

# Wireshark Lab 6: Ethernet and ARP v8.0

Due 12/4/22, 11:59 PM (Canvas)

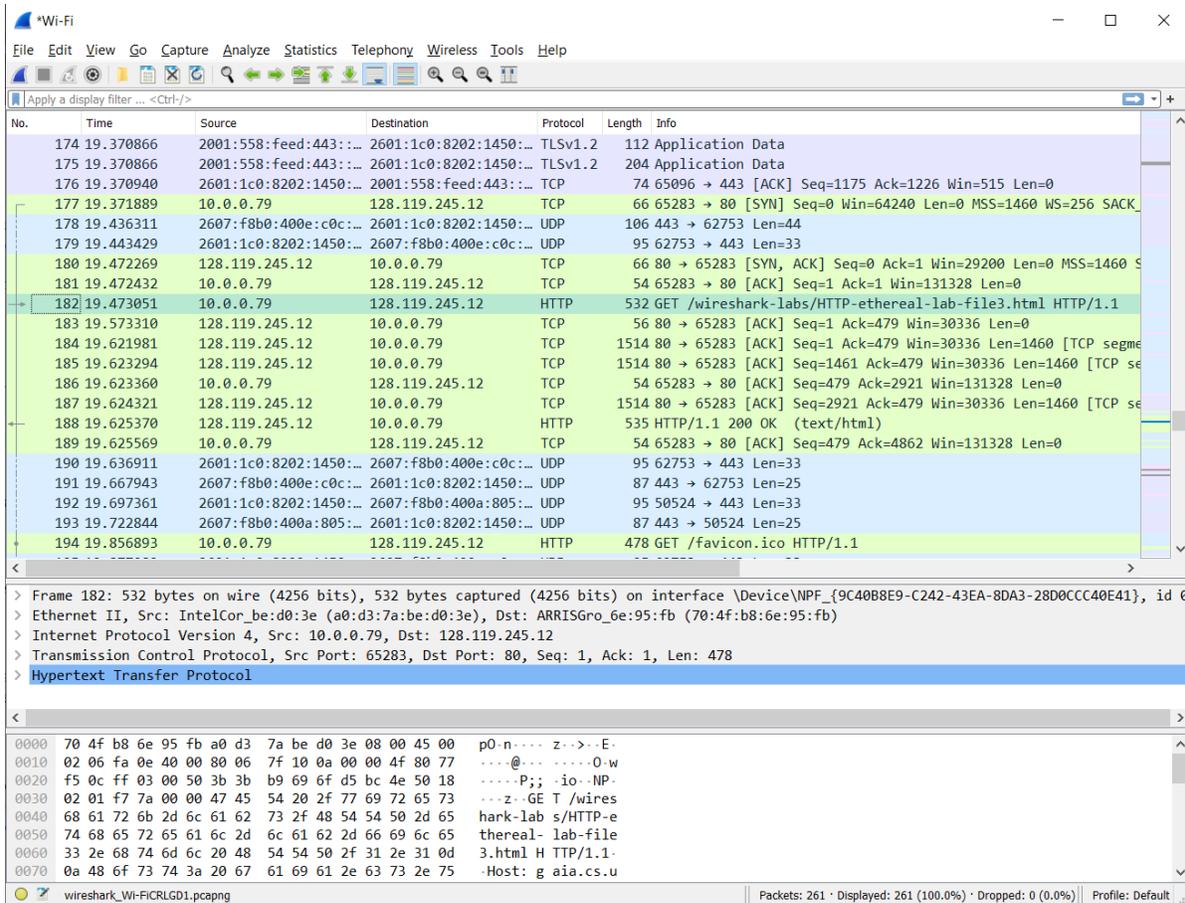
---

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. RFC 826 (<https://www.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

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

- 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 4 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 IPv4 and IPv6 box and select *OK*. You should now see a Wireshark window that looks like:

The image shows a Wireshark window titled "\*Wi-Fi". The main display area shows a list of network packets. Packet 182 is selected and highlighted in blue. The details pane below the list shows the structure of frame 182: Ethernet II, Src: IntelCor\_be:d0:3e (a0:d3:7a:be:d0:3e), Dst: ARRISGro\_6e:95:fb (70:4f:b8:6e:95:fb), Data (518 bytes). The packet contents window at the bottom shows the raw data in hexadecimal and ASCII. The ASCII portion of the data is: "p0-n...z...E...@...-O-w...P;;-io-NP...z-GE T /wires hark-lab s/HTTP-e thereal-lab-file 3.html H TTP/1.1-Host: g aia.cs.u".

No.	Time	Source	Destination	Protocol	Length	Info
176	19.370940	IntelCor_be:d0:3e	ARRISGro_6e:95:fb	0x86dd	74	IPv6
177	19.371889	IntelCor_be:d0:3e	ARRISGro_6e:95:fb	0x0800	66	IPv4
178	19.436311	ARRISGro_6e:95:fb	IntelCor_be:d0:3e	0x86dd	106	IPv6
179	19.443429	IntelCor_be:d0:3e	ARRISGro_6e:95:fb	0x86dd	95	IPv6
180	19.472269	ARRISGro_6e:95:fb	IntelCor_be:d0:3e	0x0800	66	IPv4
181	19.472432	IntelCor_be:d0:3e	ARRISGro_6e:95:fb	0x0800	54	IPv4
182	19.473051	IntelCor_be:d0:3e	ARRISGro_6e:95:fb	0x0800	532	IPv4
183	19.573310	ARRISGro_6e:95:fb	IntelCor_be:d0:3e	0x0800	56	IPv4
184	19.621981	ARRISGro_6e:95:fb	IntelCor_be:d0:3e	0x0800	1514	IPv4
185	19.623294	ARRISGro_6e:95:fb	IntelCor_be:d0:3e	0x0800	1514	IPv4
186	19.623360	IntelCor_be:d0:3e	ARRISGro_6e:95:fb	0x0800	54	IPv4
187	19.624321	ARRISGro_6e:95:fb	IntelCor_be:d0:3e	0x0800	1514	IPv4
188	19.625370	ARRISGro_6e:95:fb	IntelCor_be:d0:3e	0x0800	535	IPv4
189	19.625569	IntelCor_be:d0:3e	ARRISGro_6e:95:fb	0x0800	54	IPv4
190	19.636911	IntelCor_be:d0:3e	ARRISGro_6e:95:fb	0x86dd	95	IPv6
191	19.667943	ARRISGro_6e:95:fb	IntelCor_be:d0:3e	0x86dd	87	IPv6
192	19.697361	IntelCor_be:d0:3e	ARRISGro_6e:95:fb	0x86dd	95	IPv6
193	19.722844	ARRISGro_6e:95:fb	IntelCor_be:d0:3e	0x86dd	87	IPv6
194	19.856893	IntelCor_be:d0:3e	ARRISGro_6e:95:fb	0x0800	478	IPv4
195	19.877082	IntelCor_be:d0:3e	ARRISGro_6e:95:fb	0x86dd	95	IPv6
196	19.907726	ARRISGro_6e:95:fb	IntelCor_be:d0:3e	0x86dd	87	IPv6

> Frame 182: 532 bytes on wire (4256 bits), 532 bytes captured (4256 bits) on interface \Device\NPF...  
 > Ethernet II, Src: IntelCor\_be:d0:3e (a0:d3:7a:be:d0:3e), Dst: ARRISGro\_6e:95:fb (70:4f:b8:6e:95:fb), Data (518 bytes)

```

0000  70 4f b8 6e 95 fb a0 d3 7a be d0 3e 08 00 45 00  p0-n...z...E-
0010  02 06 fa 0e 40 00 80 06 7f 10 0a 00 00 4f 80 77  ....@...-O-w
0020  f5 0c ff 03 00 50 3b 3b b9 69 6f d5 bc 4e 50 18  ....P;;-io-NP-
0030  02 01 f7 7a 00 00 47 45 54 20 2f 77 69 72 65 73  ...z-GE T /wires
0040  68 61 72 6b 2d 6c 61 62 73 2f 48 54 54 50 2d 65  hark-lab s/HTTP-e
0050  74 68 65 72 65 61 6c 2d 6c 61 62 2d 66 69 6c 65  thereal-lab-file
0060  33 2e 68 74 6d 6c 20 48 54 54 50 2f 31 2e 31 0d  3.html H TTP/1.1-
0070  0a 48 6f 73 74 3a 20 67 61 69 61 2e 63 73 2e 75  -Host: g aia.cs.u
  
```

wireshark\_Wi-FiCRLGD1.pcapng | Packets: 261 · Displayed: 261 (100.0%) · Dropped: 0 (0.0%) | Profile: Default

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 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: 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?

## 2. 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 -a*” or “`c:\windows\system32\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 “-a” will display the contents of the ARP cache on your computer. Run the *arp* command.

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 *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.

## Observing ARP in action

Do the following:

- Clear your ARP cache, as described above.
- Next, 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-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 and IPv6 box and select *OK*. You should now see a Wireshark window that looks like:

ethernet-ethereal-trace-1

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	AmbitMic_a9:3d:68	Broadcast	ARP	42	Who has 192.168.1.1? Tell 192.168.1.10
2	0.001018	LinksysG_da:af:73	AmbitMic_a9:3d:68	ARP	60	192.168.1.1 is at 00:06:25:da:af:73
3	0.001028	AmbitMic_a9:3d:68	LinksysG_da:af:73	0x0800	62	IPv4
4	2.962850	AmbitMic_a9:3d:68	LinksysG_da:af:73	0x0800	62	IPv4
5	8.971488	AmbitMic_a9:3d:68	LinksysG_da:af:73	0x0800	62	IPv4
6	13.542974	CnetTech_73:8d:ce	Broadcast	ARP	60	Who has 192.168.1.117? Tell 192.168.1.
7	17.444423	AmbitMic_a9:3d:68	LinksysG_da:af:73	0x0800	62	IPv4
8	17.465902	LinksysG_da:af:73	AmbitMic_a9:3d:68	0x0800	62	IPv4
9	17.465927	AmbitMic_a9:3d:68	LinksysG_da:af:73	0x0800	54	IPv4
10	17.466468	AmbitMic_a9:3d:68	LinksysG_da:af:73	0x0800	686	IPv4
11	17.494766	LinksysG_da:af:73	AmbitMic_a9:3d:68	0x0800	60	IPv4
12	17.498935	LinksysG_da:af:73	AmbitMic_a9:3d:68	0x0800	1514	IPv4
13	17.500025	LinksysG_da:af:73	AmbitMic_a9:3d:68	0x0800	1514	IPv4
14	17.500069	AmbitMic_a9:3d:68	LinksysG_da:af:73	0x0800	54	IPv4
15	17.527057	LinksysG_da:af:73	AmbitMic_a9:3d:68	0x0800	1514	IPv4
16	17.527422	LinksysG_da:af:73	AmbitMic_a9:3d:68	0x0800	489	IPv4
17	17.527457	AmbitMic_a9:3d:68	LinksysG_da:af:73	0x0800	54	IPv4

< >

> Frame 1: 42 bytes on wire (336 bits), 42 bytes captured (336 bits)  
 > Ethernet II, Src: AmbitMic\_a9:3d:68 (00:d0:59:a9:3d:68), Dst: Broadcast (ff:ff:ff:ff:ff:ff)  
 > Address Resolution Protocol (request)

```

0000  ff ff ff ff ff 00 d0 59 a9 3d 68 08 06 00 01 ..... Y=h...
0010  08 00 06 04 00 01 00 d0 59 a9 3d 68 c0 a8 01 69 ..... Y=h...i
0020  00 00 00 00 00 00 c0 a8 01 01 .....
  
```

ethernet-ethereal-trace-1 | Packets: 17 · Displayed: 17 (100.0%) | Profile: Default

In the example above, the first two frames in the trace contain ARP messages (as does the 6<sup>th</sup> message).

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 <https://www.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-etherreal-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?

## What to hand in

Answer the questions above, based on YOUR Wireshark experimentation, except for question 10-15. For your reference, the pre-captured trace files are at: <http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip>

### Instructions:

1. When answering the questions above, you should print out the screenshot(s) and indicate where in the screenshot(s) you’ve found the information that answers the following questions.
2. When you hand in your assignment, annotate the output so that it’s clear where in the output you’re getting the information for your answer.