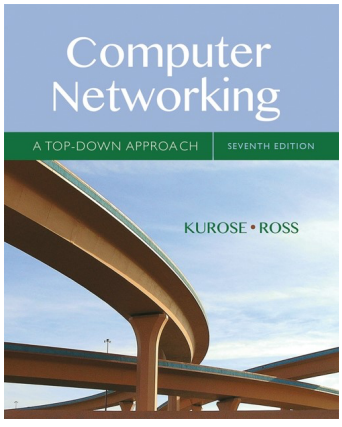
# CS 447: Network and Data Communication

## Wireshark Lab #03: Ethernet & ARP






**Note**: Make sure you produce your answers and any packet prints in PDF. Moodle will only accept PDF files.

In this lab, we'll investigate the Ethernet protocol and the ARP protocol. Before beginning this lab, you'll probably want to review sections 6.4.1 (Link-layer addressing and ARP) and 6.4.2 (Ethernet) in the text[1]. 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.

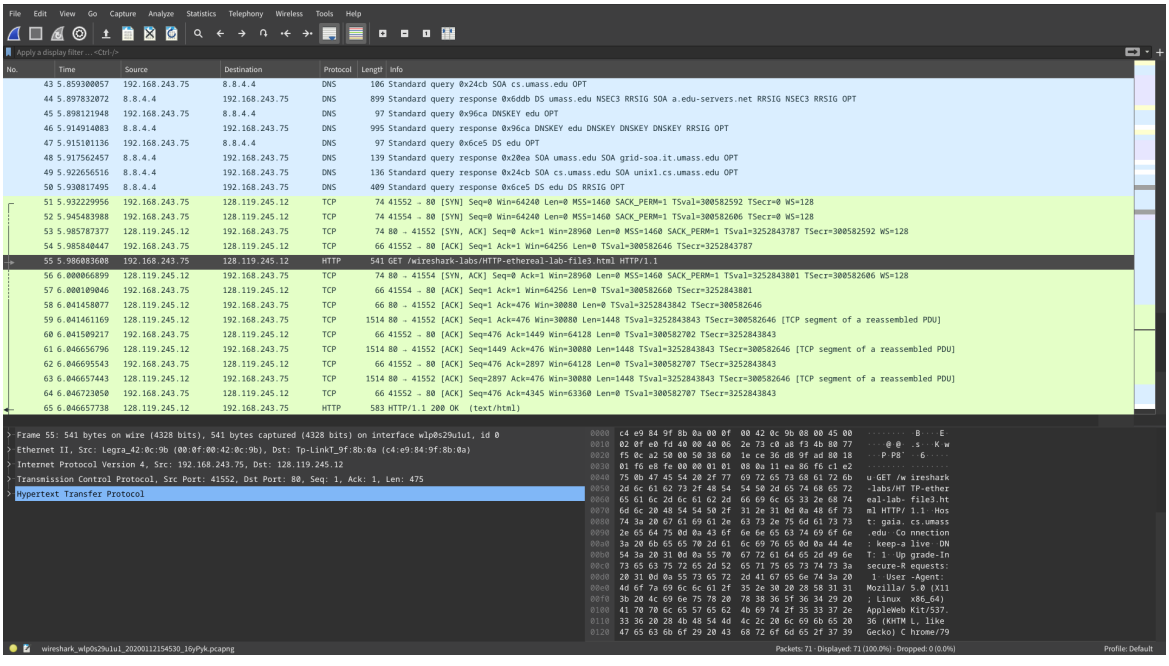
## Capturing and Analyzing Ethernet Frames

Let's begin by capturing a set of Ethernet frames to study. Do the following[2]:

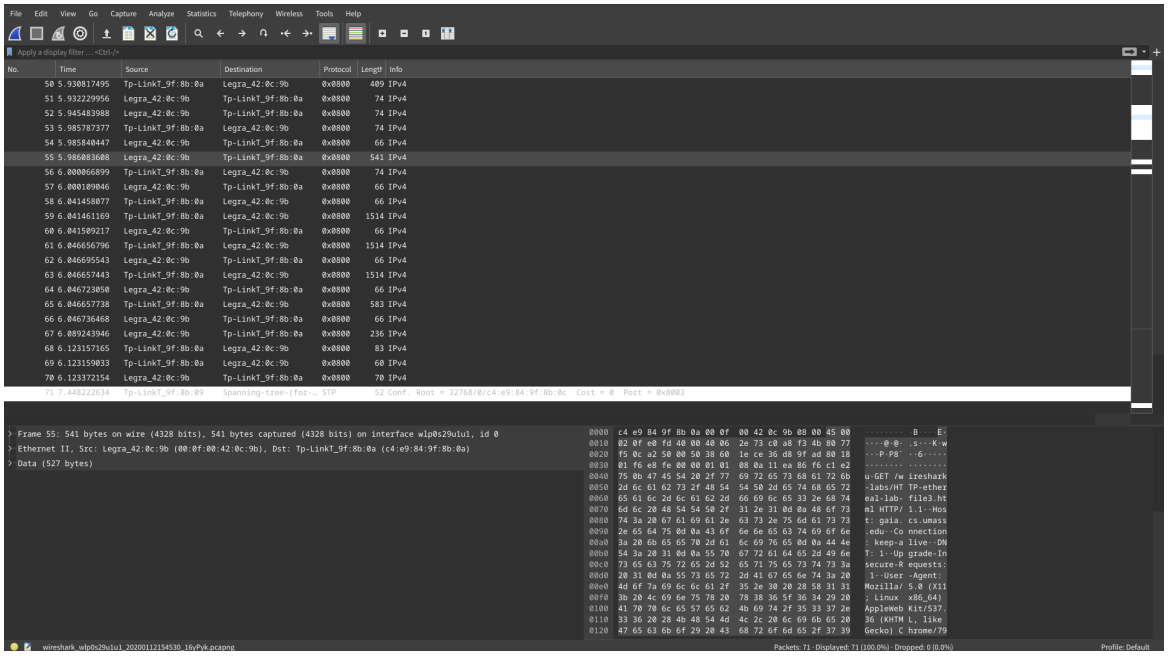
- First, make sure your browser's cache is empty.
- Start up the Wireshark packet sniffer
- Enter the following URL into your browser http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-lab-file3.html. Your browser should display the rather lengthy US Bill of Rights.
- Stop Wireshark packet capture. 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 55 in the screenshot below contains the HTTP GET message).

[1] References to figures and sections are for the 7th edition of text, *Computer Networks, A Top-down Approach, 7th ed.*, J.F. Kurose and K.W. Ross, Addison-Wesley/Pearson, 2016

[2] If you are unable to run Wireshark live on a computer, you can download the zip file http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip and extract the file *ethernet--ethereal-trace-1*. The traces in this zip file were collected by Wireshark running on one of the author's computers, while performing the steps indicated in the Wireshark lab. Once you have downloaded the trace, you can load it into Wireshark and view the trace using the *File* pull down menu, choosing *Open*, and then selecting the ethernet-ethereal-trace-1 trace file. You can then use this trace file to answer the questions below.



- 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 IPv4 box (use the filter at the top) and select OK. You should now see a Wireshark window that looks like the following:



In order to answer the following questions, you'll need to look into the packet details and packet content windows (lower left and right panels on the screenshots. You could have a different arrangement).

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. Whenever possible, when answering a question, you should hand in a printout of the packet(s) within the trace that you used to answer the question asked. Annotate the printout[3] to explain your answer. To print a packet, use *File → Print*, choose *Selected packet only*, choose *Packet summary line*, and select the minimum amount of packet detail that you need to answer the question.

1. What is the 48-bit Ethernet address of your computer?
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? [Note: this is an important question, and one that students sometimes get wrong. Re-read pages 468-469 in the text and make sure you understand the answer here.]
3. Give the hexadecimal value for the two-byte Frame type field. What upper layer protocol does this correspond to?
4. How many bytes from the very start of the Ethernet frame does the ASCII "G" in "GET" appear in the Ethernet frame?

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

5. What is the value of the Ethernet source address? Is this the address of your computer, or of gaia.cs.umass.edu (Hint: the answer is no). What device has this as its Ethernet address?
6. What is the destination address in the Ethernet frame? Is this the Ethernet address of your computer?
7. Give the hexadecimal value for the two-byte Frame type field. What upper layer protocol does this correspond to?
8. How many bytes from the very start of the Ethernet frame does the ASCII "O" in "OK" (i.e., the HTTP response code) appear in the Ethernet frame?

# The Address Resolution Protocol

In this section, we'll observe the ARP protocol in action. We strongly recommend that you 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).

[3] What is meant by "annotate"? Please highlight and annotate where you've found answers and add an explaination of what you've found in what you've highlighted.

- **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).

If on Windows try `arp -a`. If on Linux try `arp -n`. This will display the current ARP cache on your computer.

9. Write down the contents of your computer's ARP cache. What is the meaning of each column value?

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.

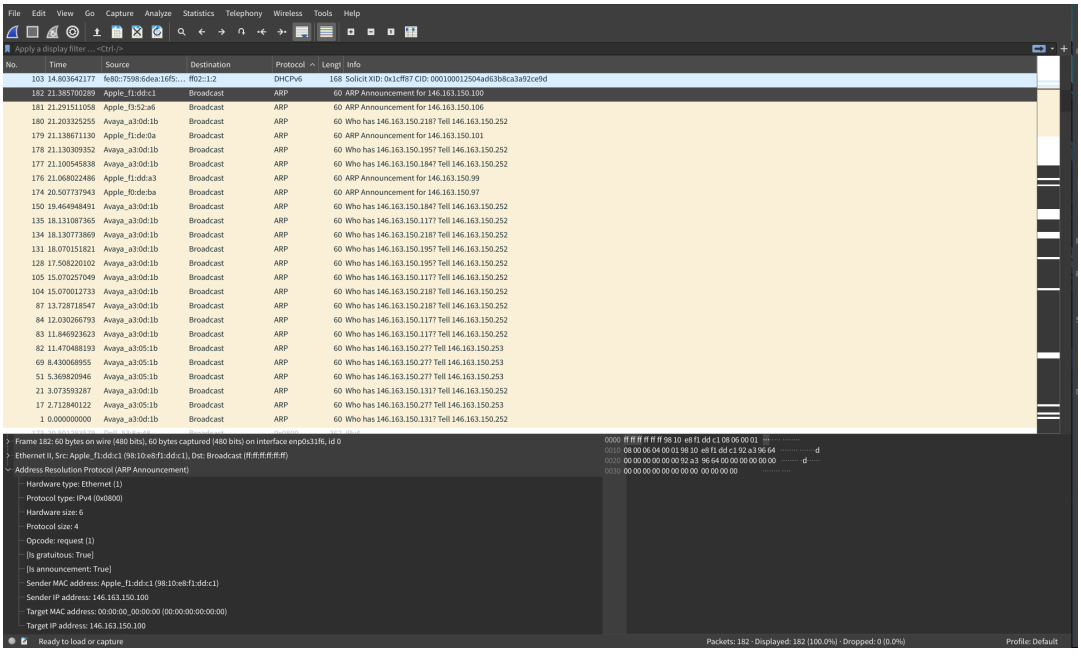
- **Linux/Unix/MacOS**. The "`ip -s -s neigh flush all`" on Linux will clear your ARP cache. You may have to elevate yourself to root to run this command.

## Observing ARP in Action

Do the following[4]:

- 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. Again, we're not interested in IP or higher-layer protocols, so 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 IPv4 box and select OK. You should now see a Wireshark window that looks like the following:

[4] The *ethernet-ethereal-trace-1* trace file in http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip was created using the steps below (in particular after the ARP cache had been flushed).



Answer the following questions:

10. What are the hexadecimal values for the source and destination addresses in the Ethernet frame containing the ARP request message?
11. Give the hexadecimal value for the two-byte Ethernet Frame type field. What upper layer protocol does this correspond to?
12. Download the ARP specification from ftp://ftp.rfc-editor.org/in-notes/std/std37.txt. A readable, detailed discussion of ARP is also at http://www.erg.abdn.ac.uk/users/gorry/course/inet-pages/arp.html.
    a. How many bytes from the very beginning of the Ethernet frame does the ARP opcode field begin?
    b. What is the value of the opcode field within the ARP-payload part of the Ethernet frame in which an ARP request is made?
    c. Does the ARP message contain the IP address of the sender?
    d. Where in the ARP request does the "question" appear – the Ethernet address of the machine whose corresponding IP address is being queried?
13. Now find the ARP reply that was sent in response to the ARP request.
    a. How many bytes from the very beginning of the Ethernet frame does the ARP *opcode* field begin?
    b. What is the value of the *opcode* field within the ARP-payload part of the Ethernet frame in which an ARP response is made?
    c. 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?
14. What are the hexadecimal values for the source and destination addresses in the Ethernet frame containing the ARP reply message?
15. Open the ethernet-ethereal-trace-1 trace file in http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip. The first and second ARP packets in this 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 is there no ARP reply (sent in response to the ARP request in packet 6) in the packet trace?

## Extra Credit

1. The arp command:

   ```
   arp -s InetAddr EtherAddr
   ```

   allows you to manually add an entry to the ARP cache that resolves the IP address `InetAddr` to the physical address `EtherAddr`. What would happen if, when you manually added an entry, you entered the correct IP address, but the wrong Ethernet address for that remote interface?

2. What is the default amount of time that an entry remains in your ARP cache before being removed. You can determine this empirically (by monitoring the cache contents) or by looking this up in your operation system documentation. Indicate how/where you determined this value.