Lab 3: Encapsulation and Network Utilities

Manthan Shah, 20658832
Heather D'Souza, 20671903

Faculty of Engineering
Department of Electrical and Computer Engineering

November 24, 2020
Course Instructor: Dr. A. Wasef

Question 1

Frame 2 (f2.txt)

The highest layer protocol is the user datagram protocol (UDP) on the transport layer.

00 00 00 00 00 00 00 00 00 00 00 00 08 00 45 00
00 ca 5c 0d 00 00 80 11 00 00 0a 20 1b a0 0a 20
7f ff eb 35 00 8a 00 b6 8e c0 11 0a 80 22 0a 20
1b a0 00 8a 00 a0 00 00 20 46 47 45 4a 46 43 46
45 46 46 45 42 45 4d 46 49 46 41 43 4e 44 49 44
45 46 46 45 42 45 4d 46 49 46 41 43 4e 44 49 44
45 44 41 44 46 44 44 41 41 00 20 46 48 45 50 46
43 45 4c 45 48 46 43 45 50 46 46 46 41 43 41 43
41 43 41 43 41 43 41 43 41 42 4e 00 ff 53 4d 42
25 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 11 00 00 06
00 00 00 00 00 00 00 00 00 e8 03 00 00 00 00 00
00 00 00 06 00 56 00 03 00 01 00 01 00 02 00 17
00 5c 4d 41 49 4c 53 4c 4f 54 5c 42 52 4f 57 53
45 00 09 04 02 00 00 00

Ethernet header (layer 2 - link):

00 00 00 00 00 00: Ethernet destination address 00 00 00 00 00 00 (unicast).
00 00 00 00 00 00: Ethernet source address 00 00 00 00 00 00 (unicast).
08 00: The payload type is IP (0x0800).

IP header (layer 3 - network):

45: This is an IP version 4 datagram.
45: The header length is 5x4 = 20 bytes - there is no options field in the given IP header.

00: This datagram has the lowest priority (routine precedence) - 0000 0000. It is used by routers to determine the priority level of a datagram, and if it can be dropped in some cases. So, datagrams with lowest priority level (like this one), get dropped by a router if needed.

Also, since bits 2, 3, and 4 (Big-Endian ordering) are all 0, the datagram's type of service (ToS) is also normal delay, normal throughput, and normal reliability (0000 0000).

00 ca: Total length of the IP datagram is 202 (0x00CA) bytes.

5c 0d: The identification of this datagram is 0x5C0D, for fragmentation purposes.

00 00: The more fragment and don't fragment flags are both not set (000<u>0</u> 0000). This means that there are no more fragments after this datagram, and this datagram can be fragmented. The offset is also 0, which means that is the only fragment of the datagram.

80: Time to live = 128 (0x80), which means that the datagram may exist for at most 128 more hops.

11: The protocol on top is UDP (0x11).

00 00: This is the header checksum of the datagram for error checking.

0a 20 1b a0: Source IP address is 10.32.27.160.

0a 20 7f ff: Destination IP address is 10.32.127.255.

UDP header (layer 4 - transport):

eb 35: The Source port is 60213, which is arbitrarily assigned by the operating system.

00 8a: The Destination port is 138.

00 b6: The length of UDP (including the header and data) is 182 bytes.

8e c0: This is the checksum of UDP.

Data (layer 5 - application):

Highlighted in orange.

Frame 12 (f12.txt)

The highest layer protocol is the transmission control protocol (TCP) on the transport layer.

00 1d 7e 46 ec 49 00 22 19 f4 30 97 08 00 45 00
00 3c cf 7e 40 00 40 06 1c 80 c0 a8 01 88 81 61
0b 2c 81 c1 1f 90 09 57 58 55 00 00 00 00 a0 02
39 08 4e ec 00 00 02 04 05 b4 04 02 08 0a 00 00
97 2b 00 00 00 00 01 03 03 06

Ethernet header (layer 2 - link):

00 1d 7e 46 ec 49: Ethernet destination address 00 1d 7e 46 ec 49 (unicast).
00 22 19 f4 30 97: Ethernet source address 00 22 19 f4 30 97 (unicast).
08 00: The payload type is IP (0x0800).

IP header (layer 3 - network):

45: This is an IP version 4 datagram.
45: The header length is 5x4 = 20 bytes - there is no options field in the given IP header.

00: This datagram has the lowest priority (routine precedence). It is used by routers to determine the priority level of a datagram, and if it can be dropped in some cases. So, datagrams with lowest priority level (like this one), get dropped by a router if needed.

Also, since bits 2, 3, and 4 (Big-Endian ordering) are all 0, the datagram's type of service (ToS) is also normal delay, normal throughput, and normal reliability.

00 3c: Total length of the IP datagram is 60 (0x003C) bytes.

cf 7e: The identification of this datagram is 0xCF7E, for fragmentation purposes.

40 00: The more fragment flag is unset, and the don't fragment flag is set (0100 0000). This means that the datagram cannot be fragmented, and there are no more fragments after this fragment. The fragment offset is also 0, which means this is the only fragment of a datagram.

40: Time to live = 64 (0x40), which means that the datagram may exist for at most 64 more hops.

06: The protocol on top is TCP (0x06).

1c 80: This is the header checksum of the datagram for error checking.

c0 a8 01 88: Source IP address is 192.168.1.136.

81 61 0b 2c: Destination IP address is 129.97.11.44.

TCP header (layer 4 - transport):

81 c1: The Source port is 33217, which is arbitrarily assigned by the operating system.

1f 90: The Destination port is 8080, which is a well-known alternative port for HTTP (hypertext transfer protocol).

09 57 58 55: The Seq. no. is 156719189.

00 00 00 00: The Ack. no. is 0.

a0: Data offset is 40 (10 x 4) bytes. This is the length of the TCP header.

02: The SYN flag is set (0000 0010). This is the synchronization flag, and is used between hosts during establishment of an initial connection (among the hosts). This is usually only set by the first packet between sender and receiver, and sets the precedence for which sequence number to accept during the initial connection process.

39 08: The receiver window size is 14600 (0x3908) bytes.

4e ec: Checksum for the whole TCP segment.

00 00: Urgent pointer (not used in this segment).

Data (layer 5 - application):

Highlighted in orange.

Question 2

ARP (address resolution protocol) is used by hosts on an IP subnetwork to figure out the MAC (physical) address of other hosts based on their IP addresses. It involves generating a table of entries containing mappings between IP address and MAC address of nodes on a network. This is useful for sending packets between nodes on a network, as a node can figure out the MAC address of another node by inspecting the ARP table, or running a ARP request to the desired host.

The *arp* command is a linux utility that can display and modify entries in the ARP table in your machine's ARP cache. The cache may contain multiple tables with multiple host IP to MAC mappings.

The arp -a command can be used to display all ARP tables currently in the cache for all interfaces (IP subnets). To display the ARP entry for a specific host, the IP address can be specified, and for a specific interface, the interface address (IP subnet address) can also be specified.

The *arp -d* command can be used to delete entries within ARP tables in the cache. An IP address for the desired entry to be deleted must be specified.

The *arp -s* command can be used to manually add entries to a ARP table, with a specific IP address, and corresponding physical (Ethernet) address.

Running the *arp -a* command on my local machine, I get the following:

```
manthanshah@Manthans-MacBook-Pro-2 ~ % arp -a
? (192.168.0.1) at 0:0:0:0:0:2 on en0 ifscope [ethernet]
? (192.168.0.14) at 50:bc:96:7:ff:e0 on en0 ifscope [ethernet]
? (192.168.0.24) at 0:56:cd:5e:43:83 on en0 ifscope [ethernet]
? (192.168.0.32) at da:fb:d6:6b:cb:7b on en0 ifscope [ethernet]
? (192.168.0.39) at 4a:7d:5b:db:27:27 on en0 ifscope [ethernet]
? (192.168.0.41) at e2:86:ce:1c:b8:6d on en0 ifscope [ethernet]
? (224.0.0.251) at 1:0:5e:0:0:fb on en0 ifscope permanent [ethernet]
manthanshah@Manthans-MacBook-Pro-2 ~ %
```

Figure 1: output when arp -a is run on personal laptop

Figure 1 shows all the IP address to MAC address mappings that my laptop has for other hosts within my subnetwork - there are a total of 7 mappings in this table. For example, inspecting the first entry, we can see that there is a mapping for IP address 192.168.0.1. As a side note, the subnet mask for my local network is 24 bits, so based on that, this IP address corresponds to the router in my home (since it is common to assign the first host address to the router)! Now, this IP address is mapped to a MAC address of 0:0:0:0:0:2, which is a 6 byte MAC address corresponding to the router. Next, it shows that this host is connected to an ethernet interface, en0 (port 0).

Question 3

The *ifconfig* command is used to view and modify any network interface configurations (using TCP/IP) that may be on your local machine or server. It can be used to assign an address to a network interface as well. Generally, this command is run during machine start-up to configure various pieces of information about each network interface such as the network IP address, broadcast address, subnet mask, and much more.

Running *ifconfig -a* on the ecetesla server, the following is generated:

```
mskshah@ecetesla0:~$ ifconfig -a
eno1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 129.97.56.23  netmask 255.255.255.0  broadcast 129.97.56.255
        inet6 fe80::43fb:5876:18f0:9943  prefixlen 64  scopeid 0x20<link>
        ether 00:25:90:5e:d0:9c  txqueuelen 1000  (Ethernet)
        RX packets 1714848002  bytes 2070769059613 (2.0 TB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 961807277  bytes 754499411349 (754.4 GB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eno2: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 00:25:90:5e:d0:9d  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 25659216  bytes 236867304120 (236.8 GB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 25659216  bytes 236867304120 (236.8 GB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figure 2: output when *ifconfig -a* is run on ecetesla server

Figure 2 shows that there are 3 network interfaces on the ecetesla server. Inspecting the first entry, eno1 indicates it is an ethernet interface and it is the first ethernet interface entry in the list. We can also see there are various flags specified. Up indicates that the interface is active with an address, broadcast means that this interface supports broadcast packets, running means that the interface is ready to transmit and receive packets (resources have been allocated for this interface), and multicast means that the interface supports multicast type of packets (sending multiple packets at once). There are also various other parameters, described in table 1:

Table 1: parameter definitions for first entry from Figure 1

Parameter	Description
MTU	Maximum transmission unit, which specifies the maximum size of packets that can be sent/received through interface.
inet	IPV4 address assigned to this interface.
netmask	Subnet mask of the IPV4 address.
broadcast	IP address to use to send broadcast packets

	among this interface.
inet6	IPV6 address assigned to this interface.
prefixLen	Length of the network prefix in bits for the IPV6 address.
scopeid	Scope ID (zone ID) used for IPV6 address.
ether	Hardware (MAC) address of the interface.
txqueuelen	Transmission queue length - number of frames per kernel transmission queue.
RX packets	Number of received packets.
RX bytes	Total data received in bytes.
RX errors	The number of erroneous packets that were received (checked through CRC bits).
RX dropped	Number of received packets that were dropped.
RX overruns	Number of packets received when the FIFO buffer queue was full.
RX frame	Number of packets that had misalignments - length not divisible by 8.
TX packets, bytes, errors, dropped, overruns, frame	Same analogy and descriptions as RX packets, bytes, errors, dropped, overruns, and frame but in the packet transmission context.

Question 4

Please note that the utilities in questions 4 to 7 were run on eceubuntu1.

The *netstat* command provides information about the Linux networking subsystem. The default information provided is a list of open sockets for all configured address families. However, if an option is specified, it can provide other information such as routing tables (-r), network interfaces (-i), masqueraded connections (-M), summary statistics for each protocol (-s), and multicast group membership information for IPv4 and IPv6 (-g).

The command *netstat -in* provides a table of all network interfaces (-i) and shows numerical addresses (-n). The output table consists of statistics for each of the currently configured interfaces (Figure 3).

```
eceubuntu1:~> netstat -in
Kernel Interface table
Iface      MTU    RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
enp0s31f  1500        0      0      0 0             0      0      0      0 BMU
enp1s0    1500 1918399506      0      0 0     694914445      0      0      0 BMRU
lo       65536 66429728      0      0 0      66429728      0      0      0 LRU
```

Figure 3: statistics for auto-configured interfaces

For each interface under the *Iface* column, the maximum transmission unit (MTU), the number of error-free packets received and transmitted (RX-OK, TX-OK), the number of damaged packets received and transmitted (RX-ERR, TX-ERR), the number of packets dropped (RX-DRP, TX-DRP), and the number of packets lost due to overrun (RX-OVR, TX-OVR) are included in this table. It is observed that interface *lo* has the largest MTU and that all packets that have been transmitted or received were error-free. Interface *enp1s0* has also only sent and received error-free packets. It has received many more packets than it has sent out. Interface *enp0s31f* has not sent or received any packets.

The command *netstat -s* provides summary statistics for each protocol (Figure 4).

a)

```
eceubuntu1:~> netstat -s
Ip:
    Forwarding: 2
    710705936 total packets received
    22 with invalid addresses
    0 forwarded
    0 incoming packets discarded
    710322104 incoming packets delivered
    761871849 requests sent out
    825 dropped because of missing route
    4 fragments received ok
    8 fragments created
Icmp:
    199345 ICMP messages received
    1112 input ICMP message failed
    ICMP input histogram:
        destination unreachable: 6536
        echo requests: 190986
        echo replies: 1823
    199356 ICMP messages sent
    0 ICMP messages failed
    ICMP output histogram:
        destination unreachable: 6538
        echo requests: 1832
        echo replies: 190986
IcmpMsg:
        InType0: 1823
        InType3: 6536
        InType8: 190986
        OutType0: 190986
        OutType3: 6538
        OutType8: 1832
Tcp:
    1522841 active connection openings
    161443 passive connection openings
    20624 failed connection attempts
    26267 connection resets received
    56 connections established
    699869309 segments received
    991532120 segments sent out
    263859 segments retransmitted
    20 bad segments received
    21514 resets sent
Udp:
    10234386 packets received
    4238 packets to unknown port received
    0 packet receive errors
    14161653 packets sent
    0 receive buffer errors
    1 send buffer errors
UdpLite:
TcpExt:
    50 SYN cookies sent
    50 SYN cookies received
    4 invalid SYN cookies received
    475 resets received for embryonic SYN_RECV sockets
    280 packets pruned from receive queue because of socket buffer overrun
    13 ICMP packets dropped because they were out-of-window
    1430130 TCP sockets finished time wait in fast timer
    425 packetes rejected in established connections because of timestamp
    1241250 delayed acks sent
    1481 delayed acks further delayed because of locked socket
```

b)

Figure 4: the output from *netstat -s*

For the sake of brevity, some statistics for *TcpExt* were truncated from Figure 4.a). An interesting point to note is that the protocols running on the server include internet protocol (IP), internet control message protocol (ICMP), transmission control protocol (TCP), and UDP (user datagram protocol). The *IP* statistics include the total number of packets received (710,705,936), the total number of packets delivered (710,322,104), and the number of packets that were dropped because of a missing route (825). For *TCP*, information such as the number of connection openings, connections established, and failed connection attempts are provided. The transport layer creates segments from a message during encapsulation and reassembles segments during decapsulation. The number of segments received (699,869,309) and sent out (991,532,120) can also be observed.

The command *netstat -r* displays the routing table (Figure 5). This table is used by the router to forward packets to the next hop.

```
eceubuntu1:~> netstat -r
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
default         exsw02-circuitn 0.0.0.0         UG        0 0          0 enp1s0
129.97.56.0     0.0.0.0         255.255.255.0   U         0 0          0 enp1s0
link-local      0.0.0.0         255.255.0.0     U         0 0          0 enp1s0
```

Figure 5: the output from *netstat -r*

There is an entry for each destination that this router knows about. For each destination, a *Gateway* is specified. This is the next hop. The *Genmask* is used to match a destination IP address to the network ID. *Iface* is the output interface of the router to which the packet will be forwarded. This interface will take the packet to the next hop. According to the routing table, a packet addressed to the link-local network or the 129.97.56.0 network will be sent to the gateway router (denoted by 0.0.0.0 in the *Gateway* column). According to the first entry in the table, a packet addressed to any other destination, will be sent to the router named exsw02-circuitn.

Question 5

The nslookup command queries the Domain Name System (DNS) and returns the IP address mapped to the given domain or the domain mapped to the given IP address.

This is the output returned for *nslookup ecelinux.uwaterloo.ca* (Figure 6). It provides the IP address of the DNS server (127.0.0.53) and the resolved IP associated with the ecelinux.uwaterloo.ca hostname (129.97.56.15). Note that the answer to the query did not come from the authoritative name server for that domain name. Instead, the answer was retrieved from the cache. This is denoted by the "non-authoritative answer" statement.

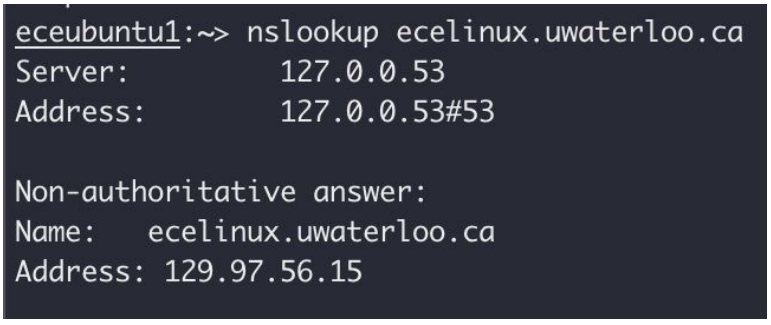

Figure 6: the output of *nslookup ecelinux.uwaterloo.ca*

This is the output for *nslookup www.mit.edu* (Figure 7). It explains that www.mit.edu is an alias for www.mit.edu.edgekey.net which is an alias for e9566.dscb.akamaiedge.net. This domain name resolves to 23.15.214.31. Two IPv6 addresses are also returned: 2600:140a:6000:293::255e and 2600:140a:6000:29f::255e. As this lookup returns three addresses in total, it seems that DNS round-robin is being used. Round-robin DNS can be used by a website for load balancing.

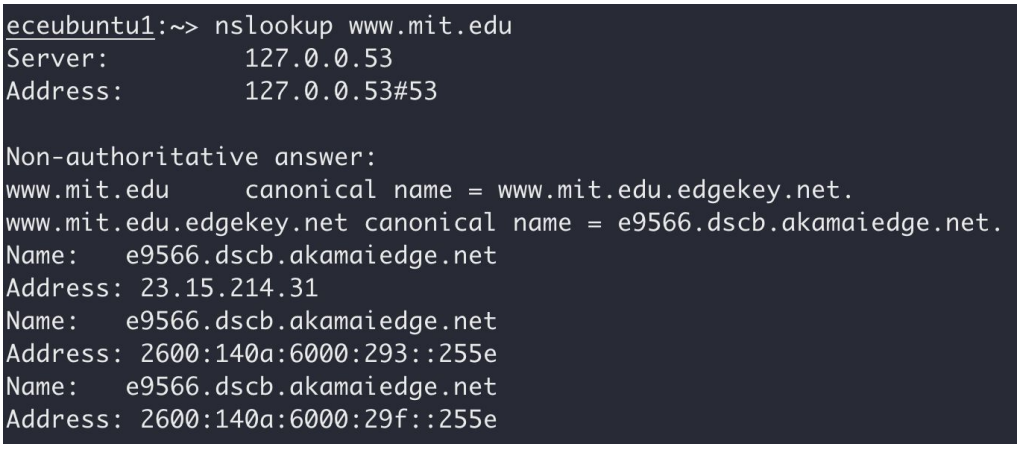

Figure 7: the output of *nslookup www.mit.edu*

This is the output for *nslookup www.gmail.com* (Figure 8). This domain name is an alias for mail.google.com which is an alias for googlemail.l.google.com. This domain name resolves to either 172.217.0.229 or 2607:f8b0:400b:808::2005.

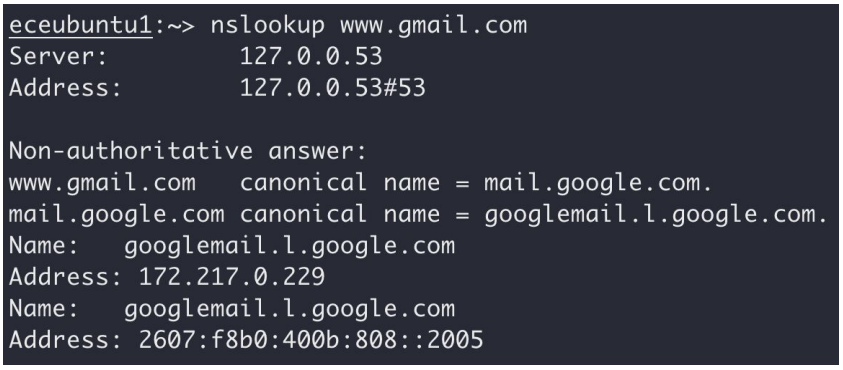

Figure 8: the output of *nslookup www.gmail.com*

This is the output for *nslookup www.facebook.com* (Figure 9). The domain name is an alias for star-mini.c10r.facebook.com. This domain name resolves to either 31.13.80.36 or 2a03:2880:f10e:83:face:b00c:0:25de.

Figure 9: the output of *nslookup www.facebook.com*

Question 6

The ping command is used to test if a host is reachable. The command uses the ICMP protocol to send an ECHO_REQUEST datagram. If it finds the target host, the host sends an ICMP_ECHO_RESPONSE.

Before pinging the following hosts, they were checked by using a web browser to connect to them. All of the hosts were up.

This is the output of *ping -c10 www.ualberta.ca* (Figure 10). The host was pinged 10 times. According to the report at the end of the output, the average round-trip time from eceubuntu1 to www.ualberta.ca was 4.164 ms. No packets were lost.

```
PING prod.cds.ualberta.cloud (13.33.165.53) 56(84) bytes of data.
64 bytes from server-13-33-165-53.yto50.r.cloudfront.net (13.33.165.53): icmp_seq=1 ttl=240 time=4.07 ms
64 bytes from server-13-33-165-53.yto50.r.cloudfront.net (13.33.165.53): icmp_seq=2 ttl=240 time=4.13 ms
64 bytes from server-13-33-165-53.yto50.r.cloudfront.net (13.33.165.53): icmp_seq=3 ttl=240 time=4.14 ms
64 bytes from server-13-33-165-53.yto50.r.cloudfront.net (13.33.165.53): icmp_seq=4 ttl=240 time=4.23 ms
64 bytes from server-13-33-165-53.yto50.r.cloudfront.net (13.33.165.53): icmp_seq=5 ttl=240 time=4.13 ms
64 bytes from server-13-33-165-53.yto50.r.cloudfront.net (13.33.165.53): icmp_seq=6 ttl=240 time=4.11 ms
64 bytes from server-13-33-165-53.yto50.r.cloudfront.net (13.33.165.53): icmp_seq=7 ttl=240 time=4.17 ms
64 bytes from server-13-33-165-53.yto50.r.cloudfront.net (13.33.165.53): icmp_seq=8 ttl=240 time=4.15 ms
64 bytes from server-13-33-165-53.yto50.r.cloudfront.net (13.33.165.53): icmp_seq=9 ttl=240 time=4.17 ms
64 bytes from server-13-33-165-53.yto50.r.cloudfront.net (13.33.165.53): icmp_seq=10 ttl=240 time=4.30 ms

--- prod.cds.ualberta.cloud ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9008ms
rtt min/avg/max/mdev = 4.074/4.164/4.304/0.102 ms
```

Figure 10: the output of *ping -c10 www.ualberta.ca*

This is the output of *ping -c10 www.lemonde.fr* (Figure 11). The host was pinged 10 times. The average round-trip time from eceubuntu1 to www.lemonde.fr was 5.344 ms. No packets were lost.

```
eceubuntu1:~> ping -c10 www.lemonde.fr
PING s2.shared.global.fastly.net (151.101.126.217) 56(84) bytes of data.
64 bytes from 151.101.126.217 (151.101.126.217): icmp_seq=1 ttl=52 time=5.31 ms
64 bytes from 151.101.126.217 (151.101.126.217): icmp_seq=2 ttl=52 time=5.56 ms
64 bytes from 151.101.126.217 (151.101.126.217): icmp_seq=3 ttl=52 time=5.57 ms
64 bytes from 151.101.126.217 (151.101.126.217): icmp_seq=4 ttl=52 time=5.44 ms
64 bytes from 151.101.126.217 (151.101.126.217): icmp_seq=5 ttl=52 time=5.22 ms
64 bytes from 151.101.126.217 (151.101.126.217): icmp_seq=6 ttl=52 time=5.32 ms
64 bytes from 151.101.126.217 (151.101.126.217): icmp_seq=7 ttl=52 time=5.23 ms
64 bytes from 151.101.126.217 (151.101.126.217): icmp_seq=8 ttl=52 time=5.32 ms
64 bytes from 151.101.126.217 (151.101.126.217): icmp_seq=9 ttl=52 time=5.20 ms
64 bytes from 151.101.126.217 (151.101.126.217): icmp_seq=10 ttl=52 time=5.23 ms

--- s2.shared.global.fastly.net ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9012ms
rtt min/avg/max/mdev = 5.208/5.344/5.574/0.138 ms
```

Figure 11: the output of *ping -c10 www.lemonde.fr*

This is the output of *ping -c10 www.ucla.edu* (Figure 12). The host was pinged 10 times. The average round-trip time from eceubuntu1 to www.ucla.edu was 79.021 ms. No packets were lost.

```
eceubuntu1:~> ping -c10 www.ucla.edu
PING gateway.lb.it.ucla.edu (164.67.228.152) 56(84) bytes of data.
64 bytes from gateway.lb.it.ucla.edu (164.67.228.152): icmp_seq=1 ttl=42 time=78.8 ms
64 bytes from gateway.lb.it.ucla.edu (164.67.228.152): icmp_seq=2 ttl=42 time=79.0 ms
64 bytes from gateway.lb.it.ucla.edu (164.67.228.152): icmp_seq=3 ttl=42 time=78.8 ms
64 bytes from gateway.lb.it.ucla.edu (164.67.228.152): icmp_seq=4 ttl=42 time=79.0 ms
64 bytes from gateway.lb.it.ucla.edu (164.67.228.152): icmp_seq=5 ttl=42 time=79.0 ms
64 bytes from gateway.lb.it.ucla.edu (164.67.228.152): icmp_seq=6 ttl=42 time=78.9 ms
64 bytes from gateway.lb.it.ucla.edu (164.67.228.152): icmp_seq=7 ttl=42 time=79.0 ms
64 bytes from gateway.lb.it.ucla.edu (164.67.228.152): icmp_seq=8 ttl=42 time=79.3 ms
64 bytes from gateway.lb.it.ucla.edu (164.67.228.152): icmp_seq=9 ttl=42 time=78.9 ms
64 bytes from gateway.lb.it.ucla.edu (164.67.228.152): icmp_seq=10 ttl=42 time=79.1 ms

--- gateway.lb.it.ucla.edu ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9011ms
rtt min/avg/max/mdev = 78.835/79.021/79.347/0.189 ms
```

Figure 12: the output of *ping -c10 www.ucla.edu*

The quickest round-trip time was for host www.ualberta.ca while the longest round-trip time was for host www.ucla.edu.

Question 7

The traceroute command provides the path a packet takes to get from the current host to another network host. It does this by sending out probe packets with a time-to-live (TTL) that initially starts at 1. When a probe packet reaches a router, the router decrements the TTL. When a TIME_EXCEEDED response is received from a router along the path to the host, the round-trip time (RTT) to that router can be recorded. The traceroute first sends probe packets with a TTL of 1 to reach the first router. Then the traceroute sends probe packets with a TTL of 2 to reach the second router. This process continues until the probe packets reach the destination router.

This is the output for */usr/sbin/traceroute www.uwaterloo.ca* (Figure 13). The traceroute reaches the host at the 10th hop. It is unclear as to why there is an entry for the 11th hop, but based on the output, there are 10 hops between eceubuntu1 and www.uwaterloo.ca.

```
eceubuntu1:~> /usr/sbin/traceroute www.uwaterloo.ca
traceroute to www.uwaterloo.ca (129.97.208.23), 30 hops max, 60 byte packets
 1  exsw02-circuitnet.uwaterloo.ca (129.97.56.1)  3.386 ms  3.360 ms  3.351 ms
 2  v490-eng-rt-e2.ns.uwaterloo.ca (172.16.32.193)  3.867 ms  3.858 ms  3.851 ms
 3  te4-3-dist-rt-phy.ns.uwaterloo.ca (172.18.7.21)  0.601 ms  0.918 ms  0.998 ms
 4  xe1-0-0-u11-dist-sa-mc-trust.ns.uwaterloo.ca (172.31.0.149)  0.413 ms  0.472 ms  0.463 ms
 5  te2-12-dist-rt-mc-global.ns.uwaterloo.ca (172.31.0.161)  1.638 ms  1.813 ms  1.918 ms
 6  te2-16-cn-rt-mc.ns.uwaterloo.ca (172.16.31.117)  0.741 ms  0.808 ms  0.737 ms
 7  e2-1-cr-rt-bb2.ns.uwaterloo.ca (172.16.16.3)  0.988 ms  1.166 ms  1.119 ms
 8  xe-1-0-1-cr-sa-bb2.nsx.uwaterloo.ca (129.97.0.59)  1.081 ms  0.991 ms  1.001 ms
 9  e1-25-20-cr-rt-mc-area2.ns.uwaterloo.ca (172.16.16.27)  1.561 ms  1.626 ms  1.641 ms
10  wms.uwaterloo.ca (129.97.208.23)  1.347 ms  1.260 ms  1.271 ms
11  wms.uwaterloo.ca (129.97.208.23)  1.367 ms !N  1.314 ms !N  1.425 ms !N
```

Figure 13: the output of */usr/sbin/traceroute www.uwaterloo.ca*

This is the output for */usr/sbin/traceroute www.youtube.com* (Figure 14). The traceroute reaches the host at the 15th hop. It is observed that at the 13th hop, two of the three probe packets were dropped. This is denoted by the asterisks.

```
traceroute to www.youtube.com (172.217.0.238), 30 hops max, 60 byte packets
 1  exsw02-circuitnet.uwaterloo.ca (129.97.56.1)  3.612 ms  3.666 ms  3.654 ms
 2  v490-eng-rt-e2.ns.uwaterloo.ca (172.16.32.193)  4.624 ms  4.615 ms  4.680 ms
 3  te4-3-dist-rt-mc.ns.uwaterloo.ca (172.18.7.17)  0.892 ms  1.011 ms  1.131 ms
 4  xe1-0-0-u10-dist-sa-mc-trust.ns.uwaterloo.ca (172.31.0.145)  0.454 ms  0.445 ms  0.437 ms
 5  te2-12-dist-rt-mc-global.ns.uwaterloo.ca (172.31.0.161)  1.422 ms  1.306 ms  1.505 ms
 6  te2-16-cn-rt-mc.ns.uwaterloo.ca (172.16.31.117)  0.794 ms  0.757 ms  0.837 ms
 7  te0-0-2-0-ext-rt-mc.ns.uwaterloo.ca (172.16.32.149)  1.458 ms  1.483 ms  1.542 ms
 8  unallocated-static.rogers.com (72.142.108.181)  1.110 ms  1.120 ms  1.087 ms
 9  24.156.146.189 (24.156.146.189)  5.001 ms  4.989 ms  4.999 ms
10  9044-cgw01.wlfdle.rmgt.net.rogers.com (209.148.230.45)  5.242 ms  5.597 ms  5.438 ms
11  209.148.233.38 (209.148.233.38)  5.477 ms  5.489 ms  5.478 ms
12  72.14.209.126 (72.14.209.126)  5.801 ms  5.793 ms  5.794 ms
13  108.170.250.225 (108.170.250.225)  4.839 ms * *
14  108.170.226.219 (108.170.226.219)  5.417 ms  5.490 ms 216.239.42.158 (216.239.42.158)  5.863 ms
15  108.170.226.217 (108.170.226.217)  5.464 ms dfw06s38-in-f14.1e100.net (172.217.0.238)  5.364 ms  5.420 ms
```

Figure 14: the output of */usr/sbin/traceroute www.youtube.com*

This is the output for */usr/sbin/traceroute www.nytimes.com* (Figure 15).

```
eceubuntu1:~> /usr/sbin/traceroute www.nytimes.com
traceroute to www.nytimes.com (151.101.125.164), 30 hops max, 60 byte packets
 1  exsw02-circuitnet.uwaterloo.ca (129.97.56.1)  3.272 ms  3.341 ms  3.332 ms
 2  v490-eng-rt-e2.ns.uwaterloo.ca (172.16.32.193)  4.012 ms  3.990 ms  4.053 ms
 3  te4-3-dist-rt-mc.ns.uwaterloo.ca (172.18.7.17)  0.923 ms  1.003 ms  1.066 ms
 4  xe1-0-0-u10-dist-sa-mc-trust.ns.uwaterloo.ca (172.31.0.145)  0.444 ms  0.436 ms  0.428 ms
 5  te2-12-dist-rt-mc-global.ns.uwaterloo.ca (172.31.0.161)  1.317 ms  1.414 ms  1.510 ms
 6  * * *
 7  te0-0-2-0-ext-rt-mc.ns.uwaterloo.ca (172.16.32.149)  1.362 ms  1.373 ms  1.386 ms
 8  unallocated-static.rogers.com (72.142.108.181)  1.102 ms  1.125 ms  1.086 ms
 9  24.156.146.189 (24.156.146.189)  7.563 ms  7.198 ms  6.941 ms
10  9044-cgw01.wlfdle.rmgt.net.rogers.com (209.148.230.45)  5.281 ms  5.466 ms  5.458 ms
11  209.148.235.214 (209.148.235.214)  5.285 ms  5.233 ms  5.281 ms
12  * * *
13  * * *
14  * * *
15  * * *
16  * * *
17  * * *
18  * * *
19  * * *
20  * * *
21  * * *
22  * * *
23  * * *
24  * * *
25  * * *
26  * * *
27  * * *
28  * * *
29  * * *
30  * * *
```

Figure 15: the output of */usr/sbin/traceroute www.nytimes.com*

The probe packets get as far as the 11th hop, but cannot pass the 12th hop. However, the command *ping -c10 www.nytimes.com* is successful as observed in Figure 16. The traceroute command is based on UDP packets by default. It is possible that the 12th hop does not respond because it blocks UDP messages.

```
eceubuntu1:~> ping -c10 www.nytimes.com
PING nytimes.map.fastly.net (151.101.125.164) 56(84) bytes of data.
64 bytes from 151.101.125.164 (151.101.125.164): icmp_seq=1 ttl=52 time=5.28 ms
64 bytes from 151.101.125.164 (151.101.125.164): icmp_seq=2 ttl=52 time=5.25 ms
64 bytes from 151.101.125.164 (151.101.125.164): icmp_seq=3 ttl=52 time=5.23 ms
64 bytes from 151.101.125.164 (151.101.125.164): icmp_seq=4 ttl=52 time=5.25 ms
64 bytes from 151.101.125.164 (151.101.125.164): icmp_seq=5 ttl=52 time=5.20 ms
64 bytes from 151.101.125.164 (151.101.125.164): icmp_seq=6 ttl=52 time=5.26 ms
64 bytes from 151.101.125.164 (151.101.125.164): icmp_seq=7 ttl=52 time=5.31 ms
64 bytes from 151.101.125.164 (151.101.125.164): icmp_seq=8 ttl=52 time=5.30 ms
64 bytes from 151.101.125.164 (151.101.125.164): icmp_seq=9 ttl=52 time=5.22 ms
64 bytes from 151.101.125.164 (151.101.125.164): icmp_seq=10 ttl=52 time=5.25 ms

--- nytimes.map.fastly.net ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9011ms
rtt min/avg/max/mdev = 5.209/5.259/5.314/0.097 ms
```

Figure 16: the output of *ping -c10 www.nytimes.com*