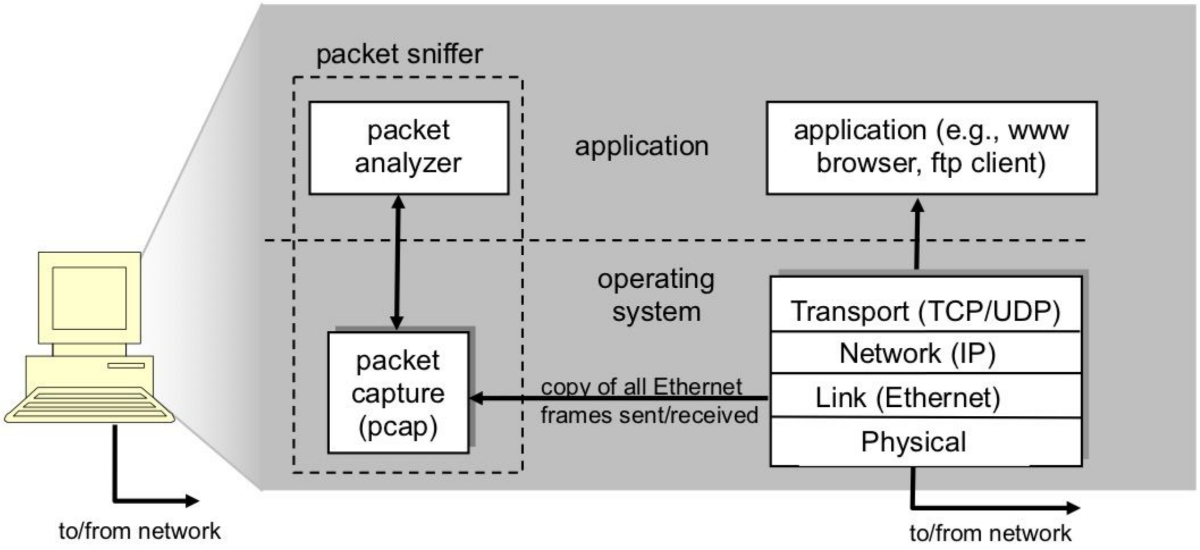
Wireshark Lab 1: Getting Started

Edited version of a similar lab from *Computer Networking: A Top-Down Approach, 7th ed.,* J.F. 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.

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.



The above figure shows the structure of a packet sniffer. At 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 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 discussion from our discussions 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. In this figure, 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 the figure. 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/

(Links to an external site.)

] 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, Mac, and Linux/Unix computer. 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/

(Links to an external site.)

), man pages (http://www.wireshark.org/docs/man-pages/

(Links to an external site.)

), and a detailed FAQ (http://www.wireshark.org/faq.html

(Links to an external site.)

), 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).

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

(Links to an external site.)

for a list of supported operating systems and download sites.

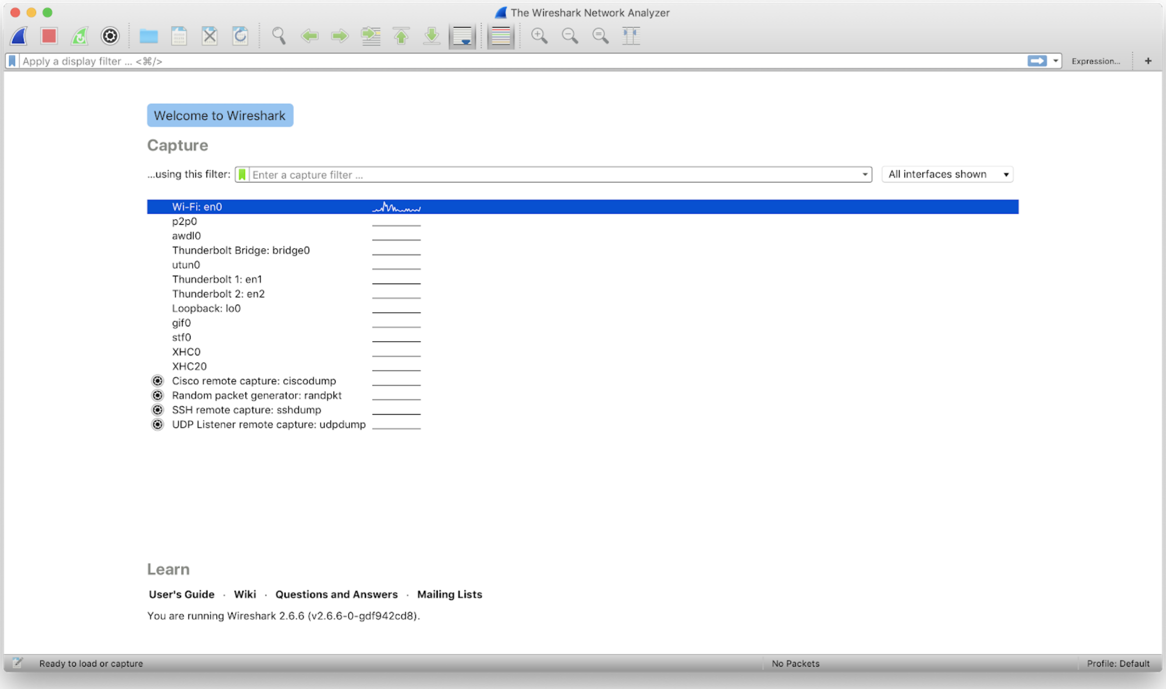
Download and install the Wireshark software:

- Go to http://www.wireshark.org/download.html
- (Links to an external site.)
- and download and install the Wireshark binary for your computer.

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'll get a startup welcoming. There's not much interesting on this screen, but note that under the Capture section, there is a list of so-called interfaces (e.g., "Wi-Fi en0," in my laptop). All packets to/from this computer will pass through the Wi-Fi interface, so it's here where we want to capture packets.


The Wireshark Network Analyzer
Apply a display filter … <⌘/>
Expression…
+
Welcome to Wireshark
Capture
Learn
User's Guide · Wiki · Questions and Answers · Mailing Lists

…using this filter: Enter a capture filter …

All interfaces shown

Wi-Fi: en0
p2p0
awdl0
Thunderbolt Bridge: bridge0
utun0
Thunderbolt 1: en1
Thunderbolt 2: en2
Loopback: lo0
gif0
stf0
XHC0
XHC20
Cisco remote capture: ciscodump
Random packet generator: randpkt
SSH remote capture: sshdump
UDP Listener remote capture: udpdump

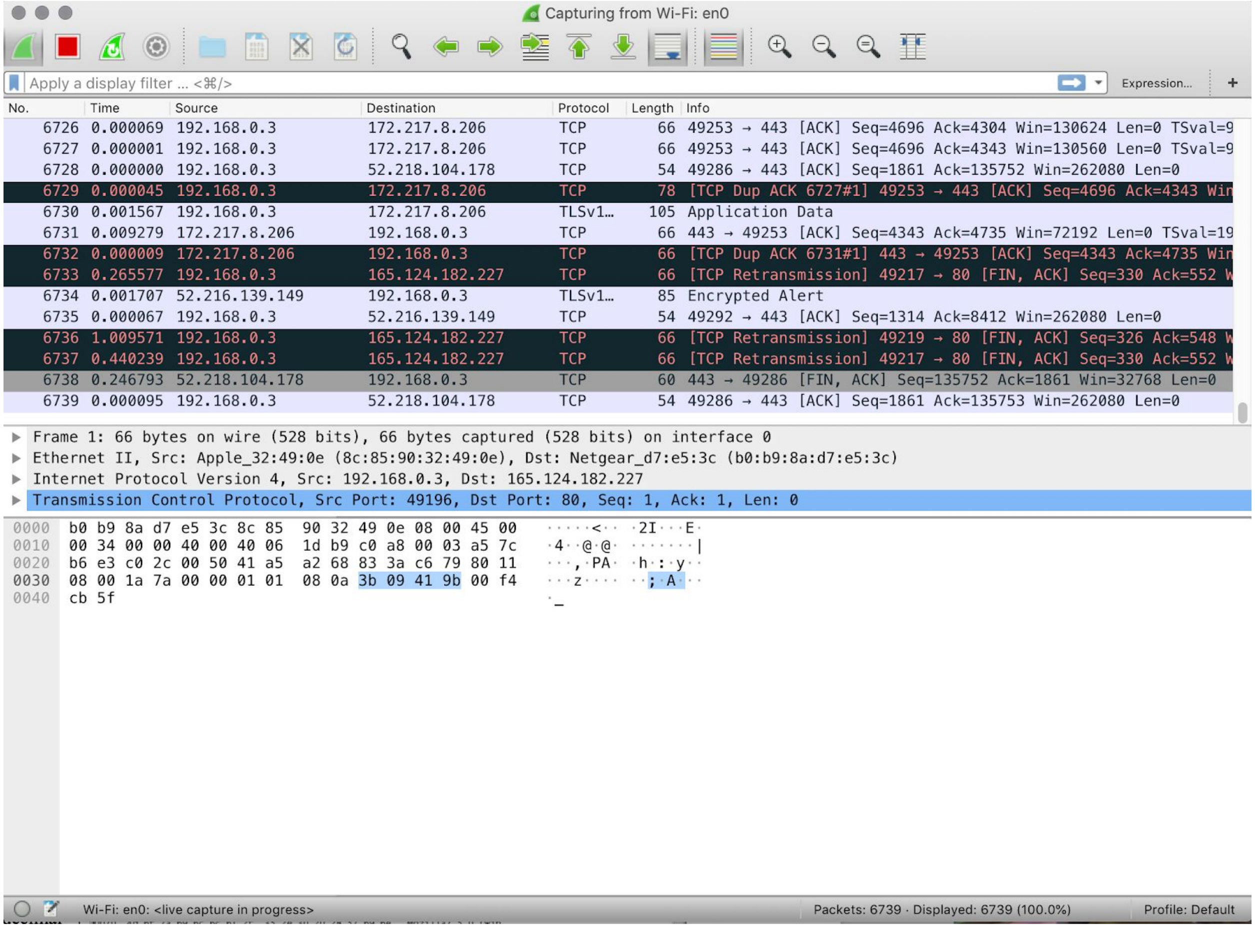
You are running Wireshark 2.6.6 (v2.6.6-0-gdf942cd8).

Ready to load or capture

No Packets

Profile: Default

On a Mac, double click on this interface (or on another computer locate the interface on startup page through which you are getting Internet connectivity, e.g., mostly likely a WiFi or Ethernet interface, and select that interface. If you click on one of these interfaces to start packet capture, a screen like the one bellow will show 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.


Capturing from Wi-Fi: en0
Apply a display filter … <⌘/>
Expression…
+
Wi-Fi: en0: <live capture in progress>
Packets: 6739 · Displayed: 6739 (100.0%)
Profile: Default

```
No.     Time     Source           Destination      Protocol  Length Info
   6726 0.000069 192.168.0.3      172.217.8.206    TCP          66 49253 → 443 [ACK] Seq=4696 Ack=4304 Win=130624 Len=0 TSval=9
   6727 0.000001 192.168.0.3      172.217.8.206    TCP          66 49253 → 443 [ACK] Seq=4696 Ack=4343 Win=130560 Len=0 TSval=9
   6728 0.000000 192.168.0.3      52.218.104.178   TCP          54 49286 → 443 [ACK] Seq=1861 Ack=135752 Win=262080 Len=0
   6729 0.000045 192.168.0.3      172.217.8.206    TCP          78 [TCP Dup ACK 6727#1] 49253 → 443 [ACK] Seq=4696 Ack=4343 Win
   6730 0.001567 192.168.0.3      172.217.8.206    TLSv1…      105 Application Data
   6731 0.009279 172.217.8.206    192.168.0.3      TCP          66 443 → 49253 [ACK] Seq=4343 Ack=4735 Win=72192 Len=0 TSval=19
   6732 0.000009 172.217.8.206    192.168.0.3      TCP          66 [TCP Dup ACK 6731#1] 443 → 49253 [ACK] Seq=4343 Ack=4735 Win
   6733 0.265577 192.168.0.3      165.124.182.227  TCP          66 [TCP Retransmission] 49217 → 80 [FIN, ACK] Seq=330 Ack=552 W
   6734 0.001707 52.216.139.149   192.168.0.3      TLSv1…       85 Encrypted Alert
   6735 0.000067 192.168.0.3      52.216.139.149   TCP          54 49292 → 443 [ACK] Seq=1314 Ack=8412 Win=262080 Len=0
   6736 1.009571 192.168.0.3      165.124.182.227  TCP          66 [TCP Retransmission] 49219 → 80 [FIN, ACK] Seq=326 Ack=548 W
   6737 0.440239 192.168.0.3      165.124.182.227  TCP          66 [TCP Retransmission] 49217 → 80 [FIN, ACK] Seq=330 Ack=552 W
   6738 0.246793 52.218.104.178   192.168.0.3      TCP          60 443 → 49286 [FIN, ACK] Seq=135752 Ack=1861 Win=32768 Len=0
   6739 0.000095 192.168.0.3      52.218.104.178   TCP          54 49286 → 443 [ACK] Seq=1861 Ack=135753 Win=262080 Len=0

▶ Frame 1: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface 0
▶ Ethernet II, Src: Apple_32:49:0e (8c:85:90:32:49:0e), Dst: Netgear_d7:e5:3c (b0:b9:8a:d7:e5:3c)
▶ Internet Protocol Version 4, Src: 192.168.0.3, Dst: 165.124.182.227
▶ Transmission Control Protocol, Src Port: 49196, Dst Port: 80, Seq: 1, Ack: 1, Len: 0

0000  b0 b9 8a d7 e5 3c 8c 85  90 32 49 0e 08 00 45 00   ·····<·· ·2I···E·
0010  00 34 00 00 40 00 40 06  1d b9 c0 a8 00 03 a5 7c   ·4··@·@· ·······|
0020  b6 e3 c0 2c 00 50 41 a5  a2 68 83 3a c6 79 80 11   ···,·PA· ·h·:·y··
0030  08 00 1a 7a 00 00 01 01  08 0a 3b 09 41 9b 00 f4   ···z···· ··;·A···
0040  cb 5f                                              ·_
```

The Wireshark interface has five major components:

- The **command menus** 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/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.

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. 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 your favorite web browser, which will display your selected homepage.

2. Start up the Wireshark software. The homepage will be displayed. Wireshark has not yet begun capturing packets.
3. To begin packet capture, click on *Capture* right below the "Welcome to Wireshark" tag. This will cause the “Wireshark: Capture Interfaces” window to be displayed.
4. 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 (in the case, the Gigabit network Connection). Packet capture will now begin - Wireshark is now capturing all packets being sent/received from/by your computer!
5. Once you begin packet capture, a new window will appear showing the packets being captured. Don’t stop packet capture yet; let’s capture some interesting packets first. For this we’ll need to generate some network traffic. Let’s do so using a web browser, which will use the HTTP protocol that we will study in detail in class to download content from a website.
6. While Wireshark is running, enter the URL: http://gaia.cs.umass.edu/wireshark-labs/INTRO-wireshark-file1.html
7. (Links to an external site.)
8. and have that page displayed in your browser. In order to display this page, your browser will contact the HTTP server at gaia.cs.umass.edu and exchange HTTP messages with the server in order to download this page. The Ethernet frames containing these HTTP messages (as well as all other frames passing through your Ethernet adapter) will be captured by Wireshark.
9. After your browser has displayed the INTRO-wireshark-file1.html page (it is a simple one line of congratulations), 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 gaia.cs.umass.edu web server should appear somewhere in the listing of packets captured. But there will be many other types of packets displayed as well. 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 “meet’s the eye”!
10. 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.

11. Find the HTTP GET message that was sent from your computer to the gaia.cs.umass.edu 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 gaia.cs.umass.edu 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[1]. 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. 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.
12. Exit Wireshark

Congratulations! You’ve now completed the first lab.

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. List 3 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 gaia.cs.umass.edu (also known as www-net.cs.umass.edu)? What is the Internet address of your computer?