This is a version of a Wireshark Lab updated for newer versions of Wireshark from

Supplement to *Computer Networking: A Top-Down Approach, 7th ed.*, J.F. Kurose and K.W. Ross

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 (https://datatracker.ietf.org/doc/html/rfc826) contains the gory details of the ARP protocol.

1. Capturing and analyzing Ethernet frames

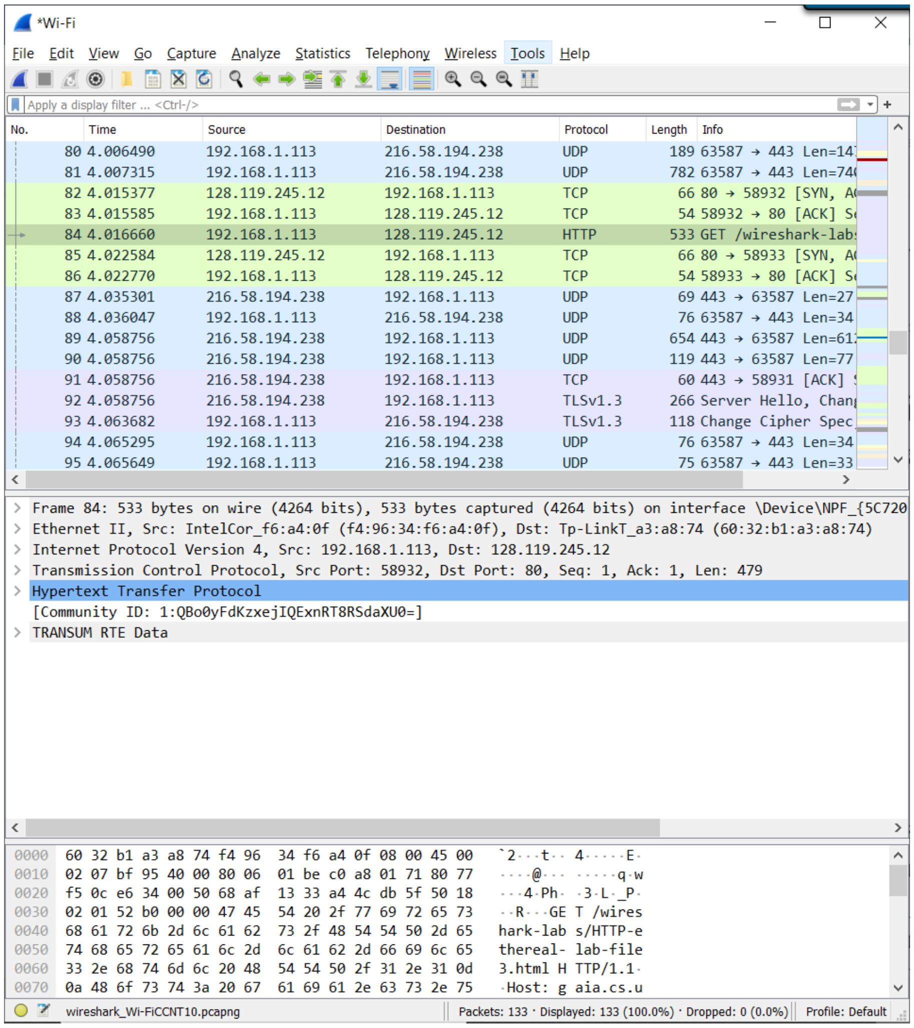
Let's begin by capturing a set of Ethernet frames to study. Do the following[2]:

- First, make sure your browser's cache is empty. To do this:
    - Under Mozilla Firefox V3, select *Tools* → *Clear Recent History* and check the box for Cache.
    - For Internet Explorer, select *Tools* → *Internet Options* → *Delete Files.*
    - For Microsoft Edge, click *[horizontal three dots in the upper right]* → *History* → *three dots* → *Clear Browsing Data* → *Clear Now*.
    - For Chrome, *[three dots, upper right]* → *History* → *History* → *Clear Browsing Data* → *Clear data* (for last hour or more).
- 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.
    - This works for wireless (802.11). Your operating system likely treats all the data as if it is Ethernet II.

[1] References to figures and sections are updated from the 7th edition of our text, *Computer Networks, A Top-down Approach, 7th ed.*, J.F. Kurose and K.W. Ross, Addison-Wesley/Pearson, 2016 to match the 8th edition, but one could use the index for the 7th to find them, or just look in the relevant section as they are identically numbered.

[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.

- 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 84, fifth from the top in the screen shot below, contains the HTTP GET message). **Hint**: Once we hide the IP information, knowing the packet numbers may aid in finding the correct packet, especially if your network has a lot of connected devices.


*Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

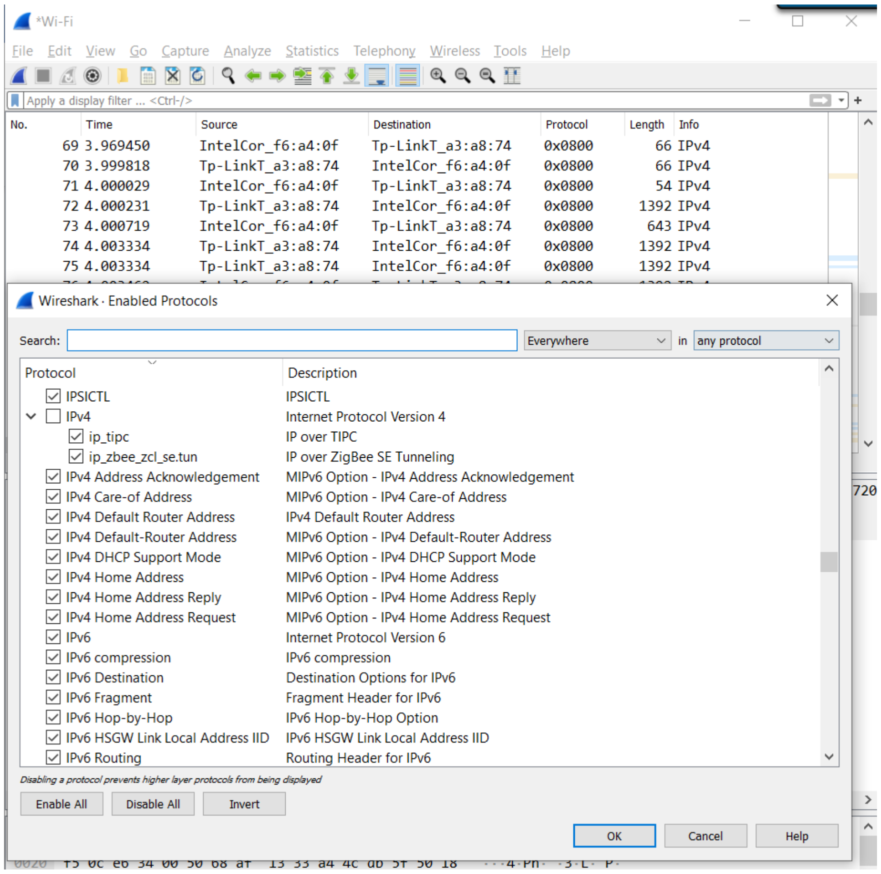
```
No.  Time      Source          Destination     Protocol  Length Info
  80 4.006490  192.168.1.113   216.58.194.238  UDP        189 63587 → 443 Len=14
  81 4.007315  192.168.1.113   216.58.194.238  UDP        782 63587 → 443 Len=74
  82 4.015377  128.119.245.12  192.168.1.113   TCP         66 80 → 58932 [SYN, A
  83 4.015585  192.168.1.113   128.119.245.12  TCP         54 58932 → 80 [ACK] S
  84 4.016660  192.168.1.113   128.119.245.12  HTTP       533 GET /wireshark-lab
  85 4.022584  128.119.245.12  192.168.1.113   TCP         66 80 → 58933 [SYN, A
  86 4.022770  192.168.1.113   128.119.245.12  TCP         54 58933 → 80 [ACK] S
  87 4.035301  216.58.194.238  192.168.1.113   UDP         69 443 → 63587 Len=27
  88 4.036047  192.168.1.113   216.58.194.238  UDP         76 63587 → 443 Len=34
  89 4.058756  216.58.194.238  192.168.1.113   UDP        654 443 → 63587 Len=61
  90 4.058756  216.58.194.238  192.168.1.113   UDP        119 443 → 63587 Len=77
  91 4.058756  216.58.194.238  192.168.1.113   TCP         60 443 → 58931 [ACK]
  92 4.058756  216.58.194.238  192.168.1.113   TLSv1.3    266 Server Hello, Chan
  93 4.063682  192.168.1.113   216.58.194.238  TLSv1.3    118 Change Cipher Spec
  94 4.065295  192.168.1.113   216.58.194.238  UDP         76 63587 → 443 Len=34
  95 4.065649  192.168.1.113   216.58.194.238  UDP         75 63587 → 443 Len=33
```

```
> Frame 84: 533 bytes on wire (4264 bits), 533 bytes captured (4264 bits) on interface \Device\NPF_{5C720
> Ethernet II, Src: IntelCor_f6:a4:0f (f4:96:34:f6:a4:0f), Dst: Tp-LinkT_a3:a8:74 (60:32:b1:a3:a8:74)
> Internet Protocol Version 4, Src: 192.168.1.113, Dst: 128.119.245.12
> Transmission Control Protocol, Src Port: 58932, Dst Port: 80, Seq: 1, Ack: 1, Len: 479
> Hypertext Transfer Protocol
  [Community ID: 1:QBo0yFdKzxejIQExnRT8RSdaXU0=]
> TRANSUM RTE Data
```

```
0000  60 32 b1 a3 a8 74 f4 96  34 f6 a4 0f 08 00 45 00   `2···t·· 4·····E·
0010  02 07 bf 95 40 00 80 06  01 be c0 a8 01 71 80 77   ····@··· ·····q·w
0020  f5 0c e6 34 00 50 68 af  13 33 a4 4c db 5f 50 18   ···4·Ph· ·3·L·_P·
0030  02 01 52 b0 00 00 47 45  54 20 2f 77 69 72 65 73   ··R···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-FiCCNT10.pcapng | Packets: 133 · Displayed: 133 (100.0%) · Dropped: 0 (0.0%) | Profile: Default

- 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. One can go *Analyze->Enabled Protocols…* and uncheck IPv4. Image below shows result *after* unchecking IPv4. **Hint 2:** Take a moment to note the structure of the HTTP packets with these settings so you can figure it out once we do this again while looking at ARP.


*Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

```
No.   Time      Source             Destination        Protocol  Length Info
   69 3.969450  IntelCor_f6:a4:0f  Tp-LinkT_a3:a8:74  0x0800        66 IPv4
   70 3.999818  Tp-LinkT_a3:a8:74  IntelCor_f6:a4:0f  0x0800        66 IPv4
   71 4.000029  IntelCor_f6:a4:0f  Tp-LinkT_a3:a8:74  0x0800        54 IPv4
   72 4.000231  Tp-LinkT_a3:a8:74  IntelCor_f6:a4:0f  0x0800      1392 IPv4
   73 4.000719  IntelCor_f6:a4:0f  Tp-LinkT_a3:a8:74  0x0800       643 IPv4
   74 4.003334  Tp-LinkT_a3:a8:74  IntelCor_f6:a4:0f  0x0800      1392 IPv4
   75 4.003334  Tp-LinkT_a3:a8:74  IntelCor_f6:a4:0f  0x0800      1392 IPv4
```

Protocol	Description
☑ IPSICTL	IPSICTL
☐ IPv4	Internet Protocol Version 4
☑ ip_tipc	IP over TIPC
☑ ip_zbee_zcl_se.tun	IP over ZigBee SE Tunneling
☑ IPv4 Address Acknowledgement	MIPv6 Option - IPv4 Address Acknowledgement
☑ IPv4 Care-of Address	MIPv6 Option - IPv4 Care-of Address
☑ IPv4 Default Router Address	IPv4 Default Router Address
☑ IPv4 Default-Router Address	MIPv6 Option - IPv4 Default-Router Address
☑ IPv4 DHCP Support Mode	MIPv6 Option - IPv4 DHCP Support Mode
☑ IPv4 Home Address	MIPv6 Option - IPv4 Home Address
☑ IPv4 Home Address Reply	MIPv6 Option - IPv4 Home Address Reply
☑ IPv4 Home Address Request	MIPv6 Option - IPv4 Home Address Request
☑ IPv6	Internet Protocol Version 6
☑ IPv6 compression	IPv6 compression
☑ IPv6 Destination	Destination Options for IPv6
☑ IPv6 Fragment	Fragment Header for IPv6
☑ IPv6 Hop-by-Hop	IPv6 Hop-by-Hop Option
☑ IPv6 HSGW Link Local Address IID	IPv6 HSGW Link Local Address IID
☑ IPv6 Routing	Routing Header for IPv6

Disabling a protocol prevents higher layer protocols from being displayed

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; (re)read section 1.5.2 in the text if you find this encapsulation a bit confusing, also in the slides). 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*, and select different options for printing, and select the minimum amount of packet detail that you need to answer the question; consider printing to file.
- One may also *File->Export Packet Dissections-> As Plain Text…* .

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 Section 6.4 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? (Ethernet II-> Type)
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?

[3] What do we mean by "annotate"? If you hand in a paper copy please highlight where in the printout you've found the answer and add some text (preferably with a colored pen) noting what you found in what you 've highlight. If you hand in an electronic copy, it would be great if you could also highlight and annotate. Consider a program like Word for highlighting.

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 Windows/MS-DOS 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 is 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:

- **Windows/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). In Windows, you can *Windows+R* -> *cmd* to load the command prompt.
- **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.

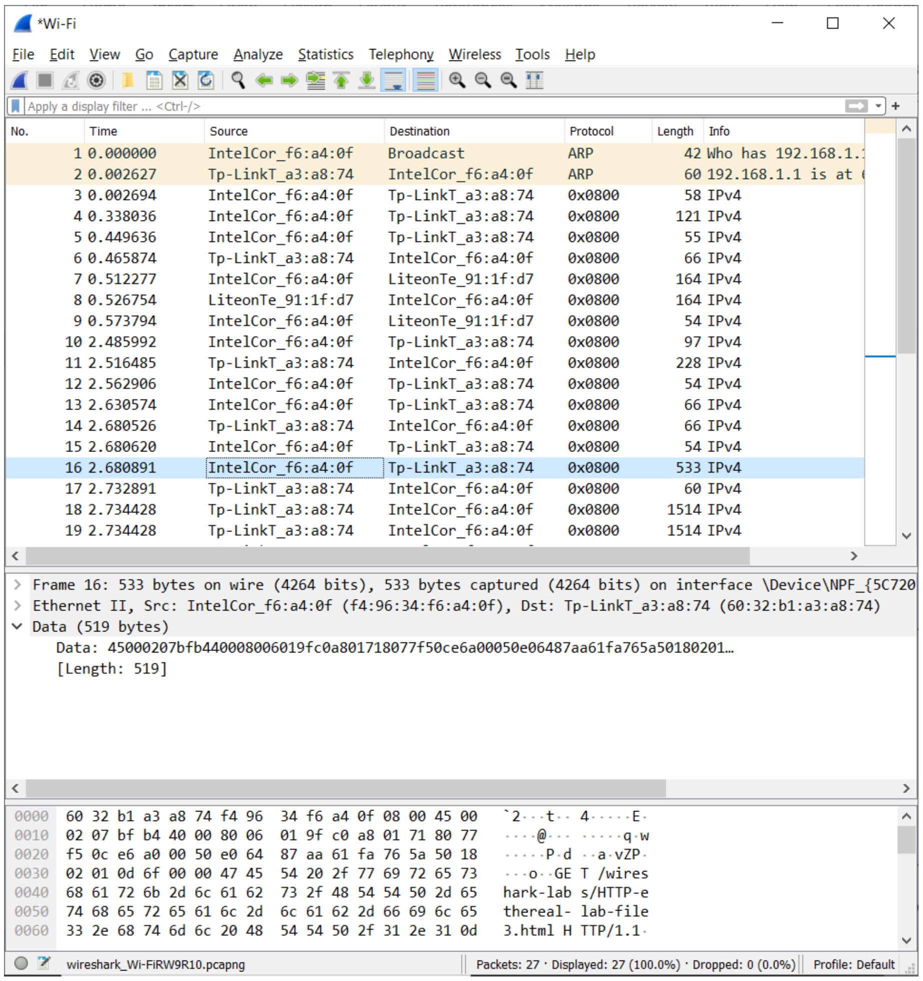
In order to observe your computer sending and receiving ARP messages, we’ll need to clear the ARP cache, since otherwise your computer is likely to find a needed IP-Ethernet address translation pair in its cache and consequently not need to send out an ARP message.

- **Windows/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. This will likely have to be run as administrator. In Windows 10, search for “cmd” in the search bar and right click on cmd and click “Run as administrator”
- **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[4]:

- Clear your ARP cache, as described above. You will likely need to run this as administrator (Windows) or sudo (Linux).
- First, make sure your browser's cache is empty. To do this:
    - under Mozilla Firefox V3, select *Tools* → *Clear Recent History* and check the box for Cache.
    - For Internet Explorer, select *Tools* → *Internet Options* → *Delete Files.*
    - For Microsoft Edge, click *[horizontal three dots in the upper right]* → *History* → *three dots* → *Clear Browsing Data* → *Clear Now*.
    - For Chrome, *[three dots,upper right]* → *History* → *History* → *Clear Browsing Data* → *Clear data* (for last hour or more).
- 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 are 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 by changing the columns if they are not still available. Then uncheck the IP box and select *OK*. You should now see a Wireshark window that looks like:


*Wi-Fi
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  IntelCor_f6:a4:0f  Broadcast          ARP           42 Who has 192.168.1.1
   2 0.002627  Tp-LinkT_a3:a8:74  IntelCor_f6:a4:0f  ARP           60 192.168.1.1 is at (
   3 0.002694  IntelCor_f6:a4:0f  Tp-LinkT_a3:a8:74  0x0800        58 IPv4
   4 0.338036  IntelCor_f6:a4:0f  Tp-LinkT_a3:a8:74  0x0800       121 IPv4
   5 0.449636  IntelCor_f6:a4:0f  Tp-LinkT_a3:a8:74  0x0800        55 IPv4
   6 0.465874  Tp-LinkT_a3:a8:74  IntelCor_f6:a4:0f  0x0800        66 IPv4
   7 0.512277  IntelCor_f6:a4:0f  LiteonTe_91:1f:d7  0x0800       164 IPv4
   8 0.526754  LiteonTe_91:1f:d7  IntelCor_f6:a4:0f  0x0800       164 IPv4
   9 0.573794  IntelCor_f6:a4:0f  LiteonTe_91:1f:d7  0x0800        54 IPv4
  10 2.485992  IntelCor_f6:a4:0f  Tp-LinkT_a3:a8:74  0x0800        97 IPv4
  11 2.516485  Tp-LinkT_a3:a8:74  IntelCor_f6:a4:0f  0x0800       228 IPv4
  12 2.562906  IntelCor_f6:a4:0f  Tp-LinkT_a3:a8:74  0x0800        54 IPv4
  13 2.630574  IntelCor_f6:a4:0f  Tp-LinkT_a3:a8:74  0x0800        66 IPv4
  14 2.680526  Tp-LinkT_a3:a8:74  IntelCor_f6:a4:0f  0x0800        66 IPv4
  15 2.680620  IntelCor_f6:a4:0f  Tp-LinkT_a3:a8:74  0x0800        54 IPv4
  16 2.680891  IntelCor_f6:a4:0f  Tp-LinkT_a3:a8:74  0x0800       533 IPv4
  17 2.732891  Tp-LinkT_a3:a8:74  IntelCor_f6:a4:0f  0x0800        60 IPv4
  18 2.734428  Tp-LinkT_a3:a8:74  IntelCor_f6:a4:0f  0x0800      1514 IPv4
  19 2.734428  Tp-LinkT_a3:a8:74  IntelCor_f6:a4:0f  0x0800      1514 IPv4
```

```
> Frame 16: 533 bytes on wire (4264 bits), 533 bytes captured (4264 bits) on interface \Device\NPF_{5C720
> Ethernet II, Src: IntelCor_f6:a4:0f (f4:96:34:f6:a4:0f), Dst: Tp-LinkT_a3:a8:74 (60:32:b1:a3:a8:74)
v Data (519 bytes)
    Data: 45000207bfb440008006019fc0a801718077f50ce6a00050e06487aa61fa765a50180201…
    [Length: 519]
```

```
0000  60 32 b1 a3 a8 74 f4 96  34 f6 a4 0f 08 00 45 00   `2···t·· 4·····E·
0010  02 07 bf b4 40 00 80 06  01 9f c0 a8 01 71 80 77   ····@··· ·····q·w
0020  f5 0c e6 a0 00 50 e0 64  87 aa 61 fa 76 5a 50 18   ·····P·d ··a·vZP·
0030  02 01 0d 6f 00 00 47 45  54 20 2f 77 69 72 65 73   ···o··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·
```

```
wireshark_Wi-FiRW9R10.pcapng    Packets: 27 · Displayed: 27 (100.0%) · Dropped: 0 (0.0%)    Profile: Default
```

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).

In the example above, the first two frames have the ARP messages. The 16th, 18th, and 19th have HTTP packets.

Answer the following questions:

9. What are the hexadecimal values for the source and destination addresses in the Ethernet frame containing the ARP request message?
10. Give the hexadecimal value for the two-byte Ethernet Frame type field. What upper layer protocol does this correspond to?
11. Where in the ARP request does the “question” appear – the Ethernet address of the machine whose corresponding IP address is being queried?
12. 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?
13. What are the hexadecimal values for the source and destination addresses in the Ethernet frame containing the ARP reply message?