WAYNE STATE UNIVERSITY
COLLEGE OF ENGINEERING

CSC 5991 Cyber Security Practice

Lab 1: Packet Sniffing and Wireshark

Introduction

The first part of the lab introduces packet sniffer, Wireshark. Wireshark is a free open-source network protocol analyzer. It is used for network troubleshooting and communication protocol analysis. Wireshark captures network packets in real time and display them in human-readable format. It provides many advanced features including live capture and offline analysis, three-pane packet browser, coloring rules for analysis. This document uses Wireshark for the experiments, and it covers Wireshark installation, packet capturing, and protocol analysis.

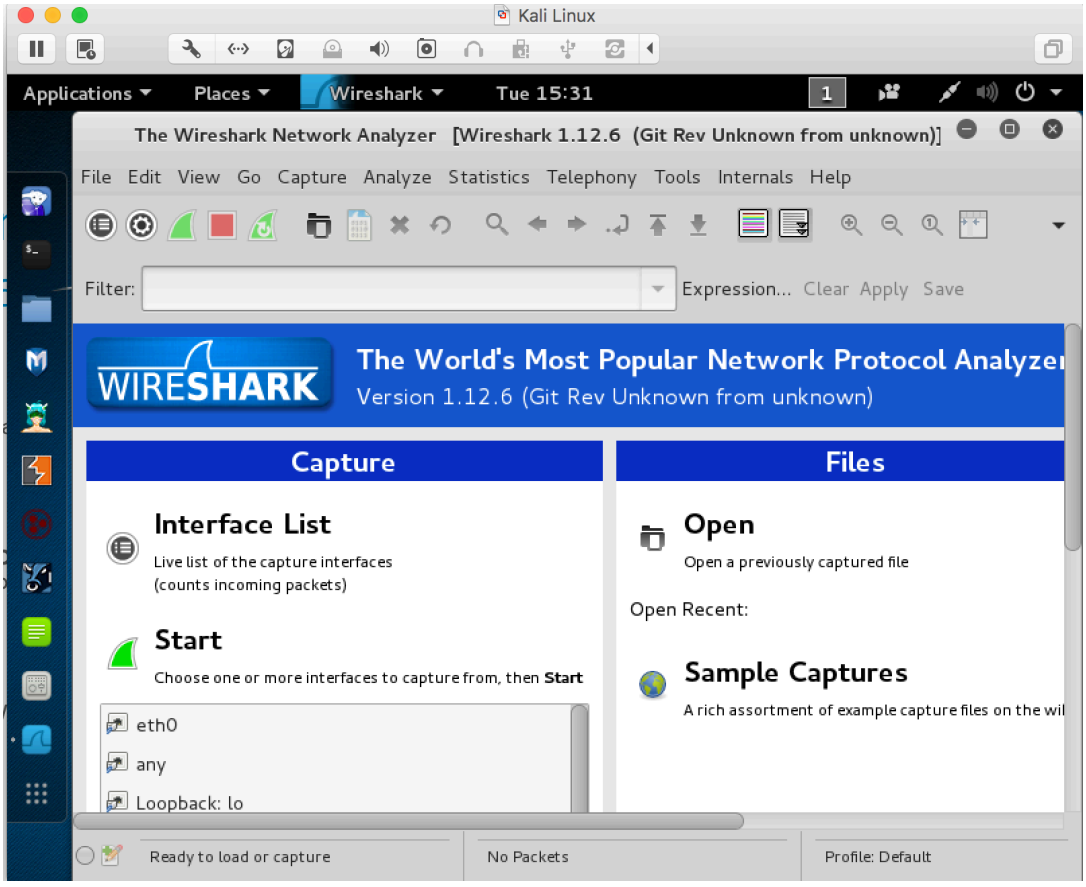


Figure 1: Wireshark in Kali Linux

Background

TCP/IP Network Stack

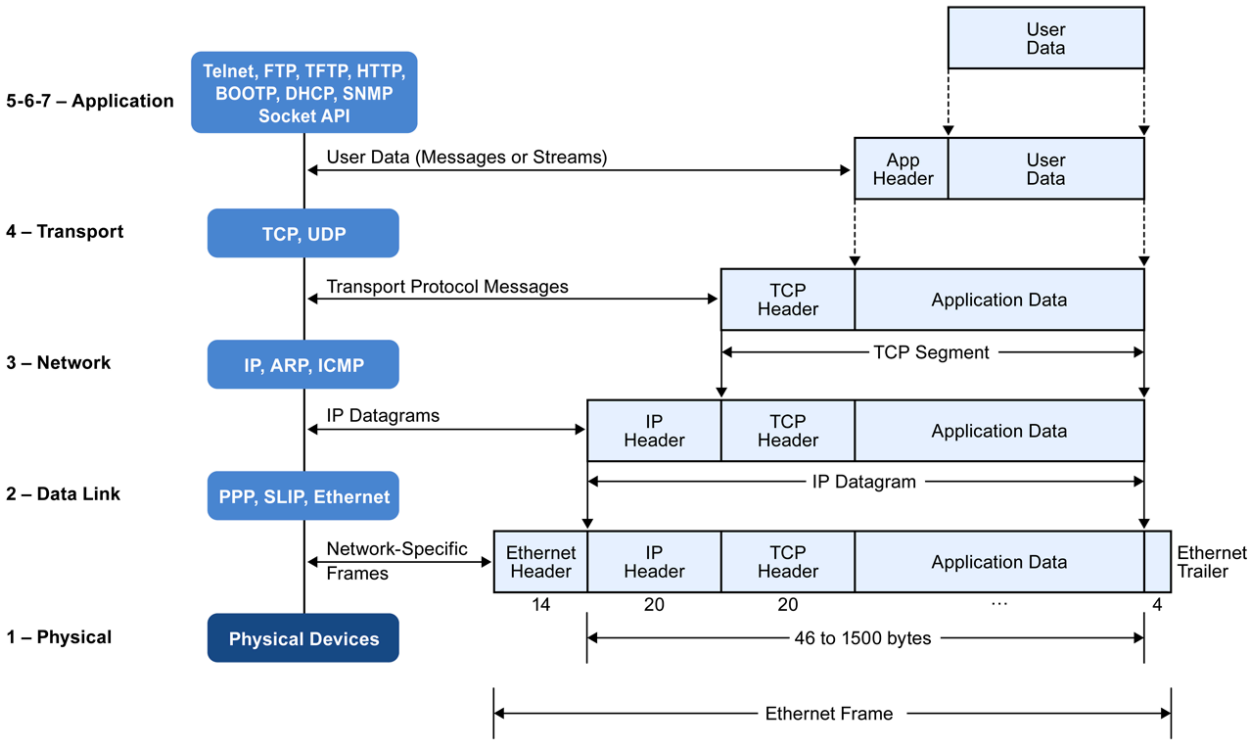


Figure 2: Encapsulation of Data in the TCP/IP Network Stack

In the CSC 4190 Introduction to Computer Networking (one of the perquisite courses), TCP/IP network stack is introduced and studied. This background section briefly explains the concept of TCP/IP network stack to help you better understand the experiments. TCP/IP is the most commonly used network model for Internet services. Because its most important protocols, the Transmission Control Protocol (TCP) and the Internet Protocol (IP) were the first networking protocols defined in this standard, it is named as TCP/IP. However, it contains multiple layers including application layer, transport layer, network layer, and data link layer.

- *Application Layer*: The application layer includes the protocols used by most applications for providing user services. Examples of application layer protocols are Hypertext Transfer Protocol (HTTP), Secure Shell (SSH), File Transfer Protocol (FTP), and Simple Mail Transfer Protocol (SMTP).

- *Transport Layer*: The transport layer establishes process-to-process connectivity, and it provides end-to-end services that are independent of underlying user data. To implement the process-to-process communication, the protocol introduces a concept of port. The examples of transport layer protocols are Transport Control Protocol (TCP) and User Datagram Protocol (UDP). The TCP provides flow-control, connection establishment, and reliable transmission of data, while the UDP is a connectionless transmission model.
- *Internet Layer*: The Internet layer is responsible for sending packets to across networks. It has two functions: 1) Host identification by using IP addressing system (IPv4 and IPv6); and 2) packets routing from source to destination. The examples of Internet layer protocols are Internet Protocol (IP), Internet Control Message Protocol (ICMP), and Address Resolution Protocol (ARP).
- *Link Layer*: The link layer defines the networking methods within the scope of the local network link. It is used to move the packets between two hosts on the same link. An common example of link layer protocols is Ethernet.

Packet Sniffer

Packet sniffer is a basic tool for observing network packet exchanges in a computer. As the name suggests, a packet sniffer captures ("sniffs") packets 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 packets. 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.

Figure 3 shows the structure of a packet sniffer. At the right of **Figure** 3 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** 3 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. 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 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 access to 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

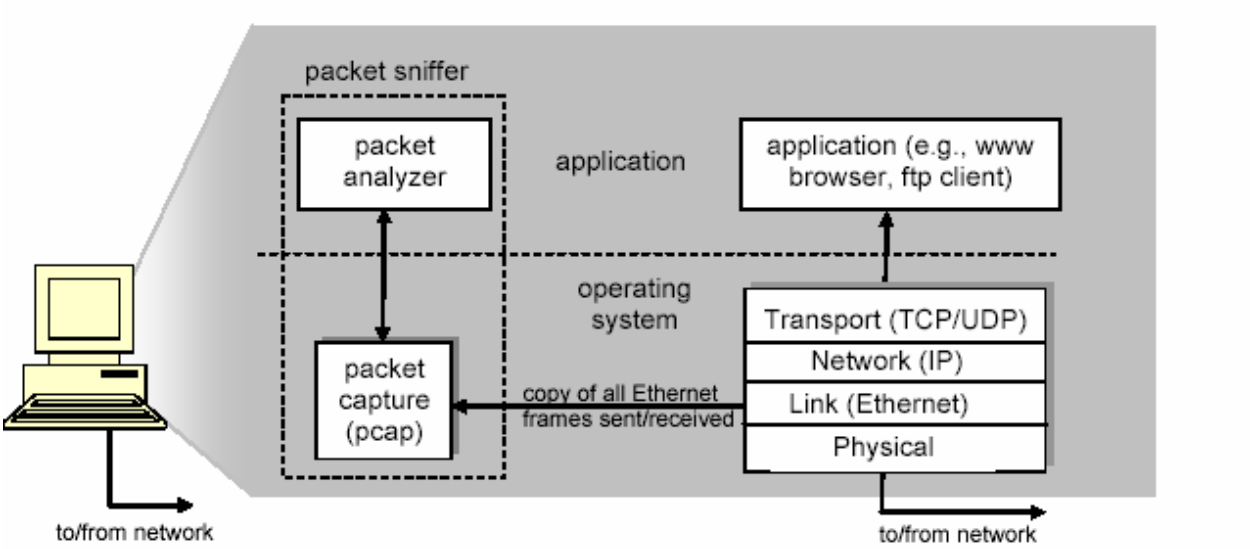


Figure 3: Packet Sniffer Structure

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** 3. 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.

Getting Wireshark

The Kai Linux has Wireshark installed. You can just launch the Kali Linux VM and open Wireshark there. Wireshark can also be downloaded from here:

https://www.wireshark.org/download.html

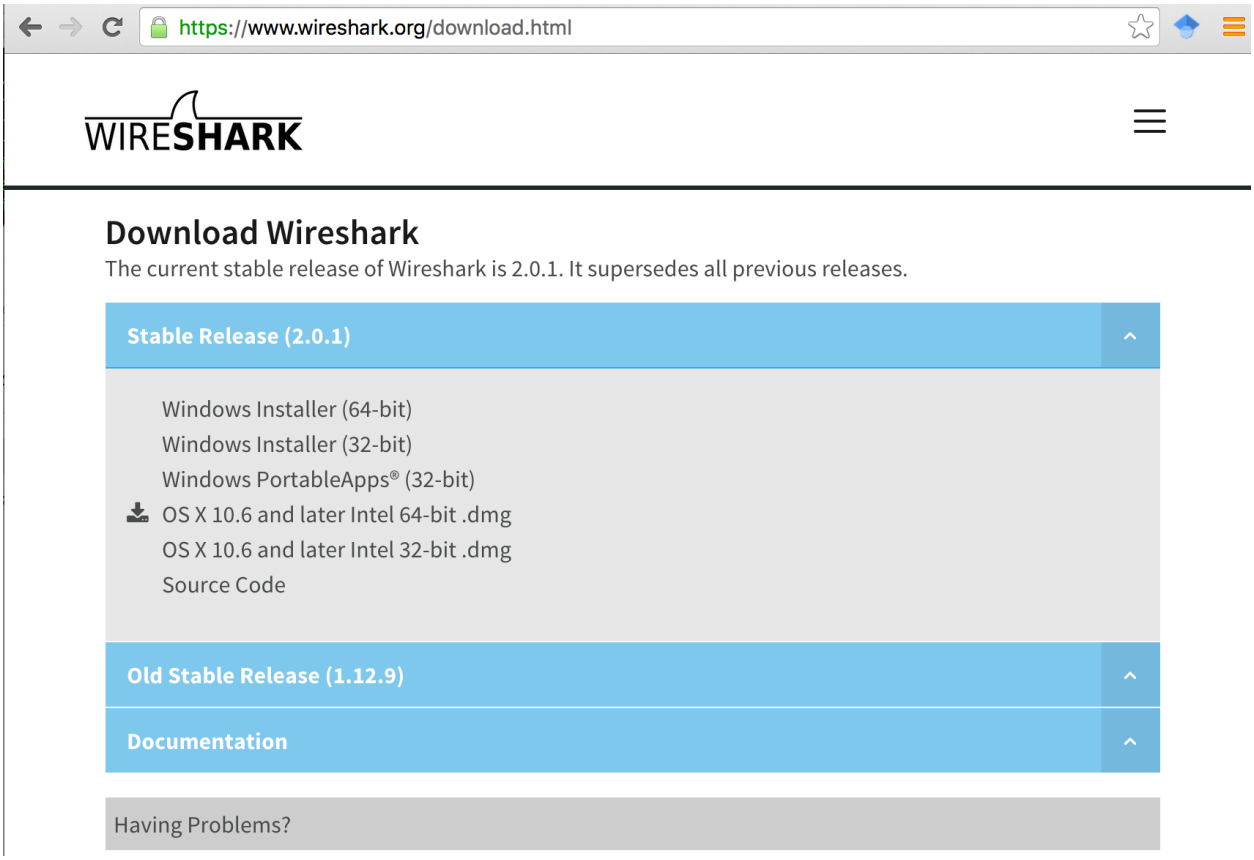


Figure 4: Download Page of Wireshark

Starting Wireshark

When you run the Wireshark program, the Wireshark graphic user interface will be shown as **Figure** 5. Currently, the program is not capturing the packets.

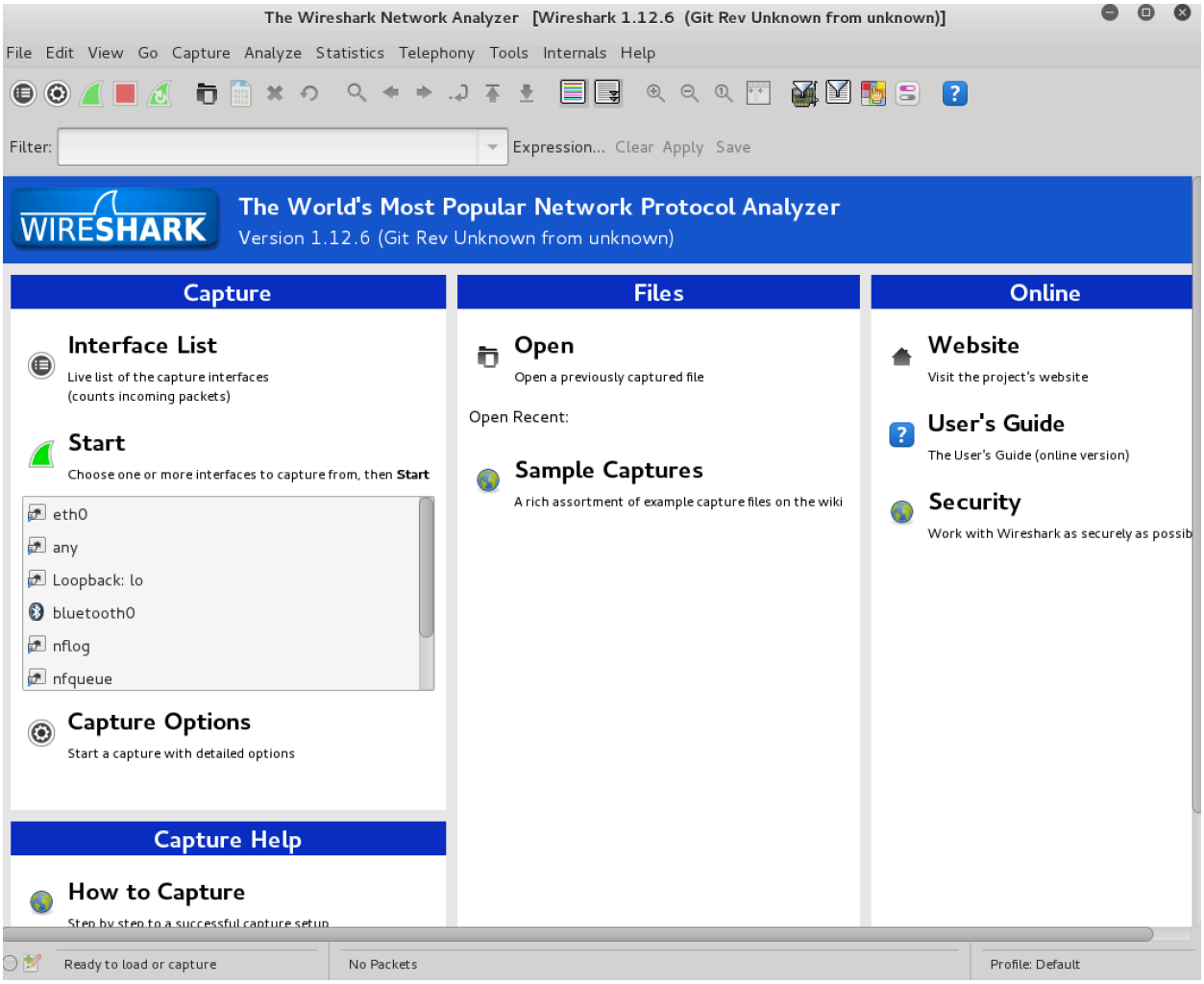


Figure 5: Initial Graphic User Interface of Wireshark

Then, you need to choose an interface. If you are running the Wireshark on your laptop, you need to select WiFi interface. If you are at a desktop, you need to select the Ethernet interface being used. Note that there could be multiple interfaces. In general, you can select any interface but that does not mean that traffic will flow through that

interface. The network interfaces (i.e., the physical connections) that your computer has to the network are shown. The attached **Figure** 6 was taken from my computer.

After you select the interface, you can click start to capture the packets as shown in **Figure** 7.

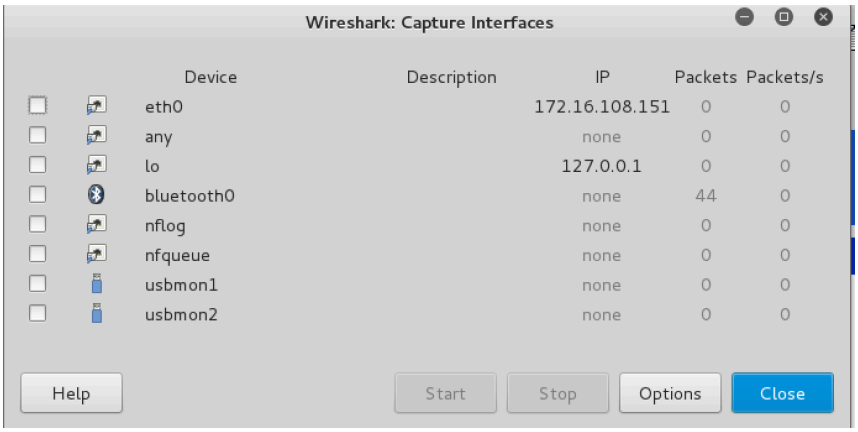


Figure 6: Capture Interfaces in Wireshark

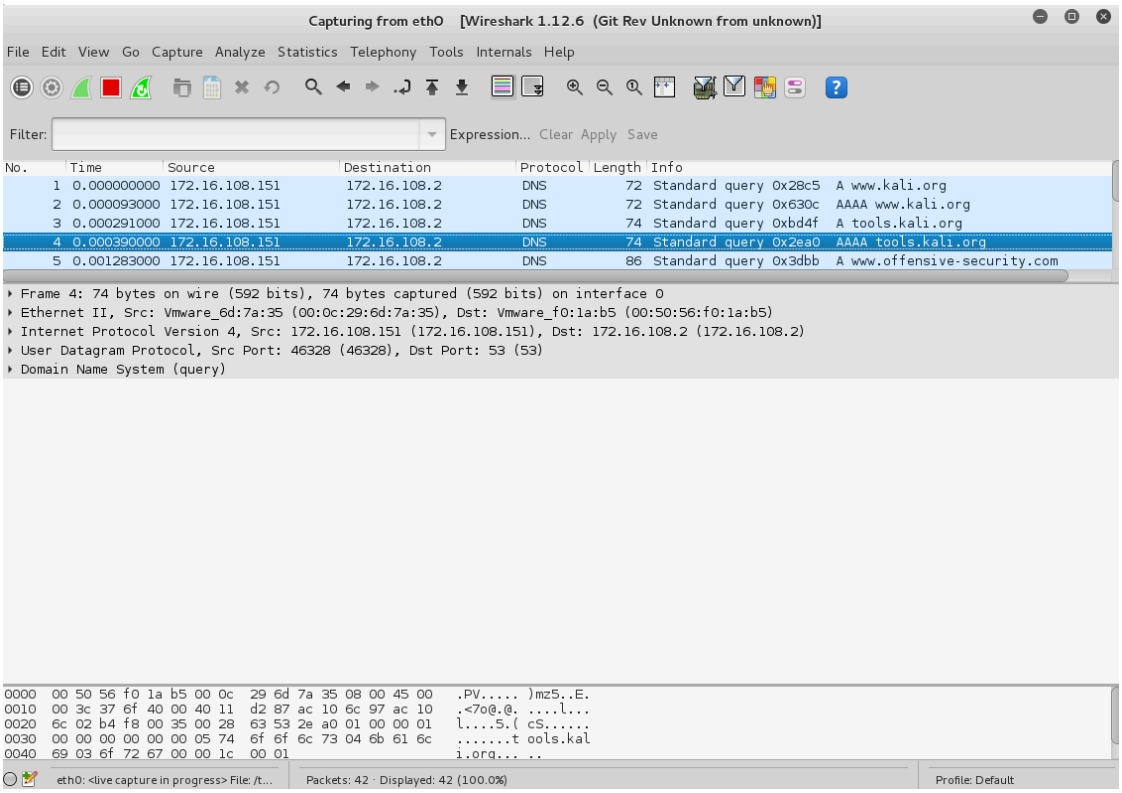


Figure 7: Capturing Packets in Wireshark

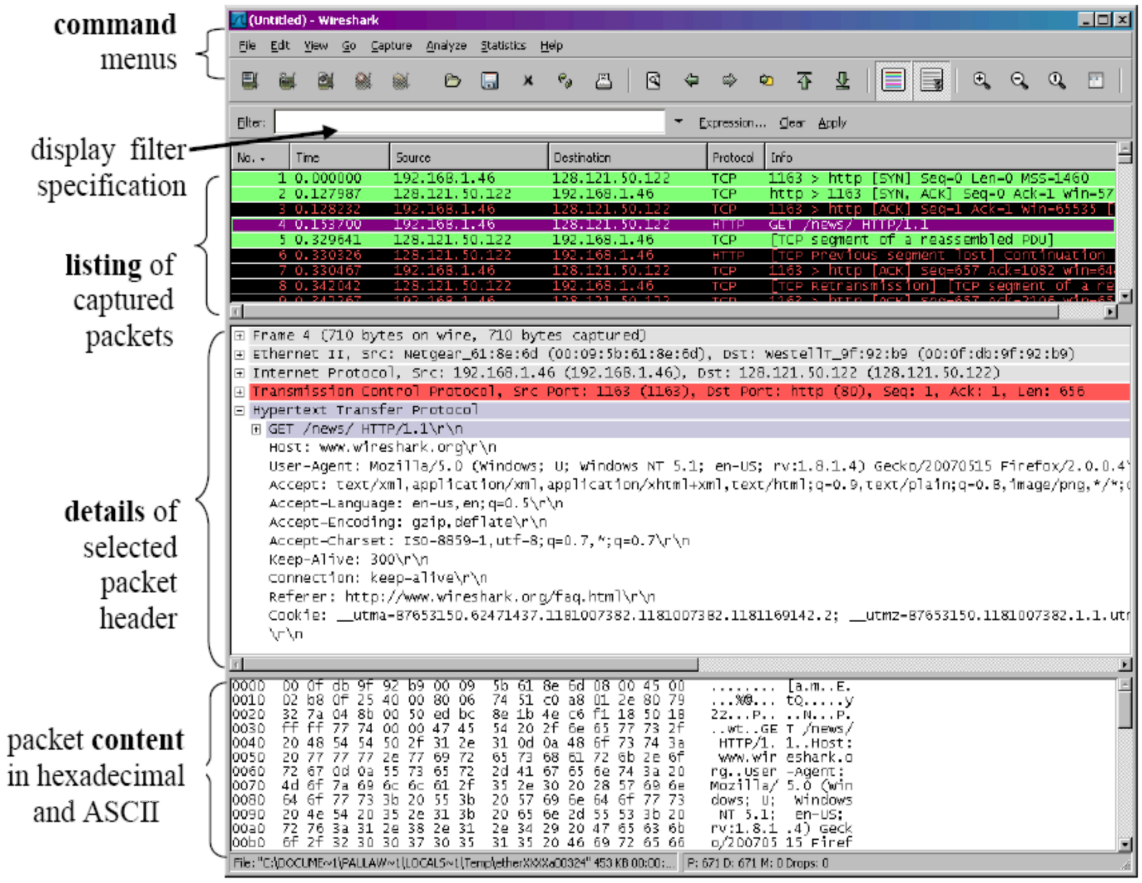


Figure 8: Wireshark Graphical User Interface on Microsoft Windows

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 is 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 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 right-pointing or down-pointing arrowhead 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.

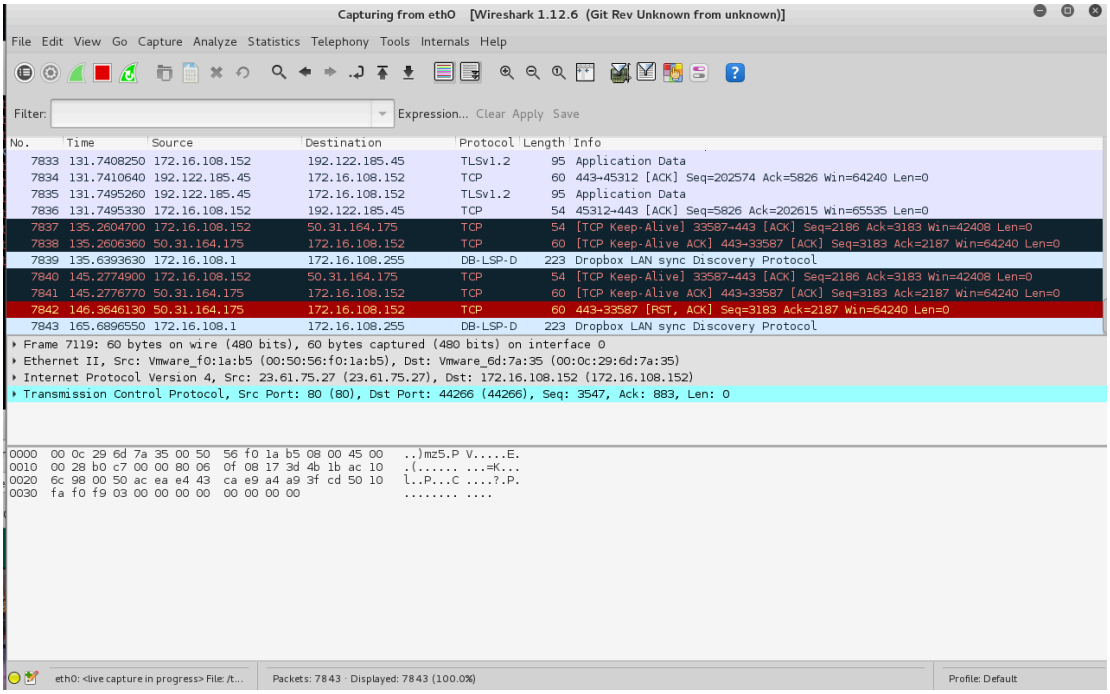
Capturing Packets

After downloading and installing Wireshark, you can launch it and click the name of an interface under Interface List to start capturing packets on that interface. For example, if you want to capture traffic on the wireless network, click your wireless interface.

Test Run

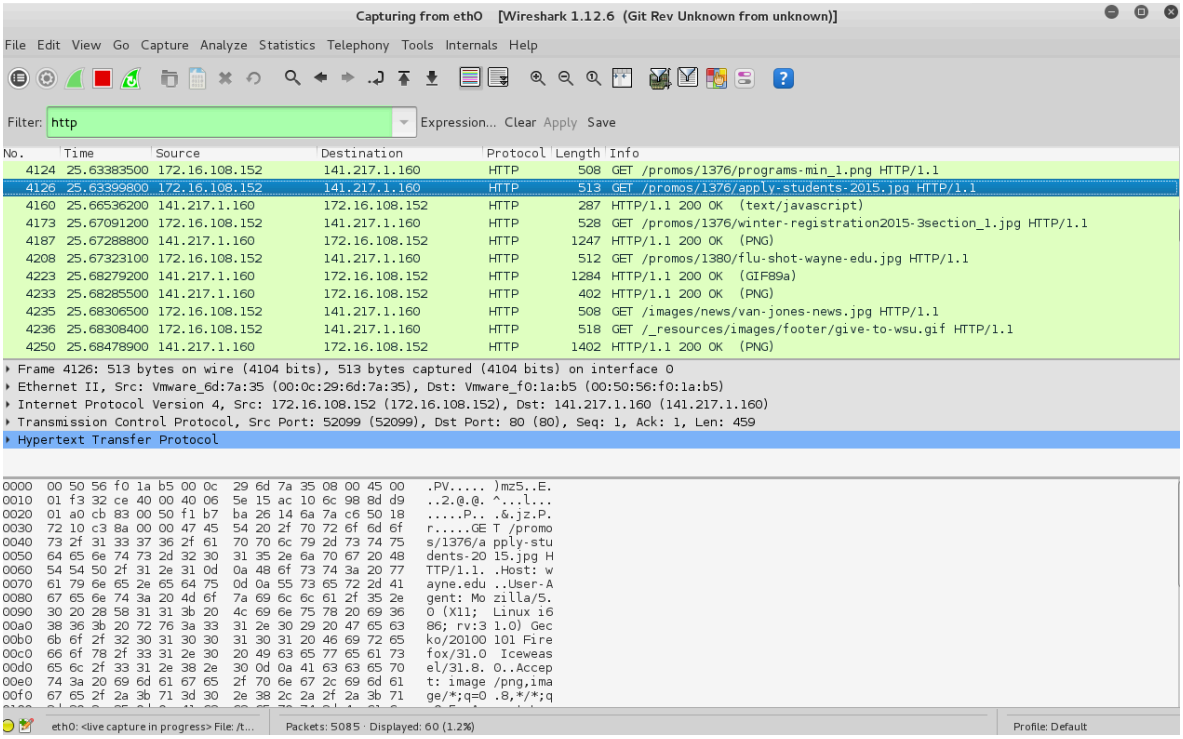
Do the following steps:

1. Start up the Wireshark program (select an interface and press start to capture packets).
2. Start up your favorite browser (ceweasel in Kali Linux).
3. In your browser, go to Wayne State homepage by typing www.wayne.edu.
4. After your browser has displayed the http://www.wayne.edu 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 see image below:



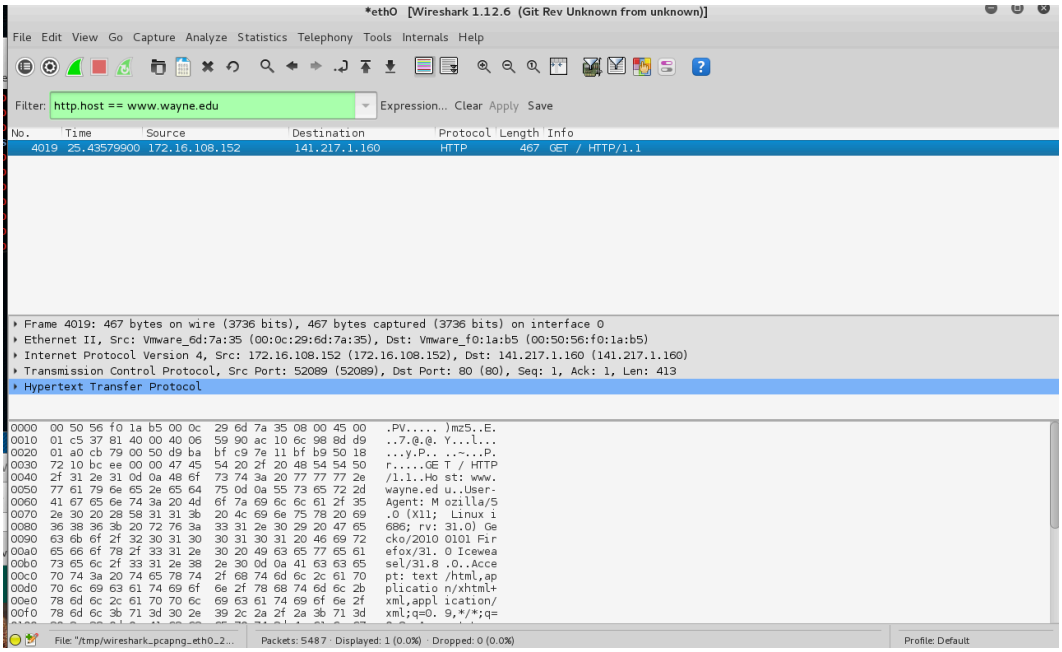
5. Color Coding: You'll probably see packets highlighted in green, blue, and black. Wireshark uses colors to help you identify the types of traffic at a glance. By default, green is TCP traffic, dark blue is DNS traffic, light blue is UDP traffic, and black identifies TCP packets with problems — for example, they could have been delivered out-of-order.

6. You now have live packet data that contains all protocol messages exchanged between your computer and other network entities! However, as you will notice the HTTP messages are not clearly shown because there are many other packets included in the packet capture. Even though the only action you took was to open your browser, there are many other programs in your computer that communicate via the network in the background. To filter the connections to the ones we want to focus on, we have to use the filtering functionality of Wireshark by typing "http" in the filtering field as shown below:

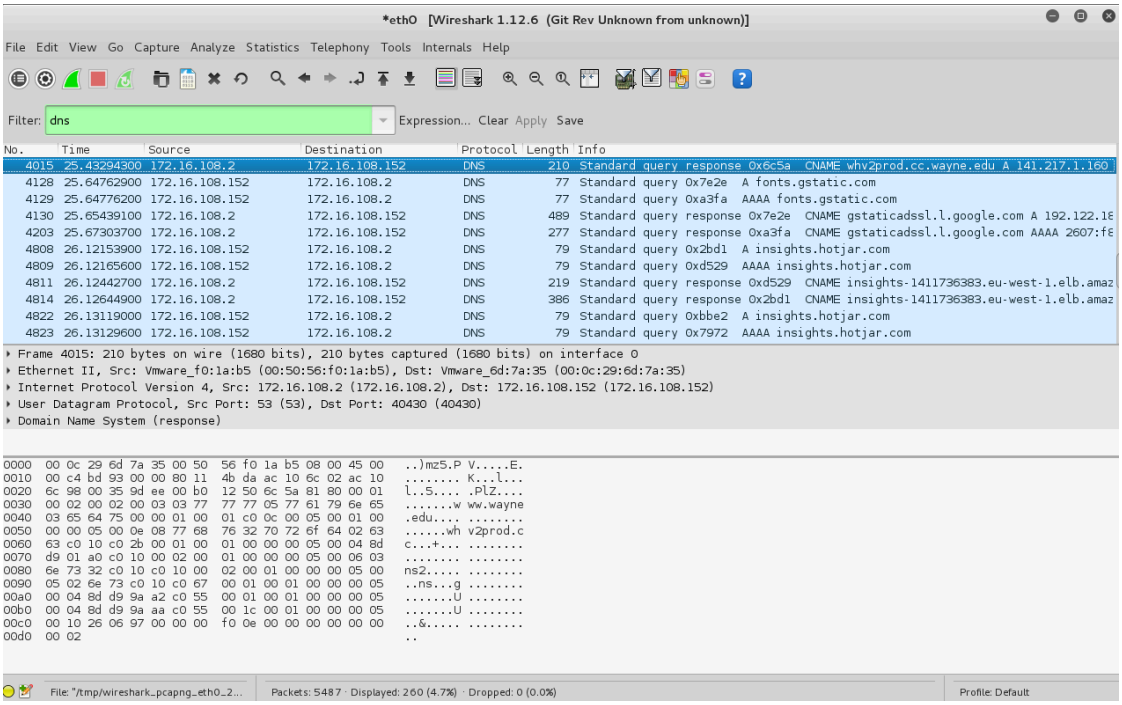


Notice that we now view only the packets that are of protocol HTTP. However, we also still do not have the exact communication we want to focus on because using HTTP as a filter is not descriptive enough to allow us to find our connection to http://www.wayne.edu. We need to be more precise if we want to capture the correct set of packets.

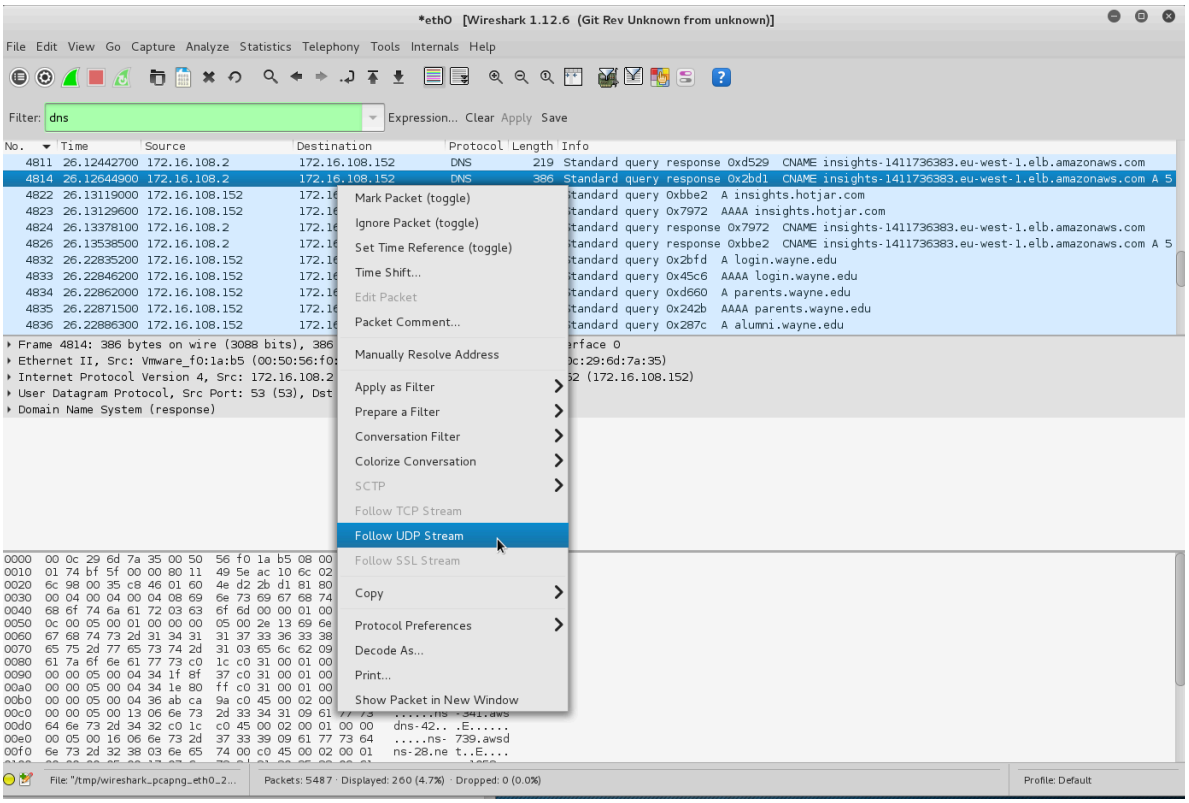
7. To further filter packets in Wireshark, we need to use a more precise filter. By setting the http.host==www.wayne.edu, we are restricting the view to packets that have as an http host the www.wayne.edu website. Notice that we need two equal signs to perform the match "==" not just one. See the screenshot below:



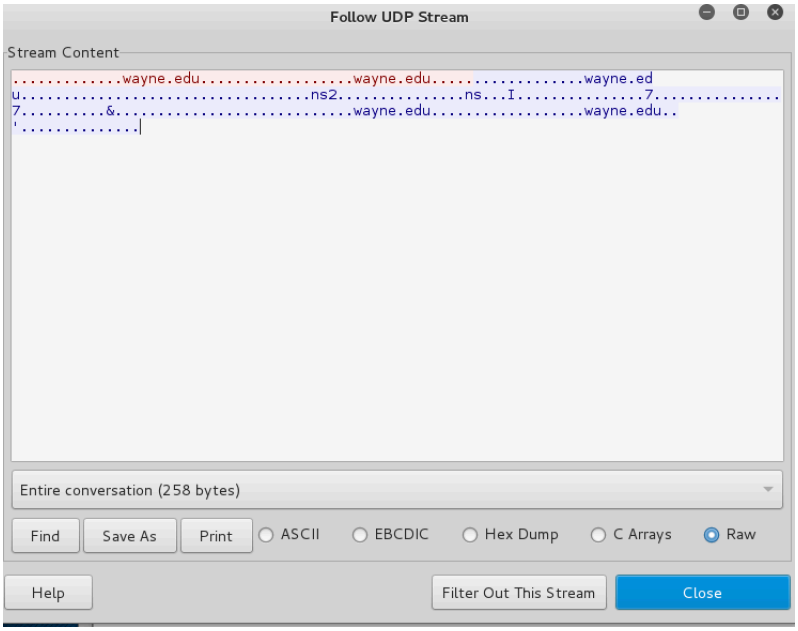
8. Now, we can try another protocol. Let's use Domain Name System (DNS) protocol as an example here.



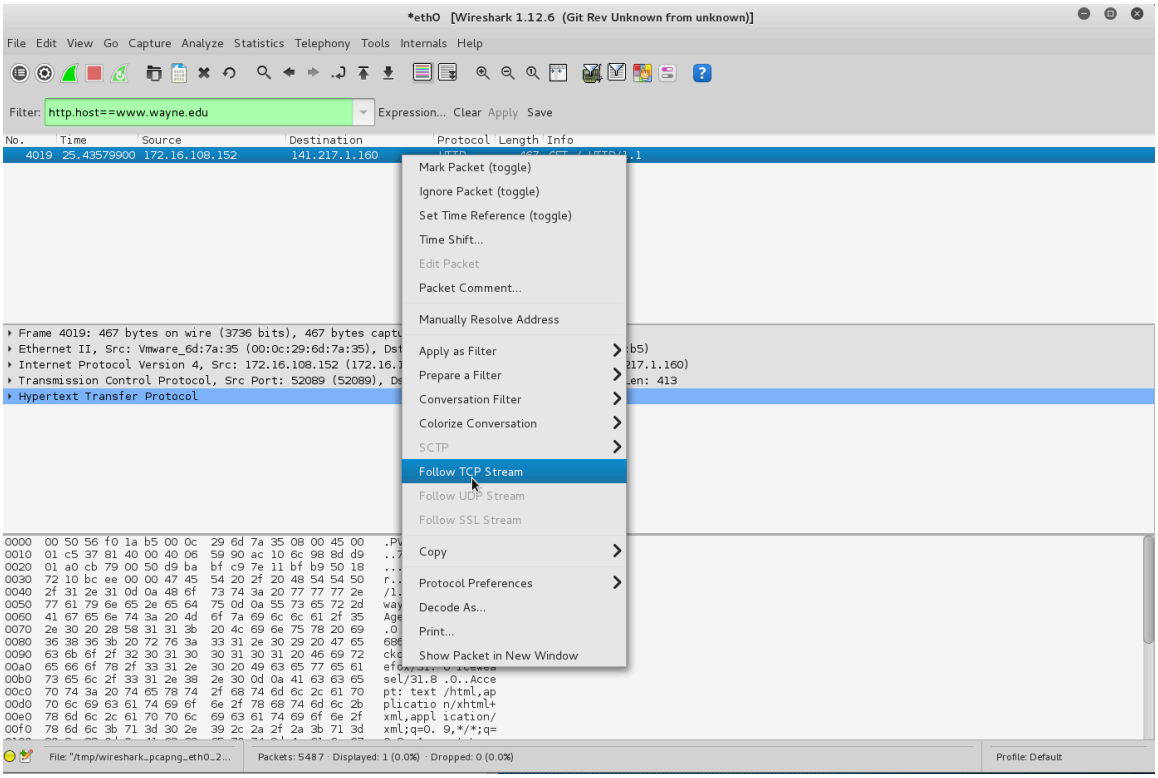
9. Let's try now to find out what are those packets contain by following one of the conversations (also called network flows), select one of the packets and press the right mouse button (if you are on a Mac use the command button and click), you should see something similar to the screen below:

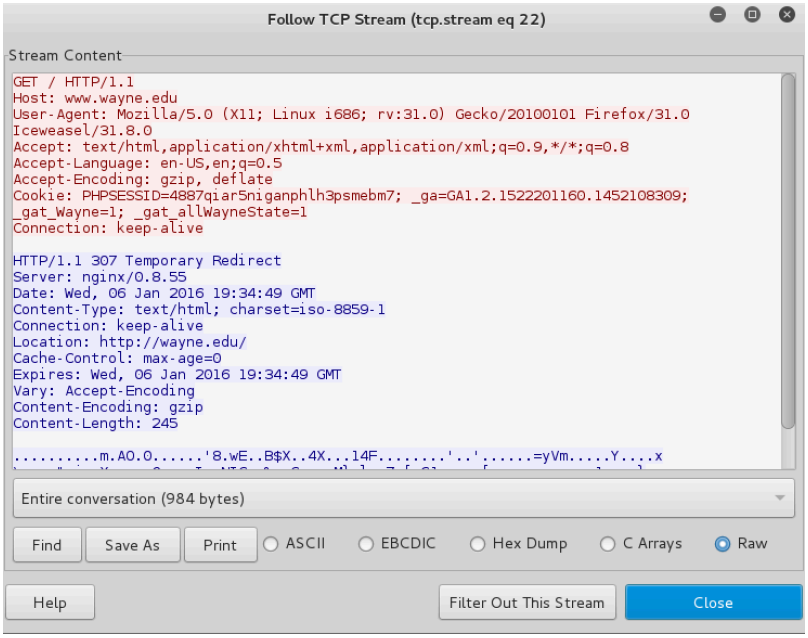


Click on **Follow UDP Stream,** and then you will see following screen.



10. If we close this window and change the filter back to "http.host==www.wayne.edu" and then follow a packet from the list of packets that match that filter, we should get the something similar to the following screens. Note that we click on **Follow TCP Stream** this time.





Questions for the Lab

1. Carefully read the lab instructions and finish all tasks above.
2. If a packet is highlighted by black, what does it mean for the packet?
3. What is the filter command for listing all outgoing http traffic?
4. Why does DNS use Follow UDP Stream while HTTP use Follow TCP Stream?
5. Using Wireshark to capture the FTP password.

   There is a FTP server installed on the Kali Linux VM. You need to use a terminal to log into the server and use Wireshark to capture the password. The username for the FTP server is csc5991-student, and the password is [TBA in the Class]. You will user the username and password to login the FTP server while Wireshark is running. Note that the FTP server is installed on the localhost, make sure you select the right interface for the capturing. You need to explain to me how you find the password and a screenshot of the password packet. Have fun!