

1. The network layer is the most important layer as it is the one that provides a number of important filters, configurations, and the firewall to protect the machines connected to the network. On the network layer, the host addressing and logical addressing for the IP occur. Also, traffic error handling, packet encapsulation and decapsulation, and routing occur on the network layer. All of this makes the network layer the most important layer of the TCP/IP stack.
2. Time-sensitive, or real-time applications are the ones that suffer the most from transmission delays. The transmission delay is the ratio of the size of packet and the link rate in transferring the packet, and in that case, large packets being sent over slow rate of service leads to an increase of transmission delays. Some of the real-time applications effected by transmission delays are VoIP and Multiplayer Video Gaming.
3. A socket is an endpoint of two-way communication link between two programs running on a network. A socket would allow the client to connect to a specific listening port where the packets gets delivered to a specific program and this program, called client, gets linked to a specific listening server with a specific port again. This is the ultimate solution to running multiple network-connected applications on the same network.
4. UDP, which is an unreliable transport protocol, has a lower overhead than a reliable transport protocol like TCP. Unreliable transport protocols are not concerned about setting up the connection or retrieving lost data or retransmissions, and therefore the transfer of data is faster with lower delays. UDP is generally used in applications that require multicasting as they can benefit well from them.
5. The total distance 10 caravan of cars need to cover is equal to 150 km, and there are 75 km between tollbooths. With that in mind, the propagation decal can be calculated as following:
 
$$75 \text{ km} / 100 \text{ km/hr} = 0.75 \text{ hr}$$

$$0.75 \text{ hr} * 60 \text{ min/hr} = 45 \text{ min}$$

The transmission delay can be calculated as following:

$$10 * 12 \text{ seconds} = 120 \text{ seconds}$$

$$120 \text{ seconds/min} / 60 \text{ second/min} = 2 \text{ min}$$

From that, we can reach the end-to-end delay by adding the propagation and transmission delays,  $45 + 2 = 47$ , multiplying that by 2, which is equal to 94, and adding 2 minutes of service delay at the third tollbooth, reaching to a total of 96 minutes of delay.

6. The throughput of a file transfer is equal to the minimum rate of all available links. The minimum of R1, R2, and R3 is R1, therefore the throughput of a file transfer is equal to 500 kbps.

7.

Frame 1: 64 bytes on wire (512 bits), 64 bytes captured (512 bits) on interface 0  
Ethernet II, Src: 00:00:00\_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00\_00:00:00 (00:00:00:00:00:00)  
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.53  
User Datagram Protocol, Src Port: 48031, Dst Port: 53  
Domain Name System (query)

The image above shows the order of the top-level protocol. As you can see from the image, the top-level protocol is the DNS, followed by the UDP, followed by IPV4, followed by Ethernet II, followed by Frame I.

8. I could not capture any multicast message sent on my local network. For two protocols relying on multicasting, I found Resource Reservation Protocol, short is RSVPm and Real-time Transport Protocol, short is RTP. RSVP is designed to serve different resources across a network of Quality of Service, short is QoS, by using the model of internet integrated services. RTP is a data transfer protocol designed to exchange real-time sensitive audio-visual data on IP-based networks.

9. Since luther.edu is currently https, I am going to look at <http://hackisu.org/>

```
▼ Hypertext Transfer Protocol
  ▼ GET / HTTP/1.1\r\n
    ► [Expert Info (Chat/Sequence): GET / HTTP/1.1\r\n]
      Request Method: GET
      Request URI: /
      Request Version: HTTP/1.1
      Host: hackisu.org\r\n
      Upgrade-Insecure-Requests: 1\r\n
      Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\n
      User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/11.1.2
      Accept-Language: en-us\r\n
      Accept-Encoding: gzip, deflate\r\n
      Connection: keep-alive\r\n
      \r\n
      [Full request URI: http://hackisu.org/]
      [HTTP request 1/7]
      [Response in frame: 4668]
      [Next request in frame: 4884]
```

This is the first HTTP GET request, out of seven other requests, to download the web page of hackisu.org. This GET request specifies the accepted language, encoding, and the browser we are using besides the OS of this device making the request.

10.

No.	Time	Source	Destination	Protocol	Length	Info
685...	1683.751018	10.28.0.1	10.28.17.9	ICMP	70	Destination unreachable (Port unreachable)
687...	1684.917264	10.28.17.9	208.67.222.222	ICMP	70	Destination unreachable (Port unreachable)
687...	1684.917264	10.28.17.9	208.67.222.222	ICMP	70	Destination unreachable (Port unreachable)
687...	1684.918579	10.28.17.9	208.67.222.222	ICMP	70	Destination unreachable (Port unreachable)
687...	1684.918580	10.28.17.9	208.67.222.222	ICMP	70	Destination unreachable (Port unreachable)
687...	1684.920927	10.28.17.9	208.67.222.222	ICMP	70	Destination unreachable (Port unreachable)
687...	1684.938090	10.28.17.9	208.67.222.222	ICMP	70	Destination unreachable (Port unreachable)
687...	1685.029975	10.28.17.9	208.67.222.222	ICMP	70	Destination unreachable (Port unreachable)
689...	1685.808959	10.28.0.1	10.28.17.9	ICMP	70	Destination unreachable (Port unreachable)
690...	1686.310346	10.28.0.1	10.28.17.9	ICMP	70	Destination unreachable (Port unreachable)
691...	1687.814017	10.28.0.1	10.28.17.9	ICMP	70	Destination unreachable (Port unreachable)
691...	1688.315024	10.28.0.1	10.28.17.9	ICMP	70	Destination unreachable (Port unreachable)
692...	1690.821286	10.28.0.1	10.28.17.9	ICMP	70	Destination unreachable (Port unreachable)
692...	1691.320017	10.28.0.1	10.28.17.9	ICMP	70	Destination unreachable (Port unreachable)
▶ Frame 68749: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface 0						0000 00 15 60 f6 88 00 a4 5e 60 ec 29 9d 08 00 45 00 ...^...)
▶ Ethernet II, Src: Apple_ec:29:9d (a4:5e:60:ec:29:9d), Dst: HewlettP_f6:88:00 (00:15:60:f6:88:00)						0010 00 38 4d b1 00 00 40 01 62 cd 0a 1c 11 09 d0 43 ...M...@...b...
▶ Internet Protocol Version 4, Src: 10.28.17.9, Dst: 208.67.222.222						0020 de de 03 03 0b 36 00 00 00 00 45 00 01 0c 65 8f ...6...E...
▼ Internet Control Message Protocol						0030 40 00 38 11 12 0b d0 43 de de 0a 1c 11 09 01 bb @.8...C.....
Type: 3 (Destination unreachable)						0040 ef 13 00 f8 00 00 .....
Code: 3 (Port unreachable)						
Checksum: 0x0b36 [correct]						
[Checksum Status: Good]						
Unused: 00000000						
▶ Internet Protocol Version 4, Src: 208.67.222.222, Dst: 10.28.17.9						
▼ User Datagram Protocol, Src Port: 443, Dst Port: 61203						
Source Port: 443						
Destination Port: 61203						
Length: 248						
[Checksum: [missing]]						
[Checksum Status: Not present]						
[Stream index: 1330]						

According to this [article](#), there are 19 types of ICMP Message Types, which are: Echo Reply, Destination Unreachable, Source Quench, Redirect, Echo Request, Router Advertisement, Router Solicitation, Time Exceeded, Parameter Problem, Timestamp Request, Timestamp Reply, Information Request (obsolete), Information Reply (obsolete), Address Mask Request, Address Mask Reply.