LABORATORY MANUAL

CE3005: Computer Networks
CZ3006: Netcentric Computing

No. 3: Understanding Network Operations and Encapsulation by Sniffing and Analysing Network Packets

SEMESTER 1
2021-2022

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

SNIFFING AND ANALYSING NETWORK PACKETS

1.   OBJECTIVE

To further understand how the Internet really works and how the concept of encapsulation is being implemented in the different layers of the TCP/IP protocol suite.

2.   LABORATORY

CE3005-Hardware Laboratory I (N4-1a-3),
CZ3006–Software Laboratory (N4-01C-06)

3.   EQUIPMENT

PC with Windows OS & Wireshark Network Analyzer.

4.   DURATION

1 hour 50 Minutes.

5.   EXERCISE 3A: PACKET SNIFFING

Network protocol analyzer, also known as packet sniffer, is a useful tool for learning about networking. An example of a good open source network protocol analyzer is **Wireshark** (downloadable freely from *http://www.wireshark.org*).

Double-click on the   Wireshark icon from the Windows OS desktop to launch the Wireshark Network Analyzer.

To start capturing network packets, click on the menu Capture → Options…
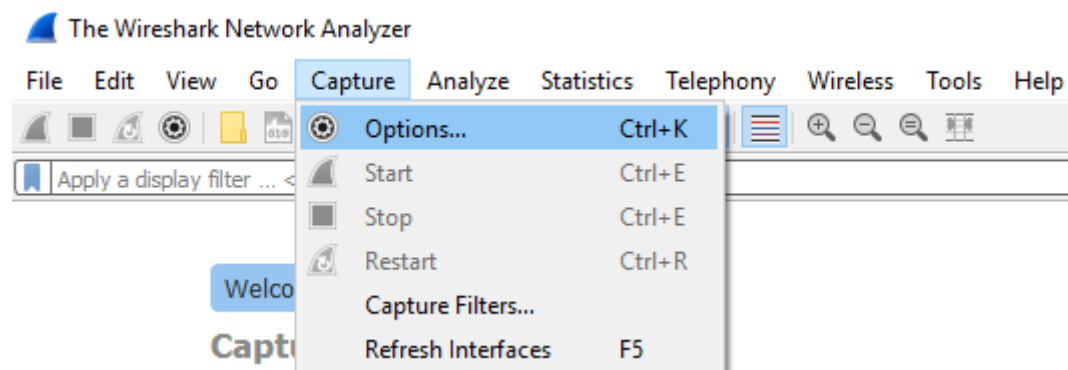
Wireshark will pop up the Capture Options dialog box shown in Figure 3.1. Select the Ethernet interface, and ensure it is highlighted in blue. To view the IP address assigned to the interface, click on the small arrow next to the interface name
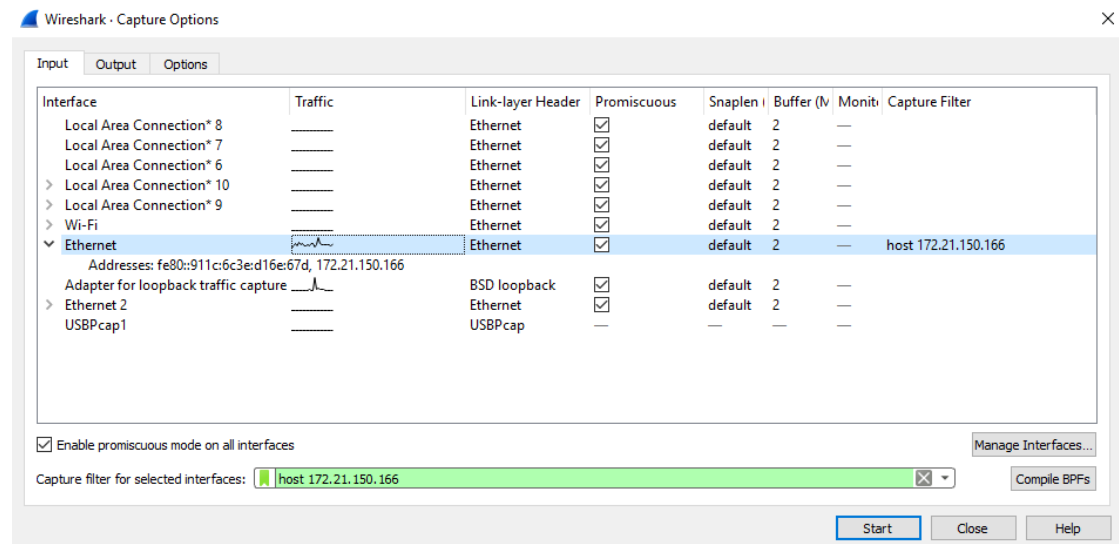

Figure 3.1 : Dialog box of Wireshark capture options

To ensure that you will have a complete picture of how the Internet works, do the followings before starting capturing:

1. Check the "Promiscuous" checkbox of the selected interface
2. Input at the "Capture Filter for selected interface" the following:
    host <IP address of your PC>
3. Clear your DNS cache. ("**ipconfig /flushdns**".)
4. Clear your ARP cache. ("***arp –d "***)
5. Get ready your Rfc865UdpClient program developed in Laboratory 2.

When ready, click the Start button to commence live capturing of network packets.

Next, similar as Laboratory 2, run your Rfc865UdpClient to request a quote of the day from the given RFC 865 UDP server, ie QoD server.

When your client received the quote from the QoD server, click the Stop button to end capturing.

Wireshark will now display all the captured network packets in three windows, or panes:

- The top pane is the packet list pane. It displays all the packets captured. By clicking on a packet in this pane, you control what is displayed in the other two panes.
- The middle pane is the tree view pane. It displays the packet selected in the top pane in more details, ie. the content of the IP and TCP fields.
- The bottom pane is the data view pane. It displays the data from the packet selected in the top pane, and highlights the field selected in the tree view pane.

Recall the networking concepts learnt in the lectures. Before your laboratory PC can actually send the quote of the day request from your Rfc865UdpClient to the given RFC 865 UDP server, additional network packets need to be sent and received.

Search the top pane of Wireshark display and list in the following table the sequence of all relevant network packets sent and received by your laboratory PC. (You shoud

be able to see the DNS, ARP and QoD request and reply. The last relevant network packet is the quote of the day packet received from the given RFC 865 UDP server as shown.). NB! To could possible set the filter in the result page to look at specific packets, eg. Setting it to DNS will display DNS packet only.

Packet	Source MAC	Source IP	Dest. MAC	Dest. IP	Purpose of Packet
1.					DNS request
2.					
...					
Last.	RFC865 UDP server		My Rfc865UdpClient		Quote of the day reply

6.   EXERCISE 3B: DATA ENCAPSULATION

TCP/IP is loosely characterised as consisting of 5 layers. For each layer, standard protocols are developed for the exchanges of messages. These messages are called protocol data units, or PDUs.

How can an application-PDU be transmitted across a physical network that does not understand the application-PDU at all? The answer lies in a technique known as **encapsulation**, where a lower level protocol accepts a PDU from a higher level protocol and places it in the data portion of the lower layer PDU.

Let us say that your Rfc865UdpClient is requesting a quote of the day from the given RFC 865 UDP server. The request data is an application-PDU, which is handled over to the transport layer via an API. The arguments for this API includes the destination computer name or IP address, the port number and the data. The transport layer appends the transport-header (TH) to the data to create a transport-PDU. This is then handled down to the network layer, which in turns creates a network-header (NH) with the transport-PDU as the data to form a network-PDU. Similarly, the data link layer appends a header (DH) and a trailer (DT) with network-PDU as the data to form a data-link-PDU. Then, this data-link-PDU is the actual frame that is being transmitted over the physical media.
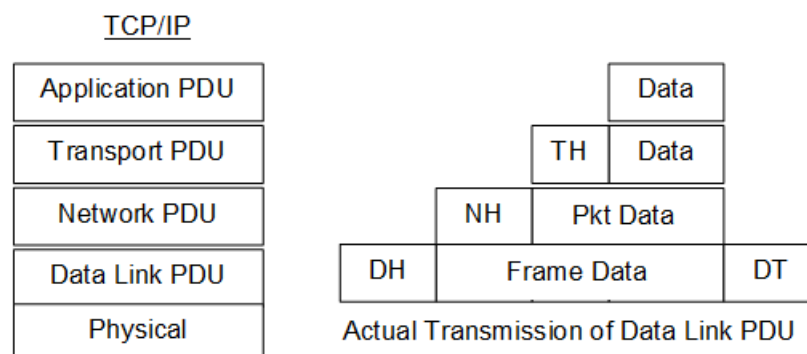

Figure 3.2: Encapsulation of PDUs

To fully understand the concept of data encapsulation between layers, let us analyse the following network packet:

At the top pane of Wireshark display, search for the network packet sent by your **Rfc865UdpClient to the given RFC 865 UDP server.** Select it, and copy the corresponding hexadecimal representation of the complete captured data below:

Complete captured data (in hexadecimal):

7. EXERCISE 3C: DATA LINK PDU - ETHERNET FRAME

Several variations of Ethernet currently exist. The widely used industry standard is Ethernet II, which was defined by the Ethernet specification created by Digital, Intel, and Xerox before the IEEE 802.3 specification. The Ethernet II frame format is also known as the DIX (Digital Intel Xerox) frame format. It is of variable length, with no frame smaller than 64 bytes.

Preamble	Dest Addr	Source Addr	Ether Protocol Type	Frame Data	CRC
8	6	6	2	46-1500	4

Figure 3.3: Ethernet Frame

The Preamble field is 64-bit long and consists of 56 bits of alternating 1s and 0s (each byte is the bit sequence 10101010) to synchronize a receiving station and a last 8-bit of 10101011 sequence that indicates the start of a frame. Note that this field is discarded at the receiving end and hence not shown in Wireshark.

The Cyclic Redundancy Check (CRC) field is 32-bit long and provides bit-level integrity verification on the bits in the Ethernet II frame. The 32-bit CRC helps the interface detect transmission errors: the sender computes the CRC as a function of the data in the frame, and the receiver recomputes the CRC to verify that the packet has been received intact. Note that this field is also not shown in Wireshark.

What type of upper layer data is the captured ethernet frame carrying? How do you know?

Determine the following from your captured data in Exercise 3B:

Destination Address   :

Source Address   :

Protocol Type:

Frame Data   :

8. EXERCISE 3D: NETWORK PDU - IP DATAGRAM

The IP datagram format is shown in Figure 3.4.

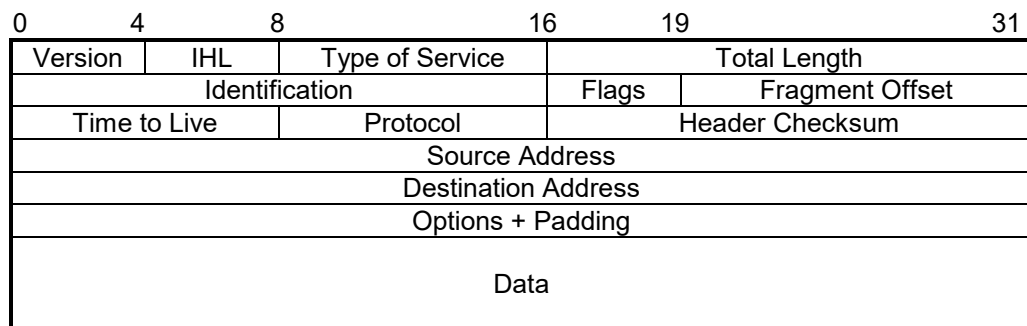| 0 4	8	16 19	31
Version	IHL	Type of Service	Total Length		
Identification			Flags	Fragment Offset	
Time to Live		Protocol	Header Checksum		
Source Address					
Destination Address					
Options + Padding					
Data					

Figure 3.4: IP Datagram

Refer to lecture slides or RFC 791 for an explaination of the fields.

What type of upper layer data is the captured IP packet carrying? How do you know?

Does the captured IP header have the field: Options + Padding? How do you know?

Determine the following from the Frame Data field in Exercise 3C:

Version	:
Total Length	:
Identification	:
Flags	:
Fragment Offset Protocol:	:
Source Address	:
Destination Address	:
Packet Data	:

9.   **EXERCISE 3E: TRANSPORT PDU - UDP DATAGRAM**

The UDP datagram packet is shown in Figure 3.5.

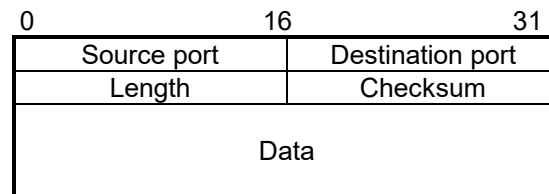| 0	16	31
Source port		Destination port
Length		Checksum
Data		

Figure 3.5: UDP Datagram

Determine the following from the Packet Data field in Exercise 3D:

Source Port   :

Destination Port   :

Length   :

Data   :

10.   **EXERCISE 3F: APPLICATION PDU**

Interpret the application data from the Data field in Exercise 3E:

Message   :

Is this the message that you have sent?