Laboratory: 1

Traffic Sniffing

Introduction

This report includes some basic methods used in Wireshark to capture and display desired packets that are transmitted over the network. For this, we set up an environment as described below and generate traffic over the network from which we sniff for the packets.

-Wireshark is a GUI tool used for packet sniffing over the network. Other similar tools are tcpdump, Capsa, Ettercap.

-Data transmitted over the network is broken down and sent in small units known as packets which reassembles at the receiver end.

Setup and Process

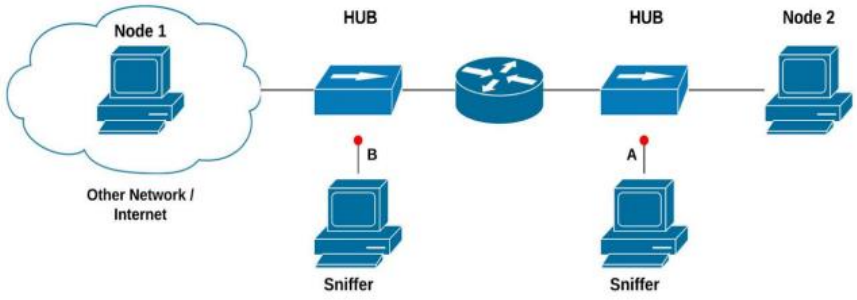

Figure 1: Network set-up

Here, as shown above, node 1 (192.168.10.10) is machine from lab in clemson and node 2 (192.168.20.13) is machine from lab in charleston. All the machines are connected over the HUB and the red dots are the sniffing points (192.168.10.9).

We login to sniffer machine using ssh and open wireshark and start the capture with enp2s0 interface. It immidiately starts to capture packets over the wire, as you can see in the screenshot below. The screenshot below displays all the packets without any filter.

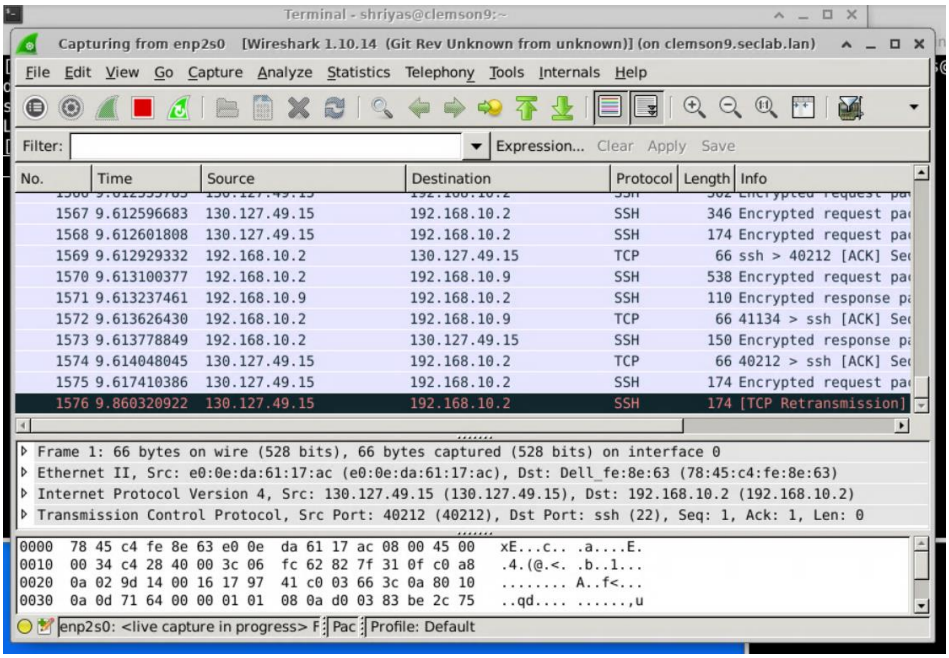

Figure 2: Packets without filter

Now, we try generating traffic over the nodes, we login to node 1 with “ssh node1” along with other credentials and use ping command (ping 192.168.20.13) to generate traffic to node2 in charleston and sniff the communication. To sniff these specific packets over the wire we can use display filter as “ICMP” since the protocol used by the ping command is ICMP. Using it as the filter in wireshark will display only that specific type of packets with ICMP as the protocol as shown in the screenshot below. The same can be used to even display packets with other specific type of protocols such as TCP, SSH etc.

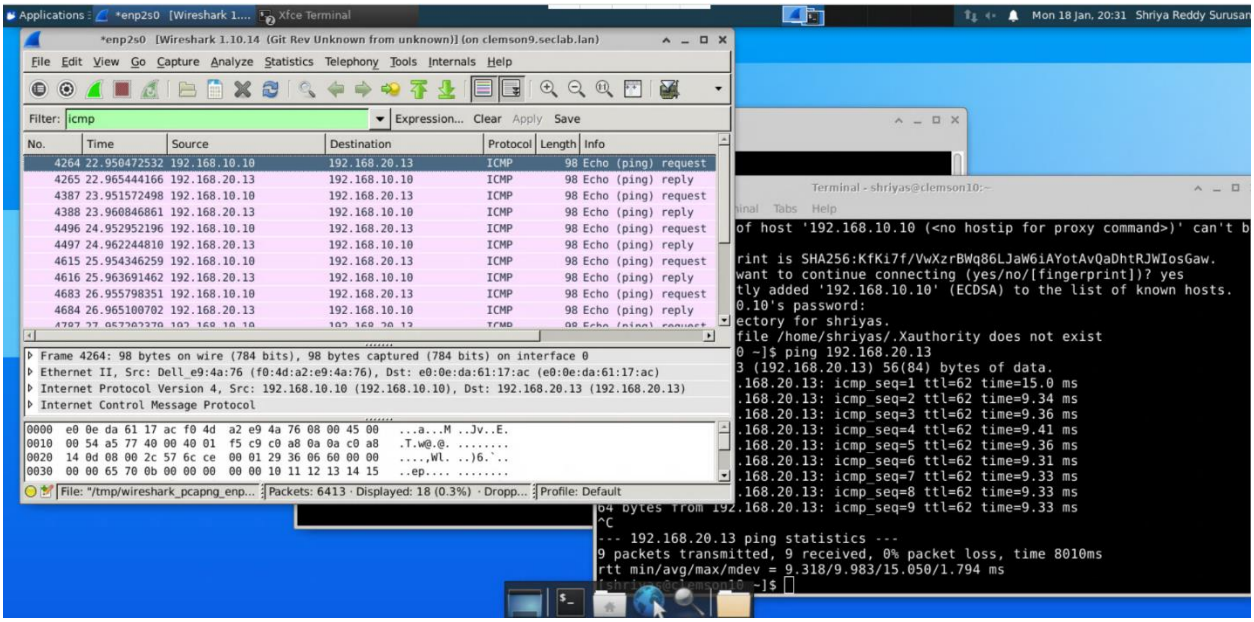

Figure 3: Packets with ICMP protocol filter

Once we could sniff the packets generated by ping, we try to generate more traffic over the network by tranferring a file over 100 MB ~ 500 MB between node1 and node2 . This is done using the following command: $ *scp <file name> <remote_username>@192.168.20.13:/remote/directory*

In my case, the command is $ *scp file.txt shriyas@192.168.20.13:/home/shriyas*

Once the file is transferred, we can see the packets coming from node1 to node2 can be seen using the filter *ip.addr==192.168.10.10 and ip.addr==192.168.20.13* (ip.addr displays the packets coming to and from that specified IP). The screenshot below shows the communication between node1 and node2 with this display filter.

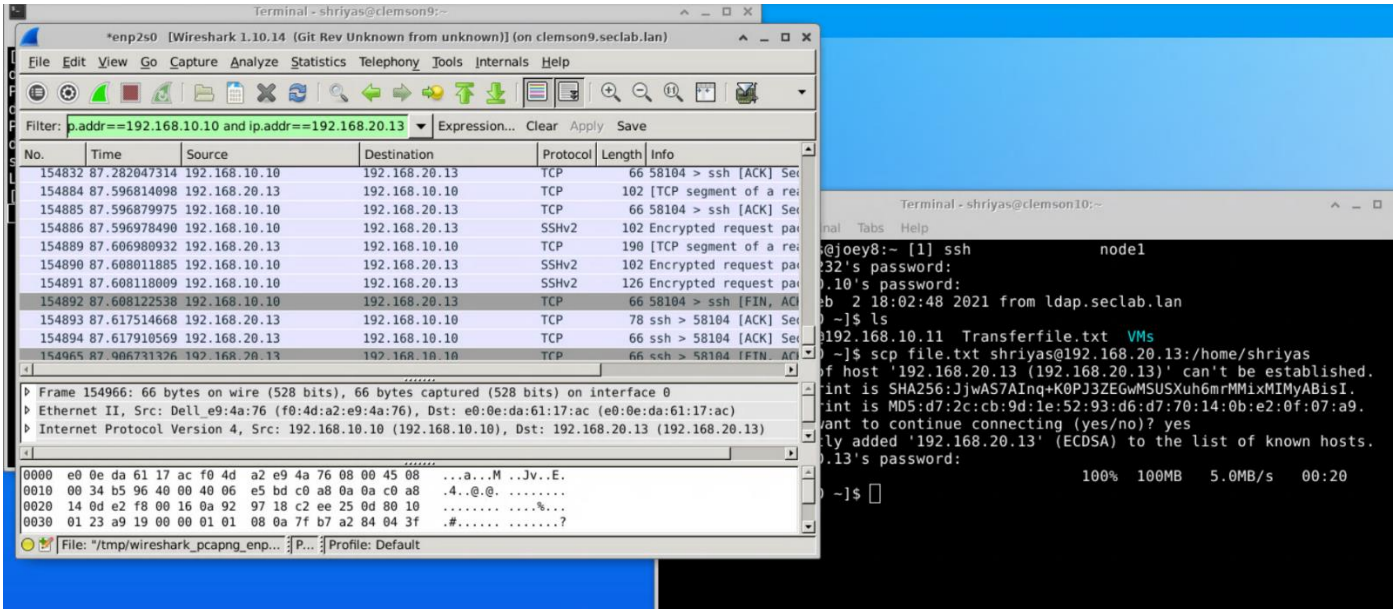

Figure 4: Displaying packets transferred between two hosts using ip.addr filter

After the file transfer, or just after generating heavy traffic over the network. We can observe that a few packets over the network are dropped. We can also find the exact number of dropped packets since wireshark displays the dropped packet count on the bottom row once the capture is stopped (you can see in the screenshot below, the drop count for my capture is 91). We can also use the display filter: *tcp.analysis.lost_segment* to detect lost segments over the network (As shown in the screenshot below). Few reasons why this drop of packets happen is, due to lack of buffer space or because wireshark tool that I used for my capture has limited functionality or just because of the errors in data transmission.

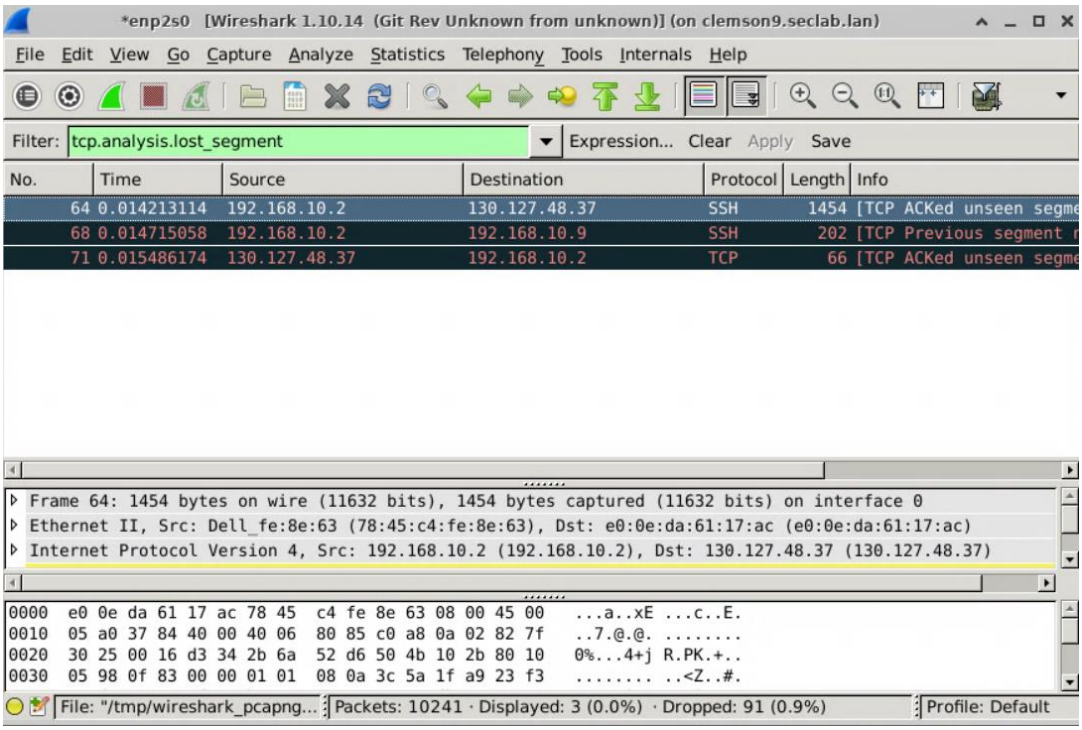

Figure 5: Display filter to detect lost segments

This capture is now saved as pcap file to analyse number of packets generated in the time period within the network for when we transferred a 100MB file over the network from node1 to node2. Time-series graph for captured packets during the the transfer is shown below.

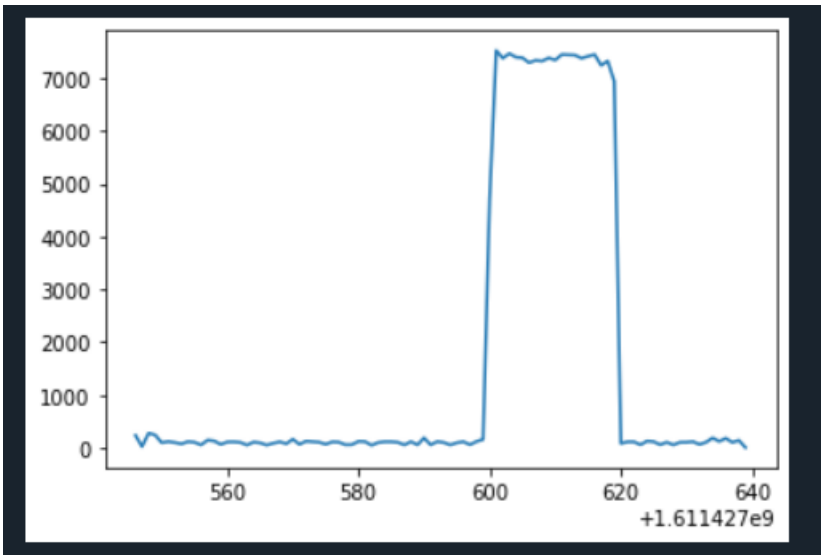

Figure 6: Time-series graph when 100MB file is transferred over the nodes

Below is the time-series graph for when we used ping command to generate traffic over the network.

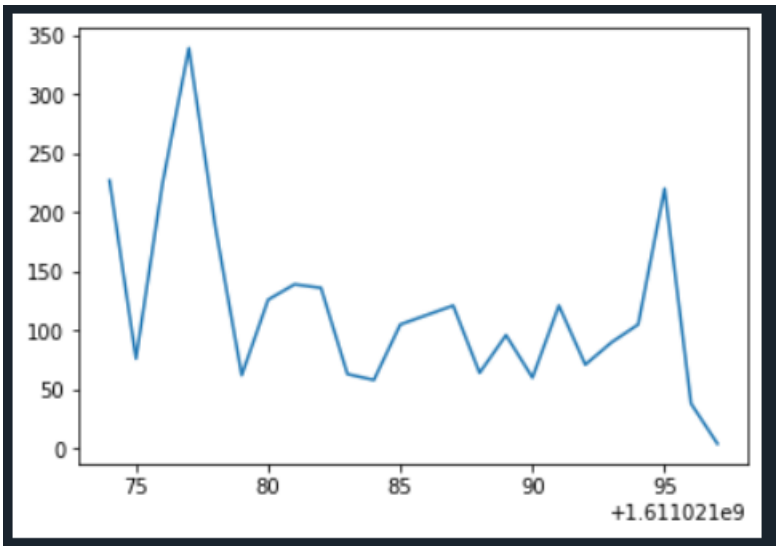

Figure 7: Time-series graph when ping 192.168.20.13 was used

Questions

Questions 1. Describe the layers and the fields of a packet captured from your network.

Layers and fields of the TCP/IP packet captured are as follows:

i). Data layer (Application layer): Consists of Upper layer information.

ii). TCP layer (Transport layer): TCP body consists of data passed by Data layer. TCP header contains the following fields:

-Source port: 2-byte number representing which client process the packet is coming from.

-Destination port: 2-byte number representing which client/server process the packet is going to.

-Sequence number: Contains the number assigned to the first bytes of data of the message.

-Acknowledgement number: Contains the next sequence number that the sender is expected to receive.

-DO: Data offset, Stores the header length information.

-RSV: 6-bit field for future use.

-Flags: Contains flags such as URG, ACK, PSH, RST, SYN, FIN.

-Window: Contains information regarding the buffer space available for sender.

-Checksum: Use to check if the header got corrupted in the transfer.

-Urgent Pointer: Points to the first urgent data byte

-Options: Has variable size. Contains TCP options.

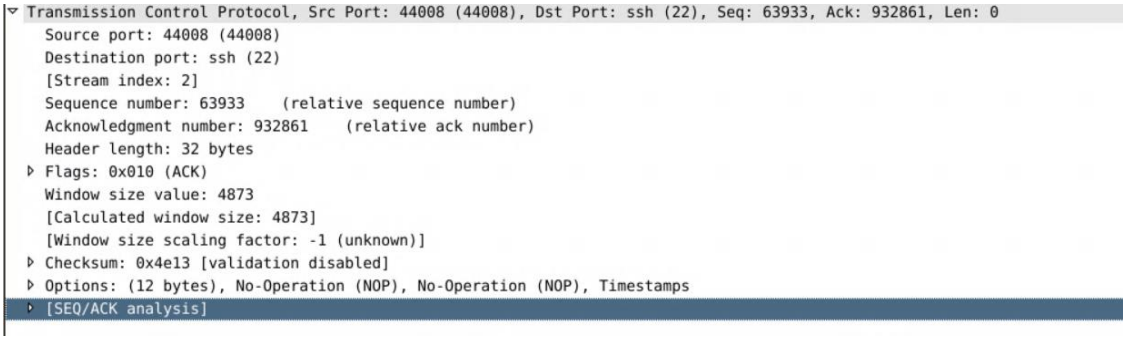

Figure 8: TCP Header fields

iii). IP layer (Internet layer): IP body consists of data passed by TCP layer. IP header contains the following fields:

-Version: Has the current IP protocol version that's being used.

-Header length: Contains length of the IP header

-Identification: Contains unique identity for datagrams which is used to reassemble the fragmented IP datagrams.

-Fragment offset: Also used to reassemble the fragmented IP datagrams.

-Padding: Makes sure the length of the IP header is multiple of 32-bits.

-Options: Lists of options that are active are represented

-Service type: It represents the type of service i.e., minimize delay/maximize throughput/maximize reliability/minimize monitory cost.

- Source address: Stores IP address of the sending machine. E.g.: 192.168.10.2

- Destination address: Stores IP address of receiving machine. E.g.: 192.168.10.9

- Protocol: Recognizes if the IP data is a TCP/UDP or any other protocol packet. (In this case it is a TCP packet)

- Flags: Used for splitting large messages into smaller packets.

- Total length: Length of the packet. Header+body.

- Header checksum: For checking if the header is damaged in the transmission.

- TTL: Time to live. Once the threshold is reached, the packet is discarded.

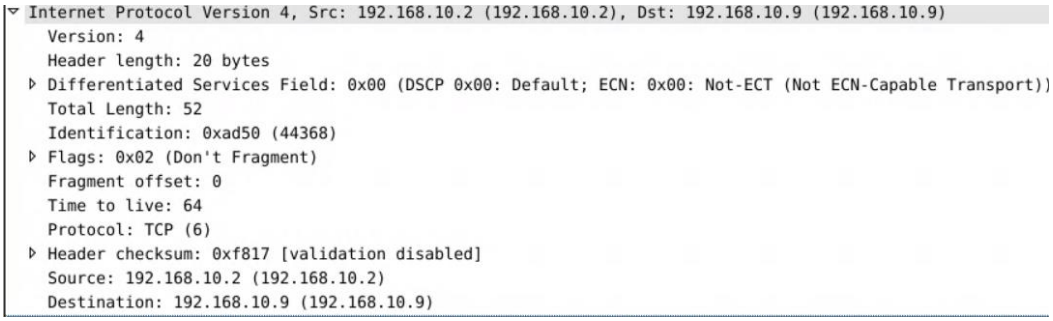

Figure 9: IP header fields

iv). Ethernet layer (Network layer): It contains information about the source machine where the packet has arrived from, the type of packet it is (i.e., IP, DECNET etc.) and also the information regarding the destination machine in the header. The body part consists of the data passed by IP layer.

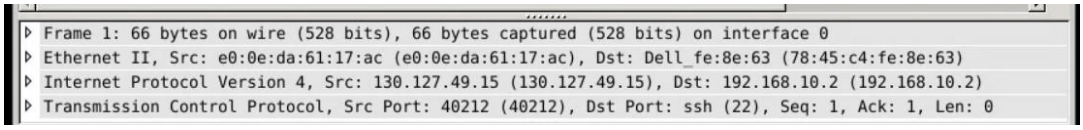

Figure 10: TCP/IP Packet layers

2. Explain the characteristics and functioning of a hub, switch and router in network science.

HUB – Hub is one of the type of devices used to connect multiple computers, network devices with each other. It can also be used to connect computers to a different network. Connection is made by plugging the cables to the port. Here, any message transmitted through hub reaches all the devices connected to it, irrespective of the destination of the message sent. It is upon the devices to check if it was intended for them. Similarly, the response sent back by the device also reaches all the other ports on the hub. It is the quickest and easiest way for setting up a small network connection. Comparatively, hub is the cheapest and least complicated than switch and router.

SWITCH – Switch is an updated type of HUB. It is more efficient and does everything that a hub does. Such as, connecting multiple computers, network devices with each other. However, switch notes the port where the incoming message is coming from initially such that when there is a response it knows which connection to send it to and doesn't parse it to every port connected to it. Similarly, once the incoming message is sent to all the connections initially. It notes which device is accepting the message and the subsequent messages that are intended for it are directly sent to the port the device is connected to, instead of sending it to all the ports in the switch.

ROUTER – Router is by far the most complicated and smartest of the three. Each device is assigned a dynamic IP address when it is first connected to the network. For basic routing, router operates as switch where it learns the location of the incoming traffic and routes it to the intended connection using the IP addresses. It also acts as a firewall blocking outside access since anything from the internet cannot initiate communication with local machines. It can only respond to the connection initiated by the local machine.

3. Why is the use of SDNs as a tool necessary for DDoS?

Software Defined Networking (SDN) is a networking architecture that was designed to be centrally controlled using software applications such that the entire network can be managed constantly irrespective of the underlying technology. Though the data plane and the application plane may seem vulnerable for DDoS attack, SDN integrated with Machine learning application is one way that can be used to defeat DDoS attacks. It leads to a self-determining network that is able to learn and act, which can also implement network monitoring practice and be prepared with a DDoS response battle plan due to the centralized nature of the controller. Weighing both pros and cons of SDN, the practical security solutions of SDNs over power by providing flexible, agile and strong defense mechanisms against DDoS. Thus, being necessary.

References

[1]: https://www.cs.ait.ac.th/~on/O/oreilly/tcpip/firewall/ch06_03.htm

[2]: https://askleo.com/whats_the_difference_between_a_hub_a_switch_and_a_router

[3]: https://www.researchgate.net/post/How-to-detection-and-prevention-DDoS-attacks-in-software-defined-networking