Wireshark Lab 0

Getting Started

One's understanding of network protocols can often be greatly deepened by "seeing protocols in action" and by "playing around with protocols" – observing the sequence of messages exchanged between two protocol entities, delving down into the details of protocol operation, and causing protocols to perform certain actions and then observing these actions and their consequences. You'll be running various network applications in different scenarios using a computer on your desk, at home, or in a lab. You'll observe the network protocols in your computer "in action," interacting and exchanging messages with protocol entities executing elsewhere in the Internet. Thus, you and your computer will be an integral part of these "live" labs. You'll observe, and you'll learn, by doing.

The basic tool for observing the messages exchanged between executing protocol entities is called a **packet sniffer**. A packet sniffer captures ("sniffs") messages being sent/received from/by your computer; it also typically store and display the contents of the various protocol fields in these captured messages. A packet sniffer itself is passive. It observes messages being sent and received by applications and protocols running on your computer, but never sends packets itself. Similarly, received packets are never explicitly addressed to the packet sniffer. Instead, a packet sniffer receives a copy of packets that are sent/received from/by application/protocols running on your machine.

Figure 1 shows the generic structure of a packet sniffer. On the right of the figure are the protocols (in this case, Internet protocols) and applications (such as a web browser or ftp client) that normally run on your computer. The packet sniffer, shown within the dashed rectangle in Figure 1 is an addition to the usual software in your computer, and consists of two parts. The packet capture library receives a copy of every link-layer frame that is sent from or received by your computer. Recall that messages exchanged by higher layer protocols such as HTTP, FTP, TCP, UDP, DNS, or IP all are eventually encapsulated in link-layer frames that are transmitted over physical media such as an Ethernet cable.

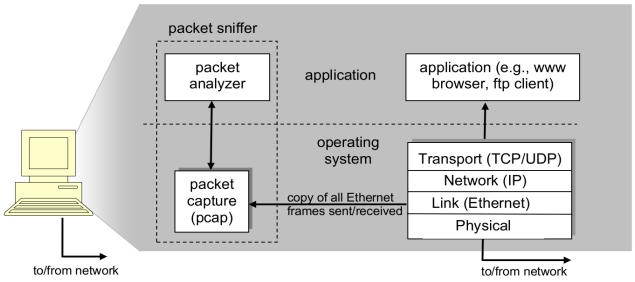


Figure 1

In Figure 1, the assumed physical media is an Ethernet, and so all upper layer protocols are eventually encapsulated within an Ethernet frame. Capturing all link-layer frames thus gives you all messages sent/received from/by all protocols and applications executing in your computer.

The second component of a packet sniffer is the **packet analyzer**, which displays the contents of all fields within a protocol message. In order to do so, the packet analyzer must "understand" the structure of all messages exchanged by protocols. For example, suppose we are interested in displaying the various fields in messages exchanged by HTTP in Fig. 1. The packet analyzer understands the format of Ethernet frames, and so can identify the IP datagram within an Ethernet frame. It understands the IP datagram format, so that it can extract the TCP segment within the IP datagram. It understands the TCP segment structure, so it can extract the HTTP message contained in the TCP segment. Finally, it understands HTTP and so, knows that the first bytes of an HTTP message will contain the string "GET," "POST," or "HEAD," headers.

We will use the Wireshark packet sniffer (www.wireshark.org) for these labs, allowing us to display the contents of messages being sent/received from/by protocols at different levels of the protocol stack. Wireshark is a free network protocol analyzer that runs on Windows, Linux, and Mac computers. It's an ideal packet analyzer for these labs – it is stable, has a large user base and good support with a user guide (www.wireshark.org/docs/wsug_html).

Getting Wireshark

Download Wireshark (Windows/64-bit) off my website from here or else visit the Wireshark home page and download the most compatible version for your computer.
Note: Ensure that you install the WinPcap 4.0 packet capture library if it is not already present on your machine. Please start WinPcap NPF as a service as well.

Starting Wireshark

After installation is complete, start the Wireshark program. You will be greeted by a initial capture screen as shown in Figure 2.

The different interfaces available on your system will be displayed here. It will be obviously be slightly different from the listing in Figure 2.

You will have to click and select the interface through which you are connected to your network. If it is a Laptop PC, it will most probably be by Wi-Fi and if it is a Desktop PC, it will be through one of the Local Area Connections. The little line graph to the right of the interface list will show you which interface is currently active. By default, my Wi-Fi is active and has been selected.

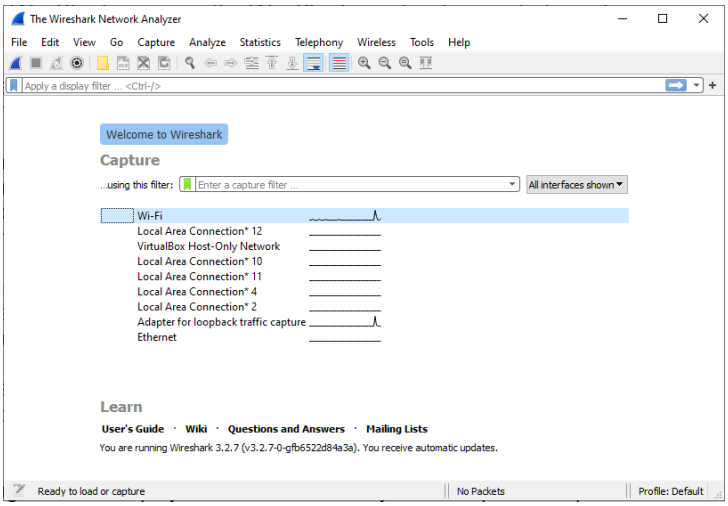


Figure 2

Performing a Capture on Wireshark

After having selected the correct interface, we now have to start a capture. The first four icons on the main toolbar controls the c  ions.

The blue shark fin icon is for starting a live capture. The second square icon is for stopping the current capture. (icon lights up as red when a capture is underway). The third icon is for aborting the current capture and restarting a capture (icon lights up as a green shark fin when a capture is underway. It will prompt you if you want to save the current capture or continue with a new capture without saving) and finally the fourth cogwheel icon is for setting capture options.

Now follow the three steps below to perform your first live capture.

1. Open your favorite browser (if you have not done so already), I would recommend you use Firefox or Chrome because these allow you to clear the browser cache easily. Also, it is recommended to keep network traffic to a minimum so ensure that you are using only one tab in your browser.

2. Enter this URL in your browser (**Do not press ENTER just yet**): **http://pages.intnet.mu/rhh**

3. Start a live capture in Wireshark by clicking on the blue shark fin icon. A new screen will appear in Wireshark. Now go to your browser and press ENTER. The above URL is from my old website hosted at Mauritius Telecom public web server. When the web page has fully loaded, only then, stop the Wireshark capture by clicking on the red square icon.

Analysing a Capture using Wireshark

You will notice that a new interface has appeared when you previously started the live capture which looks like the Figure 3 below:

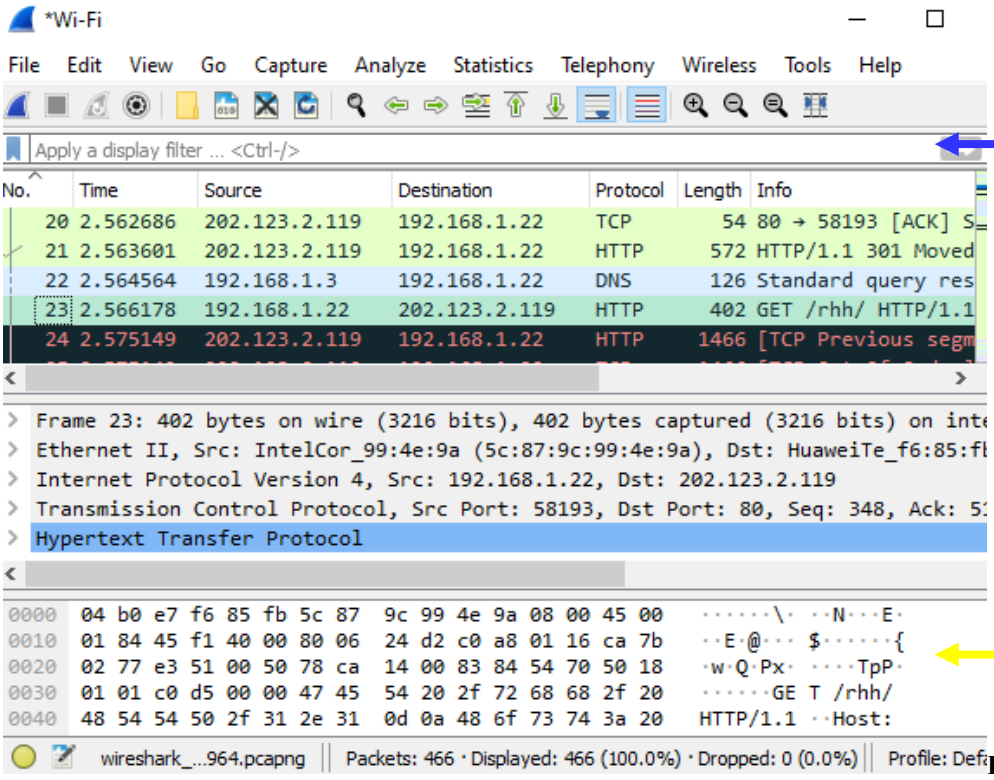


Figure 3

This is the main Wireshark interface. It consists of **four** major components:

The **packet-listing window** displays a one-line summary for each packet captured, including the packet number (assigned by Wireshark), the time at which the packet was captured, the packet's source and destination addresses, the protocol type, and protocol-specific information contained in the packet. The packet listing can be sorted according to any of these categories by clicking on a column name. The protocol type field lists the highest level protocol that sent or received this packet.

The **packet-header details window** provides details about the packet selected (highlighted) in the packet listing window. (To select a packet in the packet listing window, place the cursor over the packet's one line summary in the packet listing window and click with the left mouse button.). These details include information about the Ethernet frame and its corresponding encapsulated PDUs. The amount of details displayed can be expanded or minimized by clicking on the arrows to the left of each PDU line in the packet details window. If the packet has been carried over TCP or UDP, these can also be expanded or minimized. Finally, details of the highest level protocol that sent or received data can also be obtained.

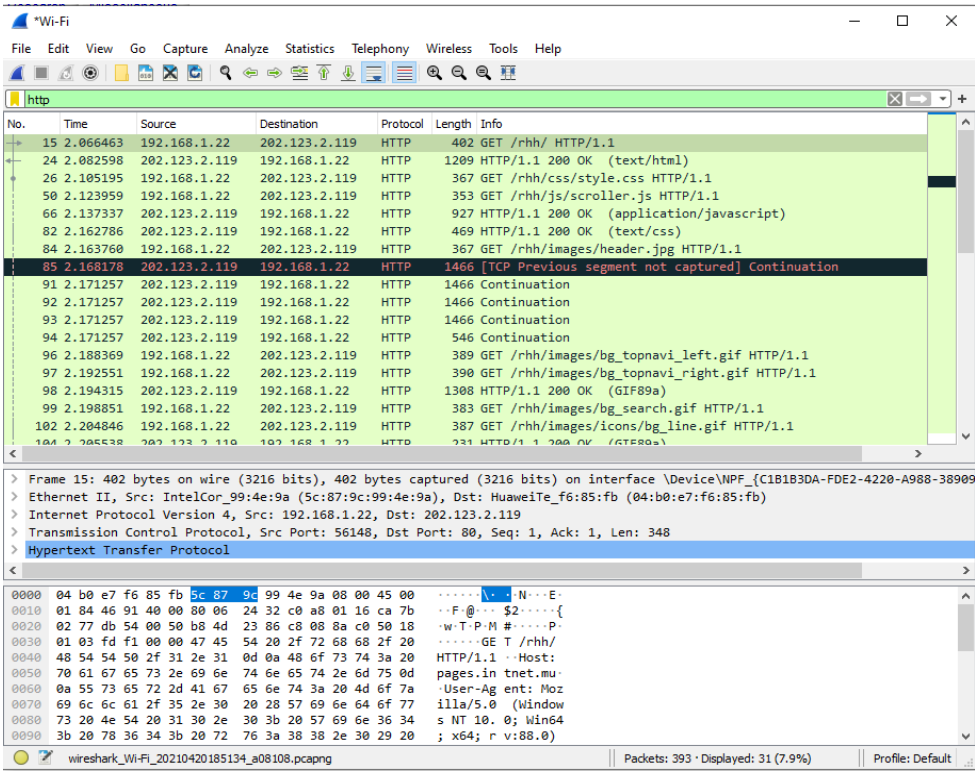
• The **packet-contents window** displays the contents of the captured frame in ASCII & Hexadecimal

• The **packet display filter field,** into which a protocol name or other information can be entered in order to filter the information displayed in the packet-listing window (and hence the packet-header and packet-contents windows). In the example below, we'll use the packet-display filter field to have Wireshark display only packets corresponding to HTTP messages.

Performing a filter using the Packet Display Filter field

You will notice that the packet-listing window consists of at many packets using numerous protocols. This can be overwhelming as sometimes you may wish to analyse only one specific kind of protocol. In this example, we will see how we can only filter HTTP messages.

1. Inside the packet display filter field, enter "http" and press Enter.
2. You should now only see HTTP messages in the packet-listing windows. It should look like below:



Answer the following questions based on your capture:

1. How long did it take from when the first HTTP GET message was sent until the respective HTTP response was received? (By default, the value of the Time column in the packet-listing window is in seconds)
2. What is the Internet address of the MT public pages webserver?
3. What is the Internet address of your computer?
4. Why were more GET messages needed to my old website?
5. What did your browser do before the first HTTP GET message was sent to the web server? *Hint: You will get the answer if you clear the filter by clicking on the cross button and examining the packets captured before the first HTTP GET messages.*