CS 145 Lab Exercise 1
Wireshark Lab: Getting Started
A.Y. 2016-2017, 2nd Semester

1 Introduction

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. This can be done in simulated scenarios or in a "real" network environment such as the Internet. In the Wireshark labs you'll be doing in this course, you'll be running various network applications in different scenarios using the computers in the laboratory (or your own computer, if you wish; do note however that the CS 145 Wireshark labs were written with the Linux OS in mind). 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 the laboratory computer will be an integral part of these "live" labs. You'll observe, and you'll learn, by doing.

In this first Wireshark lab, you'll get acquainted with Wireshark, and make some simple packet captures and observations.

The basic tool for observing the messages exchanged between executing protocol entities is called a **packet sniffer**. As the name suggests, a packet sniffer captures ("sniffs") messages being sent/received from/by your computer; it will also typically store and/or 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 and protocols executing on your machine.

Figure 1 shows the structure of a packet sniffer. At the right of Figure 1 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,

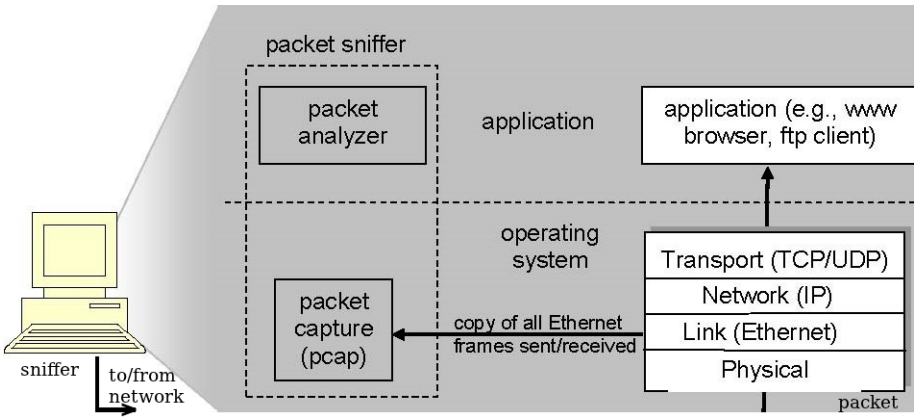

Figure 1: Packet sniffer structure.

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 from the lecture that messages exchanged by higher layer protocols (such as HTTP, FTP, TCP, UDP, DNS, or IP) are eventually encapsulated in link-layer frames that are transmitted over physical media such as an Ethernet cable. 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 the HTTP protocol in Figure 1. The packet analyzer understands the format of Ethernet frames, and so can identify the IP datagram within an Ethernet frame. It also understands the IP datagram format, so that it can extract the TCP segment within the IP datagram. Finally, it understands the TCP segment structure, so it can extract the HTTP message contained in the TCP segment. Finally, it understands the HTTP protocol and so, for example, knows that the first bytes of an HTTP message will contain the string "GET", "POST", or "HEAD".

We will be using the Wireshark packet sniffer [`http://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. (Technically speaking, Wireshark is a packet analyzer that uses a packet capture library in your computer). Wireshark is a free network protocol analyzer that runs on Windows, Linux/Unix, and Mac computers. It's an ideal packet analyzer for our labs - it is stable, has a large user base and well-documented support that includes a user-guide (`http://www.wireshark.org/docs/wsug_html_chunked/`), man pages (`http://www.wireshark.org/docs/man-pages/`), and a detailed FAQ (`http://www.wireshark.org/faq.html`), rich functionality that includes the capability to analyze hundreds of protocols, and a well-

designed user interface. It operates in computers using Ethernet, serial (PPP and SLIP), 802.11 wireless LANs, and many other link-layer technologies (if the OS on which it's running allows Wireshark to do so).

2 Getting Wireshark

In order to run Wireshark, you will need to have access to a computer that supports both Wireshark and the *libpcap* or *WinPCap* packet capture library. The *libpcap* software will be installed for you, if it is not installed within your operating system, when you install Wireshark. See `http://www.wireshark.org/download.html` for a list of supported operating systems and download sites.

If you want your own installation, download and install the Wireshark software:

- Go to `http://www.wireshark.org/download.html` and download and install the Wireshark binary for your computer.

Take note however, that DCS Teaching Laboratory computers should already have Wireshark installed.

The Wireshark FAQ has a number of helpful hints and interesting tidbits of information, particularly if you have trouble installing or running Wireshark.

3 Restrictions

Before we run Wireshark, a few words about restrictions.

As much as possible, we would like you to do most of the work in the comfort of your own home, or with your own computer (we know that you would prefer that). However, the network setup in your own home or the software setup in your own machine may be different from that of the DCS Teaching Laboratory, or the DCS Teaching Laboratory machines. For some laboratory exercises, the differences would not matter. For some, they would, significantly. To avoid confusion (and to make the checking of Laboratory Reports easier), each laboratory exercise would come with a **Restrictions** section, specifying what restrictions apply to the Laboratory exercise. We suggest that you follow these restrictions: failure to do so may result in you instantly getting a 0 in the Laboratory Report. For this laboratory exercise, the following restrictions apply:

- Trace generation must be done in a DCS Teaching Laboratory machine, using the Ethernet (not WiFi) connection.
- Trace analysis may be done in any machine with the Wireshark software installed.

4 Running Wireshark

When you run the Wireshark program, you'll get a startup screen, as shown in Figure 2.

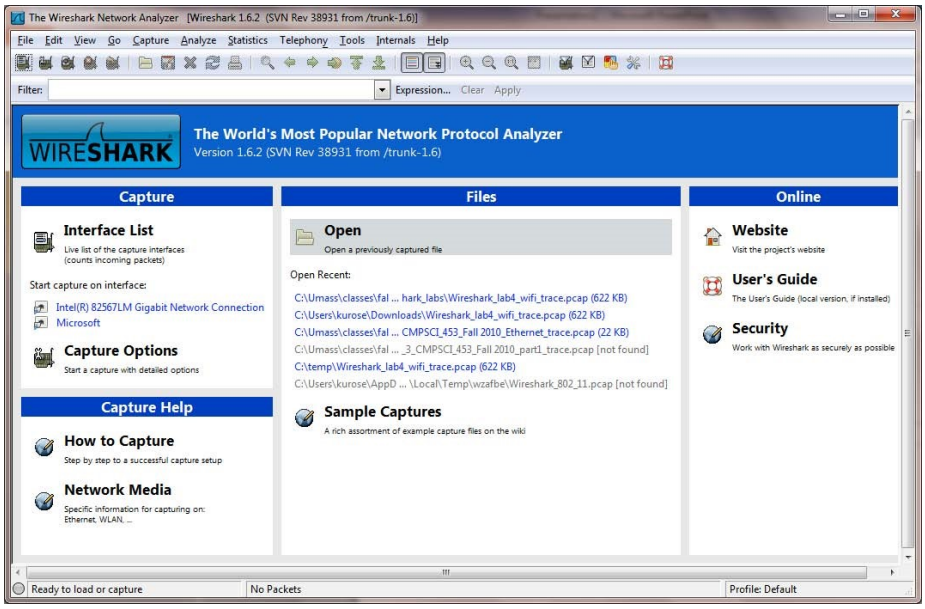

Figure 2: Initial Wireshark screen.

Take a look at the upper left hand side of the screen - you'll see an "Interface list". This is the list of network interfaces on your computer. Once you choose an interface, Wireshark will capture all packets on that interface. In the example above, there is an Ethernet interface (Gigabit network Connection) and a wireless interface ("Microsoft").

If you click on one of these interfaces to start packet capture (i.e., for Wireshark to begin capturing all packets being sent to/from that interface), a screen like the one below will be displayed, showing information about the packets being captured. Once you start packet capture, you can stop it by using the Capture pull down menu and selecting Stop.

The Wireshark interface has five major components:

- The **command menus** are standard pulldown menus located at the top of the window. Of interest to us now are the File and Capture menus. The File menu allows you to save captured packet data or open a file containing previously captured packet data, and exit the Wireshark application. The Capture menu allows you to begin packet capture.

- The **packet-listing window** displays a one-line summary for each packet captured, including the packet number (assigned by Wireshark; this is not a packet number contained in any protocol's header), 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, i.e., the protocol that is the source or ultimate sink for this packet.

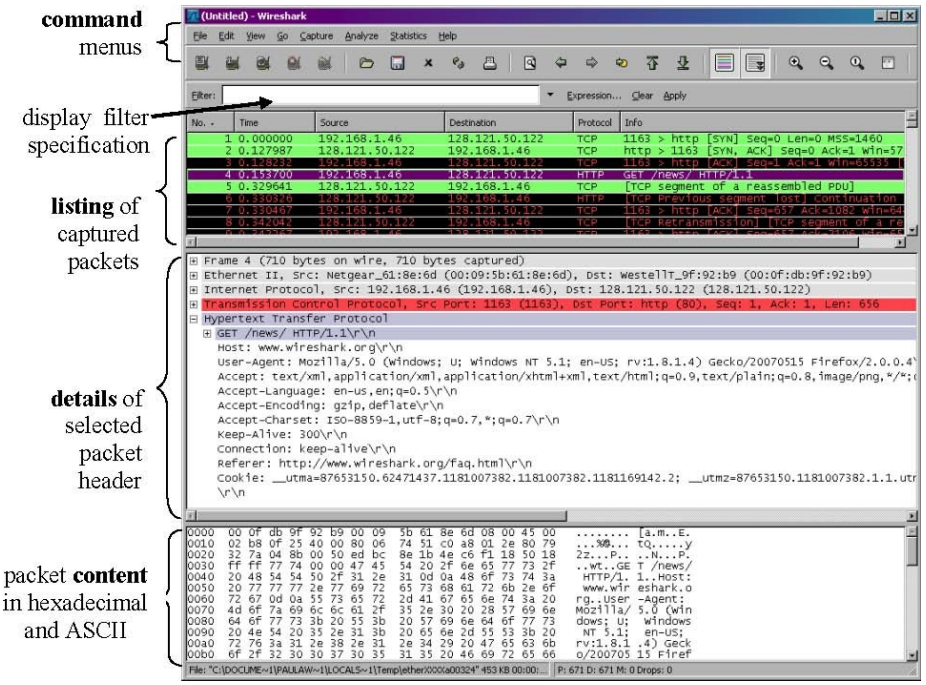

Figure 3: Wireshark Graphical User Interface, during packet capture and analysis.

- 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 (assuming the packet was sent/received over an Ethernet interface) and IP datagram that contains this packet. The amount of Ethernet and IP-layer detail displayed can be expanded or minimized by clicking on the plus minus boxes to the left of the Ethernet frame or IP datagram line in the packet details window. If the packet has been carried over TCP or UDP, TCP or UDP details will also be displayed, which can similarly be expanded or minimized. Finally, details about the highest-level protocol that sent or received this packet are also provided.

- The **packet-contents window** displays the entire contents of the captured frame, in both ASCII and hexadecimal format.

- Towards the top of the Wireshark graphical user interface, is 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 hide (not display) packets except those that correspond to HTTP messages.

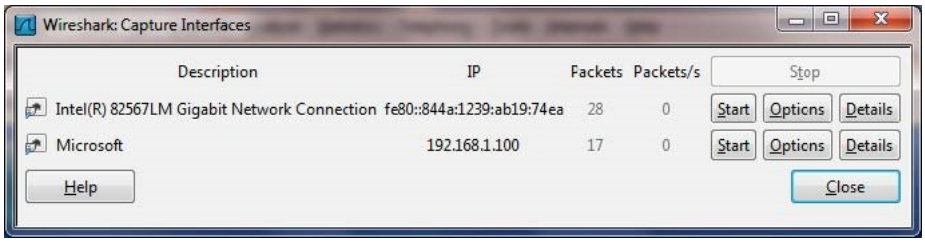

Figure 4: Wireshark Capture Interface Window.

5 Taking Wireshark for a Test Run

5.1 Part 1

The best way to learn about any new piece of software is to try it out! We'll assume that your computer is connected to the Internet via a wired Ethernet interface. Indeed, I recommend that you do this first lab on a computer that has a wired Ethernet connection, rather than just a wireless connection. Do the following

1. Start up a web browser, which will display your selected homepage.
2. Clear the browser's history - I assume that you already know how to do this. Also, throughout the exercise, avoid using multiple tabs, even if your browser is capable of tabbed browsing. That is, while doing the laboratory exercise, do **not** surf any other websites, as that would affect the trace that you would generate.
3. Start up the Wireshark software. You will initially see a window similar to that shown in Figure 2. Wireshark has not yet begun capturing packets.
4. To begin packet capture, select the Capture pull down menu and select Interfaces. This will cause the "Wireshark: Capture Interfaces" window to be displayed, as shown in Figure 4.
5. You'll see a list of the interfaces on your computer as well as a count of the packets that have been observed on that interface so far. Click on Start for the interface on which you want to begin packet capture (if you are doing this laboratory exercise in a DCS Teaching Laboratory that would be `eth0`). Packet capture will now begin - Wireshark is now capturing all packets being sent/received from/by your computer!
6. Once you begin packet capture, a window similar to that shown in Figure 3 will appear. This window shows the packets being captured. By selecting Capture pulldown menu and selecting Stop, you can stop packet capture. But don't stop packet capture yet. Let's capture some interesting packets first. To do so, we'll need to generate some network traffic. Let's do so using a web browser, which will use HTTP to download content from a website. Do not worry about what HTTP is for now; we will learn more about it in subsequent lectures. For now it is sufficient for you to know that getting websites involve exchanging HTTP messages between your computer and the web server.

7. While Wireshark is running, enter the URL `http://www.december.com/html/demo/hello.html` and have that page displayed in your browser. In order to display this page, your browser will contact the HTTP server at www.december.com and exchange HTTP messages with the server in order to download this page. You will learn more about this transaction/process in subsequent lectures. The Ethernet frames containing these HTTP messages (as well as all other frames passing through your Ethernet adapter) will be captured by Wireshark.

8. After your browser has displayed the simple web page, stop Wireshark packet capture by selecting stop in the Wireshark capture window. The main Wireshark window should now look similar to Figure 3. You now have live packet data that contains all protocol messages exchanged between your computer and other network entities! The HTTP message exchanges with the www.december.com web server should appear somewhere in the listing of packets captured. But there will be many other types of packets displayed as well (see, e.g., the many different protocol types shown in the Protocol column in Figure 3). Even though the only action you took was to download a web page, there were evidently many other protocols running on your computer that are unseen by the user. We'll learn much more about these protocols as we progress through the lectures. For now, you should just be aware that there is often much more going on than what "meets the eye"!

9. Type in "http" (without the quotes, and in lower case - all protocol names are in lower case in Wireshark) into the display filter specification window at the top of the main Wireshark window. Then select Apply (to the right of where you entered "http"). This will cause only HTTP message to be displayed in the packet-listing window.

10. Find the HTTP GET message that was sent from your computer to the www.december.com HTTP server. (Look for an HTTP GET message in the "listing of captured packets" portion of the Wireshark window (see Figure 3) that shows "GET" followed by the www.december.com URL that you entered. When you select the HTTP GET message, the Ethernet frame, IP datagram, TCP segment, and HTTP message header information will be displayed in the packet-header window. Recall from the lecture that a layer encapsulates the packet from the layer above it into a new packet. In the case of an HTTP GET message, it is encapsulated in a TCP segment, which is encapsulated in an IP datagram, which is encapsulated in an Ethernet frame. We will learn more HTTP messages, TCP segments, IP datagrams, and Ethernet frames in subsequent lectures. For now, it is sufficient for you to remember that encapsulation at the different layers (more specifically, the headers added at each layer) can be seen in Wireshark. By clicking on '+' and '-' right-pointing and down-pointing arrowheads to the left side of the packet details window, minimize the amount of Frame, Ethernet, Internet Protocol, and Transmission Control Protocol information displayed. Maximize the amount information displayed about the HTTP protocol. Your Wireshark display should now look roughly as shown in Figure 5. (Note, in particular, the minimized amount of protocol information for all protocols except HTTP, and the

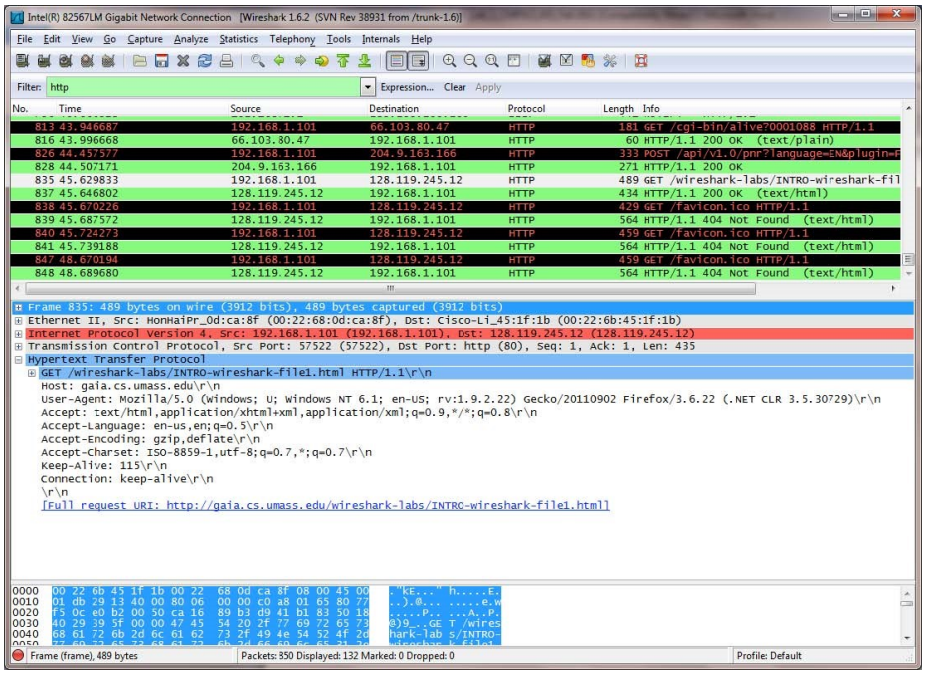

Figure 5: Wireshark window after Step 10.

maximized amount of protocol information for HTTP in the packet-header window).

11. Wireshark traces/captures can be saved for later analyses. Do this by selecting "Save" from the menu (under "File") and choosing an appropriate filename for the trace file.

12. Exit Wireshark.

5.2 Part 2

In some of the lab exercises, to facilitate uniformity in analyses, we will be providing the trace files, instead of having you generate them. It is therefore important for you to be able to open trace files that were generated and imported from other machines.

1. Start up the Wireshark software.

2. Select "Open" from the Wireshark menu (under "File") and select "`twowebsites.pcapng`", the Wireshark trace file we provided for this exercise. The Wireshark windows will be populated no differently than if the packets were captured in your machine. `twowebsites.pcapng` is the trace file captured from a machine with a web browser which visited two websites in quick succession.

3. Exit Wireshark.

6 What to hand in

The goal of this first lab was primarily to introduce you to Wireshark. The following questions will demonstrate that you've been able to get Wireshark up and running, and have explored some of its capabilities. Answer the following questions, based on your Wireshark experimentation:

1. Part 1

   (a) List 3 different protocols that appear in the protocol column in the unfiltered packet-listing window in step 8 above.

   (b) How long did it take from when the HTTP GET message was sent until the HTTP OK reply was received? (By default, the value of the Time column in the packet-listing window is the amount of time, in seconds, since Wireshark tracing began. To display the Time field in time-of-day format, select the Wireshark View pull down menu, then select Time Display Format, then select Time-of-day.)

   (c) What is the Internet address of www.december.com? What is the Internet address of your computer?

   (d) For the two HTTP messages (GET and OK) referred to in question 2 above, take a screenshot of the Wireshark window showing the relevant packet(s) and include the image(s) in the lab report which you will submit. Annotate the images using an image editor to emphasize relevant fields, values, etc. which support your answer in question 2. For an example of a properly annotated image, you may want to look at the sample laboratory report provided in the course's UVLE website.

2. Part 2

   (a) As previously mentioned, the trace file provided shows the trace for a machine/web browser which visited two websites in quick succession. What are these two websites? Provide annotated screenshots or images of the packets that support your answer.

   (b) What is the Internet address of the computer which produced the trace file? Provide annotated screenshots or images of the packet(s) that support your answer.

7 Submission

The laboratory report is due on Sunday, January 29, 2017, 2359 hours. You can submit the laboratory report via UVLE.