ECE 358: Computer Networks
Fall 2014
Project 3: Encapsulation and Network Utilities

Date of Submission: **December 1, 2014**

Submitted by:

20414514:	**Louisa Chong**	**l5chong@uwaterloo.ca**
20434209:	**Simarpreet Singh**	**s244sing@uwaterloo.ca**

Marks received:

Table of Contents

Question 1

Frame 2 (TCP):

00 e0 7d 8d bb 83 00 e0 7d 8d bb 91 08 00 45 00
00 28 f5 00 40 00 80 06 f2 e2 c0 a8 01 0a 80 2e
d1 0b 04 11 00 17 00 04 58 c9 cb 56 28 86 50 11
1d 68 2e ad 00 00

We identified 3 major headers in the frame; Ethernet header, IP header and TCP header. The green highlights indicates the ethernet header, yellow highlights indicates the IP header and the red highlights indicates the TCP header. The analysis is as following:

Ethernet header (in green highlights):
00 e0 7d 8d bb 83: Ethernet destination address is 00:e0:7d:8d:bb:83
00 e0 7d 8d bb 91: Ethernet source address is 00:e0:7d:8d:bb:91
08 00: The payload type is Internet Protocol version 4, IPv4

IP header (in yellow highlights):
45: 4<u>5</u>: The IP packet is in Internet Protocol version 4, IPv4
4<u>5</u>: The Internet Header Length is 5 x 4 = 20 bytes
00: Type of service, 0x00 = 0b00000000
0b<u>000</u>00000: The precedence of the packet is routine, CS0
0b000<u>00</u>000: Type of service, ToS, bits
0b0000<u>0</u>000: Normal Delay
0b00000<u>0</u>00: Normal throughput
0b000000<u>0</u>0: Normal Reliability
0b00000<u>00</u>: Not Used
00 28: Total IP datagram is 40 (0x0028) bytes
f5 00: Datagram identification is 62720
40 00: Fragment flags and offset 0x4000 = 0b0100000000000000
0b<u>010</u>0000000000000: Fragment flags
0b<u>0</u>100000000000000: Reserved bit, must be 0
0b0<u>1</u>00000000000000: Do not fragment set to True
0b01<u>0</u>0000000000000: More fragment set is False
0b010<u>0000000000000</u>: Fragment offset is 0
80: Time to live is 128 (0x80) more hops
06: Protocol is TCP
f2 e2: Header Checksum
c0 a8 01 0a: Source IP address is 192.168.1.10
80 2e d1 0b: Destination IP address is 128.46.209.11

TCP headers (in red highlights)
04 11: Source Port is 1041. This is assigned by the OS
00 17: Destination Port is 23. This is a port to used to send unencrypted text using telnet protocol
00 04 58 c9: Sequence Number is 284873
cb 56 28 86: Acknowledgment number is 3411421318
50 11: Data offset by 20 (5x4) bytes, 0x5011: 0b0101000000010001
0b0101000000010001: Reserved bits, must be 0
0b0101000000010001: Urgent bits, URG, is set to False
0b0101000000010001: Acknowledge bit, ACK, is True
0b0101000000010001: Push bit, PSH, is False
0b0101000000010001: Reset bit, RST, is set to False
0b0101000000010001: Synchronize sequence numbers, SYN, is set to False
0b0101000000010001: No more data from sender, FIN, is set to True
1d 68: The receiver window size is set to 7528(0x1d68) bytes
2e ad: Checksum of the TCP segment
00 00: Urgent Pointer is not used

Data:
None

Frame 6 (UDP):
00 26 16 00 00 d2 00 0c 29 50 a9 fc 08 00 45 00
00 29 00 00 40 00 40 11 b9 04 c0 a8 00 65 c0 a8
00 0a c2 f1 13 e6 00 15 74 a8 01 00 00 00 00 02
00 0d 01 00 00 75 30

We identified 3 major headers in the frame; Ethernet header, IP header and UDP header. The green highlights indicates the ethernet header, yellow highlights indicates the IP header and the red highlights indicates the UDP header. The analysis is as following:

Ethernet header (in green highlights):
00 26 16 00 00 d2: Ethernet destination address is 00:26:16:00:00:d2
00 0c 29 50 a9 fc: Ethernet source address is 00:0c:29:50:a9:fc
08 00: The payload type is Internet Protocol version 4, IPv4

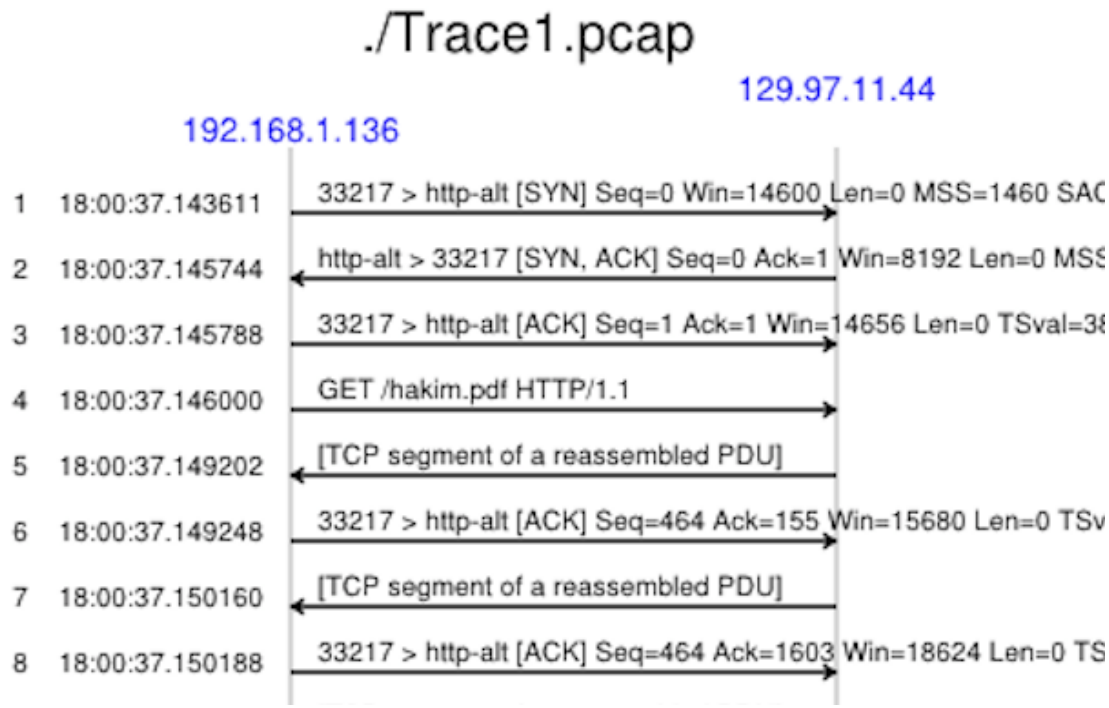
IP header (in yellow highlights):
45: 45: The IP packet is in Internet Protocol version 4, IPv4
45: The Internet Header Length is 5 x 4 = 20 bytes
00: Type of service, 0x00 = 0b00000000
0b00000000: The precedence of the packet is routine, CS0
0b00000000: Type of service, ToS, bits

0b00000000: Normal Delay
0b00000000: Normal throughput
0b00000000: Normal Reliability
0b00000000: Not Used
00 29: Total IP datagram is 41 (0x0029) bytes
00 00: Datagram identification is 0
40 00: Fragment flags and offset 0x4000 = 0b0100000000000000
0b010000000000000: Fragment flags
0b0100000000000000: Reserved bit, must be 0
0b0100000000000000: Do not fragment set to True
0b0100000000000000: More fragment set is False
0b0100000000000000: Fragment offset is 0
40: Time to live is 64 (0x40) more hops
11: Protocol is UDP
b9 04: Header Checksum
c0 a8 00 65: Source IP address is 192.168.0.101
c0 a8 00 0a: Destination IP address is 192.168.0.10

UDP headers (in red highlights)
c2 f1: Source Port is 49905. This is assigned by the OS
13 e6: Destination Port is 5094. This is assigned by the OS
00 15: Length 21 bytes
74 a8: Checksum

Data (in grey highlights)
01 00 00 00 00 02 00 0d 01 00 00 75 30: data

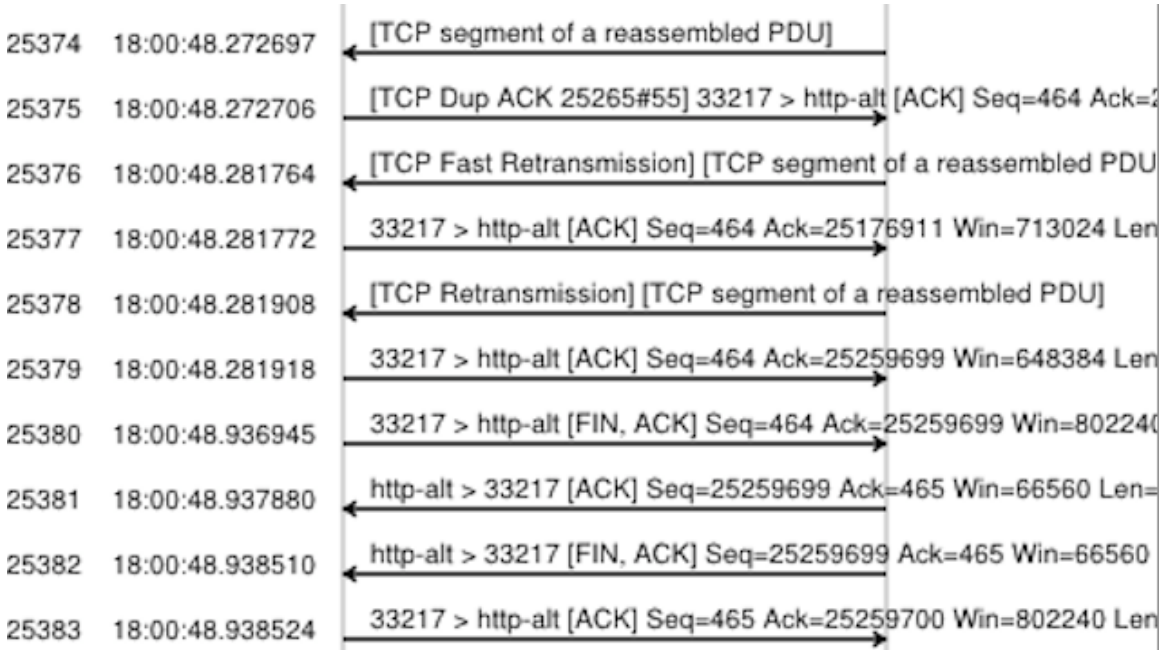
Question 2



In the above call flow diagram we can observe that the source machine (192.168.1.136) establishes a TCP connection between the two machines. This is done via a handshake packet with the SYN flag enabled. The destination machine, 129.97.11.44 responds back with a ACK or an acknowledgement informing the source machine that it has received a request for establishing a TCP connection between the two machines. This is further ACK'd by the source machine and a formal TCP request is made in call #3.

After the establishment of the TCP connection, the source machines finally sends a GET request towards the destination machine. This GET request encodes a file name which it wants to "get" from the destination machine to the source machine. There is no clear indication made by the destination machine until it receives the request about the file actually being present on the target machine (129.97.11.44). After receiving the GET request the destination machine (129.97.11.44) starts encoding the hakim.pdf file into discrete chunks. These chunks are limited by the MTU size as to how big they can be when transmitting over a TCP connection. This number in this case is at 1500 bytes. After determining this, the destination machine starts sending back packets to the source machine that generated this TCP call. One thing to keep in mind is that these TCP segments are sent out of sync and it is the job of the source machine that initiate this TCP call to make sure that the TCP segments are re-assembled properly.

...transmission continues, output truncated for brevity of flow chart…



After the constant transmission of TCP segments and the ACK's to the destination machine from the source machine, the file successfully gets across. During the course of the TCP transmission, it is possible that a same TCP segment may get sent twice to the source machine. In this case as indicated above, the source machine send a DUP ACK call to the sender saying that it received a segment that it already had. In this case the sender realizes this and tries to re-transmit a new segment that it thinks hasn't been sent yet.

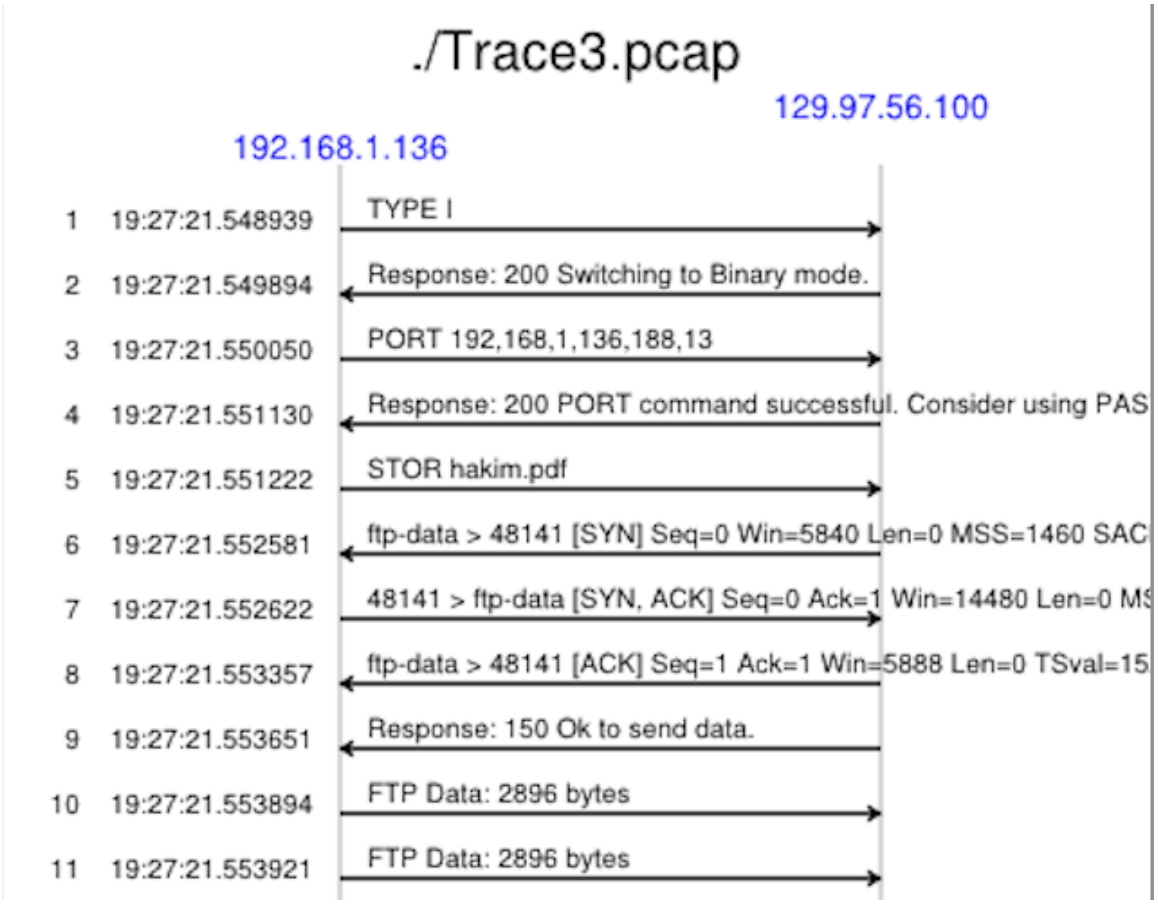
When the file is fully transferred and re-assembled, the source machine sends a FIN ACK call to the sender machine indicating that the source has received the entire file and it would now like to stop receiving anymore TCP segments. The sender machine acknowledges this fact by sending another ACK to the initial source machine and the connection is then formally initiated to be terminated by the sender as it sends a FIN ACK call to the original source machine. The source sees this call and ACK's it back. This leads to the termination of the TCP connection between the two machines and finishes the file transfer between the distant hosts.

The round trip time can be calculated as follows:

Filter: tcp.flags.syn == 1 Expression... Clear Apply

No.	Time	Source	Destination	Protocol
1	0.000000	192.168.1.136	129.97.11.44	TCP
2	0.002133	129.97.11.44	192.168.1.136	TCP

By noting the delta between the two SYN calls, we can determine the overall RTT between the two hosts. In this case the Round Trip Time was **0.002133 seconds**



In this next trace we see that the host machine is making a request to the destination machine with a TYPE I connection establishment. The response that it receives from the destination machine is 200, which stands for “OK”. The destination machines also changes the connection type to binary mode. This is done as a part of the transfer of files between the two machines. The data being transferred is a binary file hence the connection type is changed from simple ASCII to binary mode.

After switching to binary mode, the host machine makes a request for storing the hakim.pdf file to the destination machine. The destination machine responds back by sending in a synchronization flag also known as the SYN frame. This is replied back by the host using a SYN ACK segment. This means that the host ACK'd the file transfer and is now ready to send the file. After this another response for a specified data segment approval is sent by the receiver to the host machine. After this the host machine starts sending chunks of the file data across. It does so by breaking into fragments. This can be seen in the calls to follow after the connection is setup between the two machines.

...transmission continues, output truncated for brevity of flow chart…

After all the segments of the file are received at the destination, the destination send a FIN ACK call back to the sender. This indicates that the receiver has received all the parts of the file and is done with the receival process. This is ACK'd by the send using a ACK call. After this the receiver sends a 226 OK call which indicates that it is ready to terminate the line. This quickly follows a ACK and the termination from the host side which leads to the end of the call

No.	Time	Source	Destination	Protocol	Length
1	0.000000	192.168.1.136	129.97.56.100	FTP	74
2	0.000955	129.97.56.100	192.168.1.136	FTP	97

Round Trip Time = 0.000955 seconds.

Question 3 (arp)

a. Arp is used to manipulate the kernel address resolution protocol cache. The primary functionality of Arp is to clear the address mapping entries. Users can also manually set up address mappings. Arp also provides the functionality of dumping all ARP cache.

b.

Output of `/sbin/arp -a`

```
exsw02-circuitnet.uwaterloo.ca (129.97.56.1) at 00:17:A4:86:29:00 [ether] on eth0
ecemail.uwaterloo.ca (129.97.56.26) at C8:60:00:5F:BA:81 [ether] on eth0
```

From the output, we see that exsw02-circuitnet.uwaterloo.ca (129.97.56.1) has a MAC address of 00:17:A4:86:29:00

Question 4 (ifconfig)

a. ifconfig is used to query a machine's interfaces and its associated properties. Ifconfig is used to configure the kernel-resident network interfaces. It is used at boot time to set up interfaces as necessary.

b.

Output of `/sbin/ifconfig -a`

```
eth0      Link encap:Ethernet  HWaddr 04:01:2c:39:e8:01
          inet addr:162.273.2.14  Bcast:162.233.1.255  Mask:255.255.255.0
          inet6 addr: fe80::601:2dff:fe19:e891/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:14383226 errors:0 dropped:0 overruns:0 frame:0
          TX packets:5912726 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2947380427 (2.9 GB)  TX bytes:1153854301 (1.1 GB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:26782 errors:0 dropped:0 overruns:0 frame:0
          TX packets:26782 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:2794143 (2.7 MB)  TX bytes:2794143 (2.7 MB)
```

Here we can see the various properties of the interfaces available on the machine. This machines happens to have the various stats as displayed above. We see various stats like the MTU size, the local loopback interface and also the number of bytes Rx'd and Tx'd so far.

Question 5 (netstat)

a. Netstat is used to print information about the linux networking subsystem based on the arguments the user enters

b.

Output of `netstat -in`

```
Kernel Interface table
Iface       MTU Met    RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
eth0       1500   0 29644730      0      0      0 30832007      0      0      0 BMRU
lo        16436   0 13772480      0      0      0 13772480      0      0      0 LRU
```

From the result above, we observed 29644730 packets are received and 30832007 packets are transmitted by interface eth0

c.

Output of `netstat -r`

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
129.97.56.0     *               255.255.255.0   U         0 0          0 eth0
169.254.0.0     *               255.255.0.0     U         0 0          0 eth0
default         exsw02-circuitn 0.0.0.0         UG        0 0          0 eth0
```

The result we obtained from netstat is the kernel routing table. Using the destination and genmask information, we can obtain the network id. For example, Destination of 129.97.56.0 and a gen mask of 255.255.255.0, we get a network id of 129.97.56.0/24. The Gateway information tells the information of the next hop. The flags columns describes the route. For example, U is used to indicate that the interface to be used is up. UG is indicated the gateway to be used is up. MSS is the default maximum segment size for the TCP connection at the given route. Windows indicates the default window size for the TCP connection at the route. irtt indicates the initial route trip time. Lastly, Iface tell the interface which packets for the route will be sent.

Question 6 (nslookup)

a. Nslookup is a program to query Internet domain name servers. It shows the hostname and it's IP address by querying the specified DNS server.

b. In the outputs below we can the DN server that was used to retrieve the IP addresses for the specified hosts. In some cases we obtain multiple addresses as there might be several hosts mirroring the website.

1.

```
[simar@TuxCore ~ ]$ nslookup www.uwaterloo.ca
Server:         8.8.4.4
Address:        8.8.4.4#53

Non-authoritative answer:
www.uwaterloo.ca        canonical name = uwaterloo.ca.
Name:   uwaterloo.ca
Address: 129.97.149.158
```

2.

```
[simar@TuxCore ~ ]$ nslookup www.youtube.com
Server:         8.8.4.4
Address:        8.8.4.4#53

Non-authoritative answer:
www.youtube.com canonical name = youtube-ui.l.google.com.
Name:   youtube-ui.l.google.com
Address: 74.125.226.3
Name:   youtube-ui.l.google.com
Address: 74.125.226.7
Name:   youtube-ui.l.google.com
Address: 74.125.226.4
Name:   youtube-ui.l.google.com
Address: 74.125.226.1
Name:   youtube-ui.l.google.com
Address: 74.125.226.6
Name:   youtube-ui.l.google.com
Address: 74.125.226.2
Name:   youtube-ui.l.google.com
Address: 74.125.226.14
Name:   youtube-ui.l.google.com
Address: 74.125.226.0
Name:   youtube-ui.l.google.com
Address: 74.125.226.9
```

3.

```
[simar@TuxCore ~ ]$ nslookup www.gmail.com
Server:         8.8.4.4
Address:        8.8.4.4#53

Non-authoritative answer:
www.gmail.com   canonical name = mail.google.com.
mail.google.com canonical name = googlemail.l.google.com.
Name:   googlemail.l.google.com
Address: 74.125.226.21
Name:   googlemail.l.google.com
Address: 74.125.226.22
```

4.

```
[simar@TuxCore ~ ]$ nslookup www.facebook.com
Server:         8.8.4.4
Address:        8.8.4.4#53

Non-authoritative answer:
www.facebook.com        canonical name = star.c10r.facebook.com.
Name:   star.c10r.facebook.com
Address: 31.13.71.96
```

5.

```
[simar@TuxCore ~ ]$ nslookup www.brasil.gov.br
Server:         8.8.4.4
Address:        8.8.4.4#53

Non-authoritative answer:
Name:   www.brasil.gov.br
Address: 161.148.175.40
```

Question 7 (ping)

a. Ping is used to send ICMP request to a network hosts. The request made is called a ECHO_REQUEST. The datagram used in a ECHO_REQUEST consists of an IP, ICMP header and a time value following arbitrary number of bytes used to fill up the packet

b.

1.

```
PING uwaterloo.ca (129.97.149.158) 56(84) bytes of data.
64 bytes from wms.uwaterloo.ca (129.97.149.158): icmp_seq=1 ttl=122 time=1.16 ms
64 bytes from wms.uwaterloo.ca (129.97.149.158): icmp_seq=2 ttl=122 time=1.08 ms
64 bytes from wms.uwaterloo.ca (129.97.149.158): icmp_seq=3 ttl=122 time=1.23 ms
64 bytes from wms.uwaterloo.ca (129.97.149.158): icmp_seq=4 ttl=122 time=1.03 ms
64 bytes from wms.uwaterloo.ca (129.97.149.158): icmp_seq=5 ttl=122 time=1.07 ms

--- uwaterloo.ca ping statistics ---
```

```
5 packets transmitted, 5 received, 0% packet loss, time 4000ms
rtt min/avg/max/mdev = 1.031/1.115/1.231/0.077 ms
```

From the ping result above, we see that we sent out 5 ECHO_REQUESTs to the www.uwaterloo.ca (129.97.149.158). From the responses. we see that we the 5 requests have sequence from 1 to 5. The requests we made also have a time to live of 122 hops. The time associated with each request shows the round trip time. The time varies between 1.03ms to 1.23ms. Finally, ping provided with statistic of the requests. It shows that we sent 5 packets and all of them are transmitted successfully. The statistic also concluded the minimum round trip time was 1.031ms. The average round trip time was 1.114ms. The maximum round trip time was 1.231ms. Finally, the difference between the maximum and minimum round trip time is 0.077ms.

2.

```
PING youtube-ui.l.google.com (71.19.173.112) 56(84) bytes of data.
64 bytes from 71-19-173-112.dedicated.allstream.net (71.19.173.112): icmp_seq=1 ttl=57
time=2.44 ms
64 bytes from 71-19-173-112.dedicated.allstream.net (71.19.173.112): icmp_seq=2 ttl=57
time=2.37 ms
64 bytes from 71-19-173-112.dedicated.allstream.net (71.19.173.112): icmp_seq=3 ttl=57
time=2.30 ms
64 bytes from 71-19-173-112.dedicated.allstream.net (71.19.173.112): icmp_seq=4 ttl=57
time=2.34 ms
64 bytes from 71-19-173-112.dedicated.allstream.net (71.19.173.112): icmp_seq=5 ttl=57
time=2.43 ms

--- youtube-ui.l.google.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4000ms
rtt min/avg/max/mdev = 2.302/2.378/2.447/0.069 ms
```

From the ping result above, we see that we also sent out 5 ECHO_REQUESTs to the www.youtube.com (71.19.173.112). We observed a similar result from the ping. The time to live for the the requests are 56 hops instead of 122 hops. The round trip time ranges between 2.302ms to 2.447ms. The response time increased compared with the responses for www.uwaterloo.ca. This is because the server for youtube is further than the university of waterloo one. From the ping statistic provided, it concluded 5 packets were sent and all of them are transmitted successfully. The statistic also concluded the minimum round trip time was 2.301ms. The average round trip time was 2.378ms. The maximum round trip time was 2.447ms. Finally, the difference between the maximum and minimum round trip time is 0.069ms.

3.

```
PING googlemail.l.google.com (173.194.43.117) 56(84) bytes of data.
64 bytes from yyz08s10-in-f21.1e100.net (173.194.43.117): icmp_seq=1 ttl=54 time=3.00 ms
64 bytes from yyz08s10-in-f21.1e100.net (173.194.43.117): icmp_seq=2 ttl=54 time=2.99 ms
64 bytes from yyz08s10-in-f21.1e100.net (173.194.43.117): icmp_seq=3 ttl=54 time=3.18 ms
64 bytes from yyz08s10-in-f21.1e100.net (173.194.43.117): icmp_seq=4 ttl=54 time=2.99 ms
64 bytes from yyz08s10-in-f21.1e100.net (173.194.43.117): icmp_seq=5 ttl=54 time=2.99 ms

--- googlemail.l.google.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 3999ms
rtt min/avg/max/mdev = 2.991/3.033/3.184/0.083 ms
```

Similar to the previous questions, we made 5 ICMP requests to www.gmail.com. The time to live for this is 54 hops. The round trip time varies from 2.00ms to 3.18ms. All 5 packets we transmitted was received. The minimum round trip is 2.991ms. The average round trip time is 3.033. The maximum round trip time is 2.991ms. Finally, time difference between the maximum and the minimum round trip time is 0.083ms.

4.

```
PING star.c10r.facebook.com (31.13.74.144) 56(84) bytes of data.
64 bytes from edge-star-shv-10-ord1.facebook.com (31.13.74.144): icmp_seq=1 ttl=86 time=12.2
ms
64 bytes from edge-star-shv-10-ord1.facebook.com (31.13.74.144): icmp_seq=2 ttl=86 time=12.2
ms
64 bytes from edge-star-shv-10-ord1.facebook.com (31.13.74.144): icmp_seq=3 ttl=86 time=12.2
ms
64 bytes from edge-star-shv-10-ord1.facebook.com (31.13.74.144): icmp_seq=4 ttl=86 time=12.4
ms
64 bytes from edge-star-shv-10-ord1.facebook.com (31.13.74.144): icmp_seq=5 ttl=86 time=12.4
ms

--- star.c10r.facebook.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4000ms
rtt min/avg/max/mdev = 12.221/12.318/12.427/0.113 ms
```

We made 5 ICMP requests to facebook.com. The time to live for this is 86 hops. The round trip time varies from 12.22ms to 12.43ms. All 5 packets we transmitted was received. The minimum round trip is 12.221ms. The average round trip time is 12.318. The maximum round trip time is 12.427ms. Finally, time difference between the maximum and the minimum round trip time is 0.113ms.

5.

```
PING www.brasil.gov.br (161.148.175.40) 56(84) bytes of data.
From 200-193-232-14.bsace301.ipd.brasiltelecom.net.br (200.193.232.14) icmp_seq=2 Packet
filtered

--- www.brasil.gov.br ping statistics ---
5 packets transmitted, 0 received, +1 errors, 100% packet loss, time 3999ms
```

When we ping www.brasil.gov.br, we observed a different result. All icmp packets were lost. One of the packet indicates that the ping request was filtered. A filtered ping request may be a result of blocked ping request. The server return with a special respond to inform the sender that the request did not go through.

Question 8 (traceroute)

a. traceroute tracks the route packets taken from an IP network on their way to a given host.

b

1.

It took around 10+ hops to reach www.uwaterloo.ca from my devserver. After that it seems that the probe was lost. In this case its possible that the packet took around 27 hops to reach.

```
[simar@TuxCore Trace5 ]$ traceroute www.uwaterloo.ca
traceroute to www.uwaterloo.ca (129.97.149.158), 30 hops max, 60 byte packets
 1  107.170.14.253 (107.170.14.253)  3.849 ms  3.824 ms  3.793 ms
 2  192.241.164.241 (192.241.164.241)  0.222 ms 192.241.164.253 (192.241.164.253)  0.349 ms
192.241.164.241 (192.241.164.241)  3.791 ms
 3  66.110.96.21 (66.110.96.21)  0.449 ms  0.497 ms 66.110.96.25 (66.110.96.25)  0.594 ms
 4  63.243.128.121 (63.243.128.121)  1.629 ms  1.598 ms  1.675 ms
 5  if-5-5.tcore1.NYY-New-York.as6453.net (216.6.90.5)  1.313 ms  1.291 ms  1.226 ms
 6  216.6.90.26 (216.6.90.26)  0.888 ms  1.050 ms  1.062 ms
 7  10ge4-11.hcap9-tor.bb.allstream.net (199.212.169.202)  24.061 ms  23.914 ms  23.816 ms
 8  10ge4-11.hcap9-tor.bb.allstream.net (199.212.169.202)  23.526 ms  23.528 ms  23.729 ms
 9  216-191-167-38.dedicated.allstream.net (216.191.167.38)  26.051 ms  25.615 ms  25.862 ms
10  * * *
11  * * *
12  * * *
13  * * *
14  * * *
15  * * *
16  * * *
17  * * *
18  * * *
19  * * *
20  * * *
21  * * *
22  * * *
23  * * *
24  * * *
25  * * *
26  * * *
27  * * *
```

2.

It took 10 hops to get to www.youtube.com from my personal computer.

```
[simar@TuxCore ~ ]$  traceroute www.youtube.com
traceroute: Warning: www.youtube.com has multiple addresses; using 74.125.226.104
traceroute to youtube-ui.l.google.com (74.125.226.104), 64 hops max, 52 byte packets
 1  192.168.1.1 (192.168.1.1)  1.212 ms  0.881 ms  0.794 ms
 2  10.125.103.1 (10.125.103.1)  8.153 ms  7.698 ms  7.981 ms
 3  69.63.242.5 (69.63.242.5)  11.874 ms  10.763 ms  12.194 ms
 4  64.71.241.97 (64.71.241.97)  17.055 ms  17.594 ms  11.744 ms
 5  ae0_2110-bdr04-tor.teksavvy.com (69.196.136.36)  35.675 ms  41.993 ms
    ae1_2120-bdr04-tor.teksavvy.com (69.196.136.68)  13.888 ms
 6  72.14.212.134 (72.14.212.134)  15.047 ms  28.277 ms  27.710 ms
 7  216.239.47.114 (216.239.47.114)  113.171 ms  14.306 ms  13.017 ms
 8  209.85.250.207 (209.85.250.207)  13.840 ms  12.376 ms  13.973 ms
 9  yyz08s13-in-f8.1e100.net (74.125.226.104)  13.972 ms  11.817 ms  30.373 ms
```