**Name of the student: Shubham Mankar**

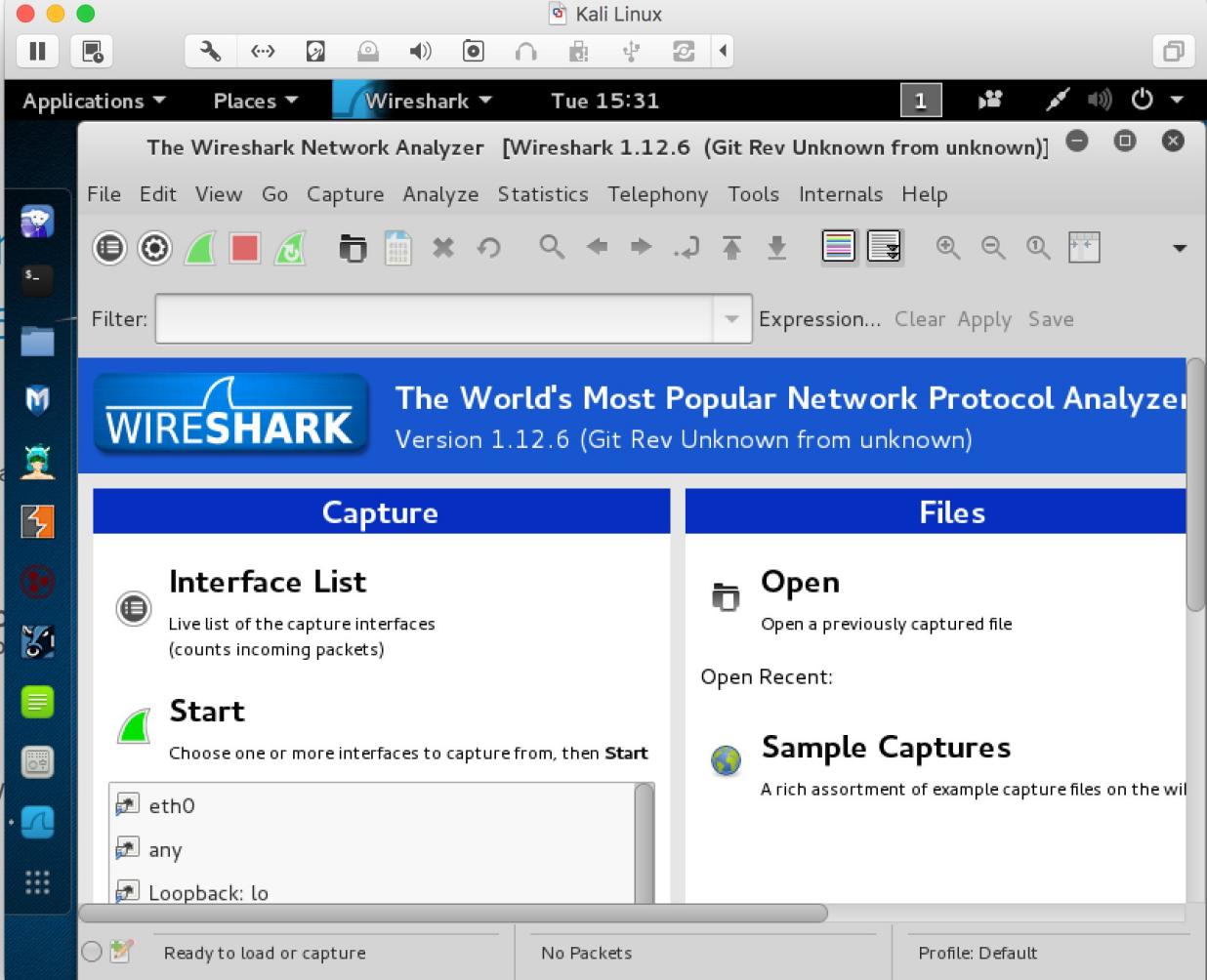
**Roll No. of the student: A73**

**Batch: A4**

**EXPERIMENT NO. 11**

**Packet Sniffing and Wireshark**

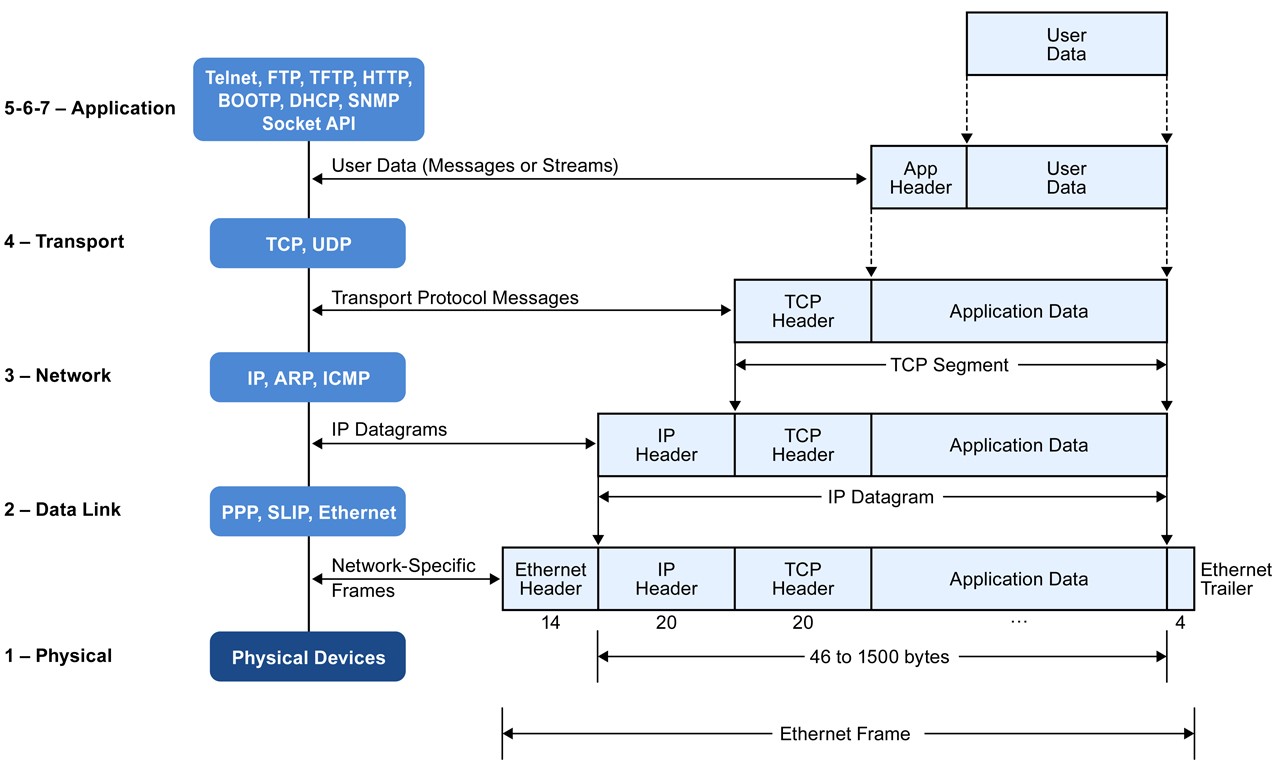
# *Introduction*

The first part of the lab introduces packet sniffer, Wireshark. Wireshark is a free opensource 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

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.

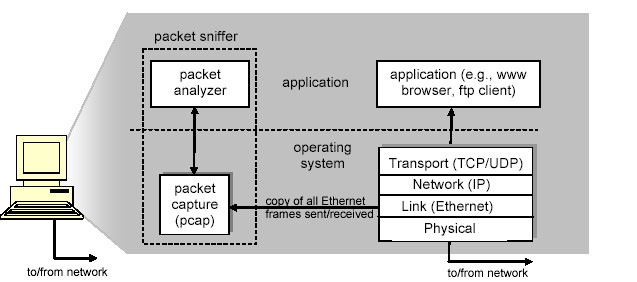
*Transport Layer*: The transport layer establishes process-to-process connectivity, and it provides end-to-end services that are independent of underlying user data.

*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.

*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.

## 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.

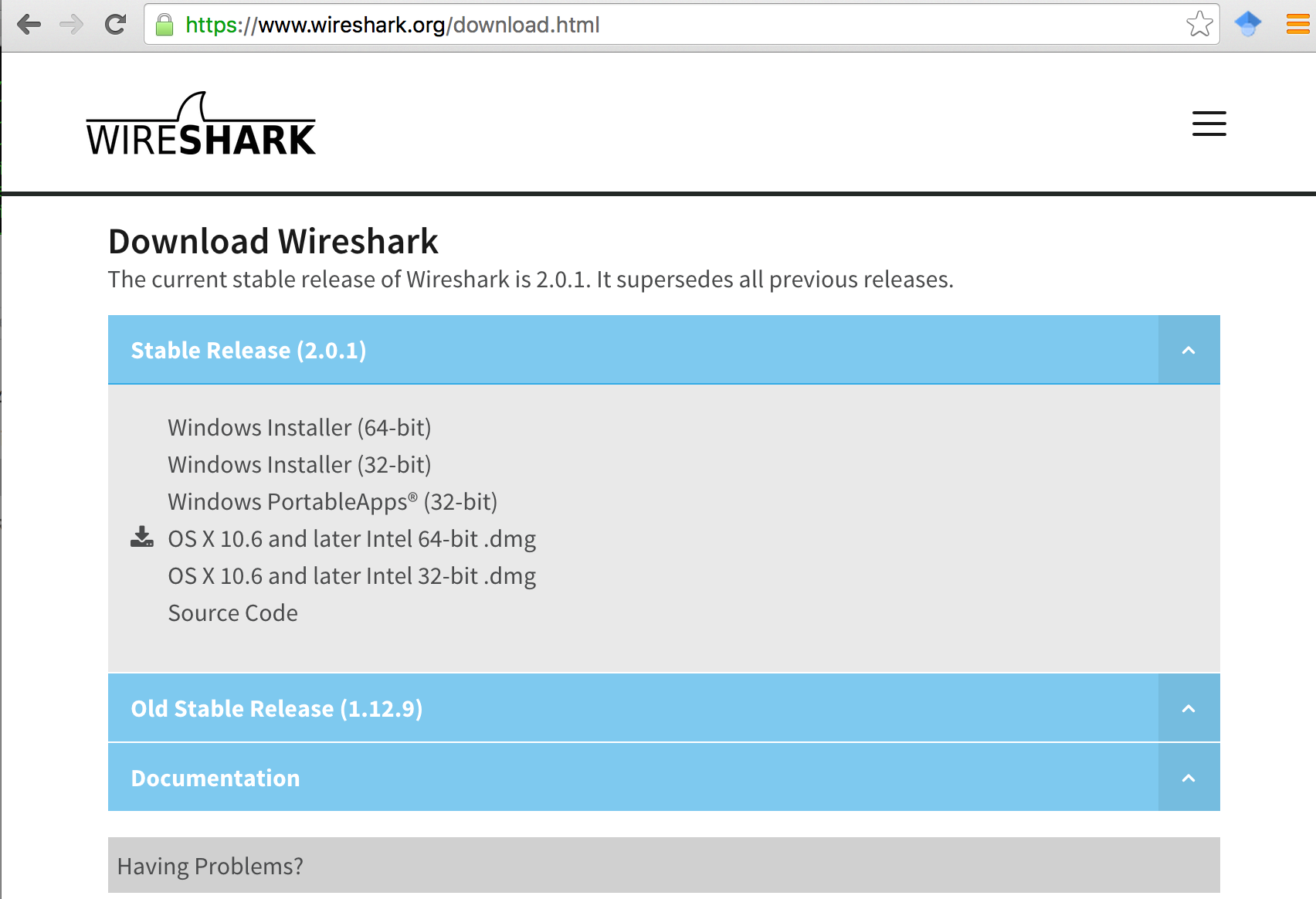


**Figure 3**: Packet Sniffer Structure

# 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:

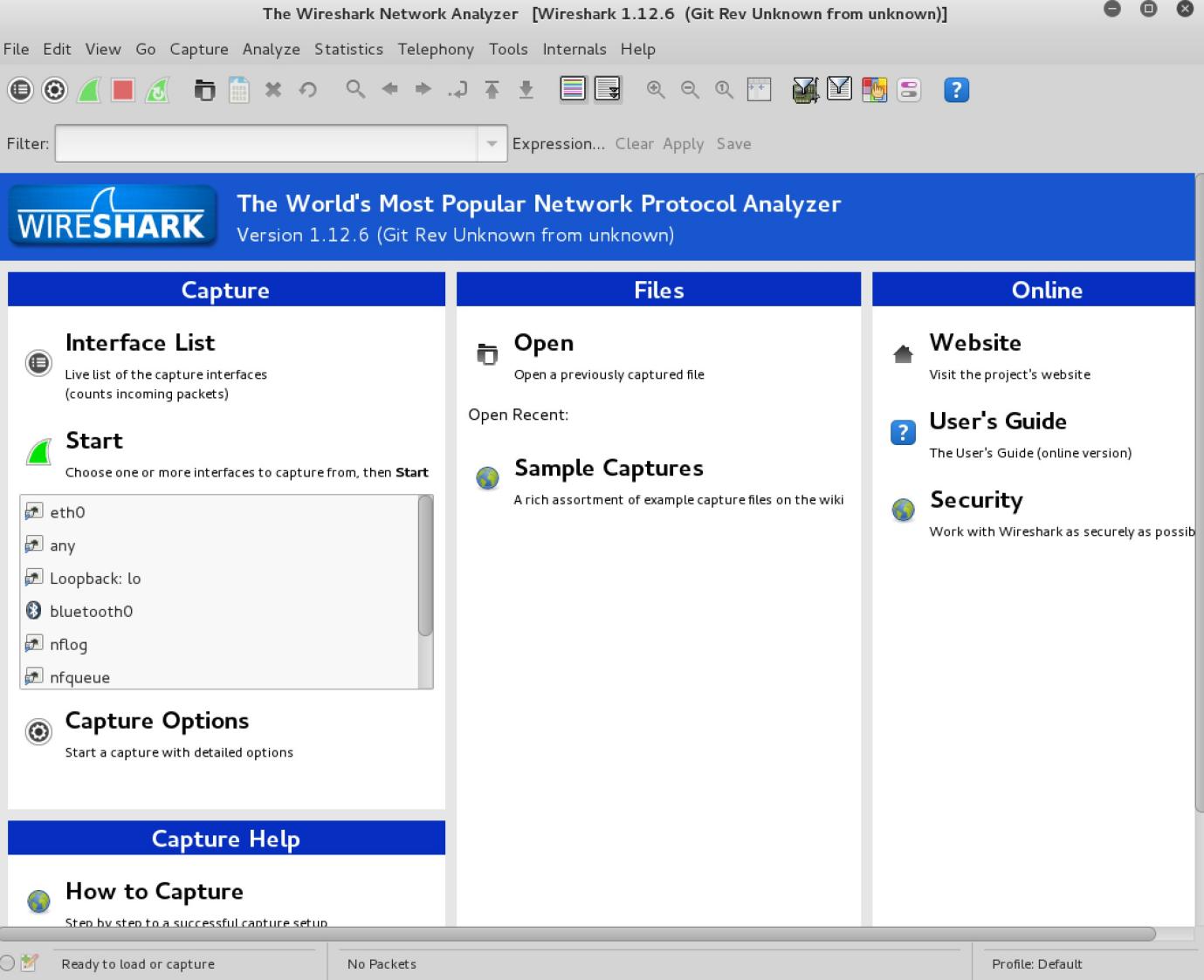
https://www.wireshark.org/download.html



**Figure** 4: Download Page of Wireshark

# Starting Wireshark

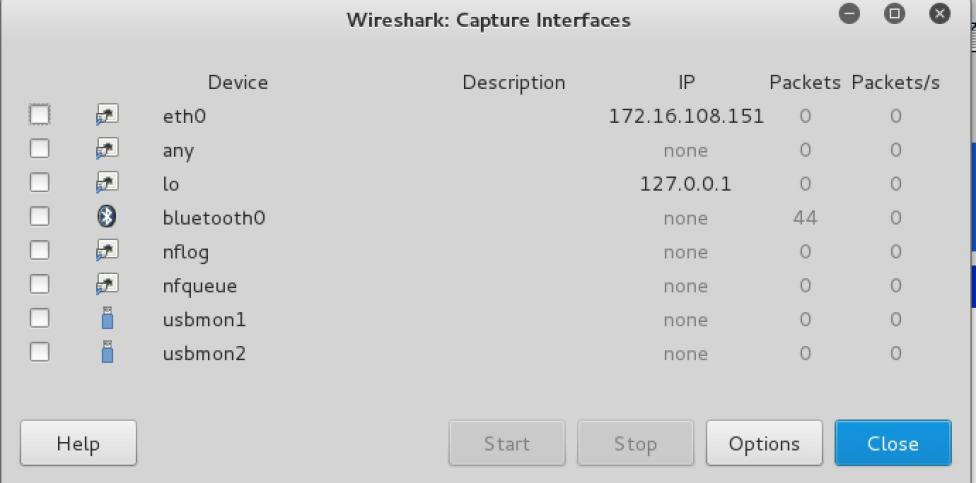
When you run the Wireshark program, the Wireshark graphic user interface will be shown as **Figure** 5. Currently, the program is not capturing the packets.



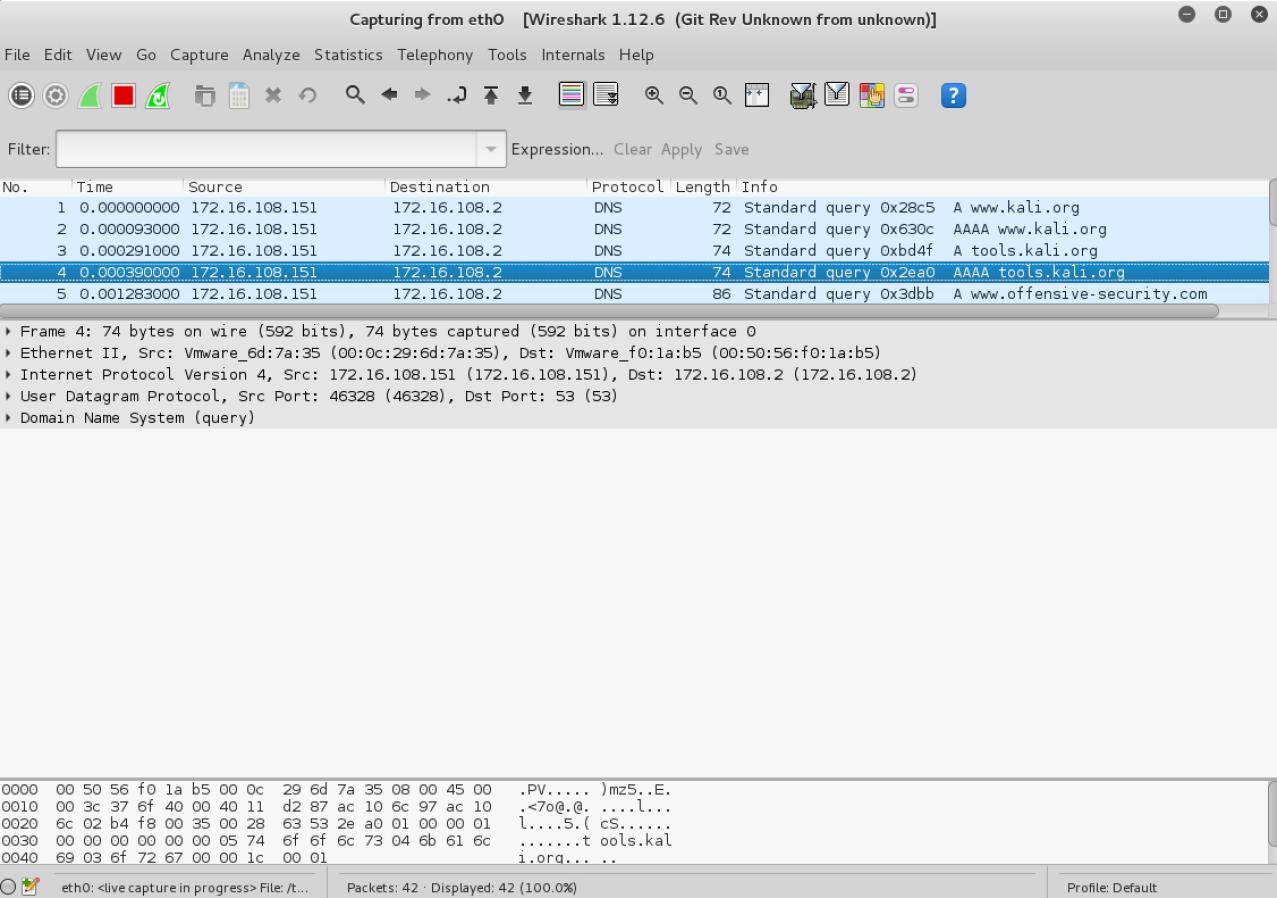
**Figure** 5: Initial Graphic User Interface of Wireshark

Then, you need to choose an interface

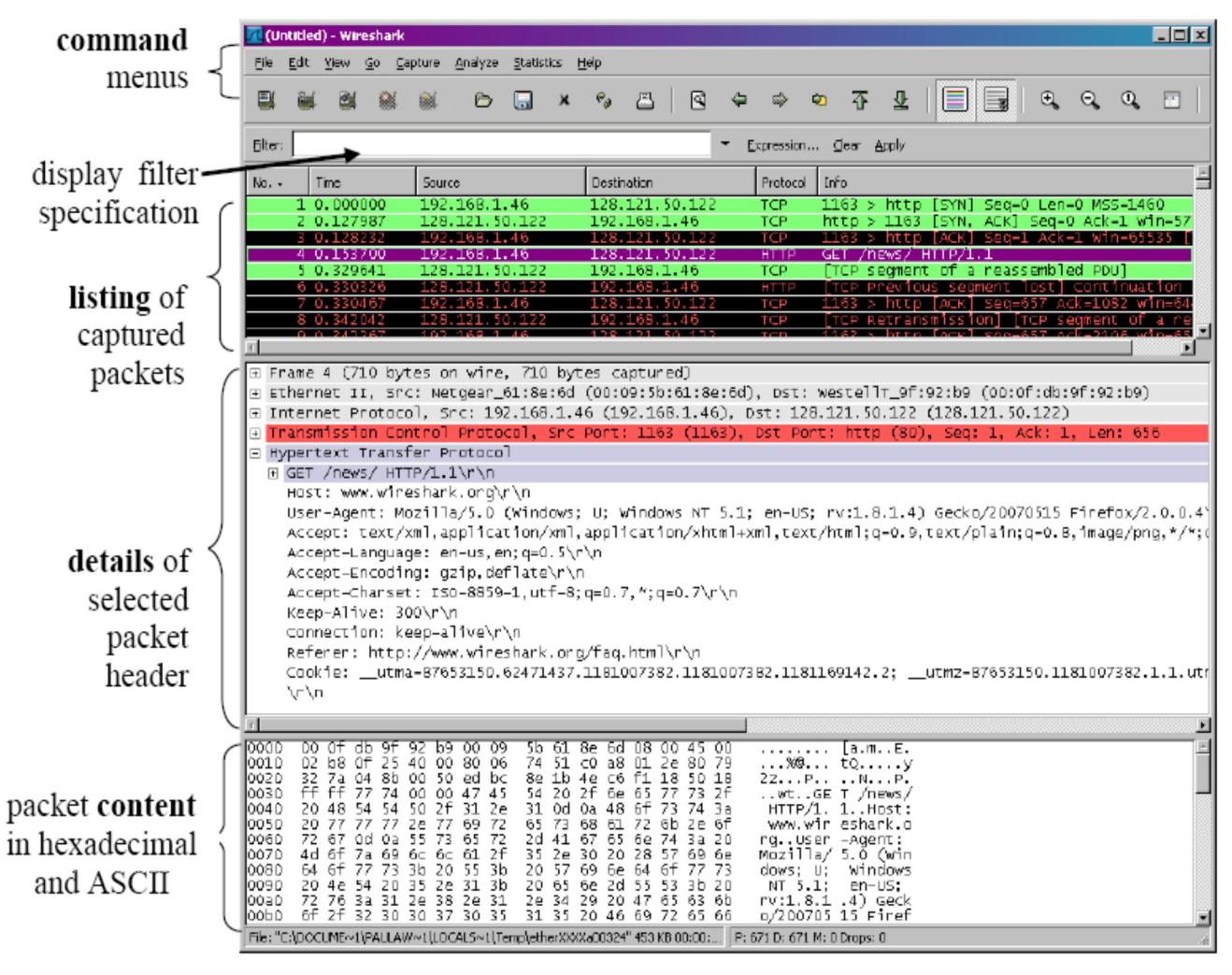
After you select the interface, you can click start to capture the packets as shown in **Figure** 7.



**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 **packet-listing window** displays a one-line summary for each packet captured, including the packet number, 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-header details window** provides details about the packet selected (highlighted) in the packet-listing window. These details include information about the Ethernet frame and IP datagram that contains this packet. 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.

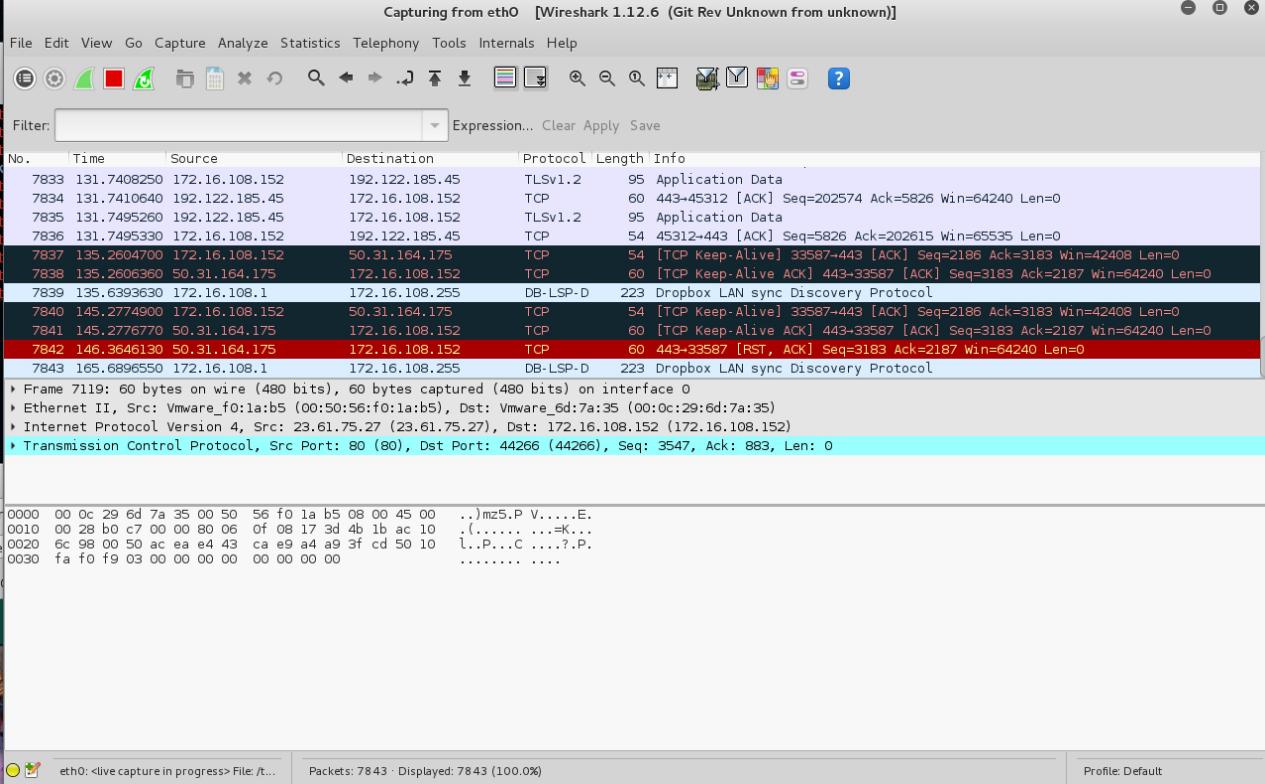
# Capturing Packets

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.

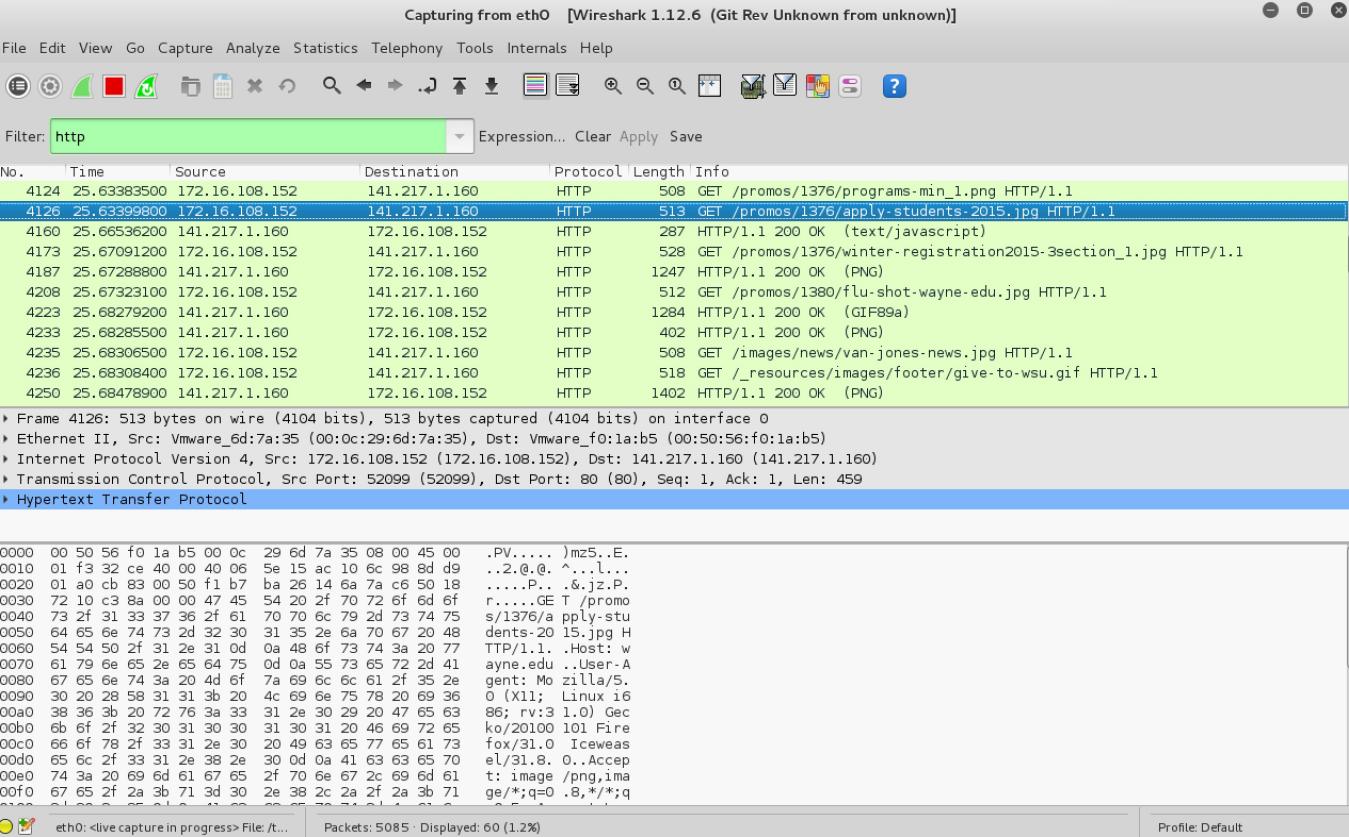
## Test Run

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:

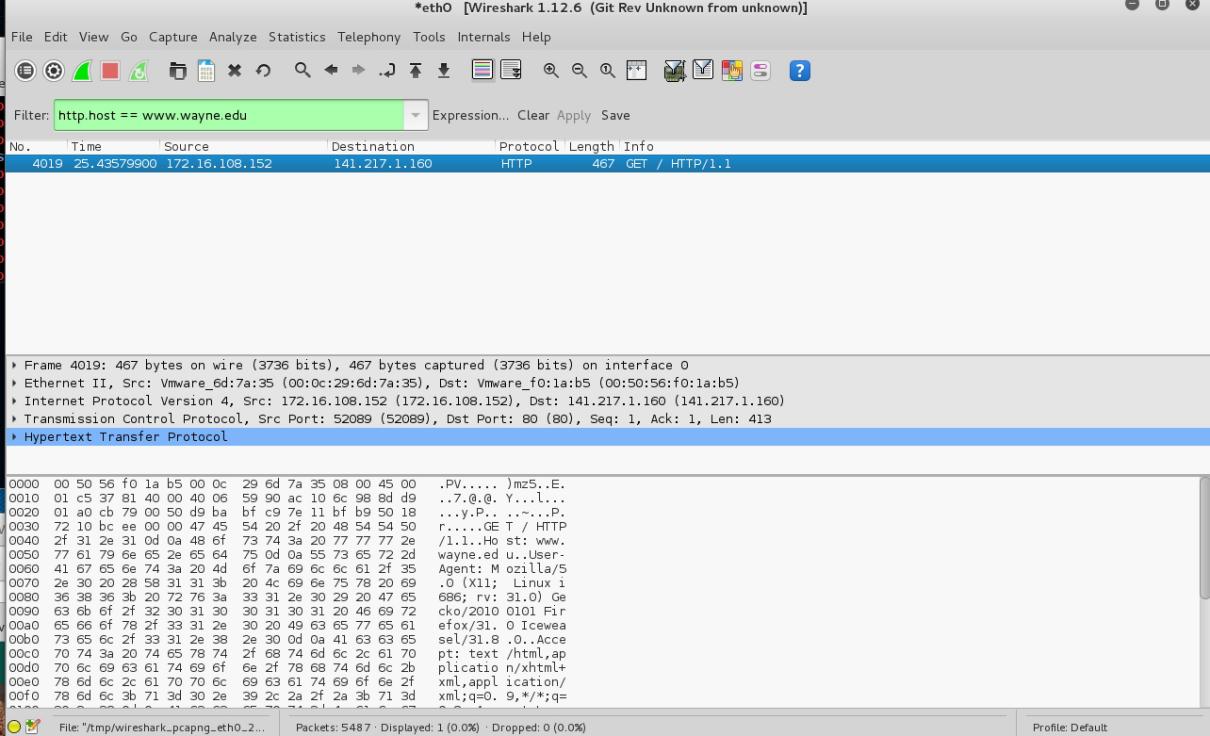


1. Color Coding: You’ll probably see packets highlighted in green, blue, and black. Wireshark uses colors to help you identify the types of traffic at a glance. By default, green is TCP traffic, dark blue is DNS traffic, light blue is UDP traffic, and black identifies TCP packets with problems — for example, they could have been delivered out-of-order.
2. 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:

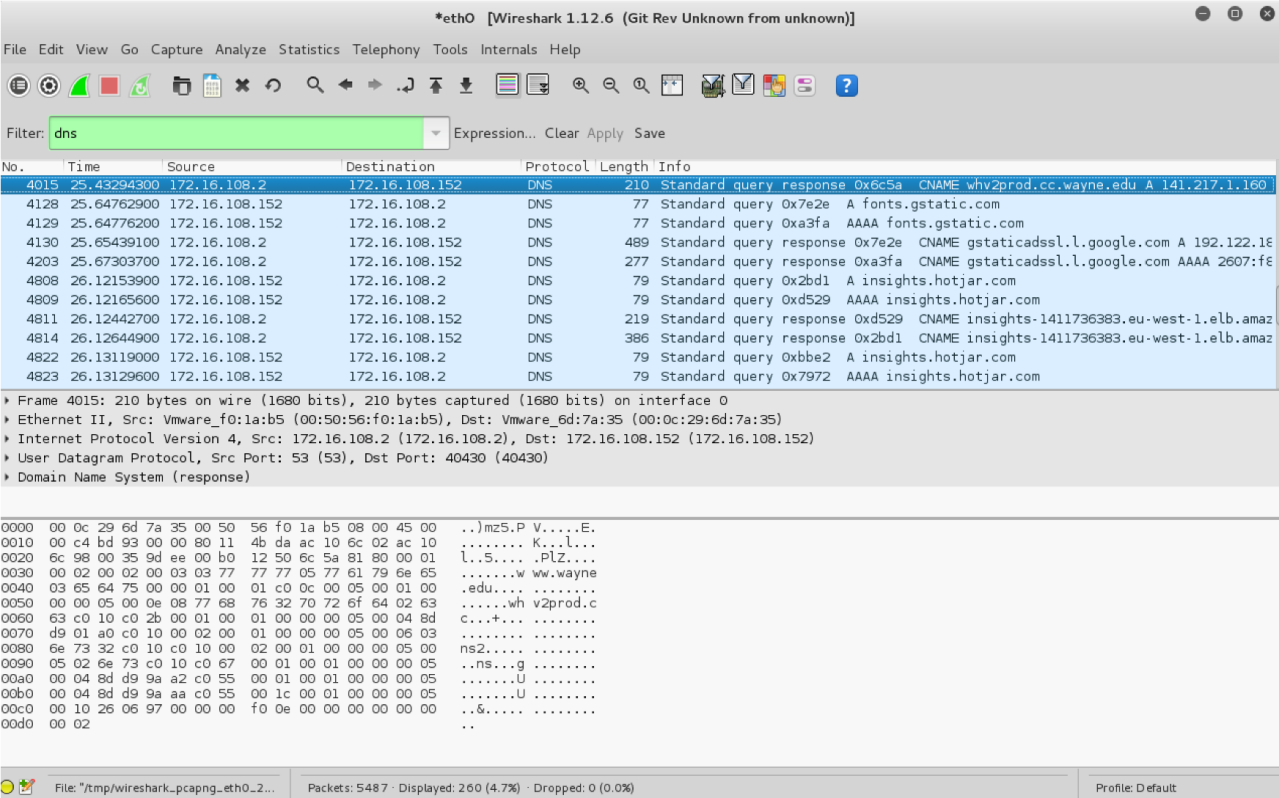


Notice that we now view only the packets that are of protocol HTTP. However, we also still do not have the exact communication we want to focus on because using HTTP as a filter is not descriptive enough to allow us to find our connection to http://www.wayne.edu. We need to be more precise if we want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

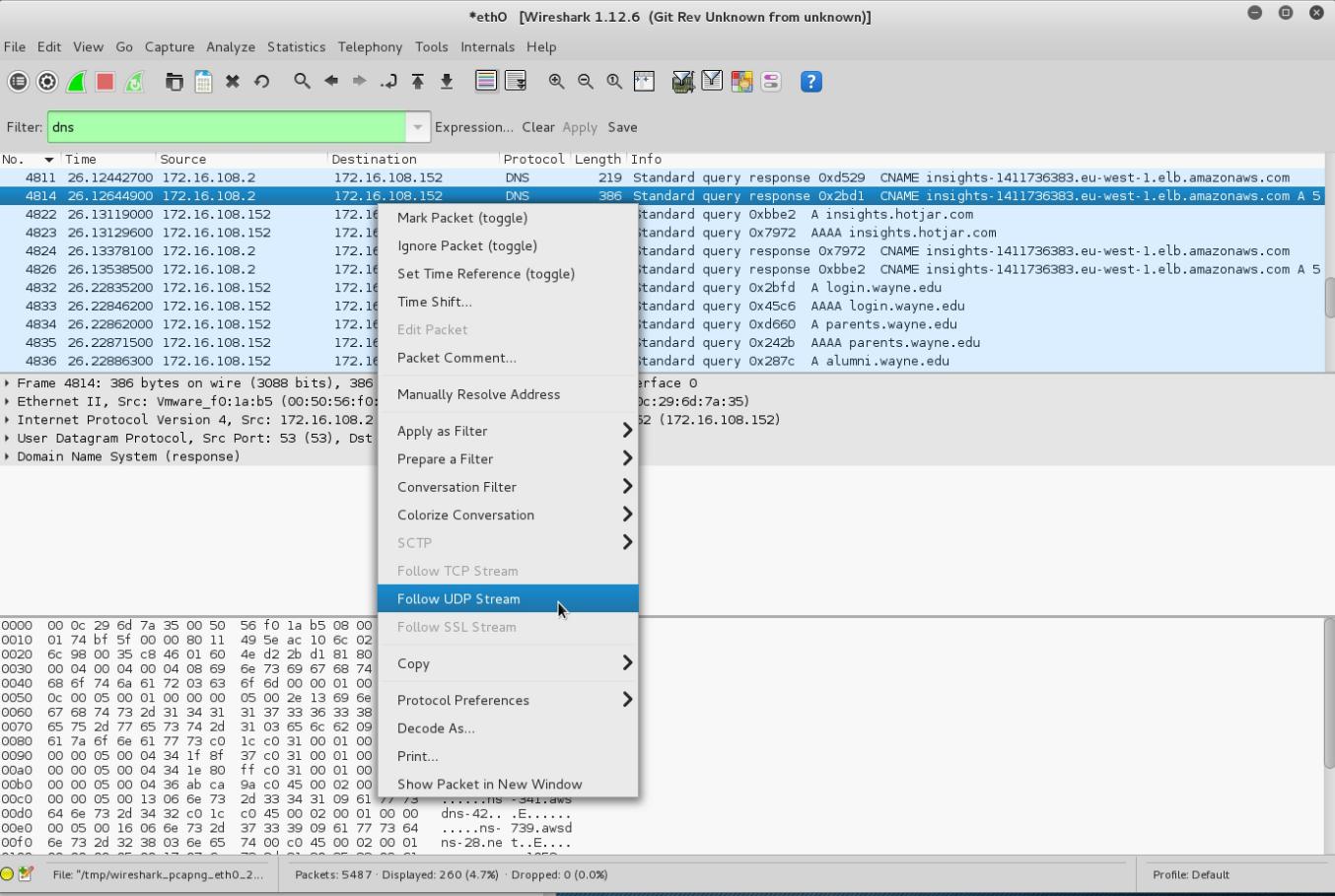
1. To further filter packets in Wireshark, we need to use a more precise filter. By setting the http.host==www.wayne.edu, we are restricting the view to packets that have as an http host the www.wayne.edu website. Notice that we need two equal signs to perform the match “==” not just one. See the screenshot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



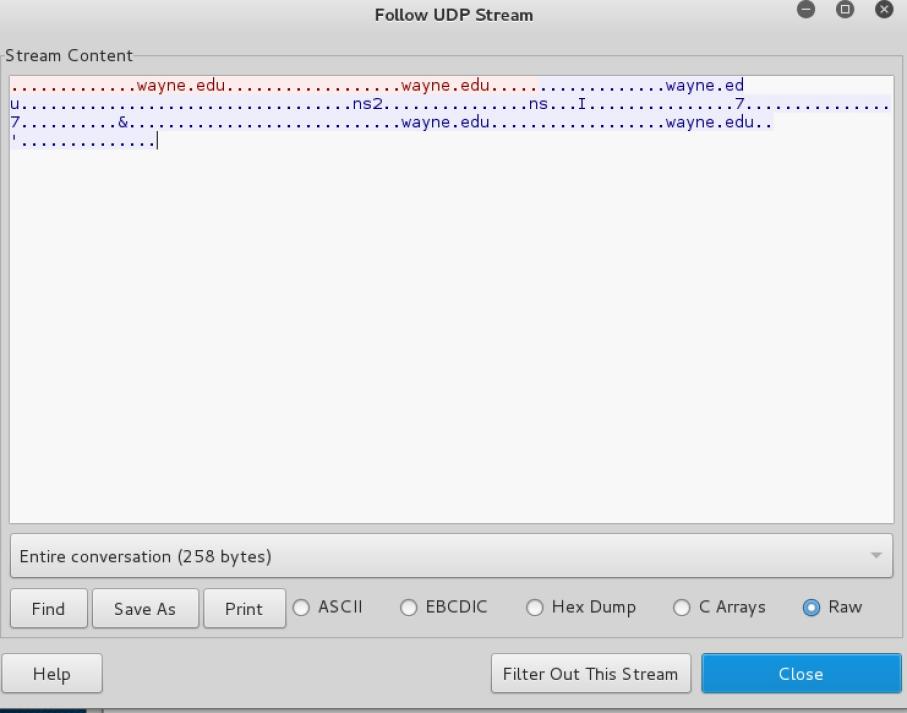
1. Now, we can try another protocol. Let’s use Domain Name System (DNS) protocol as an example here.



1. 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.



1. 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.

