Twisha Sharma
CSE 150
Parsa

Lab 3 - Building a Router and Dive into TCP

Introduction: In Prelab3, we learned about Software Defined Networking (SDN) and the POX Controller to design our own Firewall. In this Lab, we are going to learn and experiment more using the POX Controller to build a router. This lab again assumes you will spend time reading the documentation and learning about Mininet, OpenFlow and the POX controller.
In the second half of the assignment we explore the internal workings of the TCP protocol.

[50 pts] Part 1: Router
Your goal will be to allow or block traffic between the different devices based on the 4 Network Topology Rules given below. In this assignment you cannot use flooding (i.e., you cannot use flooding: of.OFPP_FLOOD)! Instead, you will need to specify specific ports for all traffic and the forwarding must be done by subnet, not by enumerating all of the IP addresses. You might consider a method to determine if an IP Address is valid on a particular subnet. Notice that the rules are written according to the subnet.

You may implement your router however you choose—although as a suggestion, you may find it easiest to determine the correct destination port by using
● the source and destination IP addresses
● or the source port on the switch from which the packet originated.

These provided files will get you started—you will need to modify both:
`lab3_topo_skel.py` / `lab3_controller_skel.py`

We will use the same topology as prelab3 and shown in Figure 1; however, now the departments have been organized into independent subnets. You will need to do the following: ● Finish the topology file by adding all the hosts (be sure to manually specify the MAC address, IP address and subnet for each host), switches and links
● Implement the controller based on Rule 1, Rule 2, Rule 3 and Rule 4.
Reminder: The same naming convention for the switches and hosts used in prelab3 must be followed here.

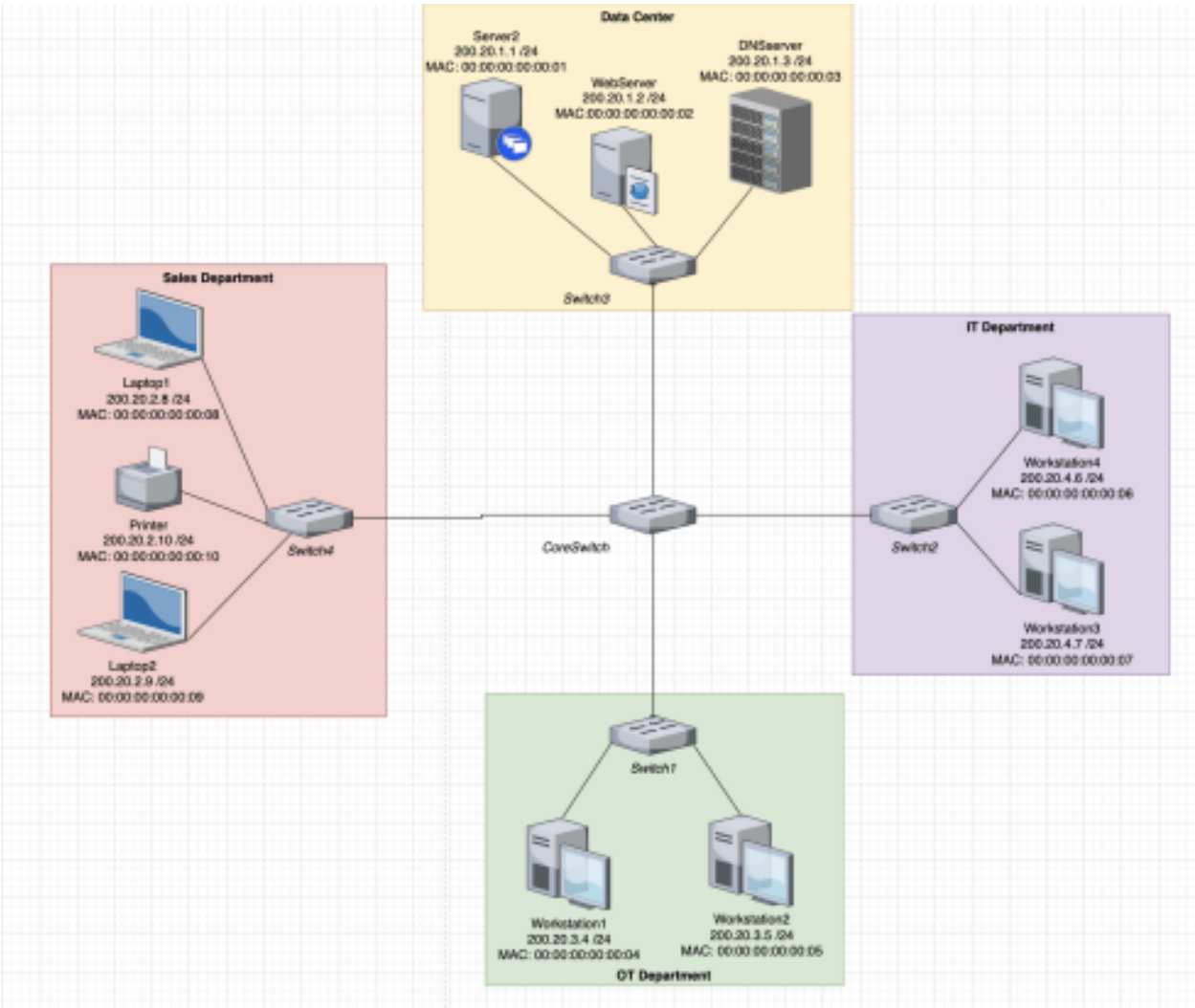


Figure 1

Network Topology and Rules to be implemented:

● Rule 1: ICMP traffic is forwarded only between the Sales Department and IT Department subnets or between devices that are on the same subnet.

● Rule 2: TCP traffic is forwarded only between the Datacenter, IT Department and OT Department subnets or between devices that are on the same subnet.

● Rule 3: UDP traffic is forwarded only between the OT Department and Datacenter subnets or IT Department and Datacenter subnets or between devices that are on the same subnet.

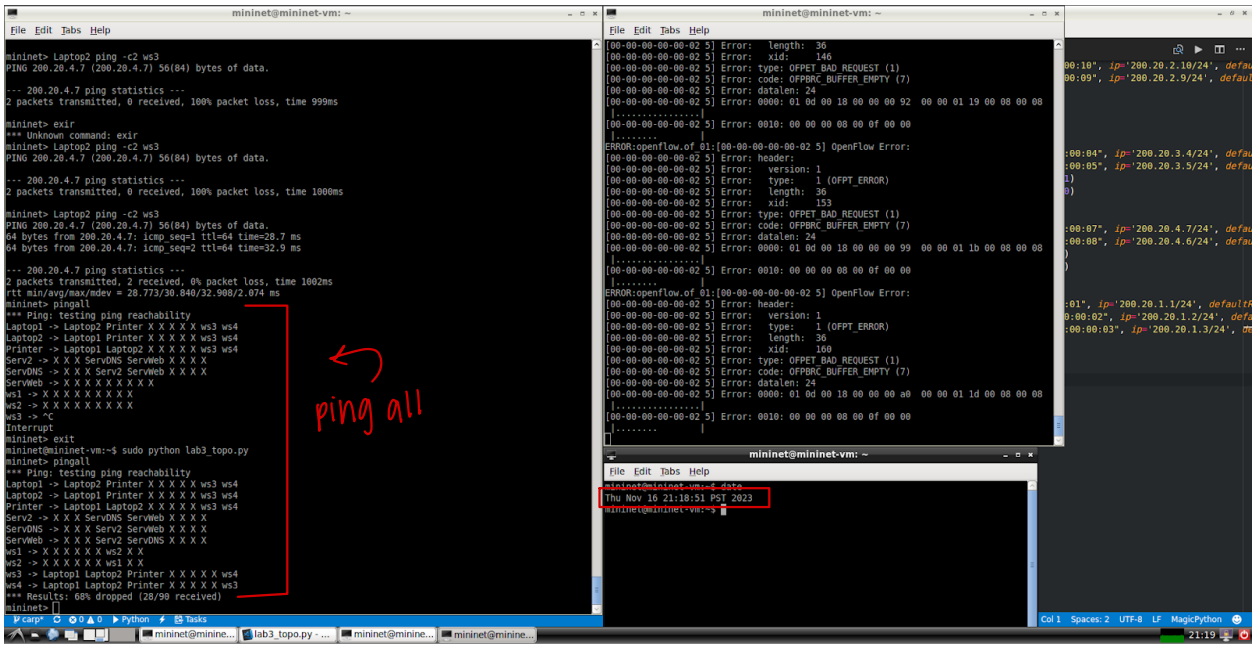
● Rule 4: All other traffic should be dropped.

Questions and Rule Verification:

1. Annotate the topology in Figure 1 with ports numbers associated with the end of each link and include a screenshot of the detailed image of Figure 1. Then, implement the topology in the skeleton file `lab3_topo_skel.py`.
2. Explain how packets are forwarded differently in this assignment compared to Prelab3. Think about your accept() function in Prelab 3.

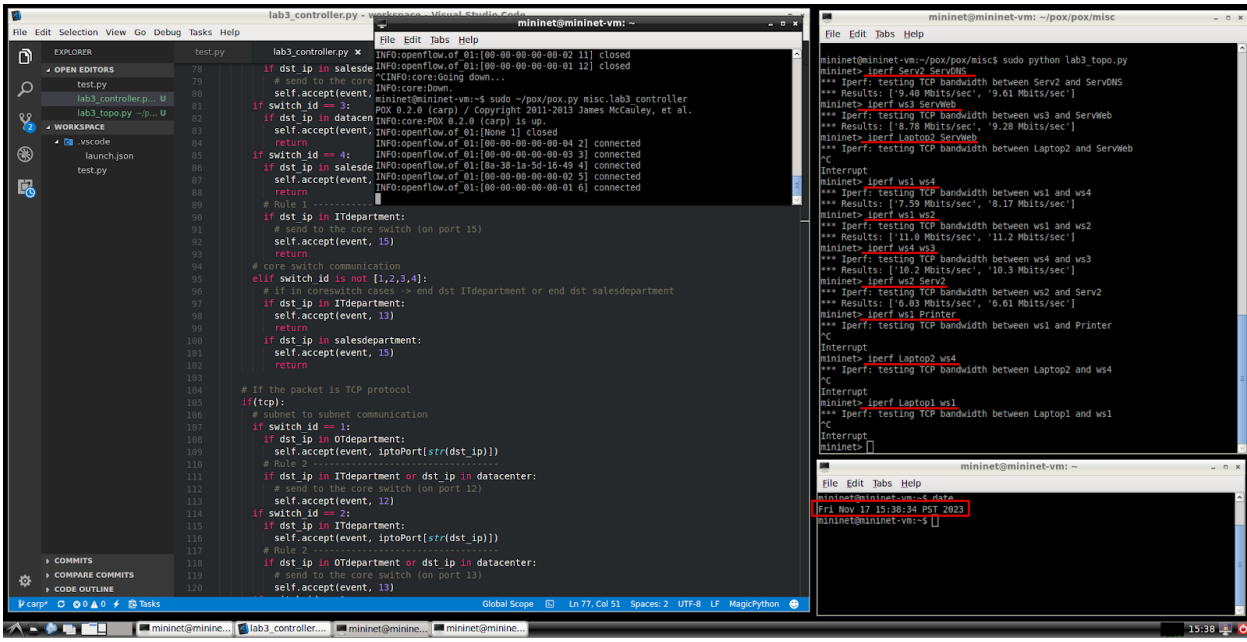
In this lab the packet forwarding is done using a routing method. This means that within the accept() function, when the message is matched, the flag that the port value is set to is different from the prelab.

3. Verify Rule 1 with pingall. Circle in red the lines of your output that indicate the inter-departmental communication is working as expected. Explain your answer. Include a timestamped screenshot and refer to it in your explanation.
    a. The ping all screenshot shows on the very first line itself, of a successful ping between the devices Laptop1, Laptop2, and the Printer. Itthen moves to the core switch where packets are naturally dropped and then shows a successful ping on workstation and workstation4. This demonstrates interdepartmental communication (Laptop1, Laptop2, and Printer) and cross departmental communication (Laptop1 to ultimately ws4)

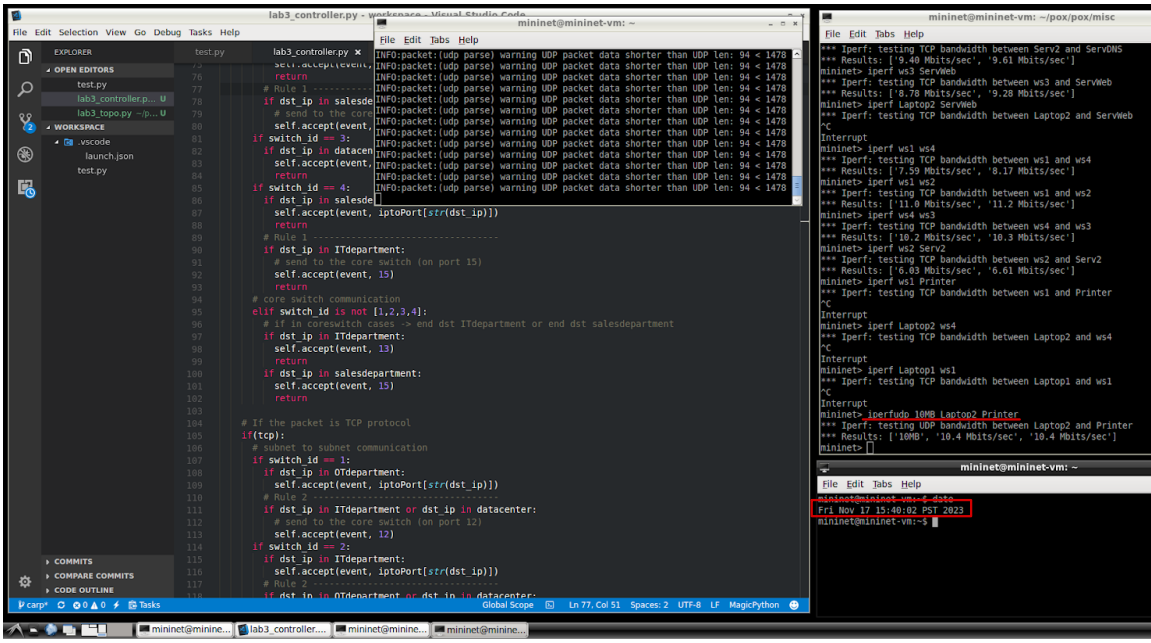


4. Verify Rule 2 with iperf. Are the individual results what you expect? Why? Include a timestamped screenshot and refer to it in your explanation after running iperf between these pairs of nodes:
    a. Server2 and DNS server (Yes) - iperf uses TCP protocols, and per Rule 2 TCP traffic is allowed between the Datacenter, IT department, and OT department, and so the traffic should be allowed because Server2 is a host in the Datacenter and the DNS server is also a host within the data center, and Rule 2 also specifies that there is allowed communication internally within each subnet. Since both the hosts are in the same subnet, it makes sense that the communication is allowed.

b. Workstation 3 and WebServer (Yes) - iperf uses TCP protocols, per Rule 2, workstation3 which is in the IT department and WebServer which is in the Datacenter department would be allowed to communicate.
c. Laptop2 and WebServer (No) - Laptop2 is within the Sales Department while WebServer is within the Data Center. This means that per rule 2 and iperfs use of TCP protocols, the 2 hosts would not be allowed to communicate and traffic between them is dropped.
d. Workstation1 and Workstation4 (Yes) - Worstation1 is within the OT Department and Workstation4 is within the IT Department so per Rule 2, which allows communication between the IT and OT departments, traffic is allowed between the 2 hosts.
e. Workstation1 and Workstation2 (Yes) - Workstation 1 and Workstation 2 are both hosted within the OT Department. Iperf uses TCP protocols, and Rule2 allows devices on the same subnet to communicate so Workstation 1 and Workstation 2 are allowed to communicate.
f. Workstation4 and Workstation3 (Yes) - Workstation 4 and Workstation 2 are both hosted within the IT Department. Iperf uses TCP protocols, and Rule2 allows devices on the same subnet to communicate so Workstation 1 and Workstation 2 are allowed to communicate.
g. Workstation2 and Server2 (Yes) - Workstation2 is hosted in the OT Department, and Server2 is hosted within the Data Center. Per Rule 2, TCP traffic between the data center and the OT department is allowed, and iperf uses TCP protocols.
h. Workstation 1 and Printer (No) - iperf uses TCP protocols, and Rule 2 allows TCP traffic between the Data Center, IT, and OT Departments. This means that traffic from workstation one hosted in the OT department and the printer hosted in the Sales department is not allowed.
i. Laptop2 and Workstation 4 (No) - iperf uses TCP protocols, and per rule 2 TCP traffic can only be allowed between the Datacenter, IT department, or OT department, and Laptop2 is a host within the Sales Department subnet and workstation4 is a host with IT department subnet.
j. Laptop1 and Workstation 1 (No) - iperf uses TCP protocols and per rule 2, traffic is only forwarded between the OT department, IT department, and datacenter subnets, meaning that Laptop1 which s in the sales department would not be allowed to communicate with ws1 which is a host in the OT department.
k. Important: If iperf takes longer than around 15 sec to run, this usually means the hosts can’t reach each other; cancel with ctrl-c and include the result in your screenshot.



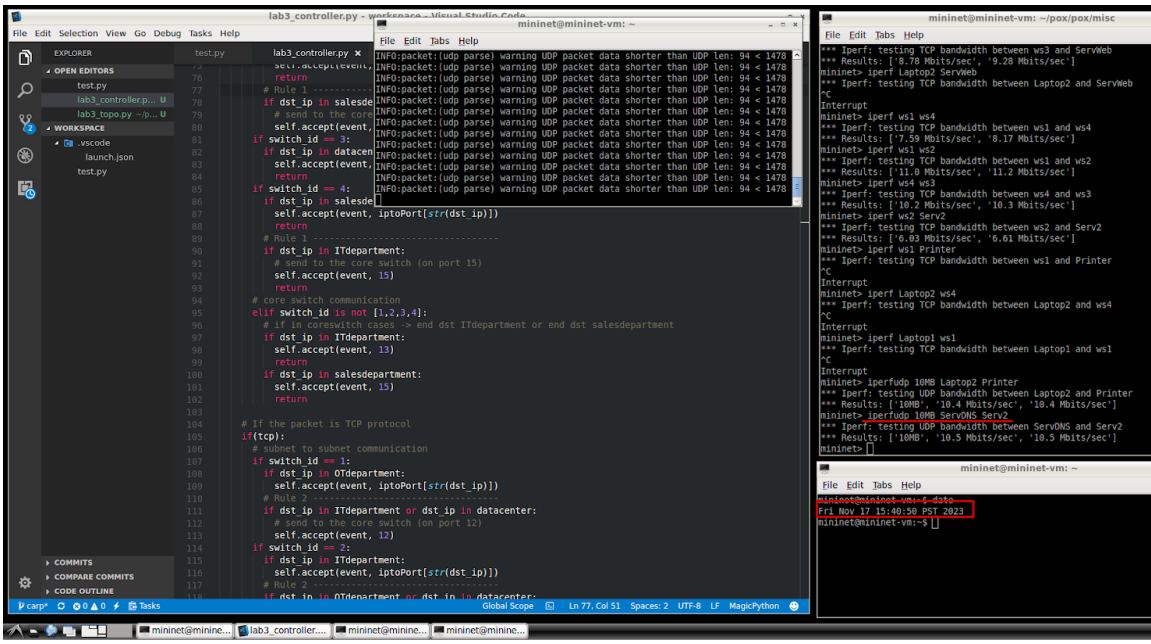
5. Verify Rule 3 with the command (iperfudp 10M <host1> <host2>). Are the individual results what you expect? Why? Include a timestamped screenshot and refer to it in your explanation after running iperfudp between these pairs of nodes: - Laptop1 and Web Server
    a. Laptop2 and Printer
        i. Yes I expected Iperf between Laptop2 and Printer to be successful. Even though invoked to use udp protocols, Rule 3, allows UDP traffic between devices on the ame subnet, and Laptop2 and the Printer are both hosted by the Sales Department, meaning traffic between them is allowed.
        ii.



    b. DNS server and Server2

i. Yes I expected Iperf between DNS Server and Server2 to be successful. Even though invoked to use udp protocols, Rule 3, allows UDP traffic between devices on the same subnet, and DNS Server and Server2 are both hosted by the Data Center, meaning traffic between them is allowed.

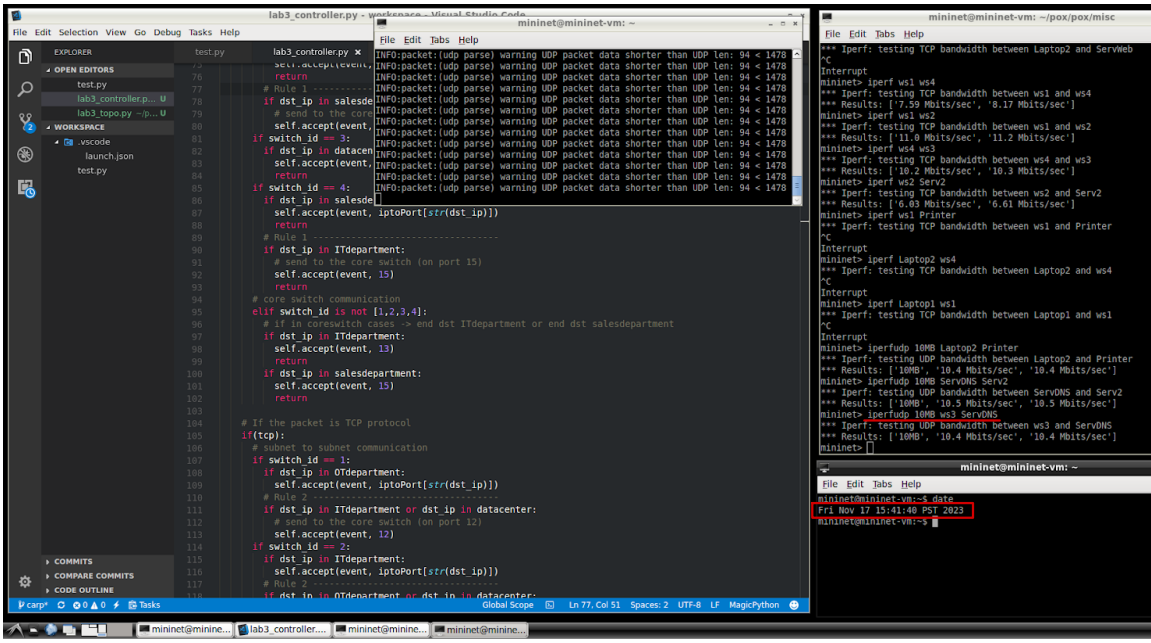
ii.

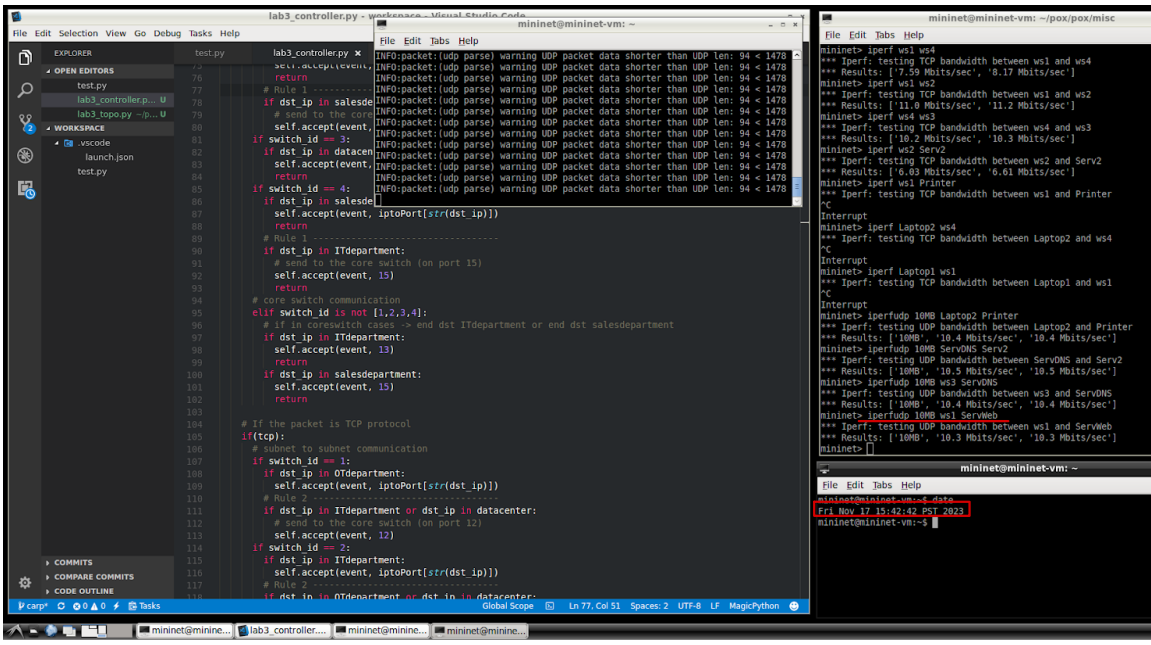


c. Workstation3 and DNS server

i. Yes I expected Iperf between Workstation3 and DNS Server to be successful. Even though invoked to use udp protocols, Rule 3, allows UDP traffic between devices from the It department to the Data Center or vice versa. Workstation3 is hosted in the IT Department, and the DNS Server is hosted by the Data Center, meaning traffic between them is allowed.

ii.



d. Workstation 1 and WebServer
   i. Yes I expected Iperf between Workstation1 and WebServer to be successful. Even though invoked to use udp protocols, Rule 3, allows UDP traffic between devices from the OT Department to the Data Center and vice versa. Workstation1 is hosted in the Ot Department and the WebServer is hosted in the DataCenter, meaning traffic between them is allowed.
   ii. 

e. Important: A UDP test is considered successful if the iperfudp output displays two UDP bandwidth measurements between hosts. If one of the UDP bandwidth measurements is missing, it means that UDP traffic is not flowing. For example:
   Pass: *** Results: ['10M', '10.1 Mbits/sec', '10.1 Mbits/sec']
   Failed: *** Results: ['10M', '', '10.0 Mbits/sec']

[50 pts] Part 2: TCP In this section we explore several aspects of the TCP protocol. First, we will upload a file to a webserver and observe the flow of packets over the TCP connection. Next, we will download a large file and introduce losses on the link to observe TCP's reliable data delivery service in action. You will be TCP experts by the end!

A. Upload a Great Story to a Web Server

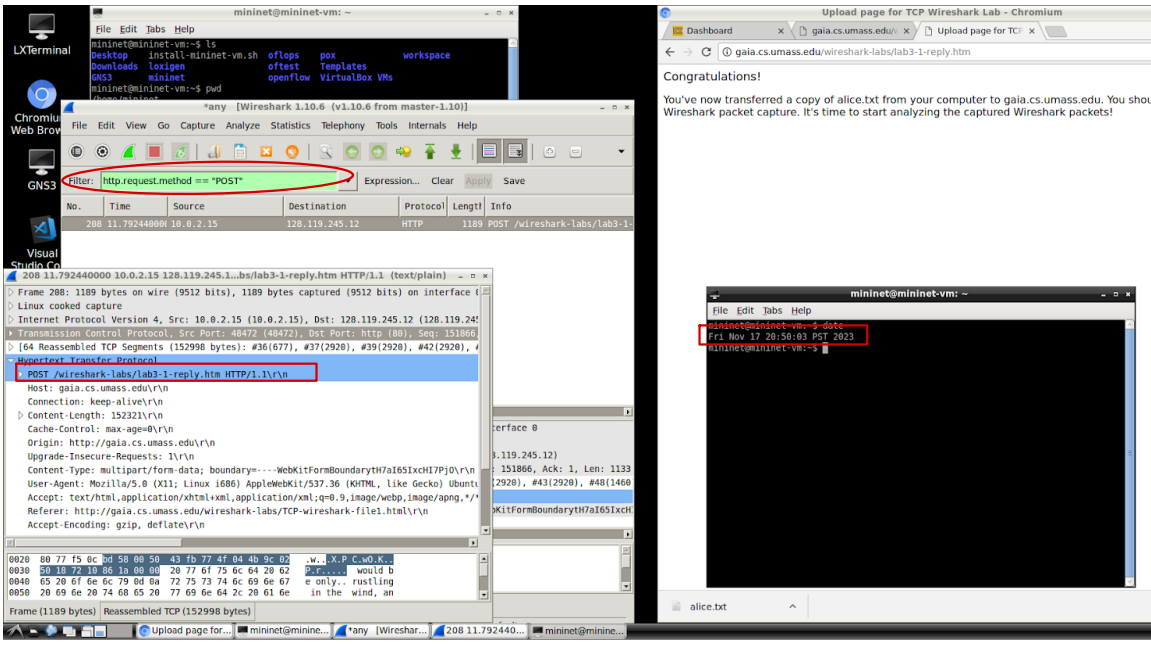
In this section you will observe TCP uploading a text file. We recommend doing this section on your own computer as the TCP transfer on the VM is tricky. (If you do work on the VM, disable HTTP in the Enabled Protocols). You should be able to find packets being uploaded, similar to the screenshot below.

Follow these instructions in the order they are given:

● Download the ASCII copy of Alice in Wonderland http://gaia.cs.umass.edu/wiresharklabs/alice.txt and store the file on your computer. (Read if you have time- it's a great story!)

● Load the page http://gaia.cs.umass.edu/wireshark-labs/TCP-wireshark-file1.html ● Start Wireshark and begin packet capture.

● As you use the buttons displayed in the web page to upload the Alice in Wonderland file, observe the URL bar in the browser. (See Q6 below)
● Once the file has been uploaded, a short congratulations message will be displayed in your browser window

6. Observe the URL in your browser during the file upload. Does it change to reflect the upload process? Investigate HTTP data and form upload methods. Identify the HTTP method used for this upload and justify your choice based on your findings. Make sure to state the reasons you think a particular method was used.
    a. The packet to observe within wireshark was the HTTP “POST” packet, which is the HTTP method used for file uploads. I found the packet using the wireshark filter http.request.method == “POST”. The url on the top of the browser does change during the file upload, indicating a change in the url and a full page resubmission after the file is uploaded.
    i. 

TCP Basics

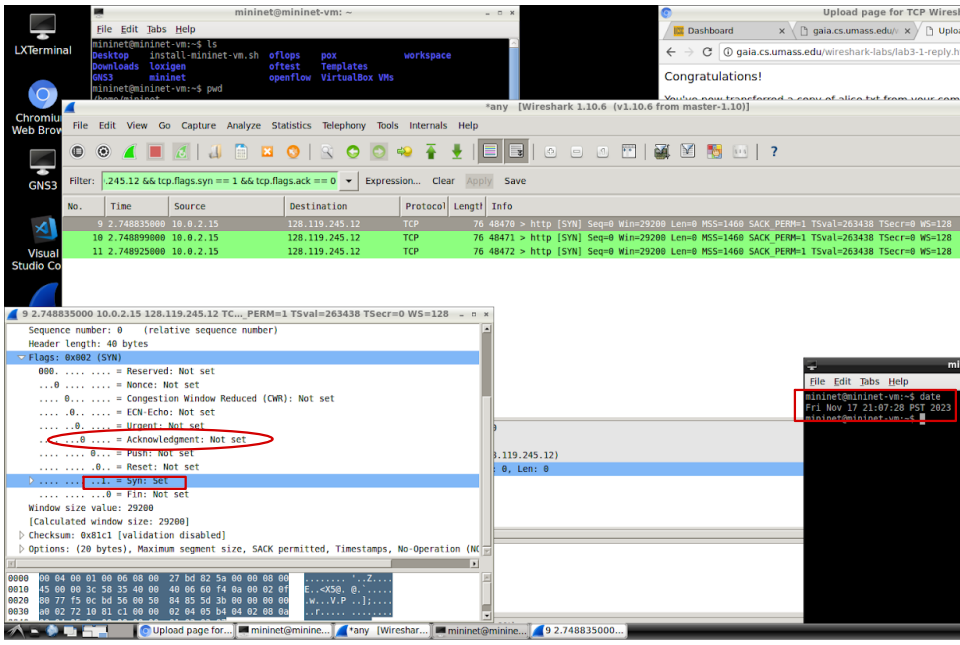
Answer the following questions for the TCP segments you observe in Wireshark:

7. TCP Connection Setup:
    a. Illustrate the Handshake:
    Using a drawing program, construct a timing diagram to illustrate the TCP 3-way handshake process. Ensure your diagram includes:
    ● Two endpoints labeled Host A (your computer) and Host B (the server), including their IP Addresses
    ● The SYN packet sent from Host A and Host B
    ● The SYN-ACK packet sent from Host B to Host A
    ● The final ACK packet sent from Host A to Host B
    ● For each packet, include the relative and raw sequence number, relative and raw acknowledgement number, and the Reciever’s Advertised Window size.

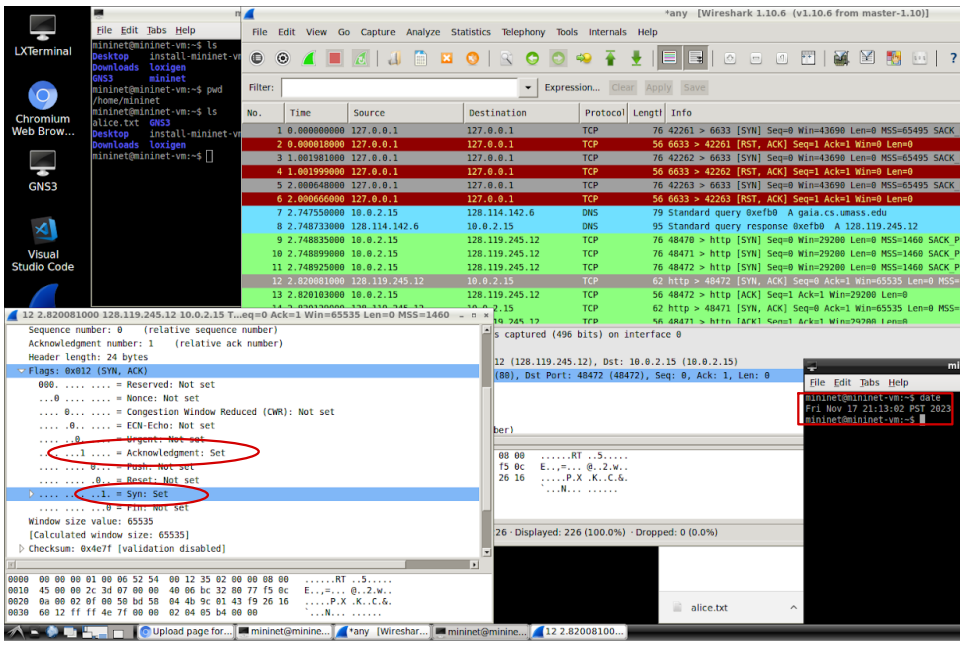
Clearly label all sequence, acknowledgment, and window numbers.
Include 3 timestamped screenshots and circle all the information used in the diagram.

i. TCP 3 WAY HANDSHAKE DRAWING

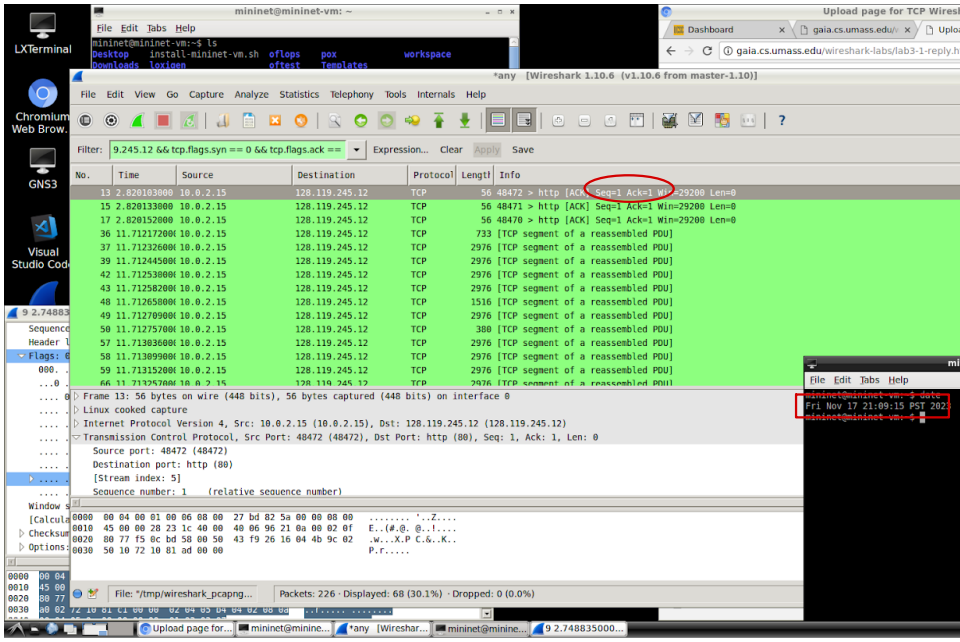
1. The SYN = 1 and the ACK = 0

a. 

2. The SYN = 1 and the ACK = 1

a. 

3. The SYN = 0 and the ACK = 1

a. 

b. Purpose of Sequence Numbers and Receiver's Window Size:
Explain why the initial SYN packet must include a sequence number and its continued role throughout the handshake process. Explain the purpose of the window size number.
   i. The initial SYN packet includes a sequence number that is used to establish the start of the data. This is used as a reference point so that both sides of the network can track the order of transmission.

c. SYN-ACK Segment Details:
Why are the sequence numbers different in the SYN and SYN-ACK packets?
Explain how the acknowledgment number is calculated in the SYN-ACK packet.
   i. The sequence numbers in the SYN and the SYN-ACK packets are different because each host chooses its own initial sequence number . It is independent. They are also randomly selected. The acknowledgment number of the SYN-ACK packet is calculated based on the sequence number in the SYN packet that's received from host A, and is set to the received number +1. This is to show the expectation on the next sequence number to be received from host A.

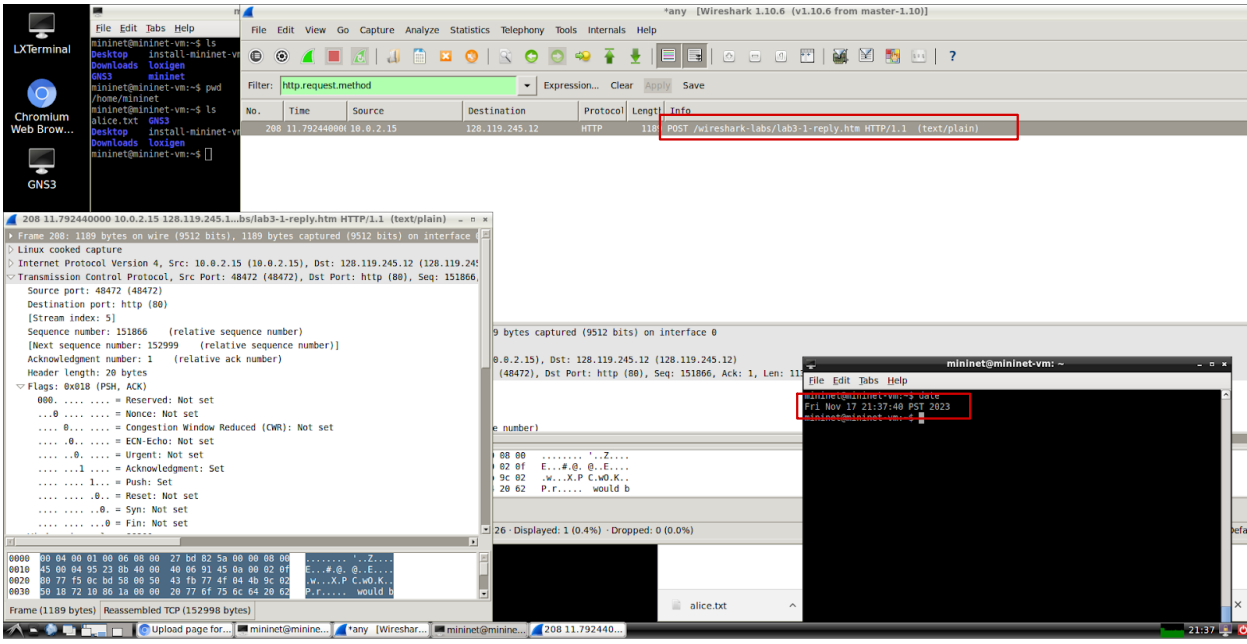
d. ACK Segment Details:
Explain how the sequence number in the final ACK packet is determined. What is the acknowledgment number in the final ACK packet referencing?
   i. The sequence number of the final ACK packet is determined by incrementing the sequence number from the previous packet by one. The acknowledgment number in the final ACK packet is referencing the sequence number from the SYN-ACK packet received from Host B.

<u>TCP File Upload</u>

8. Beginning the File Upload:

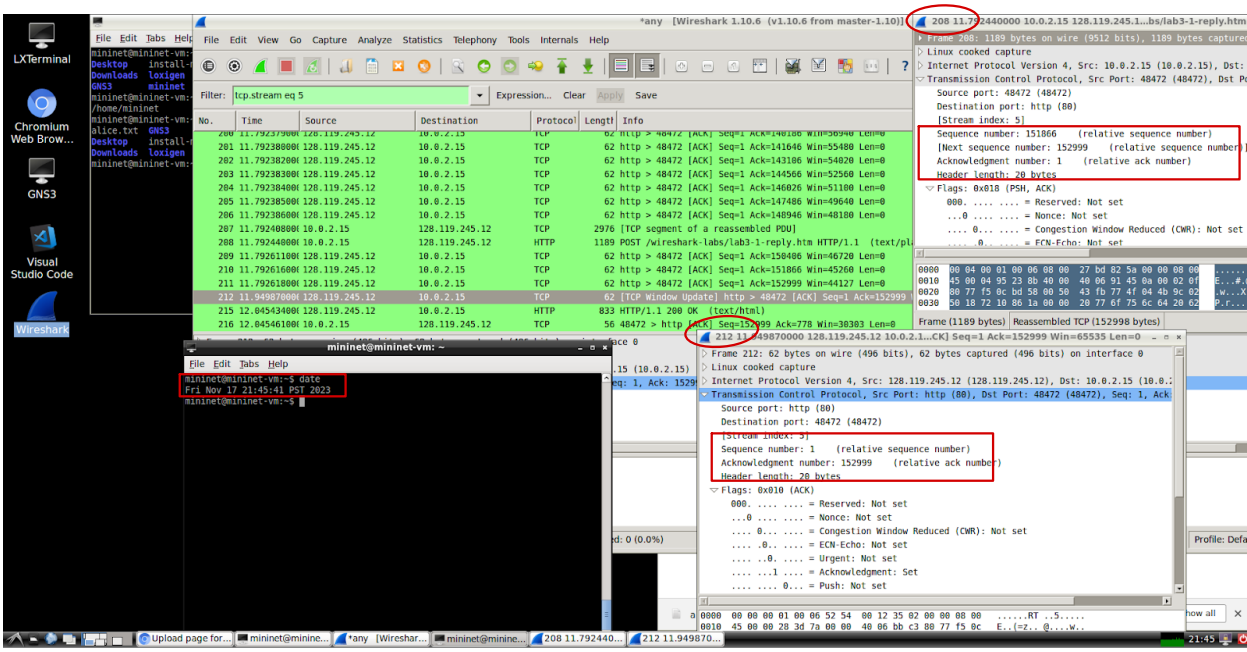
a. HTTP Request:
According to Wireshark, which HTTP method was used to initial the file upload (look for the HTTP packet)?
Circle the method in a timestamped screenshot.

i. The HTTP method was POST



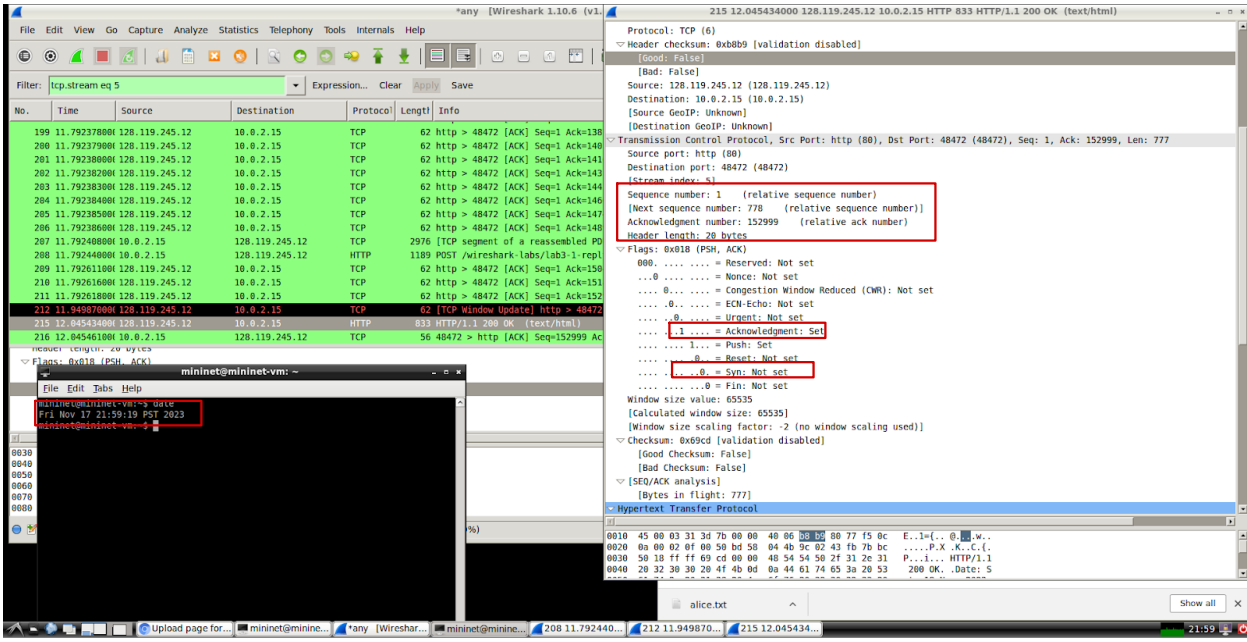
Hunting for the TCP Payload (b) and (c):

b. Find the first TCP segment (look for the TCP packet) that carries this HTTP message for the file upload in (a) and convince yourself that the payload of this packet is the HTTP request message by analyzing this TCP packet. Find the bytes of the payload as we showed in the lab.) What is the TCP segment length and relative sequence number of this TCP segment? Circle the HTTP method that was used (visible in the payload bytes) in a timestamped screenshot.



i. The first packet, frame 208, was the HTTP request method POST. has an ACK 1 and the next seq number of 152999. The second packet, frame 212, can be verified as the corresponding TCP segment, through its ack number, 152999, which is the same as the “next seq number” of the HTTP request packet. The size of the TCP packet on frame 212 is of size window size 65535.

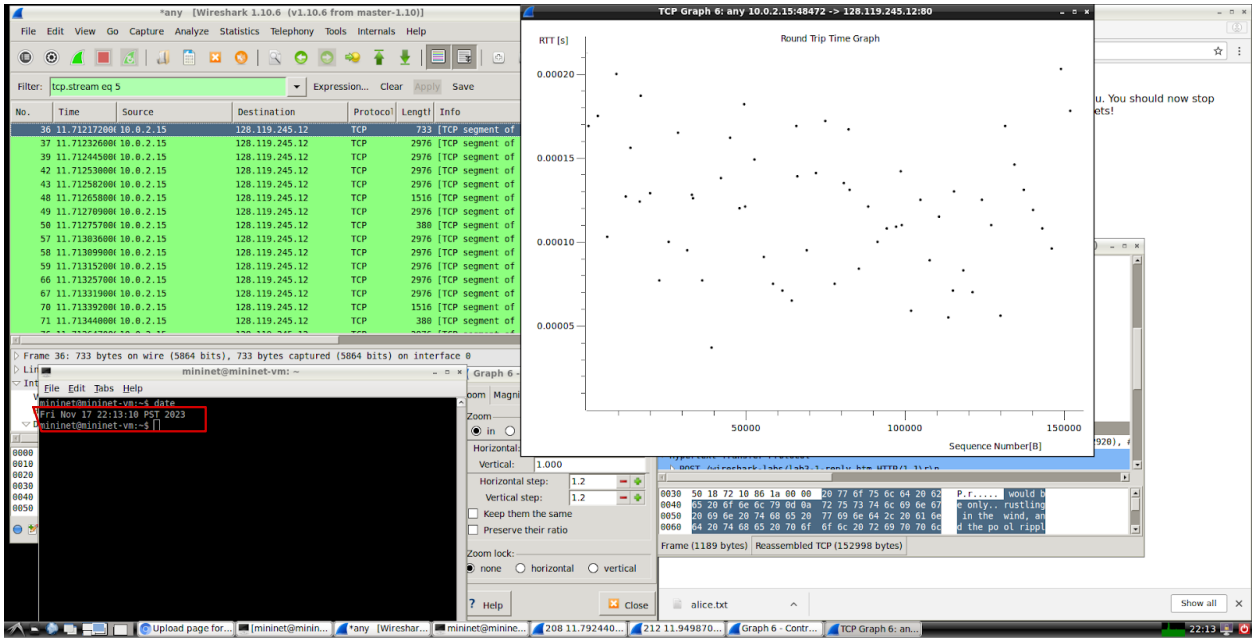
c. Now find the first TCP segment of the file transfer – it carries the beginning portion of the actual Alice in Wonderland text. Convince yourself that the payload of this packet is the contents of the Alice in Wonderland file in your screenshot. Analyze this TCP packet and find the bytes of the payload as we showed in the lab.
Circle the contents of Alice in Wonderland (visible in the payload bytes) in a timestamped screenshot.
What is the TCP segment length and relative sequence number of this TCP used below).



Sequence number = 1
Next sequence number = 778
Acknowledgement number = 152999
The packet on frame 215 has a sequence number of 1 and its next sequence number is 778. It also has an acknowledgement number of 15299 which shows that this is the next packet for the one in (b) where the acknowledgement number of this packet matches the sequence number of the last packet.

d. Look at the TCP segment length and relative sequence numbers for several subsequent segments. Can you discern a pattern? Explain how sequence numbers are calculated in subsequent segments and what they represent in the context of the TCP stream.
    i. The subsequent packets all have matching lengths of 62. The sequence numbers are used to track and manage data segments. They are used to ensure reliable and ordered delivery of data between the sender and the receiver.

9. Graphing the RTT (Round Trip Time):
Wireshark has a nice feature that allows you to plot the RTT for each of the transmitted TCP segments. Click on a TCP segment that is being sent from the client to the gaia.cs.umass.edu server.
Navigate to: Statistics -> TCP Stream Graphs -> Round Trip Time to visualize the RTT data. Note: The direction of the transfer is important! Ensure multiple segments are displayed (click switch direction if you only see a single point in the plot)



a. How would the values shown in this graph be used as a part of TCP's retransmission timeout (RTO) calculation?
    i. They can be used as a part of the TCP's retransmission timeout calculations. This is an important parameter in TCP that determines how long the sender waits before retransmitting a segment that has not been acknowledged by the receiver. The calculation for the retransmission timeout calculation is calculated based on the observed round trip times to estimate how long it takes for a segment to travel from the sender to the receiver and back.

b. Discuss your observations in your RTT Graph. (Think about the discussion in Lecture 18 related to TCP's RTO calculation and the graph that was discussed.) Include a timestamped screenshot of this graph.
    i. My observation of the graph is that it all stays within the same segment of RTT. The fact that there is no real or discernible pattern on the RTT graph can also imply that the network conditions were stable, there was low latency. The lack of pattern in the RTT graph suggests that the network was free of congestion and packet loss and other issues that typically lead to varying RTT values.

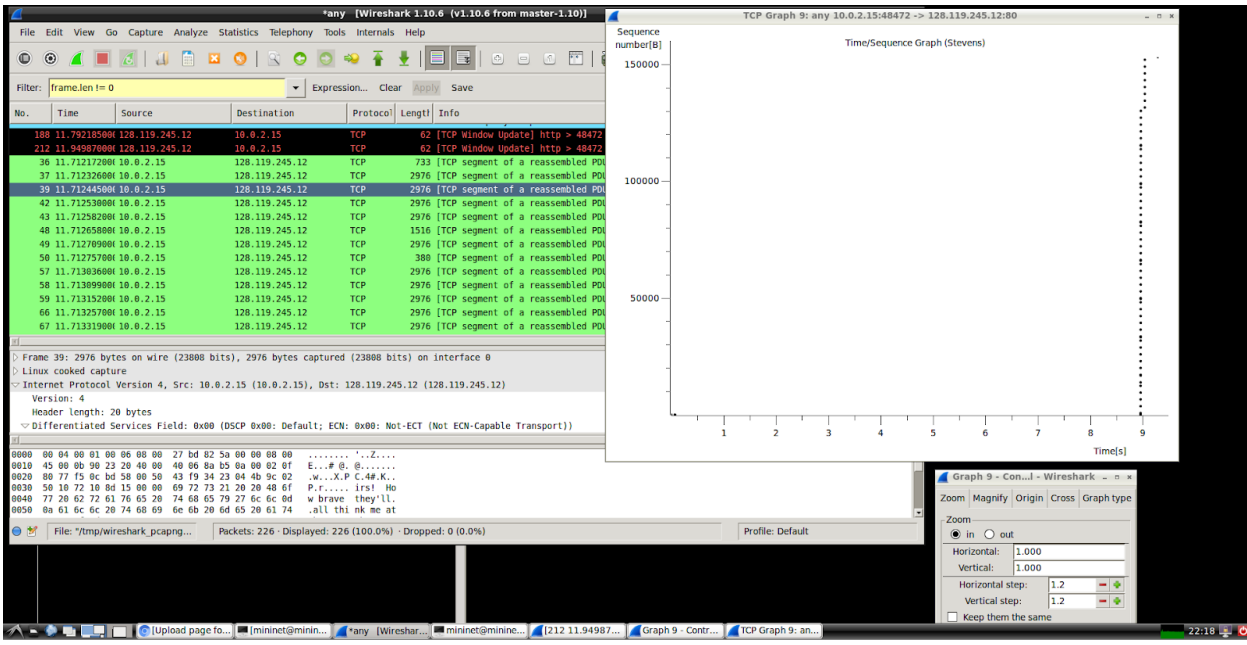
TCP Congestion Control

10. Time-Sequence-Graph(Stevens): Select a TCP packet with len != 0 that has been transmitted by the client and use the Time-Sequence-Graph(Stevens) plotting tool to create a graph.
    a. How many bytes are in the entire TCP segment you selected? (payload + header)
        i. 2976 bytes
    b. Explain what is displayed on the x and y-axis in the graph. What does each dot represent?

i. The x axis represents the time and the y axis represents the sequence numbers of the TCP packet. Each dot on the graph represents a TCP packet and its sequence number at a specific point in time.

c. Put a red circle around any region where a slow start begins.

d. Are there any retransmitted segments in your graph? If so, circle them in red on the graph. Make sure your graph shows the entire transfer and the lost segments, if any. Include a timestamped screenshot of this graph.



11. Average Transmission Rate for File Upload

a. Based on the Time-Sequence-Graph(stevens), what is the average transmission rate (over the duration of the file upload)?

b. Explain how you calculated this value (your process).

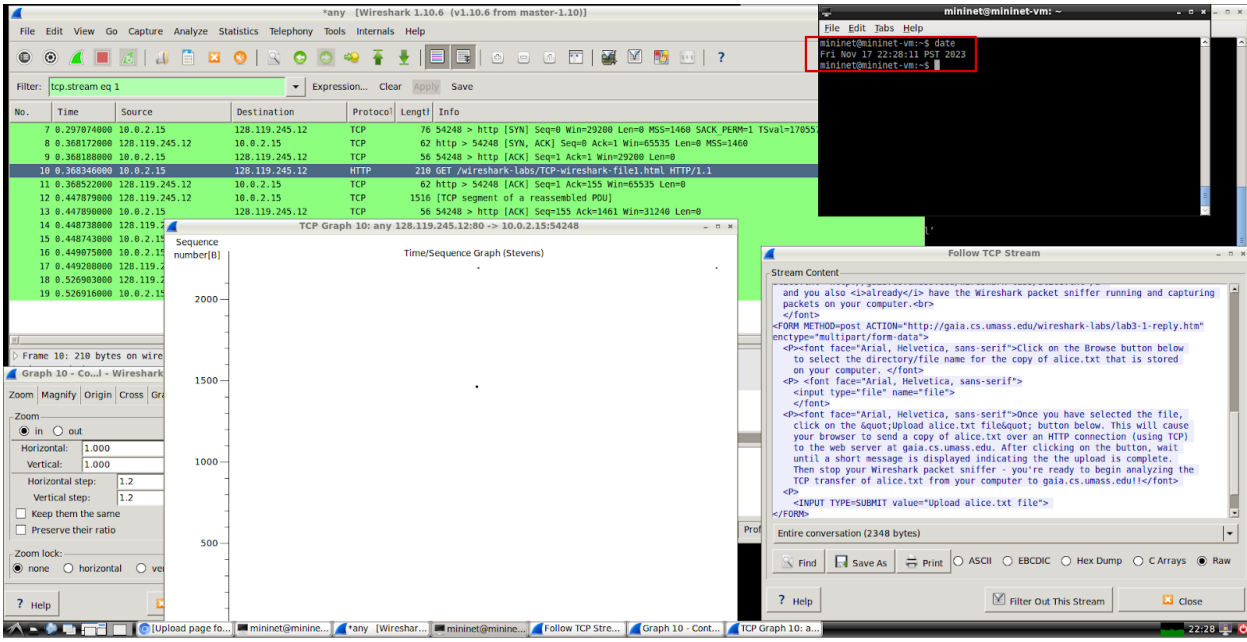
<u>B. Downloading a File from a Server (with packet loss) Environment: Use the VIRTUAL MACHINE for these problems.</u>

In this section we will examine how loss affects a TCP connection. Loss will be simulated with the tc qdisc command to apply a loss rate to an incoming interface.

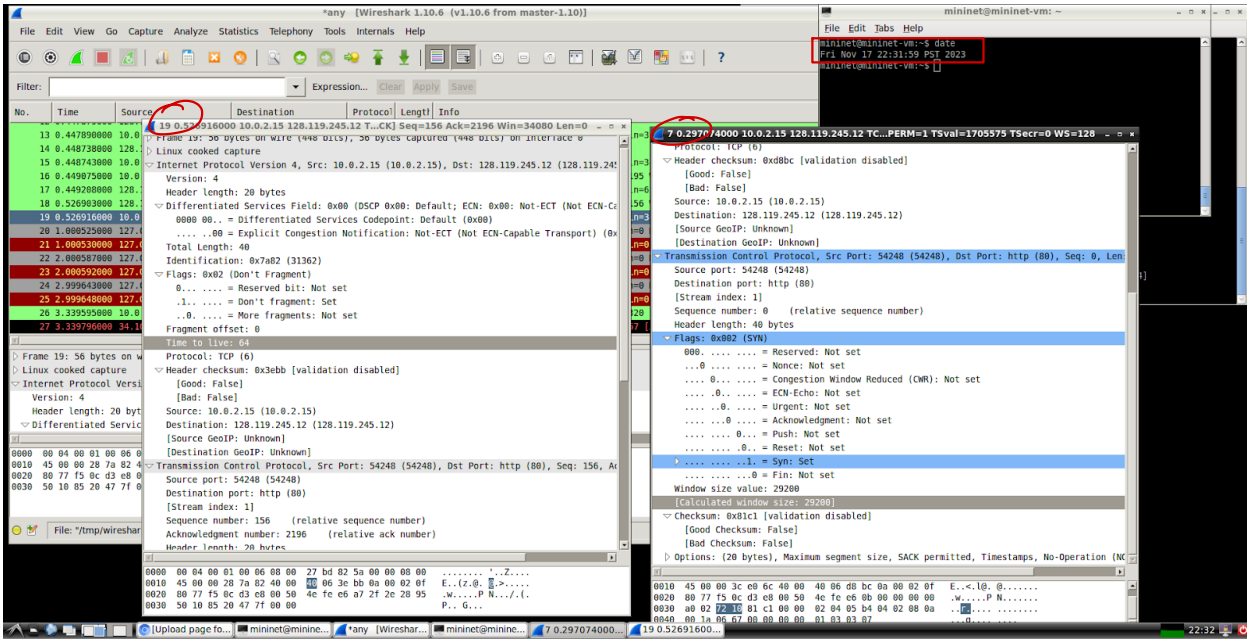
No introduced losses:

12. In this problem we will not intentionally introduce any loss. Start Wireshark and use wget to download this 10MB file.

a. Select a TCP data segment received by your client and create TimeSequenceGraph(Stevens). Include a timestamped screenshot of the graph and highlight important areas of the graph such as transfer begin/end, packet loss (if any), retransmitted packets etc



b. Average throughput: Calculate the average throughput (consider the entire duration until the file transfer completes). Show your calculations and support with your timestamped Wireshark capture with two separate highlights, one highlighting the first packet and one second highlighting the last packet.



Introducing Loss:

Now we will simulate loss by using the command tc qdisc on an interface. When the command is first used, you must use add dev for the interface being changed. After adding the interface, use change dev to set the loss rate.

The following sequence of commands are used to simulate loss on the eth0 interface:

- sudo tc qdisc add dev eth0 root netem loss 0%
- Change loss to 100%

sudo tc qdisc change dev eth0 root netem loss 100%

● Change loss back to 0%

sudo tc qdisc change dev eth0 root netem loss 0%

100% loss event:

Read through this paragraph before starting:

● First start Wireshark

● Then open 2 terminals and have these commands typed and ready before you begin: ○ In one terminal, download the 10MB.zip file using wget

○ While the download is in progress, change loss to 100%. After a second, change loss back to 0%.

13. Answer the following questions found in the Wireshark trace. Take a screenshot and highlight the requested information below:
    Find a TCP data segment received by your client and create a TimeSequenceGraph(Stevens) graph with this packet selected.
    a. At what time in your graph does packet loss begin? At what time does it end? Attach a timestamped screenshot of the graph and circle in red the region where 100% loss begins and ends in the graph.
    b. Search for an area of the graph where there is a retransmission. What Sequence Number is retransmitted? Identify the retransmitted packet (label it) in your screenshot.
    c. Calculate the average throughput (consider the entire duration until the file transfer completes). Show your calculations and support with your timestamped Wireshark capture by circling in red the first and last packet used for your calculations.

20% loss event:

14. Restart Wireshark and retry the experiment, this time with loss 20% for a second and then return to loss 0%. Find a TCP data segment received by your client and create a TimeSequenceGraph(Stevens) graph with this packet selected. Note: you can use the Zoom function to zoom in on an area of the graph to understand what is happening.

    Attach a screenshot of the trace and circle in red the periods of loss.
    a. Mark a few times when packet loss begins and ends. At what time does the region which is experiencing 20% loss end? Attach a timestamped screenshot and circle in red the region where 20% loss begins and ends in the graph.
    b. Examine the areas in the graph at the beginning of the data transfer and after the period of loss. What do you notice about these regions in the graph?
    c. Search for an area of the graph where there is a retransmission. What Sequence Number is retransmitted? Identify the retransmitted packet (label it) in your screenshot.
    d. Calculate the average throughput (consider the entire duration until the file transfer completes). Show your calculations and support with your timestamped Wireshark capture by circling in red the first and last packet used for your calculations

Comparison of TCP Performance:

15. Make a table comparing the loss rates in the exercises above, your throughput calculation and the total time it took to download the 10MB file.
    Compare your throughput results for no introduced loss, 100% loss and 20% loss.
    Do your results make sense to you? What does the difference in throughput tell you about the operation of TCP during periods of congestion or lossy links?

[5 pts] Extra Credit Questions:

UDP Performance:

16. Assume UDP was used for this packet transfer.
    a. What is a fundamental difference between UDP segment transmission compared to TCP?
    b. How does the UDP sender behave if there is a packet loss during the file transfer?
    c. From the UDP sender's perspective, does time to deliver the file change when there is packet loss? Explain your reasoning.
    d. What does the UDP receiver do about packet loss?