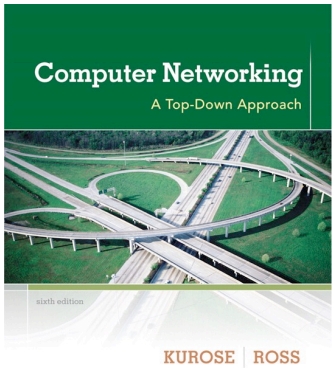
COMP 9121 Week 6
Wireshark Experiment on Ethernet Protocol and ARP

Supplement to *Computer Networking: A Top-Down Approach, 6th ed.,* J.F. Kurose and K.W. Ross

http://www-net.cs.umass.edu/wireshark-labs/



In this lab, we'll investigate the Ethernet protocol and the ARP protocol by Wireshark, a Free and open source packet analyzer. Before the lab, please read the brief introduction of Wireshark at https://en.wikipedia.org/wiki/Wireshark.

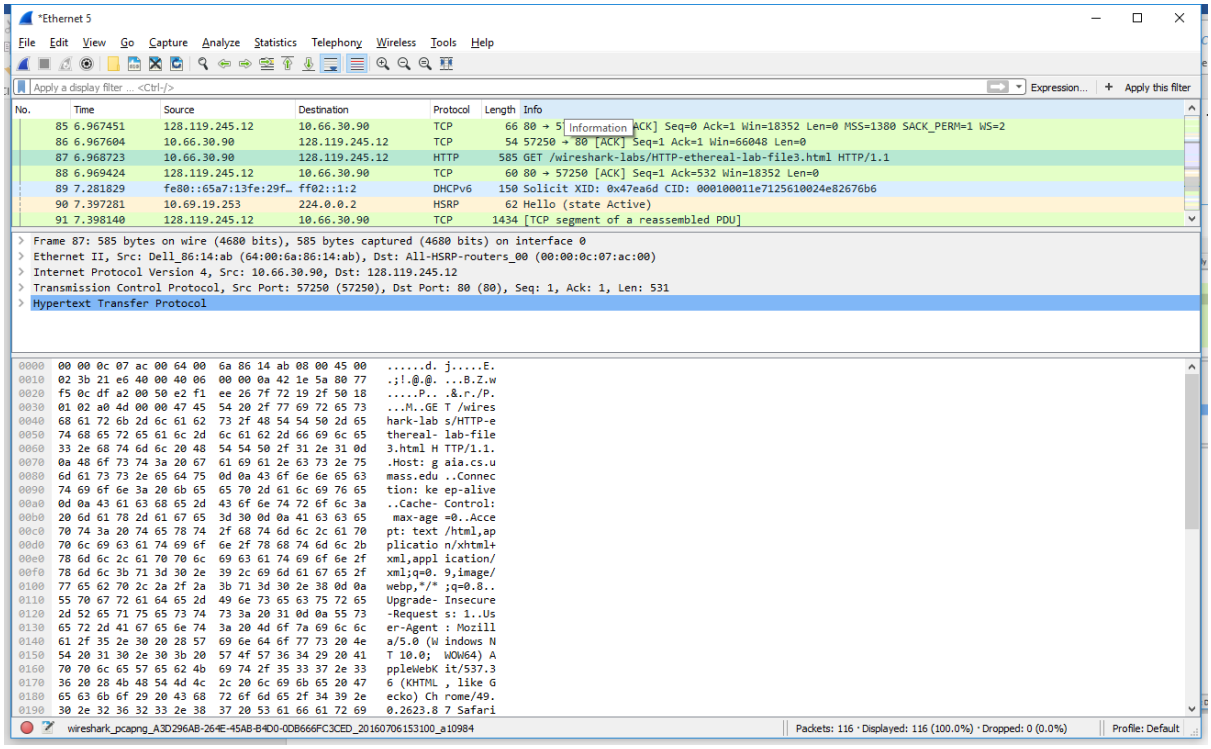
Let's begin by capturing a set of Ethernet frames to study.

First, make sure your browser's cache is empty. To do this under Mozilla Firefox, select *Tools->Clear Recent History* and check the box for Cache. For Internet Explorer, select *Tools->Internet Options->Delete Files.* Start up the Wireshark packet sniffer

Enter the following URL into your browser http://gaia.cs.umass.edu/wireshark-labs/HTTP-ethereal-lab-file3.html Your browser should display the rather lengthy US Bill of Rights.

Stop Wireshark packet capture. First, find the packet numbers (the leftmost column in the upper Wireshark window) of the HTTP GET message that was sent from your computer to gaia.cs.umass.edu, as well as the beginning of the HTTP response message sent to your computer by gaia.cs.umass.edu. You should see a screen that looks something like this (where packet 87 in the screen shot below contains the HTTP GET message).

If you are unable to run Wireshark live on a computer, you can download trace file lab-trace-1 in the Canvas.



In order to answer the following questions, you'll need to look into the packet details and packet contents windows (the middle and lower display windows in Wireshark).

Select the Ethernet frame containing the HTTP GET message. (HTTP GET message is carried inside of a TCP segment, which is carried inside of an IP datagram, which is carried inside of an Ethernet frame). Expand the Ethernet II information in the packet details window. Note that the contents of the Ethernet frame (header as well as payload) are displayed in the packet contents window.

Answer the following questions, based on the contents of the Ethernet frame containing the HTTP GET message.

1. What is the 48-bit Ethernet address of your computer?
In the trace, 00:d0:59:a9:3d:68
2. What is the 48-bit destination address in the Ethernet frame? Is this the Ethernet address of gaia.cs.umass.edu? (Hint: the answer is *no*). What device has this as its Ethernet address?
No. First-hop router.
3. Give the hexadecimal value for the two-byte Frame type field. What upper layer protocol does this correspond to?
IPv4, 0x0800

Go to https://aruljohn.com/mac.pl, check the Vendor of the network interface card of your computer.
AMBIT MICROSYSTEMS CORP.

ARP Caching
Recall that the ARP protocol typically maintains a cache of IP-to-Ethernet address translation pairs on your computer. The *arp* command (in both MSDOS and Linux/Unix) is used to view

and manipulate the contents of this cache. Since the *arp* command and the ARP protocol have the same name, it's understandably easy to confuse them. But keep in mind that they are different - the *arp* command is used to view and manipulate the ARP cache contents, while the ARP protocol defines the format and meaning of the messages sent and received, and defines the actions taken on message transmission and receipt.

Let's take a look at the contents of the ARP cache on your computer:
• **MS-DOS.** The *arp* command is in c:\windows\system32, so type either "*arp*" or "*arp -a"* in the MS-DOS command line (without quotation marks).
• **Linux/Unix/MacOS.** The executable for the *arp* command can be in various places. Popular locations are /sbin/arp (for linux) and /usr/etc/arp (for some Unix variants).

The Windows *arp* command with no arguments will display the contents of the ARP cache on your computer. Run the *arp* command.

In order to observe your computer sending and receiving ARP messages, we'll need to clear the ARP cache, since otherwise your computer is likely to find a needed IP-Ethernet address translation pair in its cache and consequently not need to send out an ARP message.

• **MS-DOS.** The MS-DOS *arp –d* * command will clear your ARP cache. The *–d* flag indicates a deletion operation, and the * is the wildcard that says to delete all table entries.

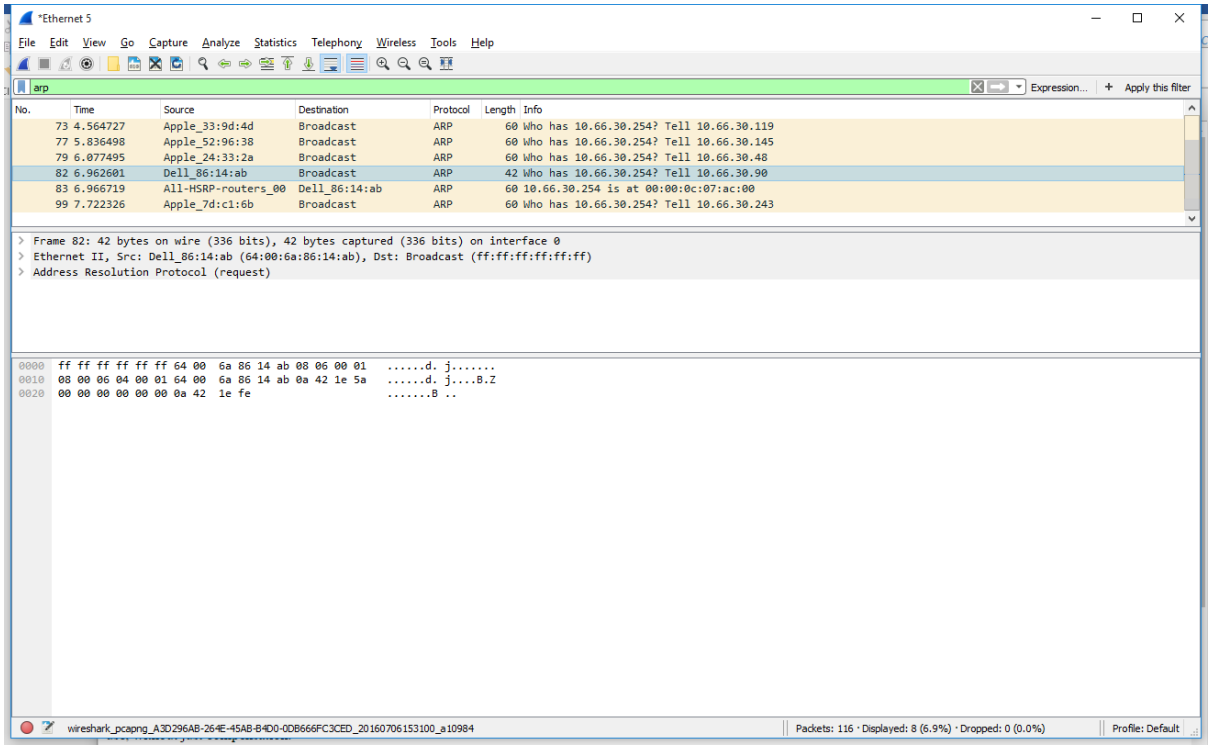
• **Linux/Unix/MacOS.** The *arp –d* * will clear your ARP cache. In order to run this command you'll need root privileges. If you don't have root privileges and can't run Wireshark on a Windows machine, you can skip the trace collection part of this lab and just use the trace discussed in the earlier footnote.

Clear your ARP cache, as described above.
• Next, make sure your browser's cache is empty.
• Start up the Wireshark packet sniffer
• Enter the following URL into your browser
http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-lab-file3.html
Your browser should again display the rather lengthy US Bill of Rights.

• Stop Wireshark packet capture.

Since this lab is about Ethernet and ARP, so let's change Wireshark's "listing of captured packets" window so that it shows information only about protocols below IP. To have Wireshark do this, input ARP at the display filter.

You should now see a Wireshark window that looks like:



4. What are the hexadecimal values for the source and destination addresses in the Ethernet frame containing the ARP request message? Give the hexadecimal value for the two-byte Ethernet Frame type field. What upper layer protocol does this correspond to? Does the ARP message contain the IP address of the sender?

Source: Mac address of your computer.
Destination: ff:ff:ff:ff:ff:ff
ARP: 0x0806
Yes.
(In the trace, you can check Packet #1)

5. Now find the ARP reply that was sent in response to the ARP request. Where in the ARP message does the “answer” to the earlier ARP request appear – the IP address of the machine having the Ethernet address whose corresponding IP address is being queried?

You know the IP-MAC mapping from ARP reply (“sender IP, sender MAC”)
(In the trace, you can check Packet #2)