Wireshark Lab: Ethernet and ARP

In this last lab, we'll investigate the Ethernet protocol and the ARP protocol. Before beginning this lab, you'll probably want to review sections 6.4.2 (Ethernet) and 6.4.1 (link-layer addressing and ARP) in the text. RFC 826 (ftp://ftp.rfc-editor.org/in-notes/std/std37.txt) contains the gory details of the ARP protocol, which is used by an IP device to determine the IP address of a remote interface whose Ethernet address is known.

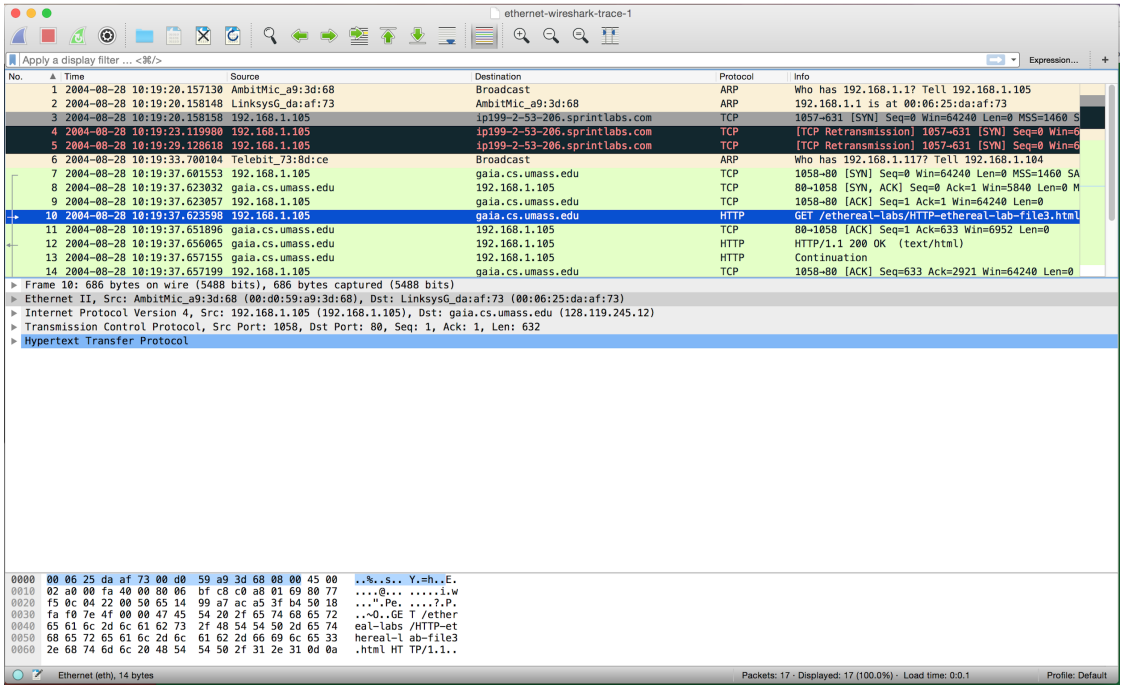
1. Capturing and analyzing Ethernet frames

The trace file that you will use for this lab was created by the following steps:

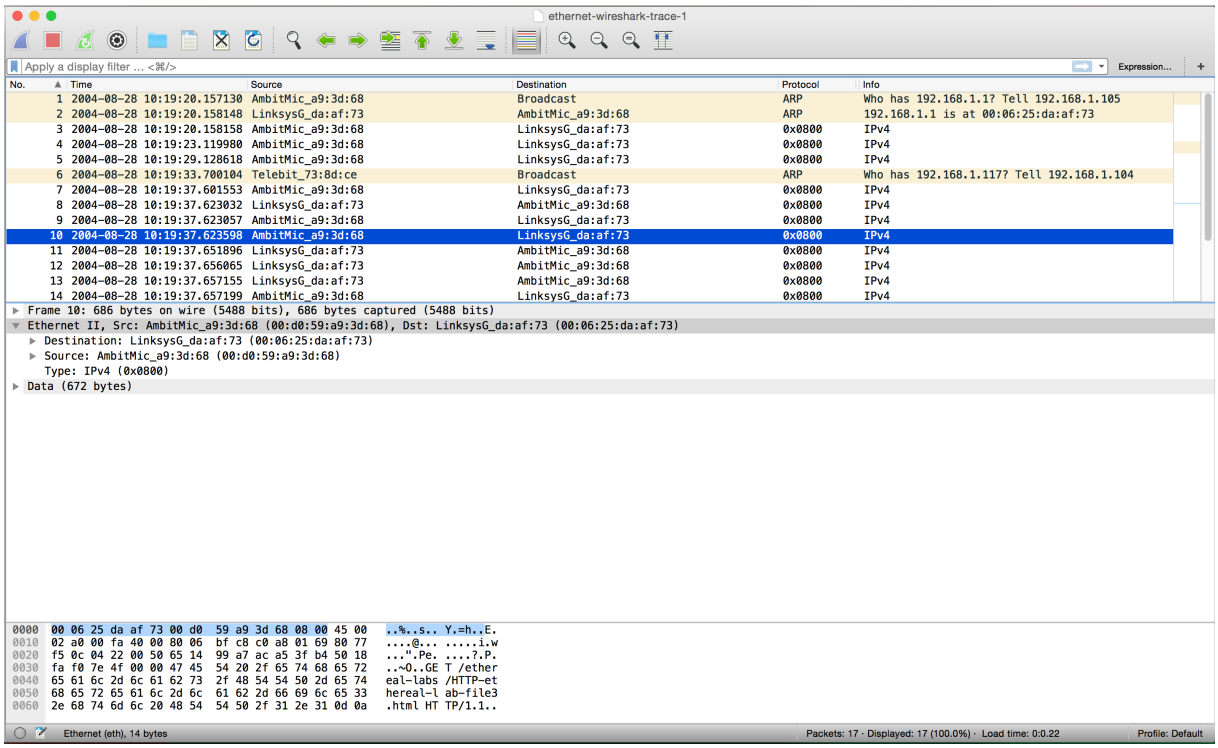
- The browser's cache where Wireshark was running was emptied. (This can be done on Firefox by selecting *Tools->Clear Recent History* and check the box for Cache.)
- The Wireshark packet sniffer was started.
- The following URL was entered into the browser: http://gaia.cs.umass.edu/Wireshark-labs/ HTTP-Wireshark-lab-file3.html
- The Wireshark packet capture was stopped.

Steps for you to take:

- First, find the packet numbers (the leftmost column in the upper Wireshark window) of the HTTP GET message that was sent from the client computer to gaia.cs.umass.edu, as well as the beginning of the HTTP response message sent to the client computer by gaia.cs.umass.edu. You should see a screen that looks like this (where packet 10 in the screen shot below contains the HTTP GET message)



- Since this lab is about Ethernet and ARP, we're not interested in IP or higher-layer protocols. 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, select *Analyze->Enabled Protocols.* Then uncheck the IP box and select *OK*. You should now see an Wireshark window that looks like:



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. (Recall that the 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; reread section 1.5.2 in the text if you find this encapsulation a bit confusing). 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 the client computer?
2. What is the 48-bit destination address in the Ethernet frame? Is this the Ethernet address of gaia.cs.umass.edu?

Next, answer the following questions, based on the contents of the Ethernet frame containing the first byte of the HTTP response message.

3. What is the value of the Ethernet source address? What device has this as its Ethernet address?
4. What is the destination address in the Ethernet frame? Is this the Ethernet address of the client computer?

2. The Address Resolution Protocol

In this section, we'll observe the ARP protocol in action. You should re-read section 6.4.1 in the text before proceeding.

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 "*c:\windows\system32\arp*" 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). On the csil.cs machine, typing "/usr/sbin/arp" will allow you to see the arp table of a csil machine.

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

5. Write down one entry of your computer's ARP cache. What is the meaning of each column value?

Observing ARP in action

The trace you are analyzing was actually generated by the following steps:

- The ARP cache at the client computer was cleared (through the command *arp –d* *, which you need root privileges to run).
- The browser's cache was emptied.
- The Wireshark packet sniffer was started.
- The following URL was entered into the browser: http://gaia.cs.umass.edu/Wireshark-labs/ HTTP-Wireshark-lab-file3.html
- Wireshark packet capture was stopped.

Steps for you to take:
Find the ARP Request and Reply messages. Answer the following questions:

6. What are the hexadecimal values for the source and destination addresses in the first Ethernet frame containing the ARP request message?
7. Note that the first and second ARP packets in the trace correspond to an ARP request sent by the computer running Wireshark, and the ARP reply sent to the computer running Wireshark by the computer with the ARP-requested Ethernet address. But there is yet another computer on this network, as indicated by packet 6 – another ARP request. Why do you think there is no ARP reply (sent in response to the ARP request in packet 6) in the packet trace?