Assignment 0 - Wireshark

Instructions:

Please complete this assignment by individuals. Both graduate and undergraduate students are expected to complete this programming assignment. This assignment is designed to evaluate your basic networking skills.

Getting started:

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, 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. Note that here messages that are 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. (Don't worry about HTTP or TCP as we have not talked about it. This lab contains minimal exposure to these two protocols.) 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.

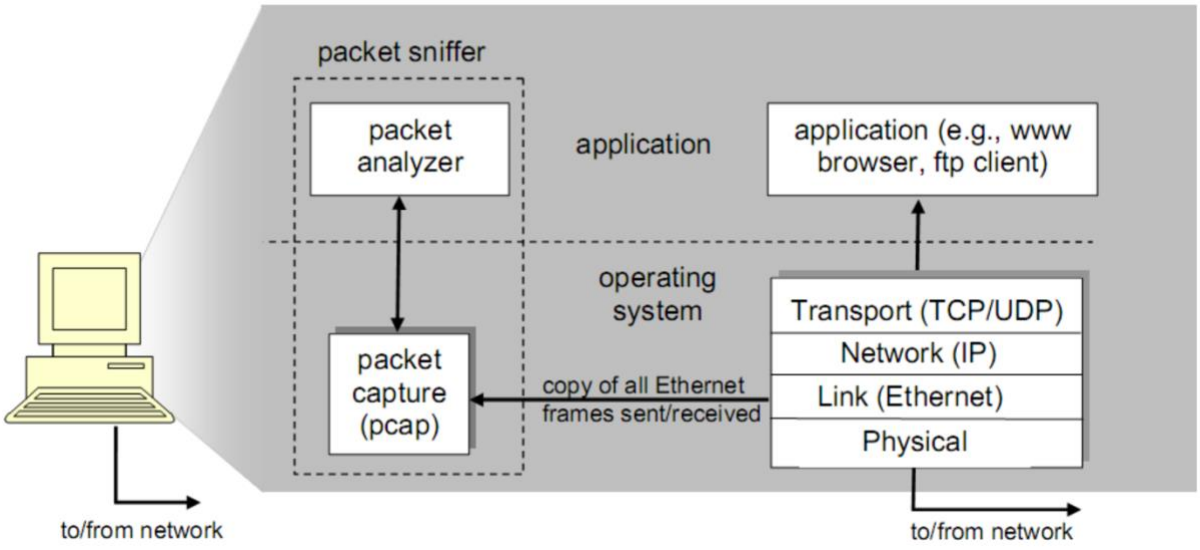

Figure 1. Packet sniffer

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 UDP 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 UDP segment within the IP datagram. It will then display the contents of UDP packets for your analysis.

We will be using the Wireshark packet sniffer [https://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

- Documentation
- 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, 802.11 wireless LANs

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 https://www.wireshark.org/#download for a list of supported operating systems and download sites Download and install the Wireshark software:

Go to https://www.wireshark.org/#download and download and install the Wireshark binary for your computer.

Running Wireshark:

When you run the Wireshark program, the Wireshark graphical user interface shown in Figure 2 will be displayed. Initially, no data will be displayed in the various windows.

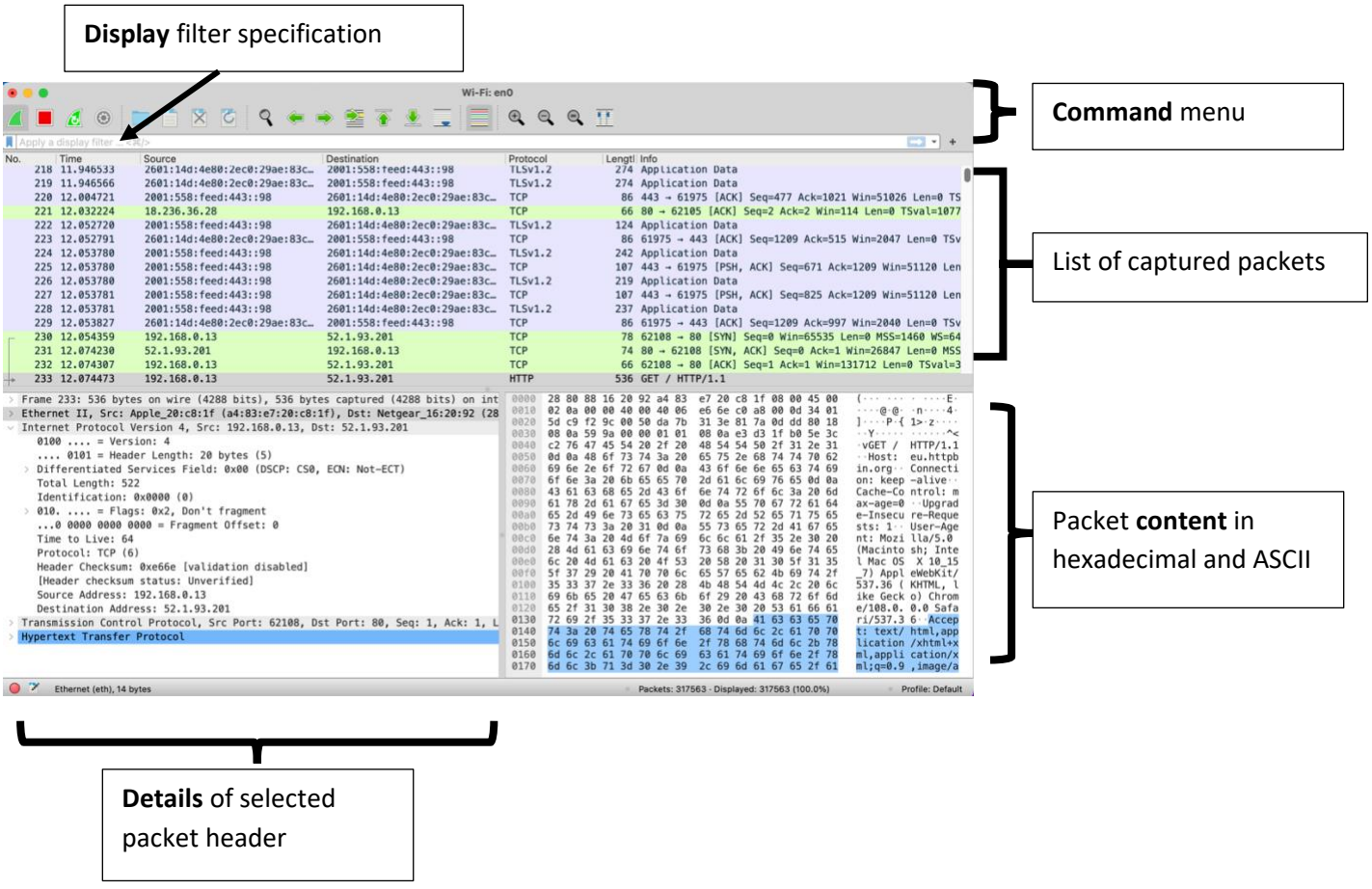

Figure 2. Wireshark graphical user interface

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/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-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 Test Run:

Let's begin our journey with a simple ICMP adventure by capturing the packets generated by the Ping program. (Just like what we did in class) You may recall that the Ping program is simple tool that allows anyone (for example, a network administrator) to verify if a host is live or not. The Ping program in the source host sends a packet to the target IP address; if the target is live,

the Ping program in the target host responds by sending a packet back to the source host. As you might have guessed, based on the previous lab, both of these Ping packets are ICMP packets.

Open your terminal (for MAC or Linux users) or Windows Command Prompt application (which can be found in your Accessories folder).

Start up the Wireshark packet sniffer, and begin Wireshark packet capture on the interface that is connected to the Internet. Figure below shows the Wireshark upon opening. **Make sure to choose the interface which is connected to the internet. ** For me is the WiFi interface: en0.

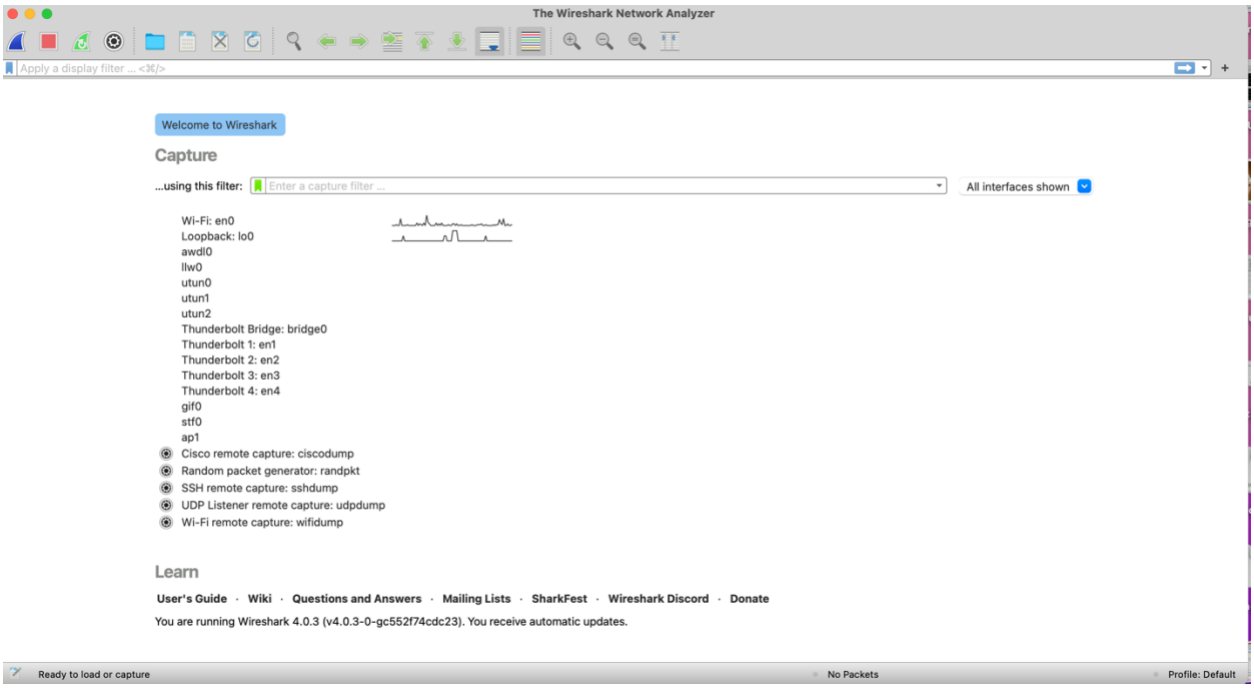

Figure 3. Wireshark main window

If you are a Linux/ Mac user, in the terminal, try the ping command like below:

ping –c 10 hostname

If you are a Windows user, in the terminal, try the ping command like below:

ping –n 10 hostname

where hostname may be a host on another continent. You may choose any website you want. We use google.com as an example. Here, "-n 10" indicates that 10 ping messages should be sent. Then run the Ping program by typing return.

When the Ping program terminates, stop the packet capture in Wireshark. In the filter area write "icmp" so that the Wireshark only shows the ICMP packets. At the end of the experiment, your Wireshark window could look something like the following.

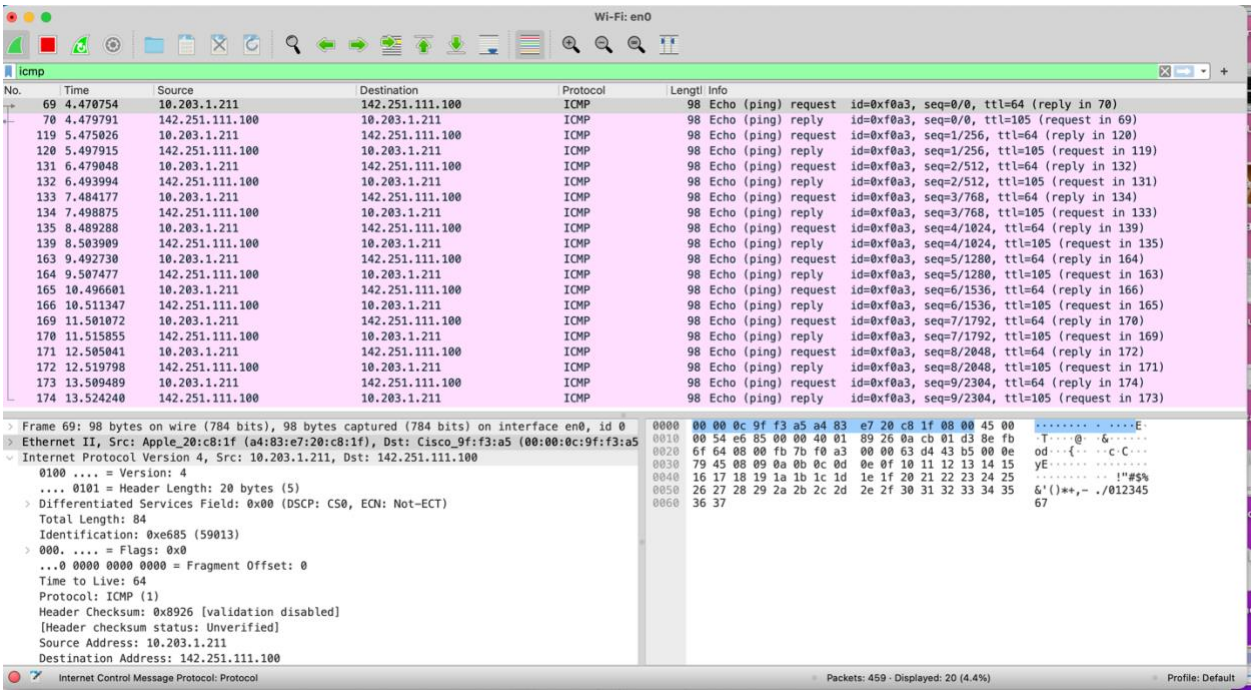

Figure 4. Wireshark packet capture window

The above figure provides a screenshot of the Wireshark output, after "icmp" has been entered into the filter display window. Note that the packet listing shows 20 packets: the 10 Ping queries sent by the source and the 10 Ping responses received by the source. Also note that the source's IP address is a private address (behind a NAT) of the form 10.203.1.211; the destination's IP address is that of the Web server at google.com. Now let's zoom in on the first packet (sent by the client); in the figure below, the packet contents area provides information about this packet. You will find that the IP datagram within this packet has protocol number 01, which is the protocol number for ICMP. This means that the payload of the IP datagram is an ICMP packet.

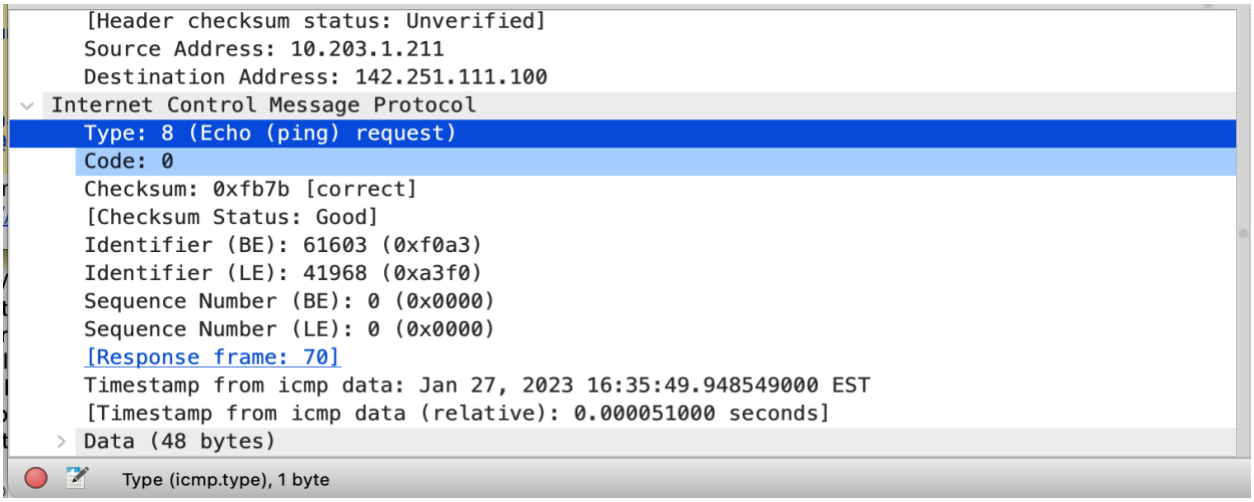

Figure 6. Details of ICMP packet

When answering the following questions, you should provide similar screenshots and use them to support your answers. The lab contains the following questions.

Questions for ICMP Part:

1. **Examine one of the ping request packets sent by your host. What are the ICMP type and code numbers? What other fields does this ICMP packet have? How many bytes are the checksum, sequence number and identifier fields? (10 points)**
2. **Examine the corresponding ping reply packet. What are the ICMP type and code numbers? What other fields does this ICMP packet have? How many bytes are the checksum, sequence number and identifier fields? (10 points)**
3. **Examine the consecutive ICMP packets. Verify the RTT time reported in the command window is the same as the timestamps you observe via Wireshark. (Providing screenshots for command windows is helpful to demonstrate your results.) (20 points)**

Test HTTP with Wireshark:

In this experiment, you will be able to capture HTTP packets. First, open your Wireshark and start sniffing the packets. Then, click on the URL below.

http://www.testingmcafeesites.com/

After your browser has displayed the web page, stop Wireshark packet capture by selecting stop in the Wireshark capture window. 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 testingmcafeesites.com 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.

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.

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 testingmcafeesites.com 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. 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 like Figure 7. Note that the maximized amount of protocol information for HTTP is displayed in the packet-header window.

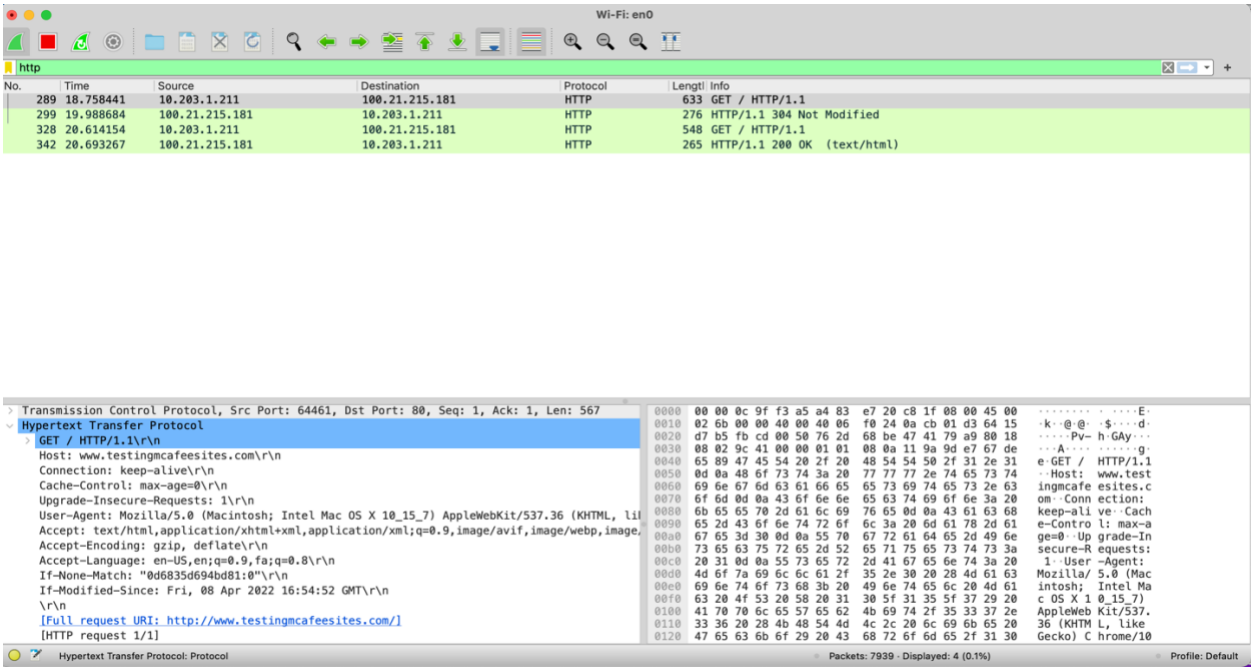

Figure 7. GET HTTP packet

Questions for HTTP Part:

1. **List up to 10 different protocols that appear in the protocol column in the unfiltered packet listing window before applying the filter. (10 points)**
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. (10 points)**
3. **What is the Internet address of the testingmcafeesites.com? What is the Internet address of your computer? (10 points)**
4. **Provide a similar screenshot to the Figure 7 with Wireshark running on your computer. (10 points)**
5. **What happens if you open google.com? Why? (10 points)**

What to turn in:

For these questions in this assignment, please prepare a report with the solutions to all the problems. Your answers should be supported by appropriate screenshots.

Please submit your pdf version of your answers to the Canvas.

Copyright: