Introduction to Wireshark

We have created lab assignments for several chapters of the textbook. We have no lab assignments for physical layer. We cannot use a standard packet sniffer, such as Wireshark, to capture bits. We cannot sniff management packets because we have normally no permission to act as a manager. In this document, we give an introduction to packet sniffing, introduce the Wireshark software.

1.1 PACKET SNIFFING

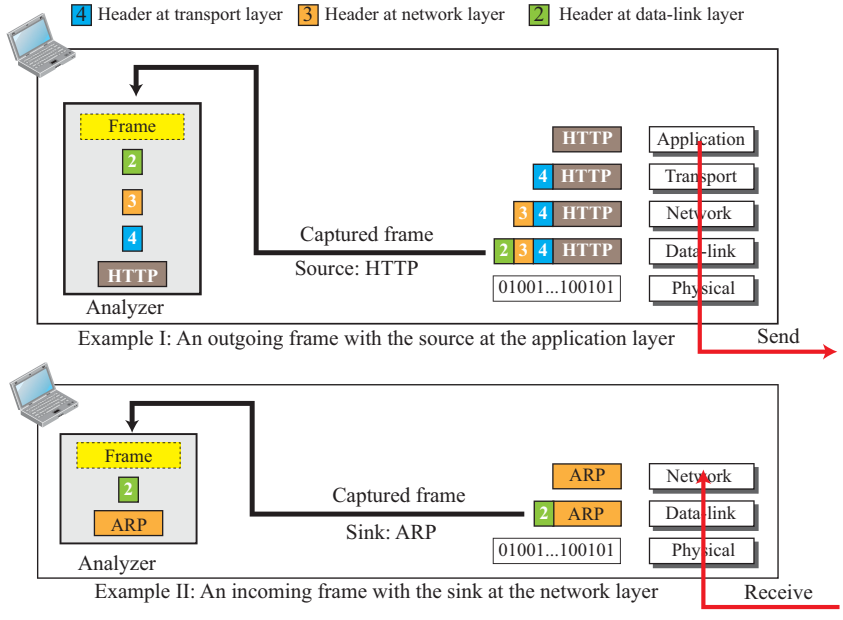
The purpose of lab assignments is to show how we can get a deeper understanding of the networking concepts by capturing and analyzing the packets sent and received from our host. One way to do so is to use a **packet sniffer**. A packet sniffer is a piece of software that should be running in parallel with the application whose packets needed to be analyzed. However, before running a packet sniffer, we need to interpret the term *packet*. As we discussed in Chapter 1 of the textbook, communication via the Internet is done using a five-layer suite. We can analyze the packets at four layers: application, transport, network, and data-link. There is no packet exchange at the physical layer; communication at this layer is done using bits.

Although it is useful to analyze the packets in each of the four upper layers of the TCP/IP protocol suite, should a packet sniffer software be designed to capture packets at each of these layers? The answer to this question can be found in Figure 2.8 in the textbook (encapsulation-decapsulation). In an outgoing situation, a packet created at any upper-layer is encapsulated in a *frame* (at the data-link layer); in an incoming situation, a packet intended for any layer is decapsulated from the received frame. This means we need to capture only outgoing or incoming frames; a packet-sniffer software can extract the packets at any layer desired to be analyzed from these frames. For this reason, a packet-sniffer software is normally has two components: a **packet-capturer** and a **packet-analyzer.** The packet-capturer captures a copy of all outgoing and incoming frames (at the data-link layer) and passes them to the packet-analyzer. The packet-analyzer can then extract different headers and the ultimate message for analysis.

Before we continue with our discussion, we need to make a point clear. Although Figure 2.8 in the textbook shows that the encapsulation starts or decapsulation ends at the application layer, a packet in the Internet can belong to any layer above the data-link layer. As we will see in future chapters, protocols at the transport or network layer

protocols also need to exchange packets. All of these packets are encapsulated in or decapsulated from the frames. A packet sniffer needs to capture all incoming and outgoing frames and show the headers of all protocols used for communication. The source or the sink of a packet is not necessarily the application layer. Figure 1.1 shows two examples.

Figure 1.1 *Role of frame capturing and packet analyzing in a packet-sniffer*



In Example I, an outgoing frames is captured. The source of the frame is the HTTP protocol at the application layer (discussed in Chapter 26 of the textbook). A copy of the frame is passed to the analyzer. The analyzer extracts the general information in the frame (the box marked frame), headers 2, 3, and 4, and the HTTP message for analysis. In Example II, an incoming frame is captured. The sink (final destination) is the ARP protocol at the network layer (discussed in Chapter 9 in the textbook). A copy of the frame is passed to the analyzer. The analyzer extracts the general information in the header (the box marked frame), header 2 and the ARP message for analysis.

1.2 WIRESHARK

In this and other lab assignments, we use a packet-sniffer called **Wireshark**. Wireshark (formerly known as ETHEREAL) is a free packet sniffer/analyzer which is available for both UNIX-like (Unix, Linux, Mac OS X, BSD, and Solaris) and Windows operating systems. It captures packets from a network interface and displays them with

detailed protocol information. Wireshark, however, is a passive analyzer. It only captures packets without manipulate them; it neither sends packets to the network nor does other active operations. Wireshark is not an intrusion-detection tool either. It does not give warning about any network intrusion. It, nevertheless, can help network administrators to figure out what is going on inside a network and to troubleshoot network problems. In addition of being an indispensable tool for network administrators, Wireshark is a valuable tool for protocol developers, who may use it to debug protocol implementations. It is also a great educational tool for computer-network students who can use it to see details of protocol operations in real time.

1.2.1 Downloading and Installing

To download the Wireshark software, connect to the Internet using the website:

http://www.wireshark.org/download.html

After the downloading is complete, install the software on your computer. If you have any problem in downloading or installing, you can consult the following site for more information:

http://wiki.wireshark.org/CaptureSetup

1.2.2 Main Window

The Wireshark main window is similar to other GUI tools as shown in Figure 1.2.

The Wireshark window is made of seven sections: **title bar**, **menu bar**, **filter bar**, **packet list pane**, **packet detail pane**, **packet byte pane**, and **status bar**. We briefly discuss the functionality of each section below:

Title Bar

The *title bar* (like the one in any GUI) shows the title of the window, the closing, maximizing, and minimizing icons.

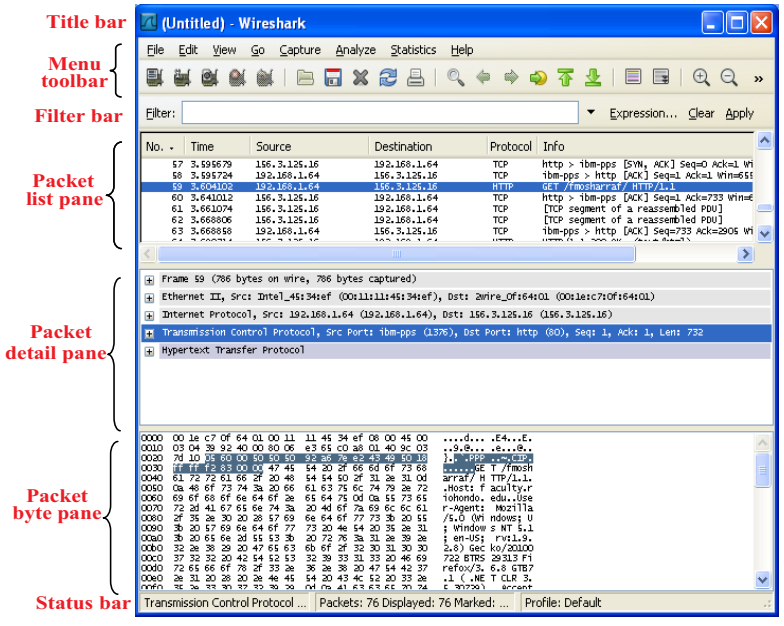
Menu Bar

The *menu bar* is made of several pulldown menus and tool bars used in most GUIs. We will using some of these menus in our lab assignments. We can use the **File** menu to perform some actions on the file itself such as saving and printing. The **Capture** menu is used to start and capturing frames. The **View** menu is useful to show or hide some of the sections in the window.

Filter Bar

The *filter bar* allows us to display packet we are interested in while hiding the rest. As we see later in this document, when we start capturing frames, Wireshark captures and analyze any outgoing and incoming frame no matter what is the source or sink protocol. Sometimes, this is not what we want. We may want to limit the analysis to a specific source or sink protocol. For example, we may want to analyze only packets sent or

Figure 1.2 *Main window of Wireshark*



receive by the HTTP protocol at the application layer or the ARP protocol at the network layer. This is called filtering in the parlance of packet sniffing. After packets have been captured, we can type the name of the protocol in lowercase and click **Apply**.

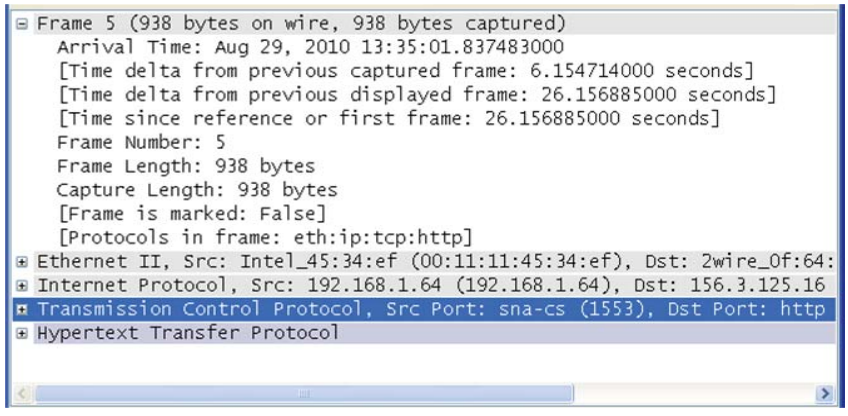
Packet List Pane

The *packet list pane* displays a one-line summary for each captured packet (actually frame). The summary includes the packet (frame) number (added by the Wireshark and not part of the packet), the time when the packet was captured, the source and destination IP addresses of the packet (at the network layer), the packet source or sink protocol, and the additional information about the packet contents. In other words, this pane shows the captured frames that will be passed for analyzing to the packet analyzer.

Packet Detail Pane

The *packet detail pane* shows the detailed analysis for each frame (Figure 1.3). The information is limited to one frame, which means we need to select one of the frames in the packet list pane for analysis. This can be done by clicking on the corresponding frame in the packet list pane. Clicking on any frame in the *packet list pane* highlights the frame and shows the details of the frame in the packet details pane. Information exhibited in this pane for each frame is made of a tree structure. However, each top branch of the tree is shown as one line as it is common in GUI trees. We can expand the branch (to see subbranches) by clicking on the plus box at the leftmost part of the line,

Figure 1.3 *Packet detail pane*



which changes the plus sign to a minus sign; the branch can be collapsed again, which changes the minus sign to the plus sign. Note that the analyzer first shows a general information at the data-link layer (frame). It then displays the information contained in each header from the data-link layer (H2) up to the source or sink protocol. It finally shows the whole message at the source or sink layer. Figure 1.3 shows an example of a packet details pane when the frame is expanded. It shows some general information and names of all protocols used in the frame (intermediate and source or sink).

Packet byte pane

The *packet byte pane* shows the entire current frame (selected in the packet list pane) in hexdump format (hexadecimal view of data) and ASCII format. The number in the left field shows the offset in the packet data; the hexdump of the packet is shown in the middle field; the corresponding ASCII characters are shown in the right field. If we need the byte (or ASCII equivalent) of any line in the packet detail pane, we can click on the line in the packet detail pane and the byte contents will be highlighted. Figure 1.4 shows an example of a packet byte pane. It shows all the bytes in the frame, but we can select the bytes in any protocol header by highlighting it in the *packet detail pane* section.

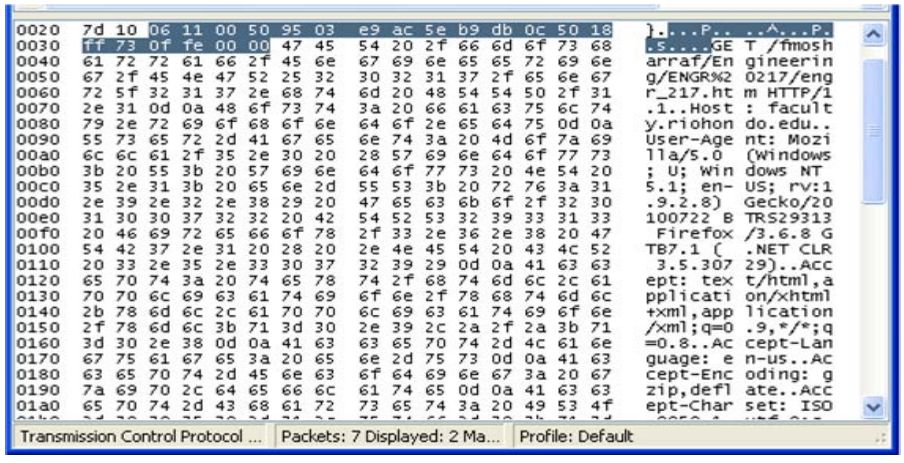
Status Bar

The last section of the window (at the bottom) is the *status bar* which shows the current protocol, the total number of packets captured, and so on.

1.2.3 Working With Wireshark

When we work with Wireshark in this and other labs, there are some actions that we need to repeat over and over. We mention the details of some of this action to avoid re-mentioning them.

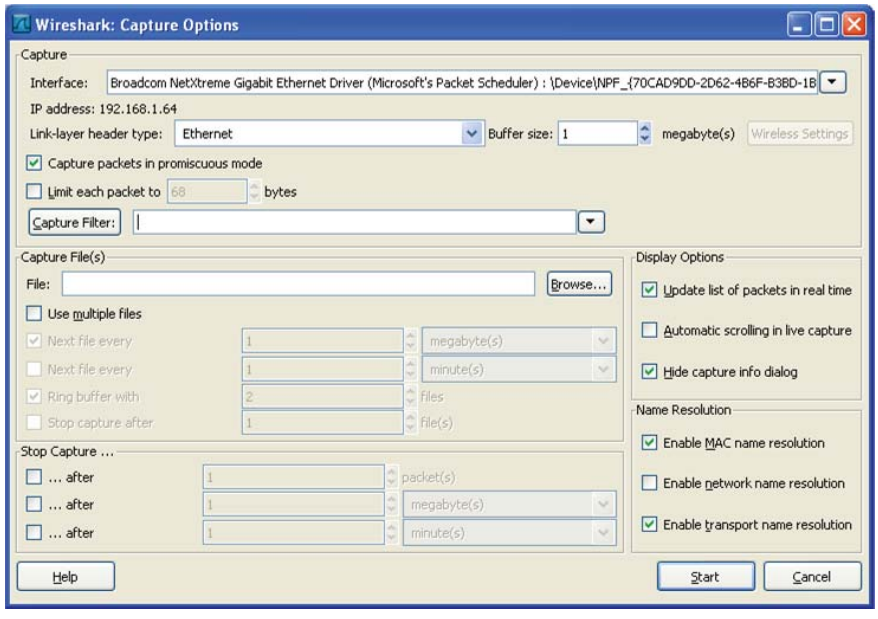
Figure 1.4 *Packet byte pane*



Start Capturing

To begin capturing, select the **Capture** from the pull down menu and click **Options** to open the Wireshark **capture options dialog box** (Figure 1.5).

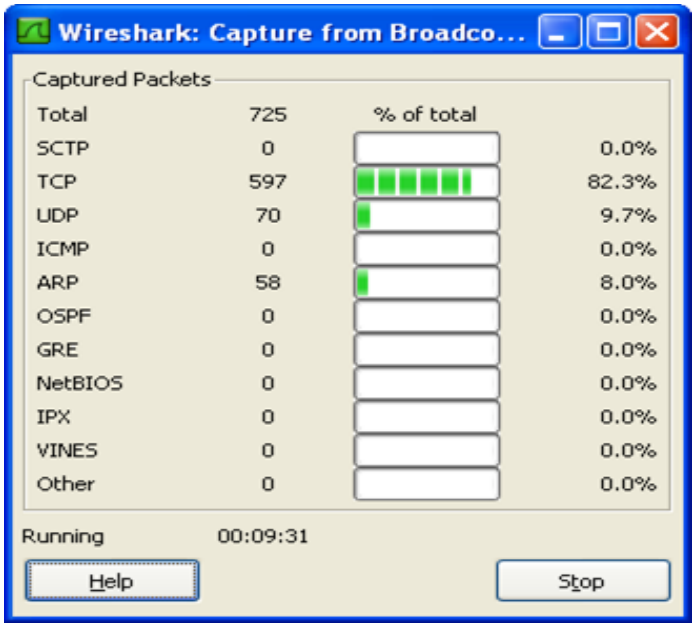
Figure 1.5 *Caption options dialog box*



There are several steps that you need to follow before you start capturing:

- You normally will use the default values in the *capture options dialog box*, but there are some options that you may need to override the default. In particular, you may want to uncheck "Hide capture info dialog." and open capture information window (Figure 1.6).

Figure 1.6 *Capture information dialog*



- The network interfaces are shown in the Interface drop-down list at the top of the dialog box. Select the network interface (or use the default interface chosen by Wireshark). If the IP address in the dialog box is unknown, you must select a different interface; otherwise, the Wireshark will not capture any packet.
- After the above two steps, click Start. Wireshark starts to captures packets that are exchanged between your computer and the network. If, after a minute, Wireshark does not capture any packet, there must be a problem; check for possible reason and troubleshooting.

Stop Capturing

Whenever you feel you have captured all the packets (frames) that you need to do your lab report, you can stop capturing. To do so, you need to use the Capture pulldown menu and click **Stop**. Wireshark stops capturing the frames

Saving the Captured File

After you have stopped capturing, you may want to save the captured file for future use.

1.2.4 Incoming and Outgoing Frames

When we see the list of the captured frames, we often wonder which frames are the incoming and which ones are outgoing. This can be found by looking at the frame in packet list pane. The packet list pane shows the source and destination addresses of the frame (generated and inserted at the network layer). If the source address is the address of the host you are working with (shown on the Capture window when you start capturing), the frame is the outgoing frame; if the destination address is the address of your host, the frame is the incoming frame.