Homework+Lab 1: Due at start of class on 25-Mar.

Please complete all three homework problems, and the four lab problems. Please turn in the HW and lab separately.

Homework Problems

1. The Old John Hancock Building is a 36-story, 495-foot structure in Boston. It is topped by a weather beacon with both red and blue lights. The weather forecast for the next day is announced by the color of light, and whether or not it is flashing or solid. The code is given by a popular rhyme,

   Steady blue, clear view.
   Flashing blue, clouds due.
   Steady red, rain ahead.
   Flashing red, snow instead.

   Using this coding scheme which can be used to send four different messages, how many bits of data are conveyed by the beacon?

2. Find the bandwidth and bit rate of at least three different communication standards in use today. Choose communication standards with different types of media (e.g. copper, fiber, and wireless). Compare and contrast the spectral efficiencies (bits per second per Hz of bandwidth) of these communication standards. [Note: The term "bandwidth" has a variety of definitions. For more information, read the paper "Bits, Symbols, Bauds, and Bandwidth" by Roger L. Freeman which is available from within the WPI network at `http://goo.gl/fSHo2`].

3. The `ping` utility (available on Windows, Mac, and Linux systems from the command line) is a tool which uses the ICMP protocol to request an "echo" from a host computer. Familiarize yourself with the ping tool, and then use it to calculate the roundtrip ping times from your computer to the two hosts www.wpi.edu and www.ucla.edu. Comment on the differences you see in the times between the two hosts.

Lab 1: Introduction to Wireshark

[adapted from J. Kurose and K.W. Ross]

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 these Wireshark labs, we'll take the latter approach. 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*[1]. 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, 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. As you may recall from reading Chapter 2, messages exchanged by higher, application 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. 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," (see Chapter 25 of the textbook, which is available as an online supplement).

We will be using the Wireshark packet packet analyzer `http://www.wireshark.org` for these

[1] Using a packet sniffer to monitor the activities of other members of the WPI community is a violation of privacy, and therefore a violation of WPI's Acceptable Use Policy. To disable this mode, please turn off "promiscuous mode" which can be found in the capture options. Please review the AUP here – `http://www.wpi.edu/offices/policies/aup.html`

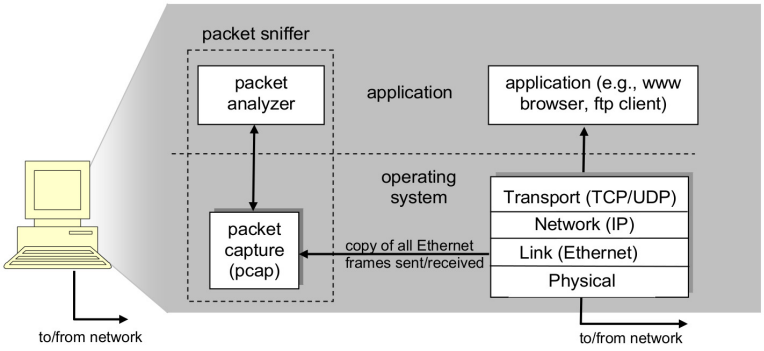

Figure 1: Packet sniffer structure

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/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 and a detailed FAQ, rich functionality that includes the capability to analyze hundreds of protocols, and a well-designed user interface. It operates in computers using Ethernet, Token-Ring, FDDI, serial (PPP and SLIP), 802.11 wireless LANs, and ATM connections (if the OS on which it's running allows Wireshark to do so).

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.

Next, download and install the Wireshark software:

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

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

Running Wireshark

When you run the Wireshark program, you must first select the network card or network interface you wish to use. Then, the Wireshark graphical user interface shown in Figure 2 will be displayed. Initially, no data will be displayed in the various windows.

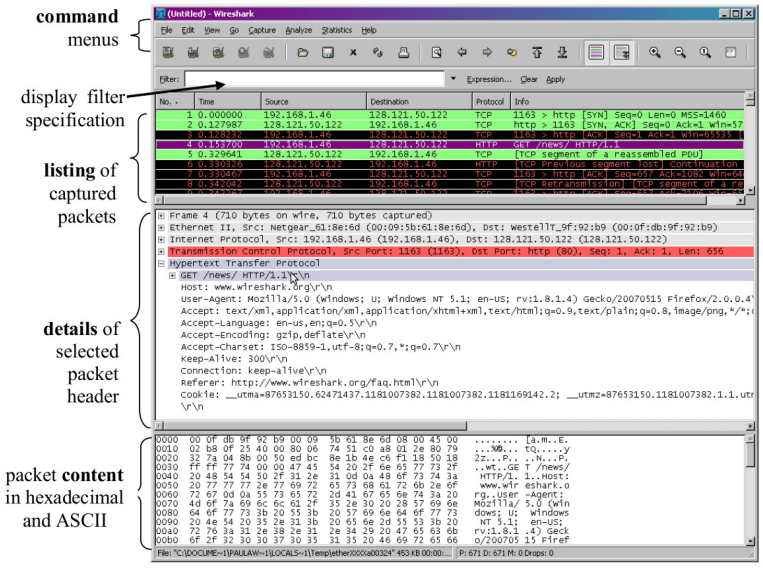

Figure 2: Wireshark Graphical User Interface

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.

- 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/receiverd 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-or-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.

Taking Wireshark for a Test Run

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. Do the following:

1. Start up your favorite web browser, which will display your selected homepage.
2. Start up the Wireshark software. You will initially see a window similar to that shown in Figure 2, except that no packet data will be displayed in the packet- listing, packet-header, or packet-contents window, since Wireshark has not yet begun capturing packets.
3. To begin packet capture, select the Capture pull down menu and select Options. This will cause the "Wireshark: Capture Options" window to be displayed, as shown in Figure 3.

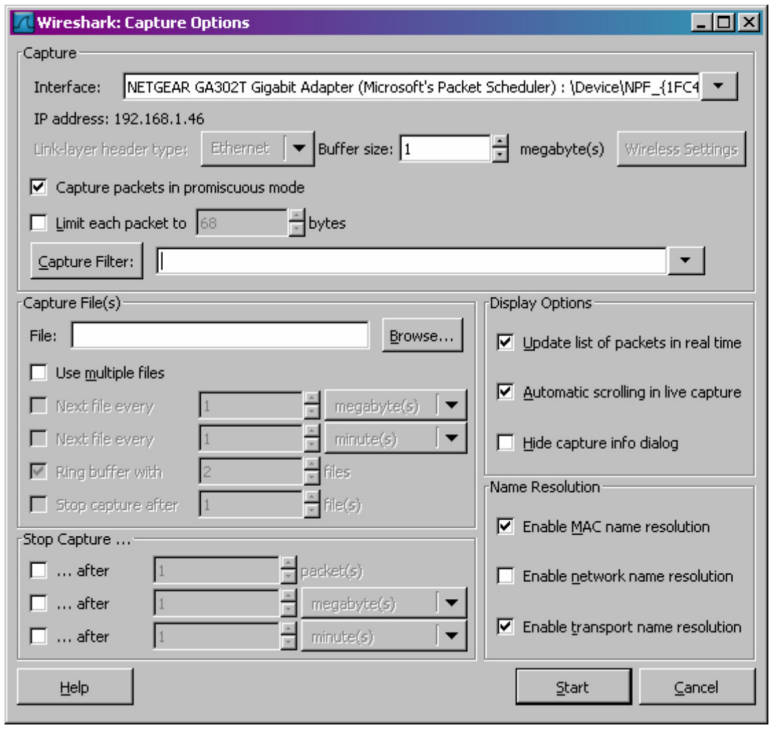

Figure 3: Wireshark Capture Options Window

4. You can use most of the default values in this window, but uncheck "Hide capture info dialog" under Display Options[2]. The network interfaces (i.e., the physical connections) that your

[2]Depending on which OS you are using, this option may instead be found under Edit→Preferences→Capture.

computer has to the network will be shown in the Interface pull down menu at the top of the Capture Options window. In case your computer has more than one active network interface (e.g., if you have both a wireless and a wired Ethernet connection), you will need to select an interface that is being used to send and receive packets (mostly likely the wired interface). After selecting the network interface (or using the default interface chosen by Wireshark), click Start. Packet capture will now begin – all packets being sent/received from/by your computer are now being captured by Wireshark.

5. Once you begin packet capture, a packet capture summary window will appear, as shown in Figure 4. This window summarizes the number of packets of various types that are being captured, and (importantly!) contains the Stop button that will allow you to stop packet capture. Don't stop packet capture yet.

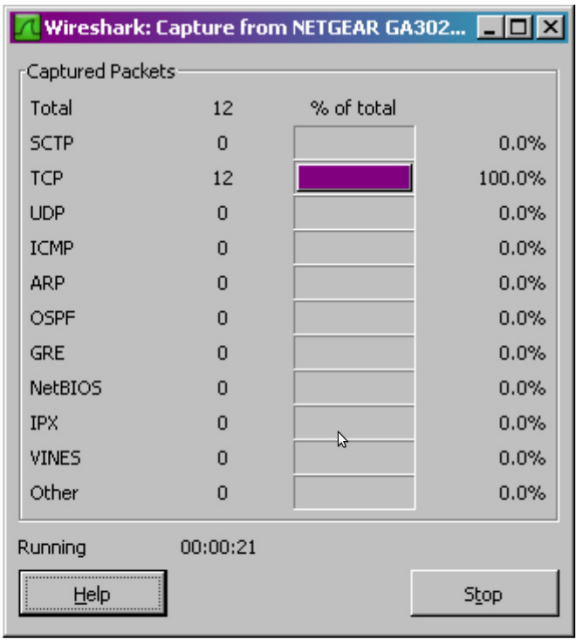

Figure 4: Wireshark Packet Capture Window

6. While Wireshark is running, enter the URL:
`http://spinlab.wpi.edu/index.html`
and have that page displayed in your browser. In order to display this page, your browser will contact the HTTP server at spinlab.wpi.edu and exchange HTTP messages with the server in order to download this page. The Ethernet frames containing these HTTP messages will be captured by Wireshark.

7. After your browser has displayed the index.html page, stop Wireshark packet capture by selecting stop in the Wireshark capture window. This will cause the Wireshark capture window to disappear and the main Wireshark window to display all packets captured since you began packet capture. The main Wireshark window should now look similar to Figure 2. 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 spinlab.wpi.edu 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 2).

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 text. For now, you should just be aware that there is often much more going on than meets the eye.

8. 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.

9. Select the first http message shown in the packet-listing window. This should be the HTTP GET message that was sent from your computer to the spinlab.wpi.edu HTTP server. 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. The HTTP GET message that is sent to the spinlab.wpi.edu web server is contained within a TCP segment, which is contained (encapsulated) in an IP datagram, which is encapsulated in an Ethernet frame. If this process of encapsulation isn't quite clear yet, review sections 2.2 and 2.3 of the text.

   By clicking plus- and-minus boxes 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 maximized amount of protocol information for

   HTTP in the packet-header window).

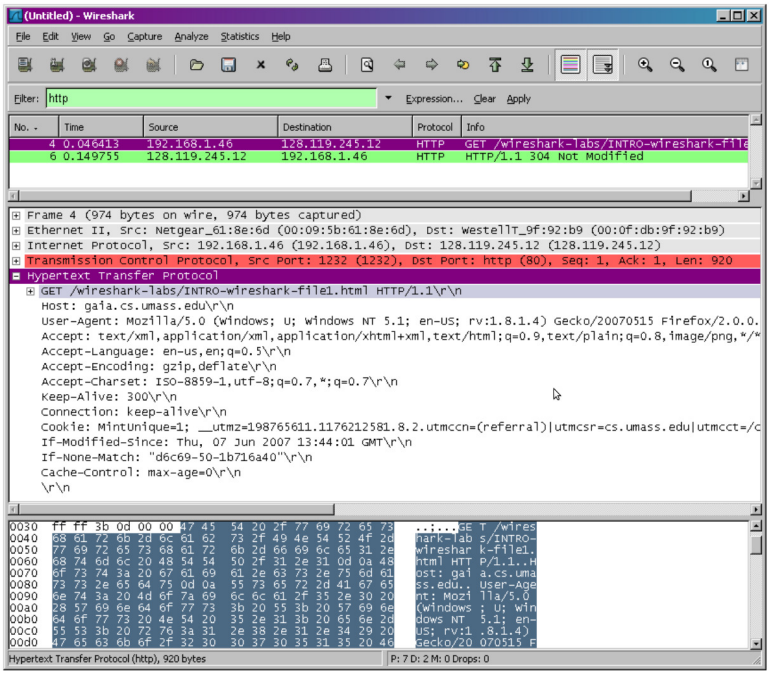

Figure 5: Wireshark display after step 9

10. Exit Wireshark

To be turned 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. List up to 10 different protocols that appear in the protocol column in the unfiltered packet-listing window in step 7 above.

2. 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.)

3. What is the Internet address of the spinlab.wpi.edu (also known as maxwell.ece.wpi.edu)? What is the Internet address of your computer?

4. Print the two HTTP messages displayed in step 9 above. To do so, select Print from the Wireshark File command menu, and select "Selected Packet Only" and "Print as displayed" and then click OK.