**Print Your Name: ______________________**

# CSC 360 Lab 4

## DHCP

In this first part of this lab, we'll take a quick look at DHCP. Recall that DHCP is used extensively in corporate, university and home-network wired and wireless LANs to dynamically assign IP addresses to hosts (as well as to configure other network configuration information).

This lab is brief, as we'll only examine the DHCP packets captured by a host. If you also have administrative access to your DHCP server, you may want to repeat this lab after making some configuration changes (such as the lease time). If you have a router at home, you most likely can configure your DHCP server. Usually, linux/Unix machines (especially those that serve many users) have a static IP address and manipulating DHCP on such machines typically requires super-user privileges.

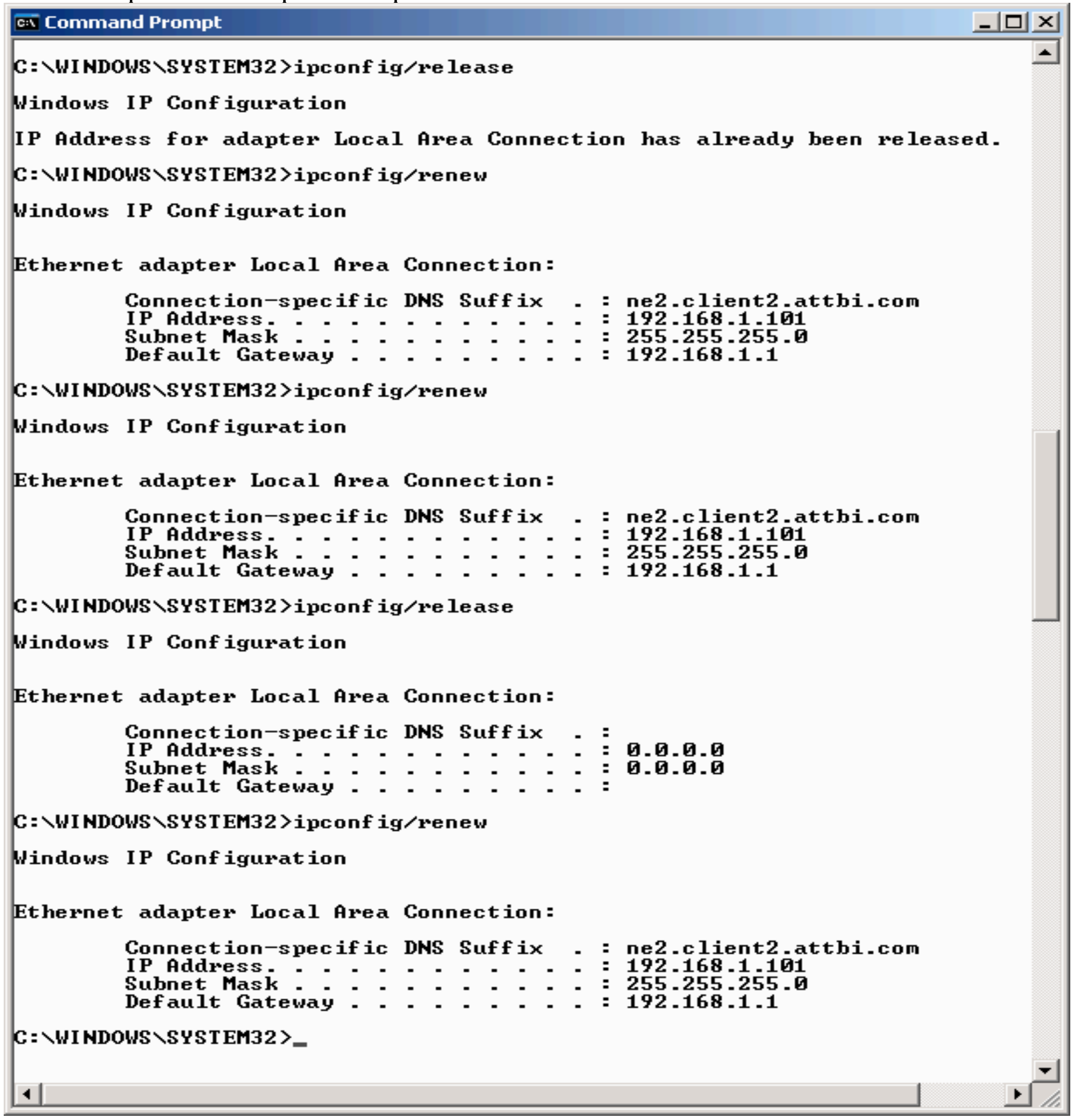
### DHCP Experiment

In order to observe DHCP in action, we'll perform several DHCP-related commands and capture the DHCP messages exchanged as a result of executing these commands. Do the following[1]:

1. Begin by opening the Windows Command Prompt application (which can be started through Start -> Run -> cmd). As shown in Figure 1, enter "*ipconfig /release*". The executable for *ipconfig* is in C:\windows\system32. This command releases your current IP address, so that your host's IP address becomes 0.0.0.0.
2. Start up the Wireshark packet sniffer, as described in the introductory Wireshark lab and begin Wireshark packet capture.
3. Now go back to the Windows Command Prompt and enter "*ipconfig /renew*". This instructs your host to obtain a network configuration, including a new IP address. In Figure 1, the host obtains the IP address 192.168.1.108
4. Wait until the "*ipconfig /renew*" has terminated. Then enter the same command "*ipconfig /renew*" again.
5. When the second *"ipconfig /renew"* terminates, enter the command "ipconfig/release" to release the previously-allocated IP address to your computer.

[1] 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 *dhcp-ethereal-trace-1*. The traces in this zip file were collected by Ethereal 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 dhcp-ethereal-trace-1 trace file. You can then use this trace file to answer the questions below.

6. Finally, enter "*ipconfig /renew*" to again be allocated an IP address for your computer.
7. Stop Wireshark packet capture.



```
C:\WINDOWS\SYSTEM32>ipconfig/release

Windows IP Configuration

IP Address for adapter Local Area Connection has already been released.

C:\WINDOWS\SYSTEM32>ipconfig/renew

Windows IP Configuration


Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . : ne2.client2.attbi.com
        IP Address. . . . . . . . . . . . : 192.168.1.101
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 192.168.1.1

C:\WINDOWS\SYSTEM32>ipconfig/renew

Windows IP Configuration


Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . : ne2.client2.attbi.com
        IP Address. . . . . . . . . . . . : 192.168.1.101
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 192.168.1.1

C:\WINDOWS\SYSTEM32>ipconfig/release

Windows IP Configuration


Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . :
        IP Address. . . . . . . . . . . . : 0.0.0.0
        Subnet Mask . . . . . . . . . . . : 0.0.0.0
        Default Gateway . . . . . . . . . :

C:\WINDOWS\SYSTEM32>ipconfig/renew

Windows IP Configuration


Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . : ne2.client2.attbi.com
        IP Address. . . . . . . . . . . . : 192.168.1.101
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 192.168.1.1

C:\WINDOWS\SYSTEM32>_
```

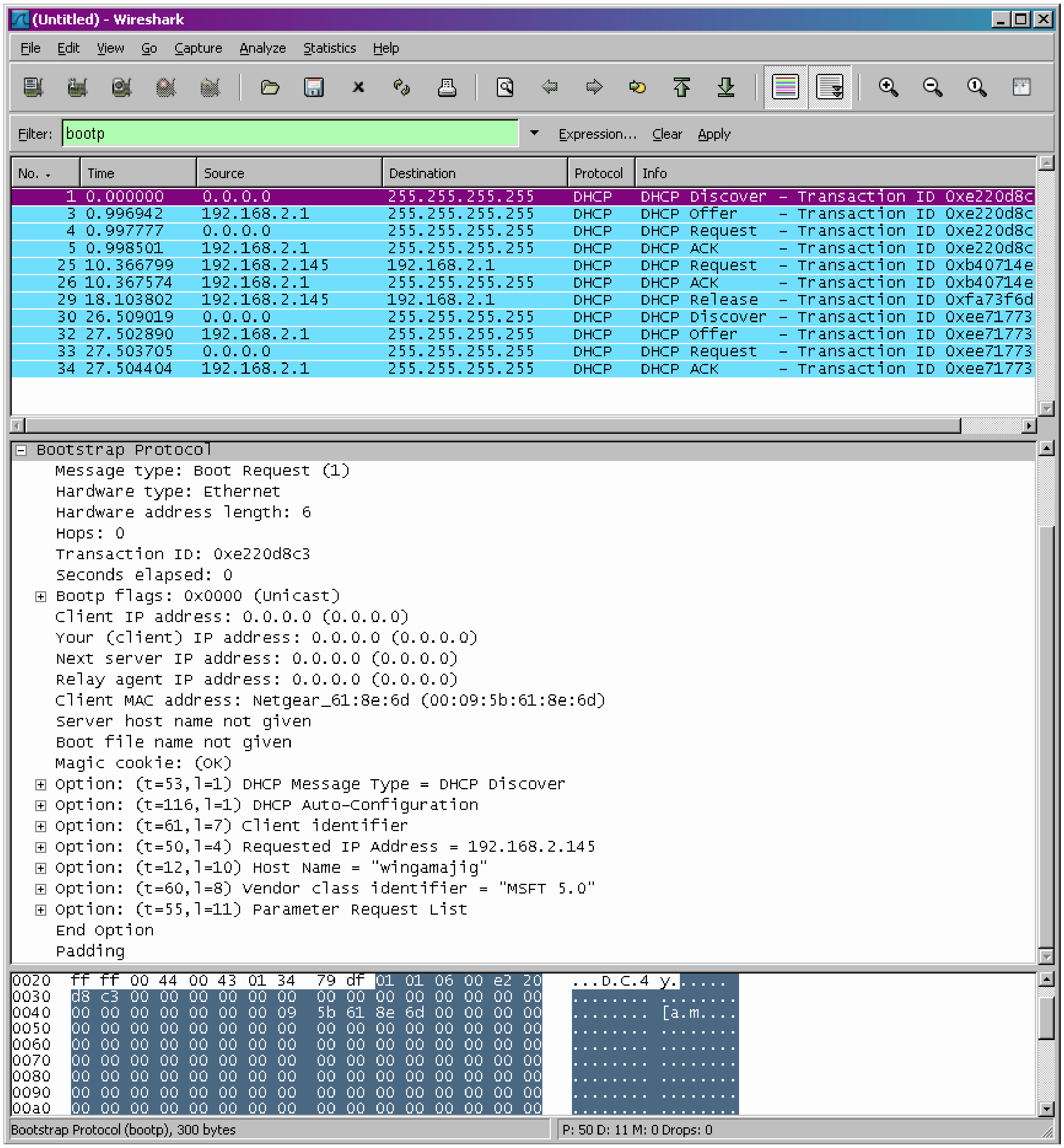
**Figure 1** Command Prompt window showing sequence of *ipconfig* commands that you should enter.

If you are using Mac, please use the following steps to renew DHCP:

1. Choose Apple menu > System Preferences, then click Network.
2. Select your connection service that uses DHCP in the list (such as Ethernet or Wi-Fi).
3. Click Advanced, click TCP/IP, then click Renew DHCP Lease.

Now let's take a look at the resulting Wireshark window. To see only the DHCP packets, enter into the filter field "ip.addr = = your IP address && bootp". (DHCP derives from an older protocol called BOOTP. Both BOOTP and DHCP use the same port numbers, 67 and 68. To see DHCP packets in the current version of Ethereal, you need to enter "bootp" and not "dhcp" in the filter.)

We see from Figure 2 that the first *ipconfig* renew command caused four DHCP packets to be generated: a DHCP Discover packet, a DHCP Offer packet, a DHCP Request packet, and a DHCP ACK packet.


(Untitled) - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter: bootp
Expression... Clear Apply

```
No.  Time       Source          Destination      Protocol Info
  1 0.000000   0.0.0.0         255.255.255.255  DHCP     DHCP Discover - Transaction ID 0xe220d8c
  3 0.996942   192.168.2.1     255.255.255.255  DHCP     DHCP Offer    - Transaction ID 0xe220d8c
  4 0.997777   0.0.0.0         255.255.255.255  DHCP     DHCP Request  - Transaction ID 0xe220d8c
  5 0.998501   192.168.2.1     255.255.255.255  DHCP     DHCP ACK      - Transaction ID 0xe220d8c
 25 10.366799  192.168.2.145   192.168.2.1      DHCP     DHCP Request  - Transaction ID 0xb40714e
 26 10.367574  192.168.2.1     255.255.255.255  DHCP     DHCP ACK      - Transaction ID 0xb40714e
 29 18.103802  192.168.2.145   192.168.2.1      DHCP     DHCP Release  - Transaction ID 0xfa73f6d
 30 26.509019  0.0.0.0         255.255.255.255  DHCP     DHCP Discover - Transaction ID 0xee71773
 32 27.502890  192.168.2.1     255.255.255.255  DHCP     DHCP Offer    - Transaction ID 0xee71773
 33 27.503705  0.0.0.0         255.255.255.255  DHCP     DHCP Request  - Transaction ID 0xee71773
 34 27.504404  192.168.2.1     255.255.255.255  DHCP     DHCP ACK      - Transaction ID 0xee71773
```

```
⊟ Bootstrap Protocol
    Message type: Boot Request (1)
    Hardware type: Ethernet
    Hardware address length: 6
    Hops: 0
    Transaction ID: 0xe220d8c3
    Seconds elapsed: 0
  ⊞ Bootp flags: 0x0000 (Unicast)
    Client IP address: 0.0.0.0 (0.0.0.0)
    Your (client) IP address: 0.0.0.0 (0.0.0.0)
    Next server IP address: 0.0.0.0 (0.0.0.0)
    Relay agent IP address: 0.0.0.0 (0.0.0.0)
    Client MAC address: Netgear_61:8e:6d (00:09:5b:61:8e:6d)
    Server host name not given
    Boot file name not given
    Magic cookie: (OK)
  ⊞ Option: (t=53,l=1) DHCP Message Type = DHCP Discover
  ⊞ Option: (t=116,l=1) DHCP Auto-Configuration
  ⊞ Option: (t=61,l=7) Client identifier
  ⊞ Option: (t=50,l=4) Requested IP Address = 192.168.2.145
  ⊞ Option: (t=12,l=10) Host Name = "wingamajig"
  ⊞ Option: (t=60,l=8) Vendor class identifier = "MSFT 5.0"
  ⊞ Option: (t=55,l=11) Parameter Request List
    End Option
    Padding
```

```
0020  ff ff 00 44 00 43 01 34  79 df 01 01 06 00 e2 20   ...D.C.4 y.......
0030  d8 c3 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0040  00 00 00 00 00 00 00 09  5b 61 8e 6d 00 00 00 00   ........ [a.m....
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
00a0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
```

```
Bootstrap Protocol (bootp), 300 bytes        P: 50 D: 11 M: 0 Drops: 0
```

**Figure 2** Wireshark window with first DHCP packet – the DHCP Discover packet – expanded.

## What to Hand In:

Answer the following questions:

1. Are DHCP messages sent over UDP or TCP?

2. Draw a timing datagram illustrating the sequence of the first four-packet Discover/Offer/Request/ACK DHCP exchange between the client and server. For each packet, indicated the source and destination port numbers. Are the port numbers the same as in the example given in this lab assignment?

3. What is the link-layer (e.g., Ethernet) address of your host?

4. What values in the DHCP discover message differentiate this message from the DHCP request message?

5. What is the value of the Transaction-ID in each of the first four (Discover/Offer/Request/ACK) DHCP messages? What are the values of the Transaction-ID in the second set (Request/ACK) set of DHCP messages? What is the purpose of the Transaction-ID field?

6. A host uses DHCP to obtain an IP address, among other things. But a host's IP address is not confirmed until the end of the four-message exchange! If the IP address is not set until the end of the four-message exchange, then what values are used in the IP datagrams in the four-message exchange? For each of the four DHCP messages (Discover/Offer/Request/ACK DHCP), indicate the source and destination IP addresses that are carried in the encapsulating IP datagram.

7. What is the IP address of your DHCP server?

8. What IP address is the DHCP server offering to your host in the DHCP Offer message? Indicate which DHCP message contains the offered DHCP address.

9. Explain the purpose of the router and subnet mask lines in the DHCP offer message.

10. In the example screenshots in this assignment, the host requests the offered IP address in the DHCP Request message. What happens in your own experiment?

11. Explain the purpose of the lease time. How long is the lease time in your experiment?

12. What is the purpose of the DHCP release message? Does the DHCP server issue an acknowledgment of receipt of the client's DHCP request? What would happen if the client's DHCP release message is lost?

13. Clear the *bootp* filter from your Ethereal window. Were any ARP packets sent or received during the DHCP packet-exchange period? If so, explain the purpose of those ARP packets.

# Ethernet and ARP

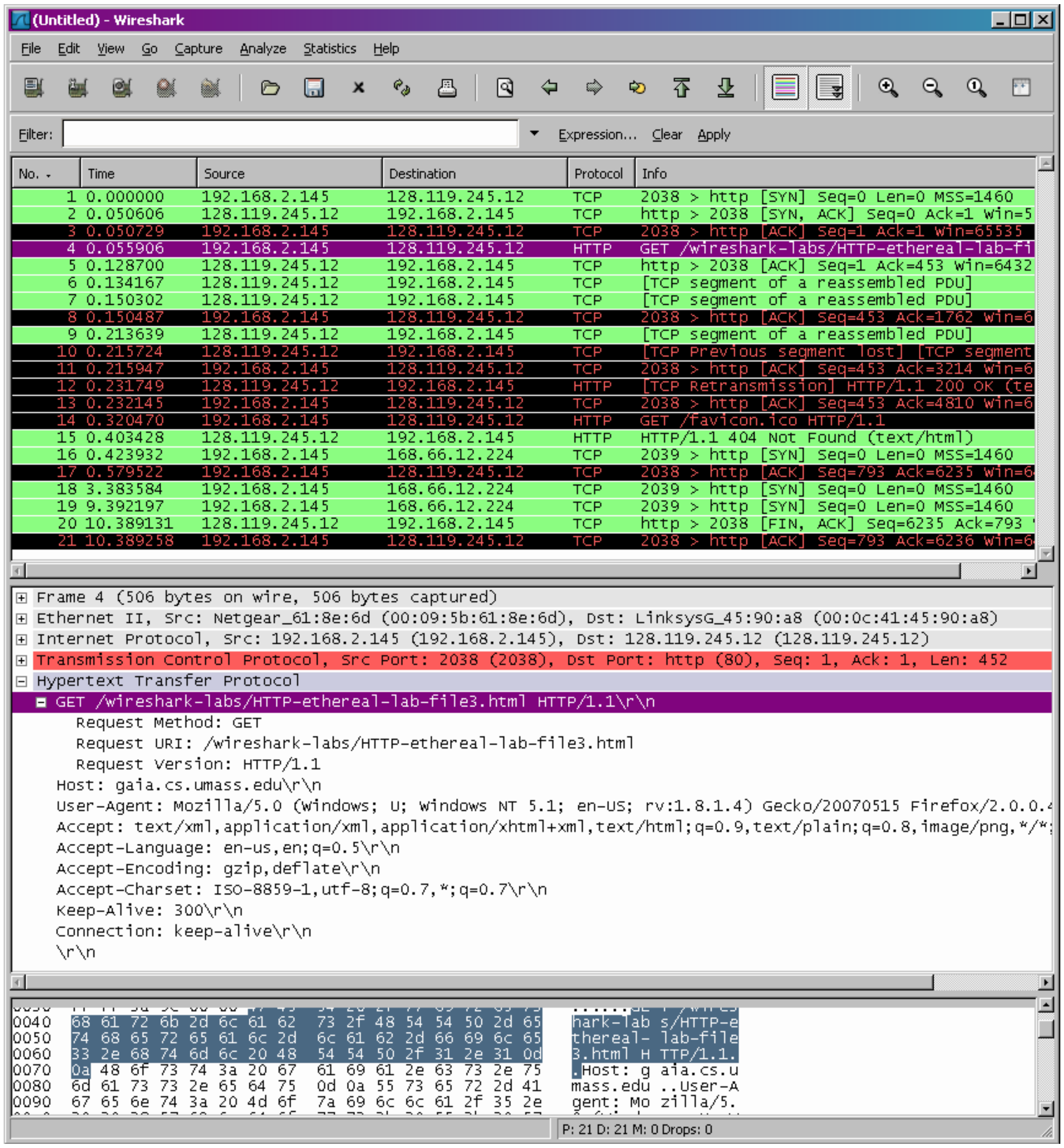
In the second part of this lab, we'll investigate the Ethernet protocol and the ARP protocol. Before beginning this lab, you'll probably want to review (Ethernet), 5.4.1 link- layer addressing and ARP in the text. RFC 826 (ftp://ftp.rfc-editor.org/in-notes/std/std37.txt) contains the details of the ARP 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[2]:

- Choose one of the following items according to your computer's operating system.
  - **Windows users** follow the instructions on https://techjourney.net/clear-delete-and-refresh-arp-cache-entry/ to clear ARP cache.
  - **Mac users** follow the instructions on https://www.ihash.eu/2015/01/clear-flush-arp-cache-os-x-yosemite/ to clear ARP cache.
  - **Linux users** follow the instructions on https://linux-audit.com/how-to-clear-the-arp-cache-on-linux/ to clear ARP cache.
- Start up the Wireshark packet sniffer
- Enter the following URL into your browser http:// http://edu.lva.virginia.gov/online_classroom/shaping_the_consti tution/doc/rights 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 Ethereal window) of the HTTP GET message that was sent from your computer to usinfo.state.gov, as well as the beginning of the HTTP response message sent to your computer by usinfo.state.gov. You should see a screen that looks something like this ( what you see may be a little bit different from this snapshot).

[2] If you are unable to run Ethereal live on a computer, you can download the Zip 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 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.



```
No.  Time       Source           Destination      Protocol  Info
  1 0.000000    192.168.2.145    128.119.245.12   TCP       2038 > http [SYN] Seq=0 Len=0 MSS=1460
  2 0.050606    128.119.245.12   192.168.2.145    TCP       http > 2038 [SYN, ACK] Seq=0 Ack=1 Win=5
  3 0.050729    192.168.2.145    128.119.245.12   TCP       2038 > http [ACK] Seq=1 Ack=1 Win=65535
  4 0.055906    192.168.2.145    128.119.245.12   HTTP      GET /wireshark-labs/HTTP-ethereal-lab-fi
  5 0.128700    128.119.245.12   192.168.2.145    TCP       http > 2038 [ACK] Seq=1 Ack=453 Win=6432
  6 0.134167    128.119.245.12   192.168.2.145    TCP       [TCP segment of a reassembled PDU]
  7 0.150302    128.119.245.12   192.168.2.145    TCP       [TCP segment of a reassembled PDU]
  8 0.150487    192.168.2.145    128.119.245.12   TCP       2038 > http [ACK] Seq=453 Ack=1762 Win=6
  9 0.213639    128.119.245.12   192.168.2.145    TCP       [TCP segment of a reassembled PDU]
 10 0.215724    128.119.245.12   192.168.2.145    TCP       [TCP Previous segment lost] [TCP segment
 11 0.215947    192.168.2.145    128.119.245.12   TCP       2038 > http [ACK] Seq=453 Ack=3214 Win=6
 12 0.231749    128.119.245.12   192.168.2.145    HTTP      [TCP Retransmission] HTTP/1.1 200 OK (te
 13 0.232145    192.168.2.145    128.119.245.12   TCP       2038 > http [ACK] Seq=453 Ack=4810 Win=6
 14 0.320470    192.168.2.145    128.119.245.12   HTTP      GET /favicon.ico HTTP/1.1
 15 0.403428    128.119.245.12   192.168.2.145    HTTP      HTTP/1.1 404 Not Found (text/html)
 16 0.423932    192.168.2.145    168.66.12.224    TCP       2039 > http [SYN] Seq=0 Len=0 MSS=1460
 17 0.579522    192.168.2.145    128.119.245.12   TCP       2038 > http [ACK] Seq=793 Ack=6235 Win=6
 18 3.383584    192.168.2.145    168.66.12.224    TCP       2039 > http [SYN] Seq=0 Len=0 MSS=1460
 19 9.392197    192.168.2.145    168.66.12.224    TCP       2039 > http [SYN] Seq=0 Len=0 MSS=1460
 20 10.389131   128.119.245.12   192.168.2.145    TCP       http > 2038 [FIN, ACK] Seq=6235 Ack=793
 21 10.389258   192.168.2.145    128.119.245.12   TCP       2038 > http [ACK] Seq=793 Ack=6236 Win=6
```

```
⊞ Frame 4 (506 bytes on wire, 506 bytes captured)
⊞ Ethernet II, Src: Netgear_61:8e:6d (00:09:5b:61:8e:6d), Dst: LinksysG_45:90:a8 (00:0c:41:45:90:a8)
⊞ Internet Protocol, Src: 192.168.2.145 (192.168.2.145), Dst: 128.119.245.12 (128.119.245.12)
⊞ Transmission Control Protocol, Src Port: 2038 (2038), Dst Port: http (80), Seq: 1, Ack: 1, Len: 452
⊟ Hypertext Transfer Protocol
  ⊟ GET /wireshark-labs/HTTP-ethereal-lab-file3.html HTTP/1.1\r\n
      Request Method: GET
      Request URI: /wireshark-labs/HTTP-ethereal-lab-file3.html
      Request Version: HTTP/1.1
    Host: gaia.cs.umass.edu\r\n
    User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8.1.4) Gecko/20070515 Firefox/2.0.0.
    Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;
    Accept-Language: en-us,en;q=0.5\r\n
    Accept-Encoding: gzip,deflate\r\n
    Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7\r\n
    Keep-Alive: 300\r\n
    Connection: keep-alive\r\n
    \r\n
```

```
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
0080  6d 61 73 73 2e 65 64 75  0d 0a 55 73 65 72 2d 41   mass.edu ..User-A
0090  67 65 6e 74 3a 20 4d 6f  7a 69 6c 6c 61 2f 35 2e   gent: Mo zilla/5.
```

P: 21 D: 21 M: 0 Drops: 0

(Untitled) - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter:
Expression... Clear Apply

- Since this lab has a lot of broadcasting info going on, we would like to filter out all the noise. Please type "ip.addr = = your computer IP address" into the filter field. 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 Ethereal).

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 in the text if you find this nesting 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 usinfo.state.gov? (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. How many bytes from the very start of the Ethernet frame does the ASCII “G” in “GET” appear in the Ethernet frame?

4. What is the hexadecimal value of the CRC field in this 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 usinfo.state.gov (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. 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?

8. What is the hexadecimal value of the CRC field in this Ethernet frame?

## 2. The Address Resolution Protocol

In this section, we'll observe the ARP protocol in action. We strongly recommend that you read section 5.4.2 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.** 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 *arp –a* command will display the contents of the ARP cache on your computer. Run the *arp –a* 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.

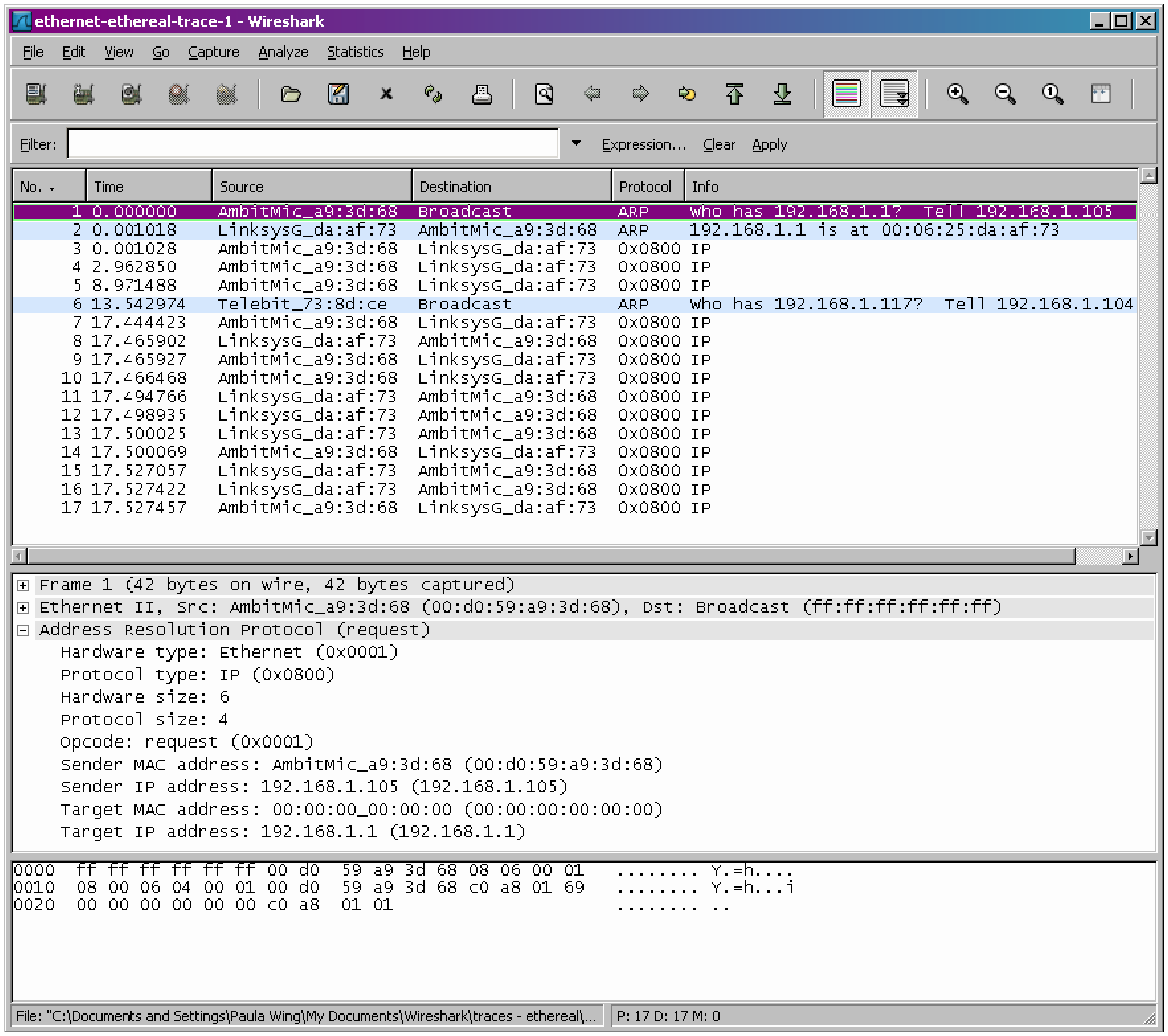
- **MS-DOS.** The MS-DOS *arp –d* * command will clear your ARP cache. The –*d* flag indicates a deletion operation, and the * is the wildcard that says to delete all table entries.
- **Linux/Unix.** The *arp –d* * will clear your ARP cache. In order to run this command you'll need root privileges. For Mac use, *sudo arp –a –d* will clear your ARP cache. 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 footnote 1.

## Observing ARP in action

Do the following[3]:

- Clear your ARP cache, as described above.
- Next, make sure your browser's cache is empty. (To do this under Firefox, select *Firefox->History->Clear History* and clear the memory and disk cache.*)*
- Start up the Wireshark packet sniffer
- Enter the following URL into your browser http://edu.lva.virginia.gov/online_classroom/shaping_the_constitution/doc/rights. Your browser should again display the rather lengthy US Bill of Rights.
- Stop Wireshark packet capture. Type "arp" into the filter field. You should now see an Ethereal window that looks like:

3 The *ethernet--ethereal-trace-* 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).


ethernet-ethereal-trace-1 - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter:
Expression... Clear Apply

```
No.  Time       Source             Destination        Protocol Info
  1 0.000000   AmbitMic_a9:3d:68  Broadcast          ARP      Who has 192.168.1.1?  Tell 192.168.1.105
  2 0.001018   LinksysG_da:af:73  AmbitMic_a9:3d:68  ARP      192.168.1.1 is at 00:06:25:da:af:73
  3 0.001028   AmbitMic_a9:3d:68  LinksysG_da:af:73  0x0800 IP
  4 2.962850   AmbitMic_a9:3d:68  LinksysG_da:af:73  0x0800 IP
  5 8.971488   AmbitMic_a9:3d:68  LinksysG_da:af:73  0x0800 IP
  6 13.542974  Telebit_73:8d:ce   Broadcast          ARP      Who has 192.168.1.117?  Tell 192.168.1.104
  7 17.444423  AmbitMic_a9:3d:68  LinksysG_da:af:73  0x0800 IP
  8 17.465902  LinksysG_da:af:73  AmbitMic_a9:3d:68  0x0800 IP
  9 17.465927  AmbitMic_a9:3d:68  LinksysG_da:af:73  0x0800 IP
 10 17.466468  AmbitMic_a9:3d:68  LinksysG_da:af:73  0x0800 IP
 11 17.494766  LinksysG_da:af:73  AmbitMic_a9:3d:68  0x0800 IP
 12 17.498935  LinksysG_da:af:73  AmbitMic_a9:3d:68  0x0800 IP
 13 17.500025  LinksysG_da:af:73  AmbitMic_a9:3d:68  0x0800 IP
 14 17.500069  AmbitMic_a9:3d:68  LinksysG_da:af:73  0x0800 IP
 15 17.527057  LinksysG_da:af:73  AmbitMic_a9:3d:68  0x0800 IP
 16 17.527422  LinksysG_da:af:73  AmbitMic_a9:3d:68  0x0800 IP
 17 17.527457  AmbitMic_a9:3d:68  LinksysG_da:af:73  0x0800 IP

⊞ Frame 1 (42 bytes on wire, 42 bytes captured)
⊞ 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)
    Hardware type: Ethernet (0x0001)
    Protocol type: IP (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (0x0001)
    Sender MAC address: AmbitMic_a9:3d:68 (00:d0:59:a9:3d:68)
    Sender IP address: 192.168.1.105 (192.168.1.105)
    Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Target IP address: 192.168.1.1 (192.168.1.1)

0000  ff 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                     ........ ..

File: "C:\Documents and Settings\Paula Wing\My Documents\Wireshark\traces - ethereal\...   P: 17 D: 17 M: 0
```

In the example above, the first two frames in the trace contain ARP messages (as does the 6th message). The screen shot above corresponds to the trace referenced in footnote 3.

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. 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?

12. 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?

13. What are the hexadecimal values for the source and destination addresses in the Ethernet frame containing the ARP reply message?

14. Open the *ethernet--ethereal-trace-* 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 Ethereal, and the ARP reply sent to the computer running Ethereal by the computer with the ARP-requested Ethernet address. But there is yet another computer on this network, as indiated 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?