Department of Electrical Engineering, UET Lahore
EE432: Computer Networks

Course Instructor: Dr. Naveed Nawaz	**Dated: 02/10/24**
Session: Fall 2024	**Semester: 7th**

LAB 7 Internet Protocol (IP)

Name	Roll. No.	Total Marks	Obtained Marks	Viva Marks
Ayesha Ahmad	2021-EE-052			

Checked on: ______________________________

Signature: ______________________________

1.1. Objectives

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.

We will also discuss IP address and discover its hierarchical nature. We will also discover the usage of subnet masks to specify the network portion of an IP address.

1.2. Instructions

1. Read carefully before starting the lab.
2. These exercises are to be done individually.
3. You are supposed to provide the answers to the questions listed at the end of this document and upload the completed report to your course's google classroom.
4. Avoid plagiarism by copying from the Internet or from your peers. You may refer to source/ text but you must paraphrase the original work. Your submitted work should be written by yourself.
5. Complete the lab half an hour before the lab ends.
6. At the end of the lab, a viva will be conducted to evaluate your understanding.

1.3. Background

2.3.1. IP Addressing

In this part, we are going to study IPv4 addressing format in which we use 32 bits to identify a host that connects to a TCP/IP network. The IPv4 address serves to standardize the logical address that is understood by all TCP/IP nodes thereby hiding the underlying heterogeneity that is a characteristic of link-layer networks.

IPv4 address is a 32 bit address containing some bits to identify the network and the remaining bits to identify a host in this network. It was decided earlier that we would have a few classes of IPv4 address according to which these 32 bits would be subdivided into network and hosts bits. These classes worked in the following way:

- Class A: 8 bits for the network and the remaining 24 bits for identifying the host.
- Class B: 16 bits for the network and the remaining 16 bits for identifying the host.
- Class C: 24 bits for the network and the remaining 8 bits for identifying the host.
- Class D was specified as a multicast class while another class, Class E was specified as an experimental class.

This simple division of IP addresses into these classes proved useful in the beginning due to its simplicity. However, such a scheme proved inflexible for the growing demands of the Internet because it led to wastage of addresses. For example, a point-to-point serial link requires only a network but it only requires for two nodes to be addressed. Using the class that has least number of host bits (Class C) still led to a lot of wastage of IP addresses---a class C IP network can be used to give addresses of 254 hosts (2 addresses of the total 2^8 addresses are reserved to refer to the network itself and to specify broadcast implying that all the network nodes should process the packet).

The next logical progress was variable length subnet masking in which instead of fixed classful addressing, we have a subnet mask in which we have more flexibility in defining the number of bits of the total 32 bits of IPv4 address that are used to identify the network. It must be pointed out that in an IP address, the network bits are contiguous and occur at the start (MSB) of an IP address. A subnet mask is a 32 bit number in which the leading bits are set to 1 commensurate to the number of network bits. For example, if 8 bits of an IP address is used to specify the network, and the remaining bits for specifying the host, then a subnet mask of 255.0.0.0 would be used (according to the decimal dotted notation explained next).

Decimal dotted notation: Since an IPv4 addresses (and subnet mask) are composed of 32 bits, it is easier to deal with 4 chunks of octets rather than with the 32 bits directly. In addition, each chunk comprising of 8 bits is represented by its decimal equivalent in the following way: 10000000 would be represented as 128. The IP address 10000000 00000000 00000000 00000001 would be represented as 128.0.0.1. Similarly, subnet masks are also described in decimal dotted notation. Subnet masks specify the demarcation between network portion and host portion, since the network bits are contiguous and occur at the leading side of an IP address, subnet masks would only have octets of the following form:

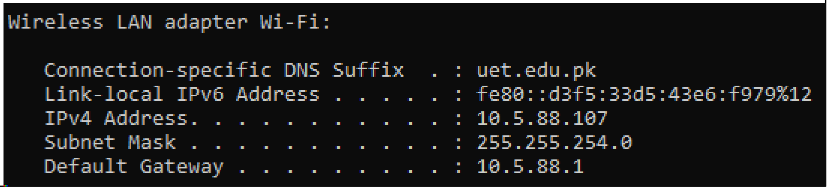
```
11111111: 255
11111110: 254
11111100: 252
11111000: 248
11110000: 240
11100000: 224
11000000: 192
10000000: 128
00000000: 0
```

As an example, a subnet mask of 255.192.0.0 would imply an IP address in which the first 10 bits specify the network part of the IP address and the remaining bits (22 bits) specify the host bits.

7.3.1.1. Questions

1. Determine the IP address of your lab machine using the ipconfig command. Also, write the subnet mask configured?

IP address of machine: 10.5.88.107
Subnet Mask: 255.255.254.0



```
Wireless LAN adapter Wi-Fi:

   Connection-specific DNS Suffix  . : uet.edu.pk
   Link-local IPv6 Address . . . . . : fe80::d3f5:33d5:43e6:f979%12
   IPv4 Address. . . . . . . . . . . : 10.5.88.107
   Subnet Mask . . . . . . . . . . . : 255.255.254.0
   Default Gateway . . . . . . . . . : 10.5.88.1
```

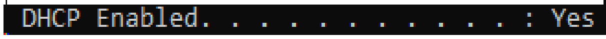
2. What is the network address of the network to which your machine belongs?
Hint: you can determine this by setting all the host bits of your IP address to 0 and obtaining the decimal dotted equivalent of the IP address; For example, the machine having IP address 10.128.232.21 having the subnet mask 255.128.0.0 belongs to the network 10.128.0.0.

IP address of machine : 10.5.88.107 -> 00001010-00000101-01011000-01101011
Subnet Mask : 255.255.254.0 -> 11111111-11111111-11111110-00000000

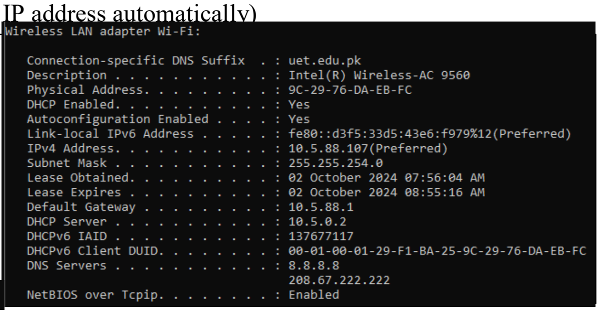
Network Address : **10.5.88.0** -> 00001010-00000101-01011000-00000000

3. Is the IP address being configured on your machine statically or dynamically? (Note, in dynamic configuration, your machine would act as the client of a DHCP server to obtain IP address automatically)

The IP address is being configured **dynamically** as a DHCP is ENABLED and a DHCP server is being configured.



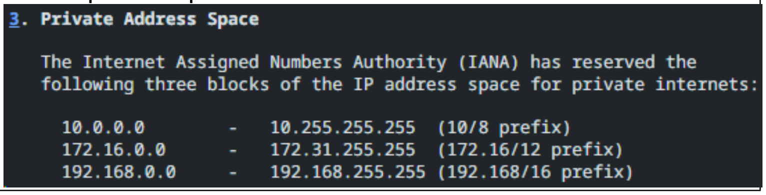
```
DHCP Enabled. . . . . . . . . . . : Yes
```



```
Wireless LAN adapter Wi-Fi:

   Connection-specific DNS Suffix  . : uet.edu.pk
   Description . . . . . . . . . . . : Intel(R) Wireless-AC 9560
   Physical Address. . . . . . . . . : 9C-29-76-DA-EB-FC
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::d3f5:33d5:43e6:f979%12(Preferred)
   IPv4 Address. . . . . . . . . . . : 10.5.88.107(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.254.0
   Lease Obtained. . . . . . . . . . : 02 October 2024 07:56:04 AM
   Lease Expires . . . . . . . . . . : 02 October 2024 08:55:16 AM
   Default Gateway . . . . . . . . . : 10.5.88.1
   DHCP Server . . . . . . . . . . . : 10.5.0.2
   DHCPv6 IAID . . . . . . . . . . . : 137677117
   DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-29-F1-BA-25-9C-29-76-DA-EB-FC
   DNS Servers . . . . . . . . . . . : 8.8.8.8
                                       208.67.222.222
   NetBIOS over Tcpip. . . . . . . . : Enabled
```

4. Is the IP address on your machine a private IP or a public IP?
(Please note that RFC 1918 describes the three IP network classes that have been dedicated for private usage; you can download this very short RFC through the URL: http://tools.ietf.org/html/rfc1918)

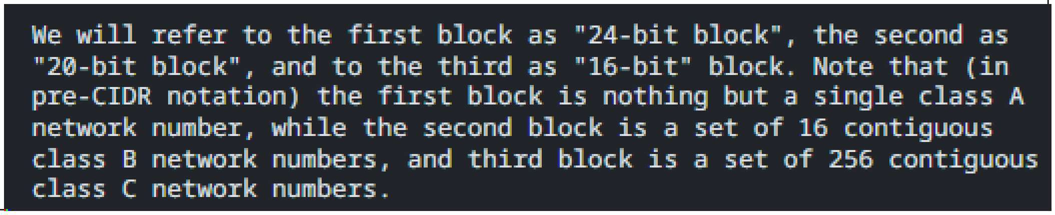
The IP address on my machine is a **PRIVATE IP** as it has 10/8 prefix.
The IP address 10.5.88.107 lies within the private space 10.0.0.0 - 10.255.255.255.



```
3. Private Address Space

   The Internet Assigned Numbers Authority (IANA) has reserved the
   following three blocks of the IP address space for private internets:

     10.0.0.0        -   10.255.255.255  (10/8 prefix)
     172.16.0.0      -   172.31.255.255  (172.16/12 prefix)
     192.168.0.0     -   192.168.255.255 (192.168/16 prefix)
```

5. If your answer to the question above is a private IP, then which IP network class (of those three defined in RFC 1918) does the IP address configured on your machine belongs to?

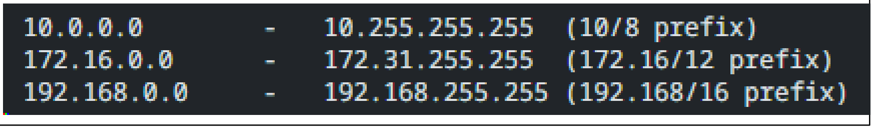
The IP of my computer belongs to the network **class A** as it belongs to the first block.



```
We will refer to the first block as "24-bit block", the second as
"20-bit block", and to the third as "16-bit" block. Note that (in
pre-CIDR notation) the first block is nothing but a single class A
network number, while the second block is a set of 16 contiguous
class B network numbers, and third block is a set of 256 contiguous
class C network numbers.
```

6. Write down all the three IP network classes that are private as defined by RFC 1918.

Class A ->	10.0.0.0	-	10.255.255.255 (10/8 prefix)
Class B ->	172.16.0.0	-	172.31.255.255 (172.16/12 prefix)
Class C ->	192.168.0.0	-	192.168.255.255 (192.168/16 prefix)

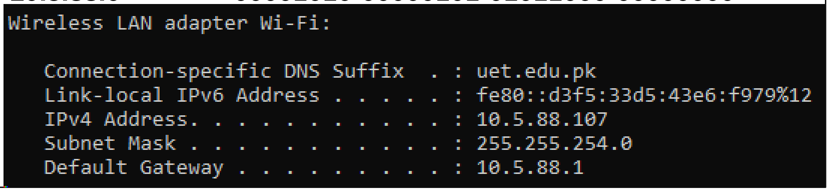


```
10.0.0.0        -   10.255.255.255  (10/8 prefix)
172.16.0.0      -   172.31.255.255  (172.16/12 prefix)
192.168.0.0     -   192.168.255.255 (192.168/16 prefix)
```

7. Note the IP address of the default gateway? Assuming the same subnet mask as is configured for your machine, determine the network address of this IP. Describe how it corresponds to the network address determined in question 2.

Default Gateway IP :	10.5.88.1	-> 00001010-00000101-01011000-00000001
Subnet Mask :	255.255.254.0	-> 11111111-11111111-11111110-00000000
Network Address :	**10.5.88.0**	-> 00001010-00000101-01011000-00000000

The network IP of the Default Gateway and the IPv4 Address are both the **same**.



```
Wireless LAN adapter Wi-Fi:

   Connection-specific DNS Suffix  . : uet.edu.pk
   Link-local IPv6 Address . . . . . : fe80::d3f5:33d5:43e6:f979%12
   IPv4 Address. . . . . . . . . . . : 10.5.88.107
   Subnet Mask . . . . . . . . . . . : 255.255.254.0
   Default Gateway . . . . . . . . . : 10.5.88.1
```

2.3.2. IP packet and IP fragmentation

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.

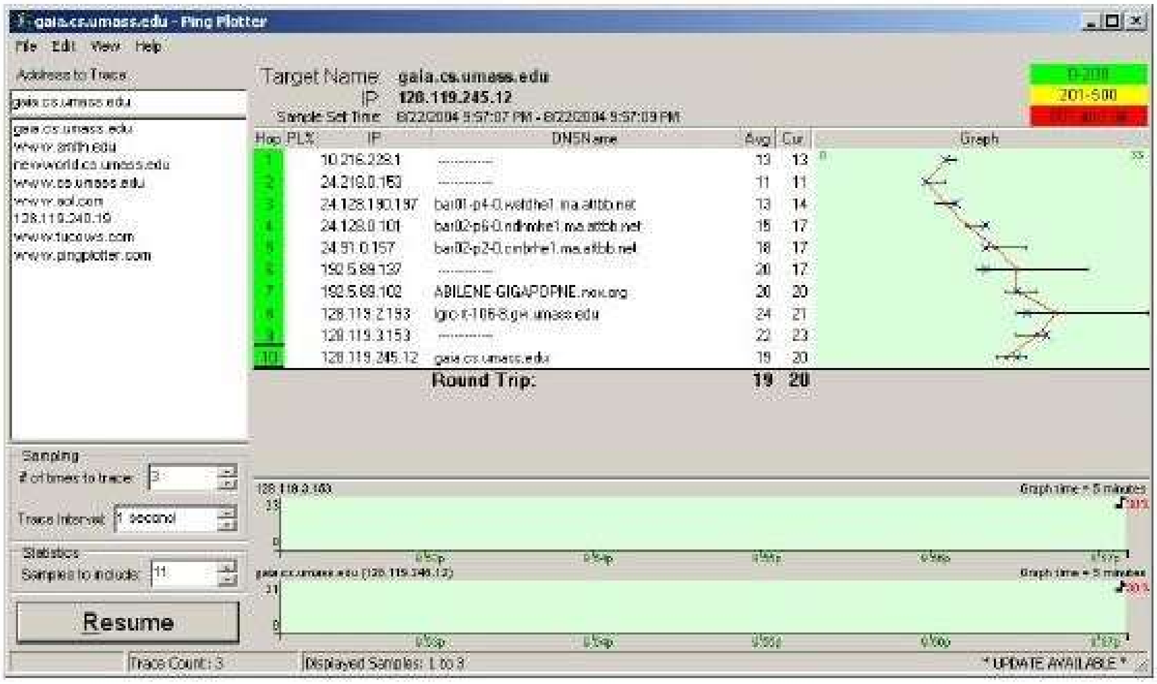
We'll want to run traceroute and have it send datagrams of various lengths.

The tracert program 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.

1.4. Procedure

1. Startup 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).
2. If you are using a Windows platform, startup 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:



Address to Trace: gaia.cs.umass.edu

gaia.cs.umass.edu
www.smith.edu
newworld.cs.umass.edu
www.cs.umass.edu
www.aol.com
128.119.240.19
www.tucows.com
www.pingplotter.com

Target Name: gaia.cs.umass.edu
IP: 128.119.245.12
Sample Set Time: 8/22/2004 9:57:07 PM - 8/22/2004 9:57:09 PM

Hop	PL%	IP	DNSName	Avg	Cur
1		10.216.228.1	----------	13	13
2		24.218.0.153	----------	11	11
3		24.128.190.197	bar01-p4-0.wsfdhe1.ma.attbb.net	13	14
4		24.128.0.101	bar02-p6-0.ndhmhe1.ma.attbb.net	15	17
5		24.91.0.157	bar02-p2-0.cmbrhe1.ma.attbb.net	18	17
6		192.5.89.137	----------	20	17
7		192.5.89.102	ABILENE-GIGAPOPNE.nox.org	20	20
8		128.119.2.193	lgrc-rt-106-8.gw.umass.edu	24	21
9		128.119.3.153	----------	22	23
10		128.119.245.12	gaia.cs.umass.edu	19	20
			Round Trip:	19	20

Sampling
# of times to trace: 3
Trace Interval: 1 second

Statistics
Samples to include: 11

128.119.3.153
gaia.cs.umass.edu (128.119.245.12)

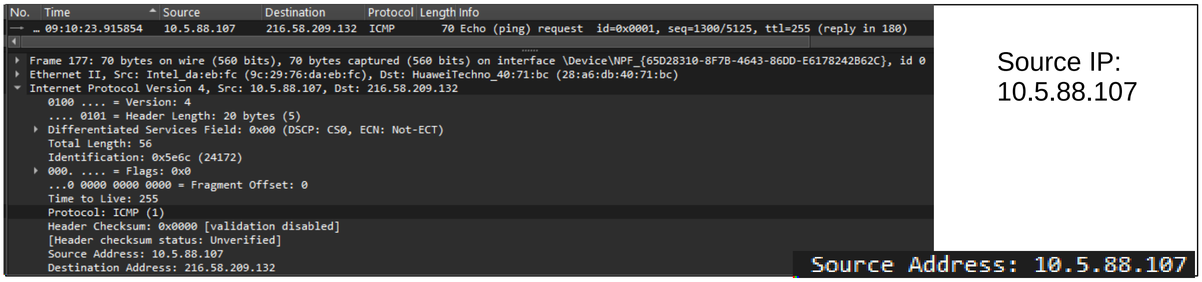
Trace Count: 3
Displayed Samples: 1 to 3

Figure 7.1: Pingplotter window

3. 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.
4. 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.
5. Stop Wireshark tracing.
6. 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 book's authors' Windows computers. 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.
7. 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.
8. In the questions below, it is assumed that you are using a Windows machine. Whenever possible, when answering a question you should hand in a printout of the packet(s) within the trace that you used to answer the question asked. Annotate the printout to explain your answer. 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.

2.4.1. Questions

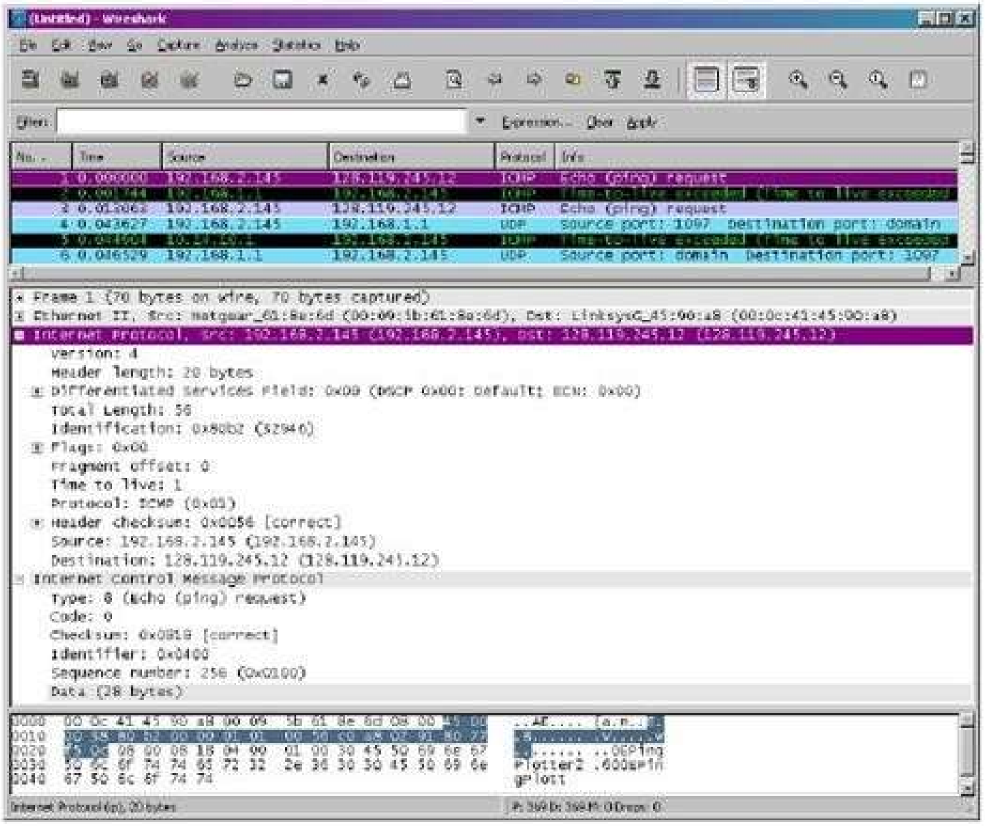
1. Select the first ICMP Echo Request message sent by your computer, and expand the Internet Protocol part of the packet in the packet details window. What is the IP address of your computer?



```
No.   Time               Source        Destination     Protocol Length Info
→ … 09:10:23.915854  10.5.88.107   216.58.209.132  ICMP       70 Echo (ping) request  id=0x0001, seq=1300/5125, ttl=255 (reply in 180)

▸ Frame 177: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface \Device\NPF_{65D28310-8F7B-4643-86DD-E6178242B62C}, id 0
▸ Ethernet II, Src: Intel_da:eb:fc (9c:29:76:da:eb:fc), Dst: HuaweiTechno_40:71:bc (28:a6:db:40:71:bc)
▾ Internet Protocol Version 4, Src: 10.5.88.107, Dst: 216.58.209.132
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 56
    Identification: 0x5e6c (24172)
  ▸ 000. .... = Flags: 0x0
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 255
    Protocol: ICMP (1)
    Header Checksum: 0x0000 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.5.88.107
    Destination Address: 216.58.209.132
```

Source IP:
10.5.88.107

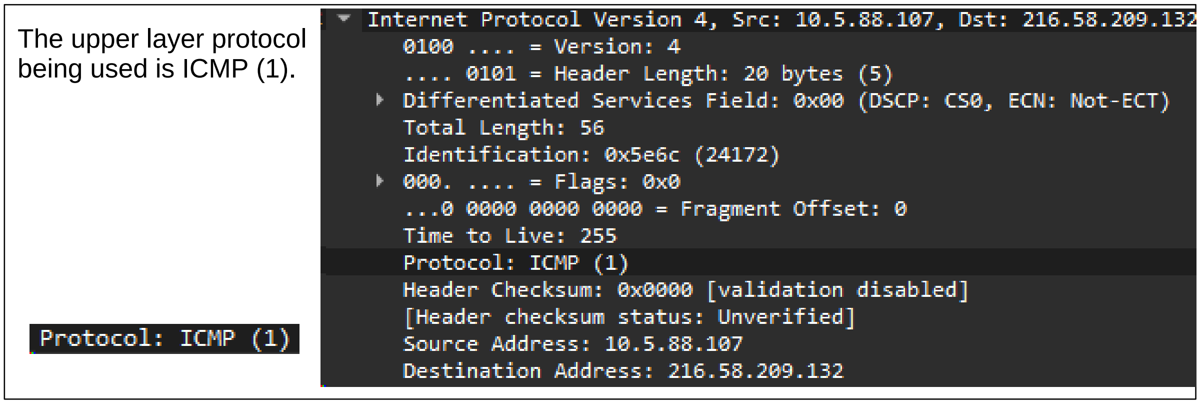
```
Source Address: 10.5.88.107
```


(Untitled) - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter:
Expression... Clear Apply

```
No. .  Time      Source           Destination      Protocol  Info
  1 0.000000  192.168.2.145    128.119.245.12   ICMP      Echo (ping) request
  2 0.001744  192.168.1.1      192.168.2.145    ICMP      Time-to-live exceeded (Time to live exceeded
  3 0.012062  192.168.2.145    128.119.245.12   ICMP      Echo (ping) request
  4 0.043627  192.168.2.145    192.168.1.1      UDP       Source port: 1097  Destination port: domain
  5 0.044908  10.19.10.1       192.168.2.145    ICMP      Time-to-live exceeded (Time to live exceeded
  6 0.046529  192.168.1.1      192.168.2.145    UDP       Source port: domain  Destination port: 1097

⊞ Frame 1 (70 bytes on wire, 70 bytes captured)
⊞ Ethernet II, Src: Netgear_61:8e:6d (00:09:5b:61:8e:6d), Dst: LinksysG_45:90:a8 (00:0c:41:45:90:a8)
⊟ Internet Protocol, Src: 192.168.2.145 (192.168.2.145), Dst: 128.119.245.12 (128.119.245.12)
    Version: 4
    Header length: 20 bytes
  ⊞ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
    Total Length: 56
    Identification: 0x80b2 (32946)
  ⊞ Flags: 0x00
    Fragment offset: 0
    Time to live: 1
    Protocol: ICMP (0x01)
  ⊞ Header checksum: 0x0056 [correct]
    Source: 192.168.2.145 (192.168.2.145)
    Destination: 128.119.245.12 (128.119.245.12)
⊟ Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0x0818 [correct]
    Identifier: 0x0400
    Sequence number: 256 (0x0100)
    Data (28 bytes)

0000  00 0c 41 45 90 a8 00 09  5b 61 8e 6d 08 00 45 00   ..AE.... [a.m..E.
0010  00 38 80 b2 00 00 01 01  00 56 c0 a8 02 91 80 77   .8...... .V.....w
0020  f5 0c 08 00 08 18 04 00  01 00 30 45 50 69 6e 67   ........ ..0EPing
0030  50 6c 6f 74 74 65 72 32  2e 36 30 30 45 50 69 6e   Plotter2 .600EPin
0040  67 50 6c 6f 74 74                                  gPlott

Internet Protocol (ip), 20 bytes                  P: 369 D: 369 M: 0 Drops: 0
```

Figure 7.2

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



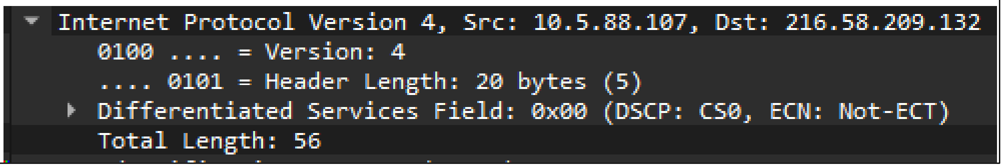
The upper layer protocol being used is ICMP (1).

```
Protocol: ICMP (1)
```

```
▾ Internet Protocol Version 4, Src: 10.5.88.107, Dst: 216.58.209.132
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 56
    Identification: 0x5e6c (24172)
  ▸ 000. .... = Flags: 0x0
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 255
    Protocol: ICMP (1)
    Header Checksum: 0x0000 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.5.88.107
    Destination Address: 216.58.209.132
```

3. 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.

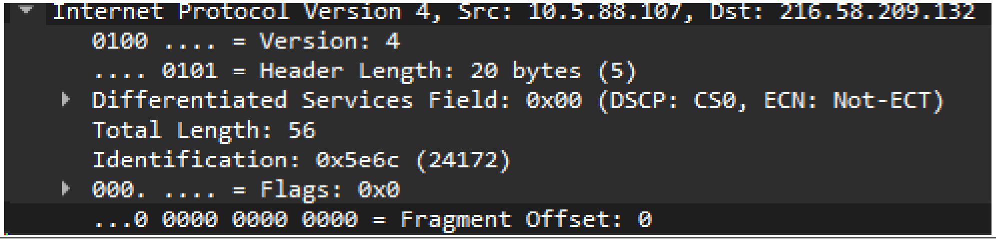
Total Length: 56 bytes
Header Length: 20 bytes
Payload Length: 56 - 20 = 36 bytes



```
▾ Internet Protocol Version 4, Src: 10.5.88.107, Dst: 216.58.209.132
     0100 .... = Version: 4
     .... 0101 = Header Length: 20 bytes (5)
   ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
     Total Length: 56
```

4. Has this IP datagram been fragmented? Explain how you determined whether or not the datagram has been fragmented.

Since the Fragment Offset is 0, so the IP datagram has NOT been fragmented.



```
▾ Internet Protocol Version 4, Src: 10.5.88.107, Dst: 216.58.209.132
     0100 .... = Version: 4
     .... 0101 = Header Length: 20 bytes (5)
   ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
     Total Length: 56
     Identification: 0x5e6c (24172)
   ▸ 000. .... = Flags: 0x0
     ...0 0000 0000 0000 = Fragment Offset: 0
```

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 my other protocols running on your computer) below this first ICMP. Use the down arrow to move through the ICMP messages sent by your computer.

5. Which fields in the IP datagram always change from one datagram to the next within this series of ICMP messages sent by your computer?

The fields of the IP datagram that always changes are:
Identification
Time to live
Header checksum (sometimes same but generally different)

6. Which fields stay constant? Which of the fields must stay constant? Which fields must change? Why?

The fields of the IP datagram that **stay constant** are:
Version Header Length Total Length
Fragment Offset Upper Layer Protocol

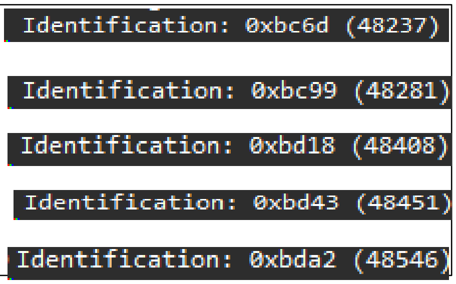
The fields that **stay constant from client to server or from server to client are**:
Source IP Destination IP Differentiated Services

The fields that **must stay constant** are:
Version as all packets are using IPv4
Header length as ICMP packets have a fixed header length
Total Length as the packet size was defined as 56 in Ping Plotter
Upper Layer Protocol as these are all ICMP packets
Fragment Offset as these packets are not fragmented

The fields that **must change** are :
Identification as IP packets must have different IDs
Time to live as the traceroute increments with each subsequent packet
Header checksum since header changes, so must the checksum

7. Describe the pattern you see in the values in the Identification field of the IP datagram.

The Identification field of the IP datagram increments with each ICMP ping request.

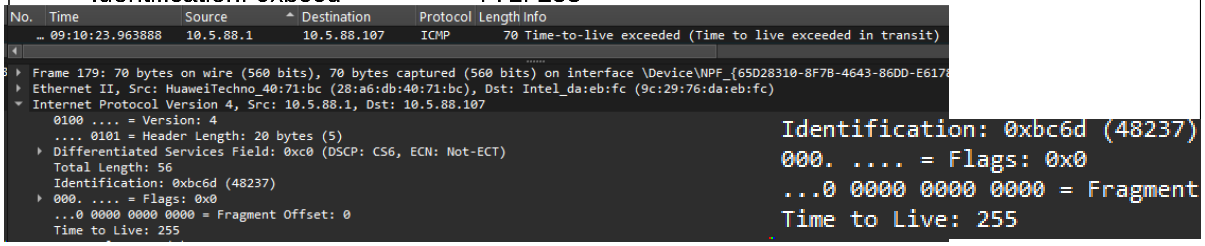


```
Identification: 0xbc6d (48237)
Identification: 0xbc99 (48281)
Identification: 0xbd18 (48408)
Identification: 0xbd43 (48451)
Identification: 0xbda2 (48546)
```

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.

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

Identification: 0xbc6d TTL: 255



```
No.  Time             Source       Destination   Protocol Length Info
… 09:10:23.963888  10.5.88.1    10.5.88.107   ICMP       70 Time-to-live exceeded (Time to live exceeded in transit)

Frame 179: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface \Device\NPF_{65D28310-8F7B-4643-86DD-E617
Ethernet II, Src: HuaweiTechno_40:71:bc (28:a6:db:40:71:bc), Dst: Intel_da:eb:fc (9c:29:76:da:eb:fc)
Internet Protocol Version 4, Src: 10.5.88.1, Dst: 10.5.88.107
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  Differentiated Services Field: 0xc0 (DSCP: CS6, ECN: Not-ECT)
    Total Length: 56
    Identification: 0xbc6d (48237)
  000. .... = Flags: 0x0
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 255
```

```
Identification: 0xbc6d (48237)
000. .... = Flags: 0x0
...0 0000 0000 0000 = Fragment
Time to Live: 255
```

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

The Identification field keeps incrementing but the TTL remains 255.

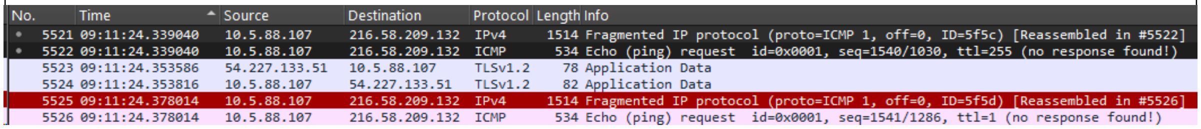
The identification field is a unique value. When two or more IP datagrams have the same identification value, then it means that these IP datagrams are fragments of a single large IP datagram. So, for all the ICMP TTL-exceeded replies the Identification field remains constant.

The TTL (Time to Live) for all the ICMP TTL-exceeded replies remains 255 as it is the maximum value of a single octet.

2.4.2. Fragmentation

1. Sort the packet listing according to time again by clicking on the Time column.
2. 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-1packet trace. If your computer has an Ethernet interface, a packet size of 2000 should cause fragmentation]

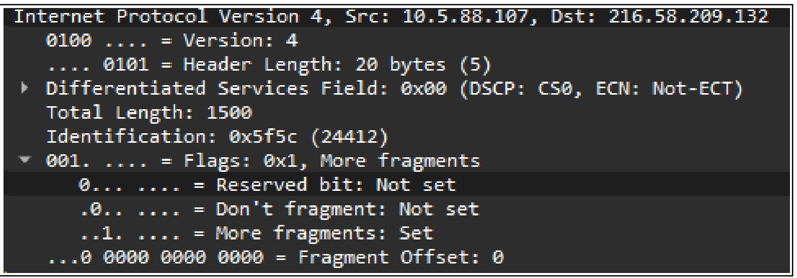
After the packet size was set to 2000, the message has been **fragmented.**



```
No.  Time             Source          Destination     Protocol Length Info
5521 09:11:24.339040  10.5.88.107     216.58.209.132  IPv4       1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=5f5c) [Reassembled in #5522]
5522 09:11:24.339040  10.5.88.107     216.58.209.132  ICMP        534 Echo (ping) request  id=0x0001, seq=1540/1030, ttl=255 (no response found!)
5523 09:11:24.353586  54.227.133.51   10.5.88.107     TLSv1.2      78 Application Data
5524 09:11:24.353816  10.5.88.107     54.227.133.51   TLSv1.2      82 Application Data
5525 09:11:24.378014  10.5.88.107     216.58.209.132  IPv4       1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=5f5d) [Reassembled in #5526]
5526 09:11:24.378014  10.5.88.107     216.58.209.132  ICMP        534 Echo (ping) request  id=0x0001, seq=1541/1286, ttl=1 (no response found!)
```

3. 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?

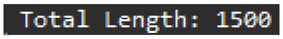
The IP header **Fragment offset is 0** while the **More Fragments Flag is set** which indicates that this is the **first** fragment.
For the latter fragment the **fragment offset is non-zero**.



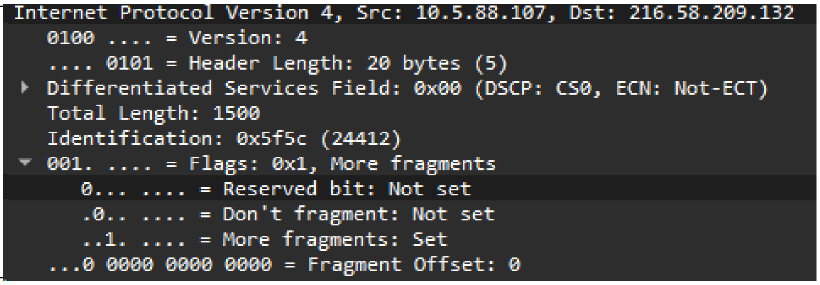
```
Internet Protocol Version 4, Src: 10.5.88.107, Dst: 216.58.209.132
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 1500
    Identification: 0x5f5c (24412)
  ▾ 001. .... = Flags: 0x1, More fragments
       0... .... = Reserved bit: Not set
       .0.. .... = Don't fragment: Not set
       ..1. .... = More fragments: Set
    ...0 0000 0000 0000 = Fragment Offset: 0
```

4. How long is this IP datagram?

The Total Length of this Datagram is **1500 bytes** including the header.



```
Total Length: 1500
```

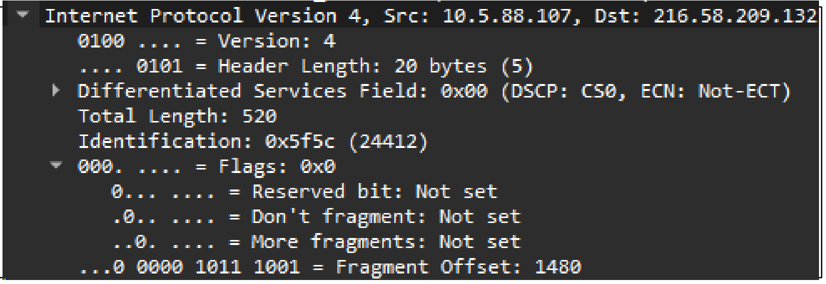


```
Internet Protocol Version 4, Src: 10.5.88.107, Dst: 216.58.209.132
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 1500
    Identification: 0x5f5c (24412)
  ▾ 001. .... = Flags: 0x1, More fragments
       0... .... = Reserved bit: Not set
       .0.. .... = Don't fragment: Not set
       ..1. .... = More fragments: Set
    ...0 0000 0000 0000 = Fragment Offset: 0
```

5. 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 is non-zero** which shows that this is not the first fragment.

There are NO more fragments as the **More Fragments Flag is not set**.



```
▾ Internet Protocol Version 4, Src: 10.5.88.107, Dst: 216.58.209.132
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 520
    Identification: 0x5f5c (24412)
  ▾ 000. .... = Flags: 0x0
       0... .... = Reserved bit: Not set
       .0.. .... = Don't fragment: Not set
       ..0. .... = More fragments: Not set
    ...0 0000 1011 1001 = Fragment Offset: 1480
```

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

The following fields change between the first and second fragment:

Total length
Flags
Fragment offset
Checksum

Assessment Rubrics for EE432: Computer Networks Lab 7

Student Name: ______________________________ Roll Number: ______________________________

Method:

Lab report evaluation and instructor observation during lab sessions.

Outcomes Assessed:

a. Ability to conduct experiments as well as to analyze and interpret data
b. Ability to adhere to safety and disciplinary rules
c. Ability to use the techniques, skills and modern engineering tools necessary for engineering practice

Performance	**Exceeds expectation (5-4)**	**Meets expectation (3-2)**	**Does not meet expectation (1)**	**Marks**
Realization of experiment (a)	Downloads and installs required software and sets up the system according to the experiment requirements	Needs guidance to set up the system according to the experiment requirements	Incapable of selecting relevant software to the experiment and unable to setup the system with required software tools	
Conducting experiment (a, c)	Carries out each procedural step in a satisfactory manner and studies outputs of the software application rigorously	Needs assistance or guidance to proceed through experiment steps, studies outputs with minor errors in interpretation	Unable to carry out procedural steps and make any useful observations of outputs	
Laboratory safety and disciplinary rules (b)	Observes lab safety rules; adheres to the lab disciplinary guidelines aptly	Observes safety rules and disciplinary guidelines with minor deviations	Disregards lab safety and disciplinary rules	
Data collection (c)	Completes data collection from the experiment setup by following procedural steps, ensures that the data is entered in the lab manual according to the specified instructions	Completes data collection with minor error and enters data in lab manual with slight deviation from guidelines	Fails at collecting data by giving proper inputs and observing output states of experiment setup, unable to fill the lab manual properly	
Data analysis (a, c)	Analyzes the data obtained from experiment thoroughly and accurately verifies it with theoretical understanding, accounts for any discrepancy in data from theory with sound explanation	Analyzes data with minor error and correlates it with theoretical values reasonably. Attempts to account for any discrepancy in data from theory	Unable to establish the relationship between practical and theoretical values and lacks the theoretical understanding to explain any discrepancy in data	