



TAIBAH UNIVERSITY



College of Computer Science and Engineering

Computer Engineering Department

COE332

Computer Networks

Student's Lab Manual

V7

Prepared by:

Dr. Mohamed ZAYED Dr. Ahmed ABDELMONEM

Dr. Abdullah AL BINALI

Lab06

Student Name: Norah Fahad Aloufi

Student ID: 4050772

Section: C8C Group:

Session (Fall / Spring / Summer): 28/3/2022

Lab-6: IP 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 (the `traceroute` program itself is explored in more detail in the Wireshark ICMP lab). We'll investigate the various fields in the IP datagram, and study IP fragmentation in detail.

Before beginning this lab, you'll probably want to review sections 1.4.3 in the text¹³ and section 3.4 of RFC 2151 [[ftp://ftp.rfc-editor.org/in-notes/rfc2151.txt](http://ftp.rfc-editor.org/in-notes/rfc2151.txt)] to update yourself on the operation of the `traceroute` program. You'll also want to read Section 4.3 in the text, and probably also have RFC 791 [[ftp://ftp.rfc-editor.org/in-notes/rfc791.txt](http://ftp.rfc-editor.org/in-notes/rfc791.txt)] on hand as well, for a discussion of the IP protocol.

1. 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*. Recall that `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 (actually, RFC 791 says that the router must decrement the TTL by *at least* one). 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 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

¹³ References to figures and sections are for the 8th edition of our text, *Computer Networks, A Top-down Approach*, 8th ed., J.F. Kurose and K.W. Ross, Addison-Wesley/Pearson, 2020.

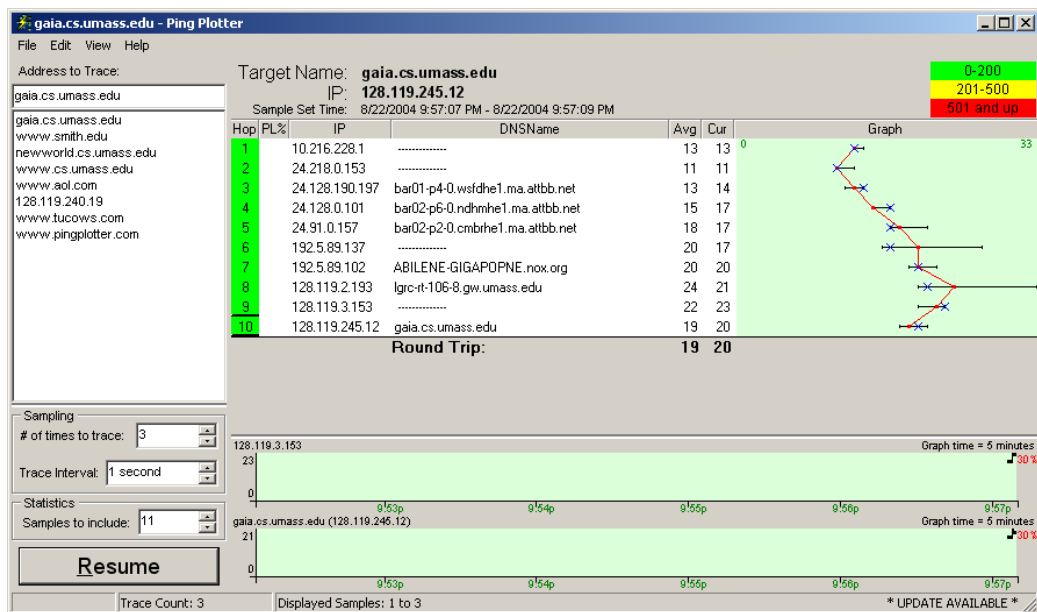
with a TTL of 1, after waiting *Trace Interval* amount of time. The value of *Trace Interval* and the number of intervals can be explicitly set in *pingplotter*.

- **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 this:



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.

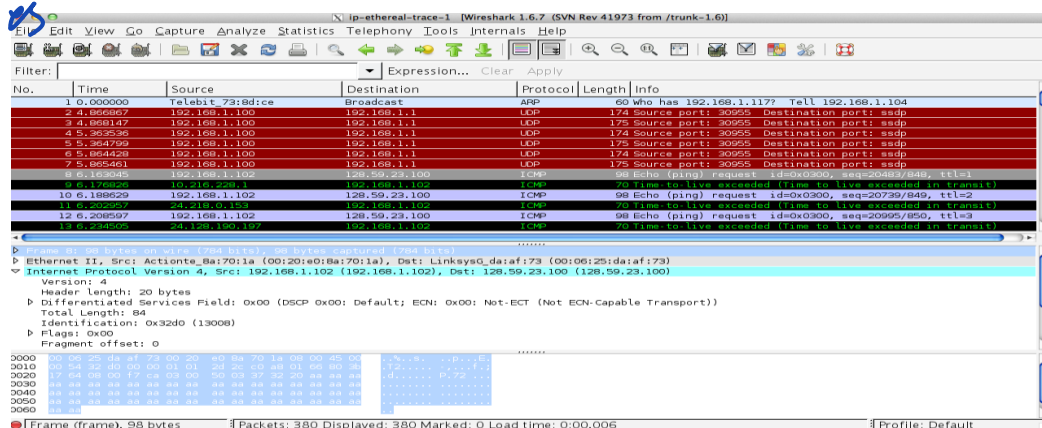
If you are unable to run Wireshark on a live network connection, you can download a packet trace file that was captured while following the steps above on one of the author's Windows computers¹⁴. You may well find it valuable to download this trace even if you've captured your own trace and use it, as well as your own trace, when you explore the questions below.

2. 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 TTL-exceeded 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. Whenever possible, when answering a question below you should hand in a printout of the packet(s) within the trace that you used to answer the question asked. When you hand in your assignment, annotate the output so that it's clear where in the output you're getting the information for your answer (e.g., for our classes, we ask that students markup paper copies with a pen, or annotate electronic copies with text in a colored font). To print a packet, use *File->Print*, choose *Selected packet only*, choose *Packet summary line*, and select the minimum amount of packet detail that you need to answer the question.

- 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.

Last pages



¹⁴ Download the zip file <http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip> and extract the file *ip-ethereal-trace-1*. The traces in this zip file were collected by Wireshark running on one of the author's computers, while performing the steps indicated in the Wireshark lab. Once you have downloaded the trace, you can load it into Wireshark and view the trace using the *File* pull down menu, choosing *Open*, and then selecting the *ip-ethereal-trace-1* trace file.

2. What is the IP address of your computer?
3. What is the IP address of your computer?
4. Within the IP packet header, what is the value in the upper layer protocol field?
5. 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.
6. 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.

7. Which fields in the IP datagram *always* change from one datagram to the next within this series of ICMP messages sent by your computer?
8. Which fields stay constant? Which of the fields *must* stay constant? Which fields must change? Why?
9. 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.

10. What is the value in the Identification field and the TTL field?
11. Do these values remain unchanged for all of the ICMP TTL-exceeded replies sent to your computer by the nearest (first hop) router? Why?

Fragmentation

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

12. 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? [Note: if you find your packet has not been fragmented, you should download the zip file <http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip> and extract the *ip-ethereal-trace-1* packet trace. If your computer has an Ethernet interface, a packet size of 2000 *should* cause fragmentation.¹⁵]
13. 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?
14. 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 there more fragments? How can you tell?

¹⁵ The packets in the *ip-ethereal-trace-1* trace file in <http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip> are all less than 1500 bytes. This is because the computer on which the trace was gathered has an Ethernet card that limits the length of the maximum IP packet to 1500 bytes (40 bytes of TCP/IP header data and 1460 bytes of upper-layer protocol payload). This 1500 byte value is the standard maximum length allowed by Ethernet. If your trace indicates a datagram longer than 1500 bytes, and your computer is using an Ethernet connection, then Wireshark is reporting the wrong IP datagram length; it will likely also show only one large IP datagram rather than multiple smaller datagrams. This inconsistency in reported lengths is due to the interaction between the Ethernet driver and the Wireshark software. We recommend that if you have this inconsistency, that you perform this lab using the *ip-ethereal-trace-1* trace file.

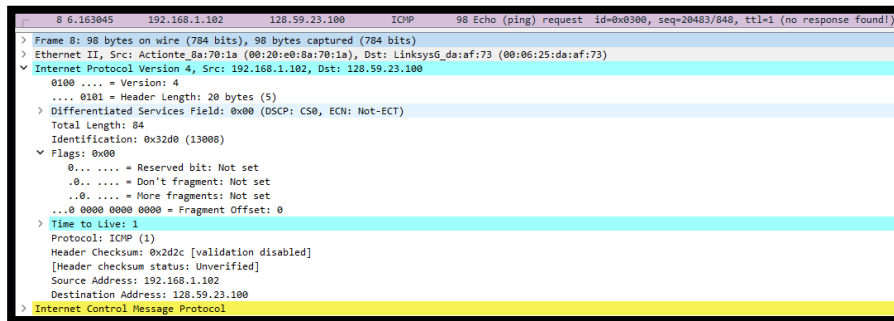
15. What fields change in the IP header between the first and second fragment?

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.

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

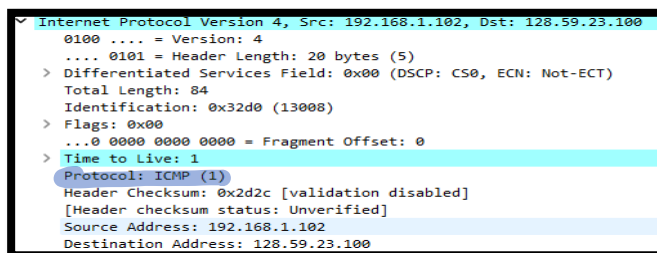
17. What fields change in the IP header among the fragments?

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.



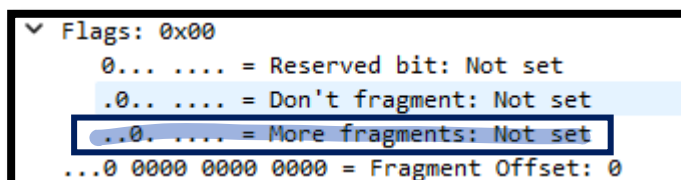
2. What is the IP address of your computer? **192.168.1.102**

4. Within the IP packet header, what is the value in the upper layer protocol field? **ICMP (1)**



5. 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. **Header length is 20 bytes and Payload = Total Length: 56 - header length: 20 = 36 bytes.**

6. Has this IP datagram been fragmented? Explain how you determined whether or not the datagram has been fragmented. **The more fragments bit = 0, so the data is not fragmented**



7. Which fields in the IP datagram *always* change from one datagram to the next within this series of ICMP messages sent by your computer? **Identification, Time to Live and Header checksum always change**

8. Which fields stay constant? Which of the fields *must* stay constant? Which fields must change? Why? **The fields that stay constant across the IP datagrams are:**

Version, header Length, source IP, destination IP, Differentiated and Upper Layer

The fields that must stay constant are:

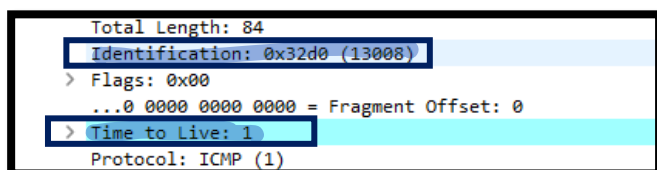
Version, header Length, source IP, destination IP, Differentiated Services and Upper Layer Protocol

The fields that must change are:

Identification, Time to Live and Header checksum

9. Describe the pattern you see in the values in the Identification field of the IP datagram **the IP header Identification fields increment with each ICMP Echo (ping) request.**

10. What is the value in the Identification field and the TTL field? **Identification: 13008 TTL: 1**



11. Do these values remain unchanged for all the ICMP TTL-exceeded replies sent to your computer by the nearest (first hop) router? Why? *The identification field changes for all the ICMP TTL-exceeded replies because the identification field is a unique value. When two or more IP datagrams have the same identification value, then it means that these IP datagrams are fragments of a single large IP datagram. The TTL field remains unchanged because the TTL for the first hop router is always the same.*

12. 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? *Yes, that message has been fragmented across more than one IP datagram.*

```

289 12.950236 192.168.1.21 128.119.245.12 IPv4 1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=bce5) [Reassembled in #290]
290 12.950236 192.168.1.21 128.119.245.12 ICMP 534 Echo (ping) request id=0x0001, seq=7058/37403, ttl=255 (reply in 303)
291 13.000639 192.168.1.21 128.119.245.12 IPv4 1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=bce6) [Reassembled in #292]
292 13.000639 192.168.1.21 128.119.245.12 ICMP 534 Echo (ping) request id=0x0001, seq=7059/37659, ttl=1 (no response found!)

> Frame 289: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface \Device\NPF_{988EAC36-8A1D-4408-9902-CF19ACF13B21}, id 0
> Ethernet II, Src: IntelCon_78:63:29 (04:d3:b0:78:63:29), Dst: HuaweiTe_8c:26:c2 (2c:97:b1:8c:26:c2)
> Internet Protocol Version 4, Src: 192.168.1.21, Dst: 128.119.245.12
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 1500
  Identification: 0xbce5 (48357)
  > Flags: 0x20, More fragments
    0... .... = Reserved bit: Not set
    .0... .... = Don't fragment: Not set
    ..1.... = More fragments: Set
    ...0 0000 0000 0000 = Fragment Offset: 0
  Time to Live: 255
  Protocol: ICMP (1)
  Header Checksum: 0xa1f9 [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 192.168.1.21
  Destination Address: 128.119.245.12
  [Reassembled IPv4 in frame: 290]
> Data (1480 bytes)

```

13. 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? *The Flags bit for (more fragments) = set is indicating that the datagram has been fragmented. The fragment offset is 0 indicating that this is the first fragment. This first datagram has a total Length of 1500, including the header.*

14. 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? *The (Fragment offset) = 1480, indicating this is the second fragment. The (more fragments) bit is clear, indicating this is the last fragment.*

```

109 28.620.. 192.168.1.102 128.59.23.100 IPv4 1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=3300) [Reassembled in #110]
110 28.621.. 192.168.1.102 128.59.23.100 ICMP 562 Echo (ping) request id=0x0300, seq=32259/894, ttl=8 (no response found!)
111 28.640.. 192.168.1.102 128.59.23.100 IPv4 1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=3301) [Reassembled in #112]

> Frame 110: 562 bytes on wire (4496 bits), 562 bytes captured (4496 bits)
> Ethernet II, Src: Actionte_8a:70:1a (00:20:e0:8a:70:1a), Dst: LinksysG_da:af:73 (00:06:25:da:af:73)
> Internet Protocol Version 4, Src: 192.168.1.102, Dst: 128.59.23.100
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 548
  Identification: 0x3300 (13056)
  > Flags: 0x00
    0... .... = Reserved bit: Not set
    .0... .... = Don't fragment: Not set
    ..0.... = More fragments: Not set
    ...0 0101 1100 1000 = Fragment Offset: 1480
  Time to Live: 8
  Protocol: ICMP (1)
  Header Checksum: 0x2373 [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 192.168.1.102
  Destination Address: 128.59.23.100
  > [2 IPv4 Fragments (2008 bytes): #109(1480), #110(528)]
> Internet Control Message Protocol

```

15. What fields change in the IP header between the first and second fragment? *The IP header fields that changed between the fragments are total length, flags, fragment offset, and checksum.*

16. How many fragments were created from the original datagram? *After switching to 3500, there are 3 packets created from the original datagram*

17. What fields change in the IP header among the fragments? *The IP header fields that changed between all the packets are fragment offset, and checksum. Between the first two packets and the last packet, we see a change in total length, and in the flags. The first two packets have a total Length of 1500, with the more fragments bit set to 1, and the last packet has a total Length of 540, with the more fragments bit set to 0.*