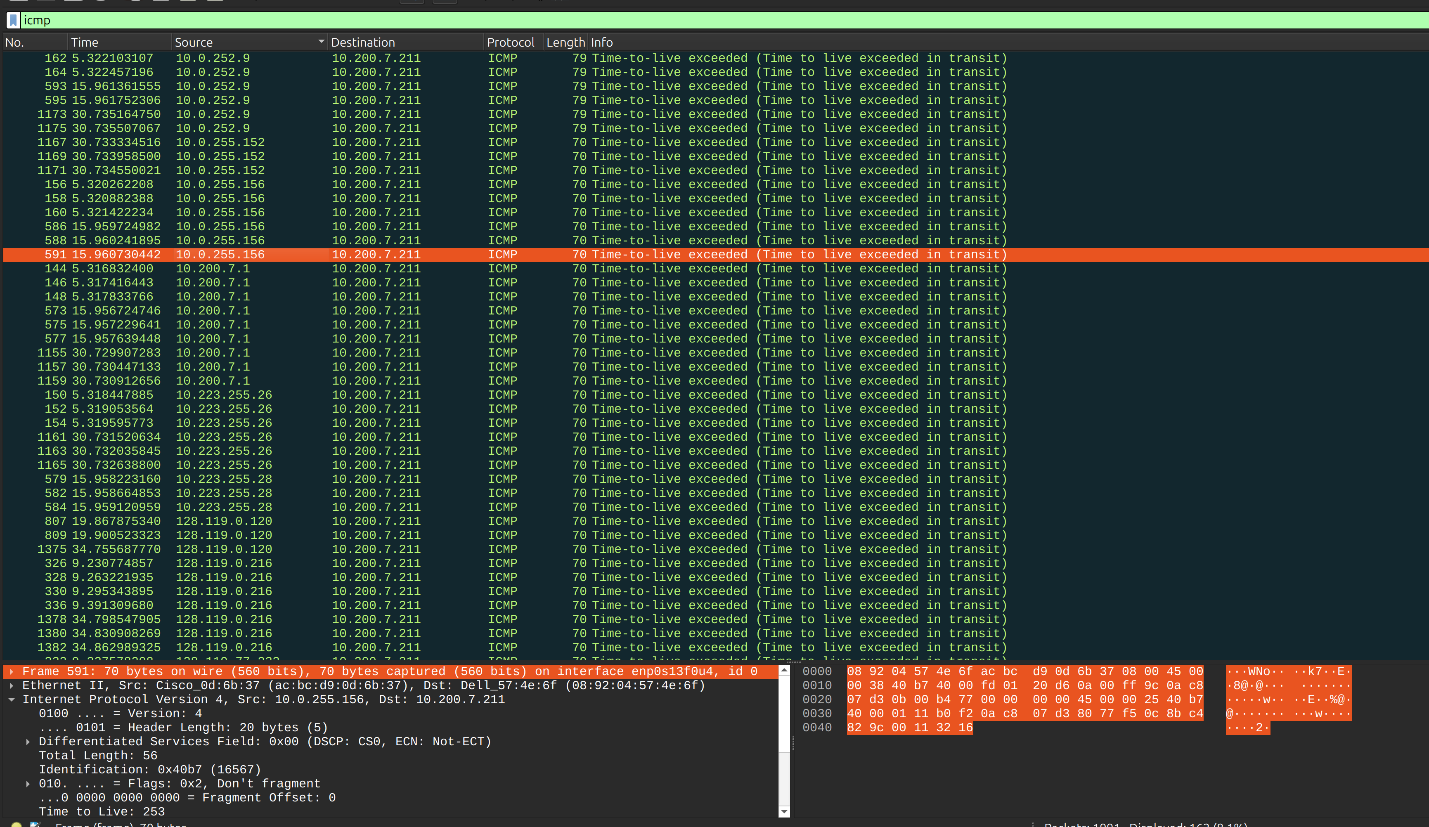
1. What is the value in the Identification field and the TTL field?



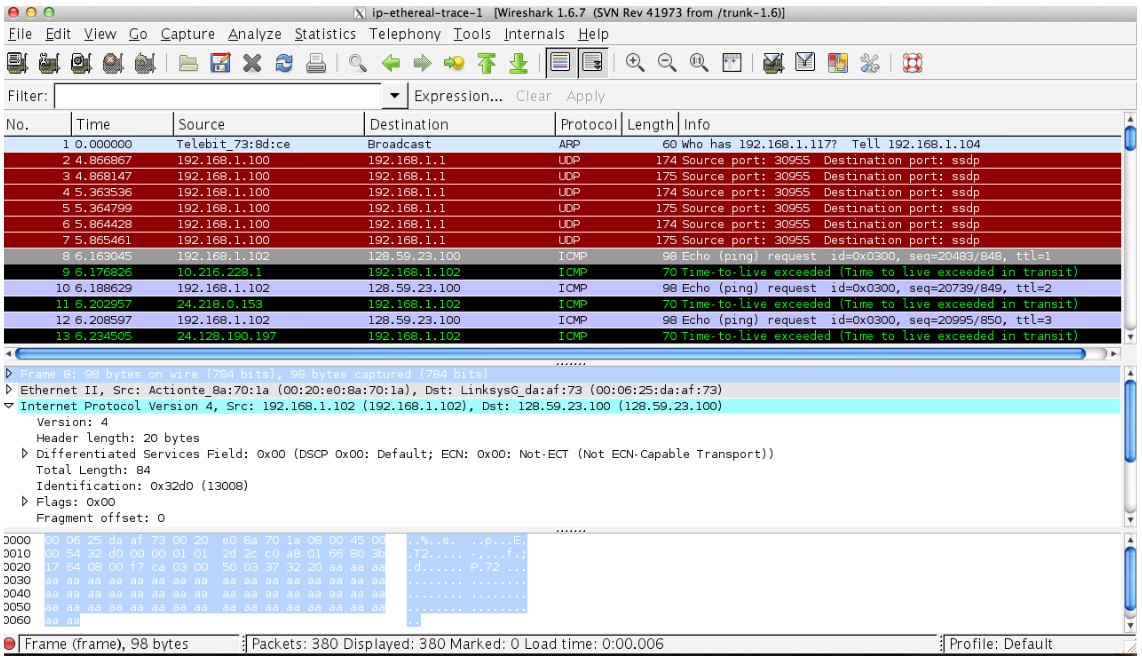
The value in the identification field and TTL field are constantly changing in my packets. I attached 2 pictures above to show the issues that I had.

1. Do these values remain unchanged for all of the ICMP TTL-exceeded replies sent to your computer by the nearest (first hop) router? Why?





No, they change between each router I would guess because the identification is going through another. This is shown in the pictures above in the previous question.



Fragmentation

Sort the packet listing according to time again by clicking the Time column.

1. Find the first ICMP Echo Request message that was sent by your computer after you changed the Packet Size in pingplotter to be 2000. Has that message been fragmented across more than one IP datagram?

I never got any ICMP Echo Requests in my trace so I am unable to answer that. I am going to attach my pcap file to show the issues I was encountering.

1. Print out the first fragment of the fragmented IP datagram. What information in the IP header indicates that the datagram been fragmented? What information in the IP header indicates whether this is the first fragment versus a latter fragment? How long is this IP datagram?

I am unsure because my trace did not work correctly.

1. Print out the second fragment of the fragmented IP datagram. What information in the IP header indicates that this is not the first datagram fragment? Are the more fragments? How can you tell?

I am unsure because my trace did not work correctly.

1. What fields change in the IP header between the first and second fragment? Now find the first ICMP Echo Request message that was sent by your computer after you changed the Packet Size in pingplotter to be 3500.

I am unsure because my trace did not work correctly.

1. How many fragments were created from the original datagram?

I am unsure because my trace did not work correctly.

1. What fields change in the IP header among the fragments?

I am unsure because my trace did not work correctly.

# Part 2: Network Address Translator

This part of the lab will be different from our other Wireshark labs, where we’ve captured a trace file at a single Wireshark measurement point. Because we’re interested in capturing packets at both the input and output sides of the NAT device, we’ll need to capture packets at two locations.

Download the “NAT-PCAP” zip file and extract the files need for this part of the lab.

* NAT home side.pcap
* NAT ISP side.pcap

## NAT Measurement Scenario

In this part of the lab, we’ll capture packets from a simple web request from a client PC in a home network to a www.google.com server. Within the home network, the home network router provides a NAT service.

The figure below shows our Wireshark trace-collection scenario: As in our other Wireshark labs, we collect a Wireshark trace on the client PC in our home network. This file is called NAT home side. Because we are also interested in the packets being sent by the NAT router into the ISP, we’ll collect a second trace file at a PC (not shown) tapping into the link from the home router into the ISP network. (The hub device shown on the ISP side of the router is used to tap into the link between the NAT router and the first hop router in the ISP). Client-to-server packets captured by Wireshark at this point will have undergone NAT translation. The Wireshark trace file captured on the ISP side of the home router is called NAT ISP side.



Open the NAT home side file and answer the following questions. You might find it useful to use a Wireshark filter so that only frames containing HTTP messages are displayed from the trace file.

1. What is the IP address of the client?

A screenshot of a computer

AI-generated content may be incorrect.



The IP address of the client is 192.168.1.100

1. The client actually communicates with several different Google servers in order to implement “safe browsing.” The main Google server that will serve up the main Google web page has IP address 64.233.169.104. In order to display only those frames containing HTTP messages that are sent to/from this Google, server, enter the expression “http && ip.addr == 64.233.169.104” (without quotes) into the Filter: field in Wireshark.

A screenshot of a computer

AI-generated content may be incorrect.

1. Consider now the HTTP GET sent from the client to the Google server (whose IP address is IP address 64.233.169.104) at time 7.109267. What are the source and destination IP addresses and TCP source and destination ports on the IP datagram carrying this HTTP GET?

A screenshot of a computer

AI-generated content may be incorrect.



The source IP is 192.168.1.100. The destination IP is 64.233.169.104. The TCP source port is 4335. The TCP destination port is 80.

1. At what time (specify using time since the beginning of the trace in seconds) is the corresponding 200 OK HTTP message received from the Google server? What are the source and destination IP addresses and TCP source and destination ports on the IP datagram carrying this HTTP 200 OK message?

A screenshot of a computer

AI-generated content may be incorrect.



The message is received at 7.158797. The source IP is 64.233.169.104. The destination IP is 192.128.1.100. The TCP source port is 80. The TCP destination pot is 4335.

1. Recall that before a GET command can be sent to an HTTP server, TCP must first set up a connection using the three-way SYN/ACK handshake. At what time is the client-to-server TCP SYN segment sent that sets up the connection used by the GET sent at time 7.109267? What are the source and destination IP addresses and source and destination ports for the TCP SYN segment? What are the source and destination IP addresses and source and destination ports of the ACK sent in response to the SYN. At what time is this ACK received at the client? (Note: to find these segments you will need to clear the Filter expression you entered above in step 2. If you enter the filter “tcp”, only TCP segments will be displayed by Wireshark).

A screenshot of a computer

AI-generated content may be incorrect.



The SYN message is sent at 7.075657. The source IP is 192.168.1.100. The destination IP is 64.233.169.104. The TCP source port is 4335. The TCP destination port is 80.

A screenshot of a computer

AI-generated content may be incorrect.



The ACK response was sent at 7.108986. The source IP is 64.233.169.104. The destination IP is 192.168.1.100. The TCP source port is 80. The TCP destination port is 4335.

A screenshot of a computer

AI-generated content may be incorrect.



The final ACK was sent at 7.109053. The source IP is 192.168.1.100. The destination IP is 64.233.169.104. The TCP source port is 4335. The TCP destination port is 80.

In the following we’ll focus on the two HTTP messages (GET and 200 OK) and the TCP SYN and ACK segments identified above. Our goal below will be to locate these two HTTP messages and two TCP segments in the trace file (NAT ISP side) captured on the link between the router and the ISP. Because these captured frames will have already been for- warded through the NAT router, some of the IP address and port numbers will have been changed as a result of NAT translation. Open the NAT ISP side. Note that the time stamps in this file and in NAT home side are not synchronized since the packet captures at the two locations were not started simultaneously. (Indeed, you should discover that the timestamps of a packet captured at the ISP link is actually less that the timestamp of the packet captured at the client PC).

1. In the NAT ISP side trace file, find the HTTP GET message was sent from the client to the Google server at time 7.109267 (where t=7.109267 is time at which this was sent as recorded in the NAT home side trace file). At what time does this message appear in the NAT ISP side trace file? What are the source and destination IP addresses and TCP source and destination ports on the IP datagram carrying this HTTP GET (as recording in the NAT ISP side trace file)? Which of these fields are the same, and which are different, than in your answer to question 3 above?

A screenshot of a computer

AI-generated content may be incorrect.



The message appears at the time 6.069168. The source IP is 71.192.34.104. The destination IP is 64.233.169.104. The TCP source port is 4335. The TCP destination port is 80.

The source IP is different from the home side which was 192.168.1.100. The destination IP, TCP source port, and TCP destination port are all the same.

1. Are any fields in the HTTP GET message changed? Which of the following fields in the IP datagram carrying the HTTP GET are changed: Version, Header Length, Flags, Checksum. If any of these fields have changed, give a reason (in one sentence) stating why this field needed to change.

A screenshot of a computer

AI-generated content may be incorrect.



NAT\_home\_side.pcap

The Version is 4. The Header Length is 20 bytes. The Flags is 0x2; Don’t fragment. The Checksum is 0xa94a.

A screenshot of a computer

AI-generated content may be incorrect.



NAT\_ISP\_side.pcap

The Version is 4. The Header Length is 20 bytes. The Flags is 0x2 Don’t Fragment. The Checksum is 0x022f.

The only difference is the checksum. This is because the source IP is different so it will result in a different checksum.

1. In the NAT ISP side trace file, at what time is the first 200 OK HTTP message received from the Google server? What are the source and destination IP addresses and TCP source and destination ports on the IP datagram carrying this HTTP 200 OK message? Which of these fields are the same, and which are different than your answer to question 4 above?

A screenshot of a computer

AI-generated content may be incorrect.



The message is received at 6.117570. The source IP is 64.233.169.104. The destination IP is 71.192.34.104. The TCP source port is 80. The TCP destination port is 4335.

The destination IP is different from the host which was 192.168.1.100. The source IP, TCP source port, and TCP destination port are all the same.

1. In the NAT ISP side trace file, at what time were the client-to-server TCP SYN segment and the server-to-client TCP ACK segment corresponding to the segments in question 5 above captured? What are the source and destination IP addresses and source and destination ports for these two segments? Which of these fields are the same, and which are different than your answer to question 5 above?

A screenshot of a computer

AI-generated content may be incorrect.



The SYN message was sent at 6.035475. The source IP is 71.192.34.104. The destination IP is 64.233.169.104. The TCP source port is 4335. The TCP destination port is 80. The source IP is the only thing that is different.

A screenshot of a computer

AI-generated content may be incorrect.



The SYN, ACK was sent at 6.067775. The source IP is 64.233.169.104. The destination IP is 71.192.34.104. The TCP source port is 80. The TCP destination port is 4335. The destination IP is the only thing that is different.

End of Lab (3)

YouTube Video:

<https://youtu.be/klHUZE4P744>