

Exam # 1

Release Date: October 7, 2020 - Due Date: October 13, 2020 at 5:00 pm

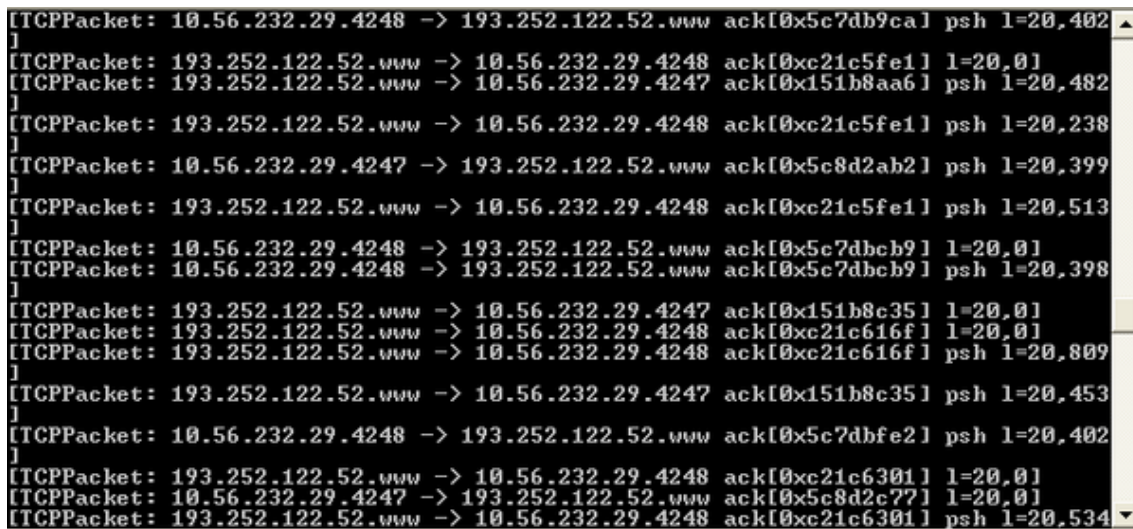
The required answer sheet must be submitted on OWL along with the commented and raw python files. Do not share your answers and codes with others. Do not put your answers and codes in any public domain.

Problem # 1 (10 Marks):

Please, include screenshots with your answers for this problem or ZERO mark will be assigned.

This problem will introduce you to packet sniffers which will allow us to take a deeper look into how the various protocols we have encountered thus far operate. You will also gain experience in utilizing packet sniffing software such as WinPcap as we as packet analyzing programs such as Wireshark.

Before getting started let us go over a brief overview of what exactly is a packet sniffer. A packet sniffer captures ("sniffs") passively all messages being sent/received from/by your computer through your network card a sample of the raw data collected by WinPcap will look similar in figure 1.



```
[TCPPacket: 10.56.232.29.4248 -> 193.252.122.52.www ack[0x5c7db9ca] psh 1=20,402
]
[TCPPacket: 193.252.122.52.www -> 10.56.232.29.4248 ack[0xc21c5fe1] l=20,0]
[TCPPacket: 193.252.122.52.www -> 10.56.232.29.4247 ack[0x151b8aa6] psh 1=20,482
]
[TCPPacket: 193.252.122.52.www -> 10.56.232.29.4248 ack[0xc21c5fe1] psh 1=20,238
]
[TCPPacket: 10.56.232.29.4247 -> 193.252.122.52.www ack[0x5c8d2ab2] psh 1=20,399
]
[TCPPacket: 193.252.122.52.www -> 10.56.232.29.4248 ack[0xc21c5fe1] psh 1=20,513
]
[TCPPacket: 10.56.232.29.4248 -> 193.252.122.52.www ack[0x5c7dbcb9] l=20,0]
[TCPPacket: 10.56.232.29.4248 -> 193.252.122.52.www ack[0x5c7dbcb9] psh 1=20,398
]
[TCPPacket: 193.252.122.52.www -> 10.56.232.29.4247 ack[0x151b8c35] l=20,0]
[TCPPacket: 193.252.122.52.www -> 10.56.232.29.4248 ack[0xc21c616f] l=20,0]
[TCPPacket: 193.252.122.52.www -> 10.56.232.29.4248 ack[0xc21c616f] psh 1=20,809
]
[TCPPacket: 193.252.122.52.www -> 10.56.232.29.4247 ack[0x151b8c35] psh 1=20,453
]
[TCPPacket: 10.56.232.29.4248 -> 193.252.122.52.www ack[0x5c7dbfe2] psh 1=20,402
]
[TCPPacket: 193.252.122.52.www -> 10.56.232.29.4248 ack[0xc21c6301] l=20,0]
[TCPPacket: 10.56.232.29.4247 -> 193.252.122.52.www ack[0x5c8d2c77] l=20,0]
[TCPPacket: 193.252.122.52.www -> 10.56.232.29.4248 ack[0xc21c6301] psh 1=20,534
]
```

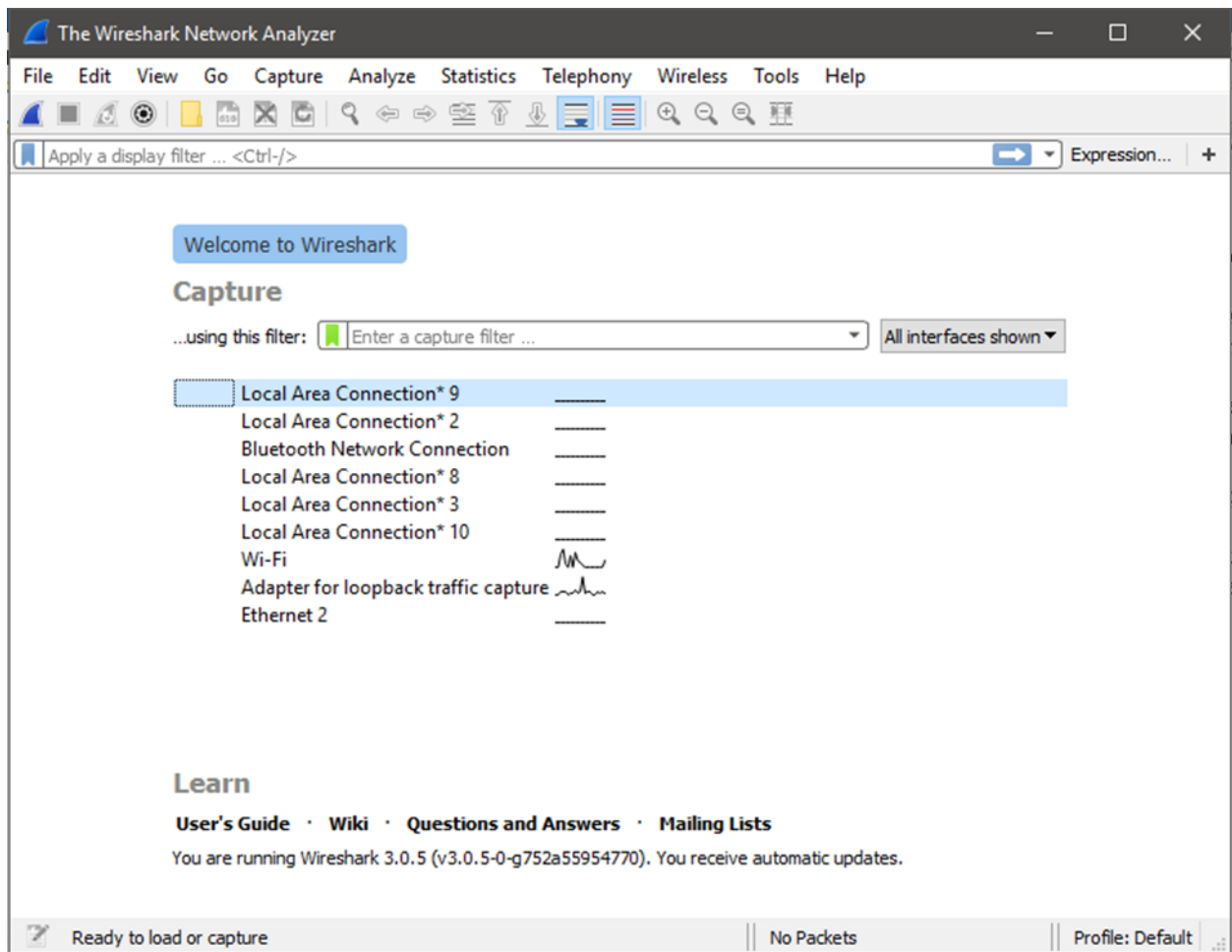
Figure 1: WinPcap raw data

After noticing the difficulties we will face in trying to sift through information in that format, it is clear there is a need for a more flexible method to view and filter the log to isolate the relevant information that tasks are handled by the packet analyzer i.e. Wireshark, which displays the contents of all fields within a

the required dependencies and install them along with `MBinpack`, including `NuGet`

Note: please make sure to terminate all other background programs (outlook, steam, whatsapp, skype,

Upon opening Wireshark for the first time it will scan all networking adaptors in your system after which



Choose the current connection you are using to connect to the internet in the figure above is clearly labeled Wi-Fi.

Start your capture session for five seconds only and then close it. you will be asked if you wish to save it choose yes and name it in the following format [student #] _ [lab1] _ [task#].

Use the lower ribbon for the overall count of your packet of the various types to answer the following questions.

Q1: What is your device IP address, are there any rows where you do not see it?

(hint: there are two forms for your IP known as IPv4 and IPv6) (1 mark)

Q2: How many packets in total were captured? (1 mark)

Q3: What are the ratios of each the UDP, and TCP packets from the overall number of packets?

(hint: use the filter function in Wireshark) (1 mark)

Start another session and access the following link <http://www.uwo.ca/> wait 1 second and hit refresh and then close the session immediately after saving it using the same method as before.

Q4: What is the IP address that corresponds to the http link requested? (1 mark)

Q5: Print the two HTTP messages (GET and OK). (1.5 marks)

To do so, select Print from the Wireshark File command menu, and select the "Selected Packet Only" and "Print as displayed" radial buttons, and then click OK.

Q6: How much time did it take to deliver the website to your device? (1.5 marks)

(hint: use the timestamps for the request and response)

Q7: How much longer did it take the second time when you made the same request for the page? (1.5 marks)

Q8: What do you attribute the change in the time between the two? (1.5 marks)

Problem #2 (10 Marks):

Please, include screenshots with your answers for this problem or ZERO mark will be assigned.

In this problem, you will have the chance to familiarize yourself with the network delays and utilities such as ping and traceroute. Ping is often used for network testing, measurement, and management. Traceroute will help you gain a better understanding of the organization of the Internet.

Ping is a basic Internet application that lets you verify that a particular network host is online and available. It got its name from the SONAR signals used to locate underwater objects. Ping can test basic connectivity between two hosts on a network. Ping makes use of the layer three Internet Control Message Protocol (ICMP) to send ICMP_ECHO requests and receive ICMP_RESPONSE replies. Since the TCP/IP protocol suite incorporates ICMP, any workstation with TCP/IP installed can reply to ping. It also provides a measurement for round trip time (RTT), making performance measurements somewhat easier to record. The command in its simplest form is `ping <hostname>`.

Traceroute is a tool to help debug network route problems. It allows users to determine the route taken by a packet from the local host to a remote host, as well as latency and reachability from the source to each hop on the route. It uses ICMP and the time-to-live (TTL) field in the IP header. The TTL field is used to prevent routing loops that can sometimes occur on networks. If the TTL is 0 or 1, the router is supposed to return the ICMP datagram to the originating host with ICMP_TIMXCEED message. The router appends its IP address as the source address for the error message. As each router on a route receives the traceroute ICMP packet, it decrements the TTL by one; effectively making TTL a hop count parameter.

Q1: Use ping on your workstations to familiarize yourself with these commands. List a few of the flags that are used with these two commands? (1 mark)

Q2: Ping your workstation's loopback interface (localhost). What command did you use? Was the ping successful? (1 mark)

Q3: Ping your default gateway and your DNS server. What command did you use? (1.5 marks)

Q4: Ping google.com, What is the IP address of the computer you pinged? What are the minimum, average, and maximum round trip times? (1.5 marks)

Q5: Use ping to measure Round Trip Time (RTT) for 10 messages 1024 bytes. Use the "-f" configuration switch to make sure that the message is not fragmented. Graph the message size versus RTT for two nodes on a WAN (alibaba.cn, google.ca). Calculate the average and standard deviation for every destination. Discuss the effects of distance and their relationship with latency. (2 marks)

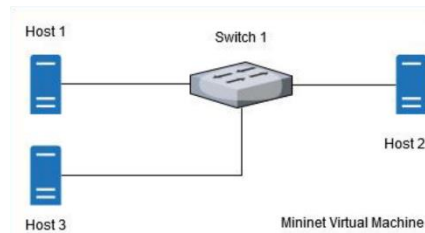
Q6: Use the tracert utility on your workstation to find the route to a host: (3 marks)

- i) In Toronto (google.ca): How many hops did it take to reach the destination host?
- ii) In China (Alibaba.cn): How many hops did it take to reach the destination host?
- iii) Why do you see "*" * *" on some of the output lines?

Problem #3(20 Marks):

Please, include screenshots with your answers for this problem or ZERO mark will be assigned.

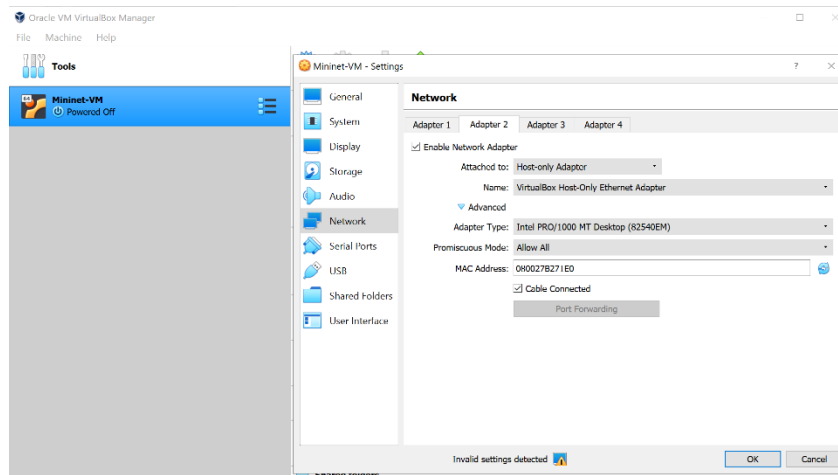
In the problem, you will create a star topology network consisting of 3 hosts and one switch using mininet.



Requirement:

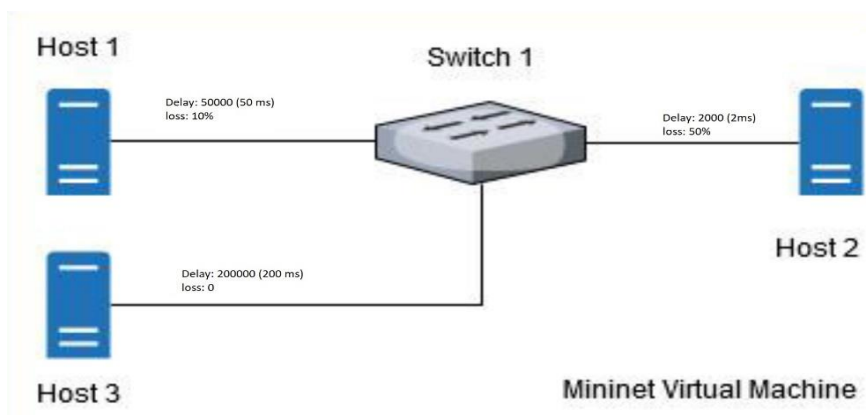
- 1- Install Virtual box to run the mininet virtual machine image.
Link: <https://www.virtualbox.org/wiki/Downloads> : Download version 6.x of virtual box
- 2- Download mininet virtual machine image.
Link: <http://mininet.org/download/#option-1-mininet-vm-installation-easy-recommended>
- 3- Import the mininet virtual machine (VM) to virtual box

- 4- Go to the VM settings after adding the image. Then add a new network adapter as Host-only adapter. As shown in the image below.



- 5- Turn on the virtual machine and access the terminal with:
Username: mininet
Password: mininet
- 6- Get the VM assigned IP by executing the following command in the terminal:
"ifconfig"
- 7- Then use your preferred ssh and ftp client to connect to the VM using the ip you got from the previous step. For Windows, you can install and use [MobaXterm](#). For MacOS, you can install [XQuartz](#) and set the SSH options to use the default SSH client.
Note: Your client should support Xterm to be able to use the mininet GUI.
- 8- Once connected to the mininet VM using your ssh and ftp client, execute the following command to run the mininet GUI:
"sudo ~/mininet/examples/miniedit.py"

Q1: create a star topology network consisting of 3 hosts and one legacy switch with the specification defined in the image below. Save and export as level2 script. Submit the *.mn and the mininet topology python script *.py with your answers. Filename Should follow the following convention StarTopology_<courseNumber>_<uwo username>_<uwo student ID> (1 mark)



Q2: To transmit packets between hosts, you will use the Netcat utility tool already available in your Mininet VM. Host 2 (h2) will act as the receiver and host 1 (h1) as the transmitter. Then, open a UDP port in host 2 and specify a name for the file to be received using the following command in the host 2 terminal:

```
"nc -l -u 34567 > uwo_rcv.jpg"
```

NOTE: The name of the file to be received by host 2 must be different from your original test file to be sent by host 1, since all hosts and network elements in Mininet share the same file system. Otherwise, your original test file will be overwritten.

Send your test file from h1 using the h2's IP address and the same UDP port using the following command in the host 1 terminal:

```
nc -u 10.0.0.2 34567 < uwo.jpg
```

During the transmission process use Wireshark to capture the data. You can open wireshark on host1 and host2 by executing the following command on a separate terminal: "wireshark &"

Pay attention to the Wireshark capture to see once the transmission has finished (no more UDP packets being captured). Force the stop of both transmitting and receiving processes by striking `Ctrl+z` in each host terminal. Also, stop the Wireshark capture.

Explore the captured information to answer:

Q2-1: How long did it take to transfer the file? (0.5 mark)

Q2-2: Are there UDP packets in both directions, i.e. from h1 to h2 and from h2 to h1? Justify your answers. (0.5 mark)

Q2-3: Is the received image the same as the original test file? Do they have the same size? Indicate the sizes. (0.5 mark)

Q3: repeat all the steps with UDP traffic from H1 and H3 this time and answer the following:

Q3-1: How long did it take to transfer the file? (0.5marks)

Q3-2: Are there UDP packets in both directions, i.e. from h1 to h3 and from h3 to h1? Justify your answers. (0.5 mark)

Q3-3: Is the received image the same as the original test file? Do they have the same size? Indicate the sizes. (0.5 mark)

Q4: In this question, you will transfer a file between two hosts through a TCP connection.

To transmit packets between hosts, you will use the Netcat utility tool already available in your Mininet VM. Host 2 (h2) will act as the receiver and host 1 (h1) as the transmitter. Then, open a TCP port in host 2 and specify a name for the file to be received using the following command in the host 2 terminal:

```
"nc -q -l -l 34567 -v > uwo_rcv.jpg"
```


NOTE: The name of the file to be received by host 2 must be different from your original test file to be sent by host 1, since all hosts and network elements in Mininet share the same file system. Otherwise, your original test file will be overwritten.

Send your test file from h1 using the h2's IP address and the same TCP port using the following command in the host 1 terminal:

```
"nc -q -l 10.0.0.2 34567 -v < uwo.jpg"
```

During the transmission process use Wireshark to capture the data. You can open wireshark on host1 and host2 by executing the following command on a separate terminal: "wireshark &"

Pay attention to the Wireshark capture to see once the transmission has finished. Force the stop of both transmitting and receiving processes by striking `Ctrl+z` in each host terminal. Also, stop the Wireshark capture.

Explore the captured information to answer:

Q4-1: How long did it take to transfer the file? (0.5 mark)

Q4-2: Are there TCP packets in both directions, i.e. from h1 to h2 and from h2 to h1? Justify your answers. (0.5 mark)

Q4-3: Is the received image the same as the original test file? Do they have the same size? Indicate the sizes. (1.5 marks)

Q4-4: Show the handshaking process of TCP in Wireshark with a screenshot. (1.5 marks)

Q4-5: What is the RTT for the TCP connection. (1.5 marks)

Q4-6: Show the packet loss of TCP in Wireshark with a screenshot. (1 mark)

Q4-7: Show the packet retransmission request of TCP in Wireshark with a screenshot. (1 mark)

Q5: repeat all the steps with TCP traffic from H1 and H3 this time and answer the following:

Q5-1: How long did it take to transfer the file? (0.5 mark)

Q5-2: Are there TCP packets in both directions, i.e. from h1 to h3 and from h3 to h1? Justify your answers. (1 mark)

Q5-3: Is the received image the same as the original test file? Do they have the same size? Indicate the sizes. (1.5 marks)

Q5-4: Show the handshaking process of TCP in Wireshark with a screenshot. (1.5 marks)

Q5-5: What is the RTT for the TCP connection. (1.5 marks)

Q5-6: Show the packet loss of TCP in Wireshark with a screenshot. (1.5 marks)

Q5-7: Show the packet retransmission request of TCP in Wireshark with a screenshot. (1 mark)

Problem #4 (35 marks):

Please, include screenshots with your answers for this problem or ZERO mark will be assigned.

In this problem, you will learn the basics of socket programming for UDP in Python. You will learn how to send and receive datagram packets using UDP sockets and also, how to set a proper socket timeout.

You will first study a simple Internet ping server written in Python, and implement a corresponding client. The functionality provided by these programs is similar to the functionality provided by standard ping programs available in modern operating systems. However, these programs use a simpler protocol, UDP, rather than the standard Internet Control Message Protocol (ICMP) to communicate with each other. The ping protocol allows a client machine to send a packet of data to a remote machine, and have the remote machine return the data back to the client unchanged (an action referred to as echoing). Among other uses, the ping protocol allows hosts to determine round-trip times to other machines.

Your task is to write the Ping Server and Client.

Server Code (17.5 marks):

You need to implement the following server program.

The server sits in an infinite loop listening for incoming UDP packets. When a packet comes in and if a randomized integer is greater than or equal to 3, the server simply capitalizes the encapsulated data and sends it back to the client. If the random integer is less than 3 the server does not respond.

UDP provides applications with unreliable transport service. Messages may get lost in the network due to router queue overflows, faulty hardware, or some other reasons. Because packet loss is rare or even non-existent in typical local networks, the server in this problem injects artificial loss to simulate the effects of network packet loss. The server creates a variable randomized integer that determines whether a particular incoming packet is lost or not.

Use this code for generating the random integer.

```
import random  
  
rand = random.randint(0, 10)
```

Client Code (17.5 marks):

You need to implement the following client program.

The client should send 10 pings to the server. Because UDP is an unreliable protocol, a packet sent from the client to the server may be lost in the network, or vice versa. For this reason, the client cannot wait indefinitely for a reply to a ping message. You should get the client to wait up to two seconds for a reply; if no reply is received within two seconds, your client program should assume that the packet was lost during transmission across the network.

Specifically, your client program should do the following:

- (1) send the ping message using UDP (Note: Unlike TCP, you do not need to establish a connection first, since UDP is a connectionless protocol.)
- (2) print the response message from the server, if any
- (3) calculate and print the round trip time (RTT), in milliseconds, of each packet, if server responses
- (4) otherwise, print "Request timed out"

Note: use localhost or 127.0.0.1 for the server and client sockets

Message Format:

The ping messages in this lab are formatted in a simple way. The client message is one line, consisting of ASCII characters in the following format:

Ping from client <sequence_number> <time>

where <sequence_number> starts at 1 and progresses to 10 for each successive ping message sent by the client, and <time> is the time when the client sends the message.

What to Hand in

You will hand in the **complete server and client codes** and **screenshots** verifying that your ping program works as required.

Naming conventions for the files:

ServerPingCode_<course number>_<uwo username>_<uwo ID>.py

ClientPingCode_<course number>_<uwo username>_<uwo ID>.py

Along with the above specifications, your program should do also the following:

1. You will need to report the minimum, maximum, average, and standard deviation RTTs at the end of all pings from the client.
2. Also, you need to calculate and show the packet loss rate (in percentage) before the client exists.

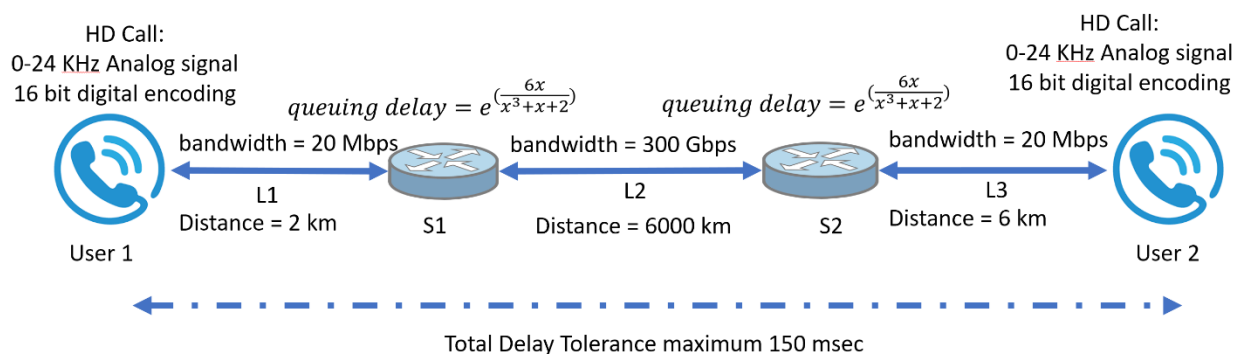
Problem #5 (25 marks):

In this problem, you will solve for design problem of a networking system.

Consider having 2 users initiating an HD voice over IP call (VOIP) call. The HD voice call analog frequency is from 0 to 24KHz. 16-bit encoder is used by the analog-to-digital process. The voice call held over a distance where 2 wan hops exist and have a connection of 300Gbps bandwidth and 6000 Km apart. The users (User 1 and User 2) have 2 Km, 6 Km distance from there access point respectively, and a bandwidth links of 20 Mbps each. The 2 hops (switches) have a queuing delay of:

$$\text{queuing delay} = e^{\left(\frac{6x}{x^3+x+2}\right)}$$

The image below describes the problem.



propagation speed: $2.8 * 10^8$ m/sec

Q1: Considering the delay tolerance for not hearing jitters by the human ear is 150 msec maximum.

Solve for variable X in the queuing delay to satisfy the call requirement. Consider all types of delays that exist in the network.

Show all your steps and the delays detailed calculations used to solve for variable X (25 marks).