Wireshark Lab 2 – Ethernet ARP

In this lab, we'll investigate the Ethernet protocol and the ARP protocol. Before beginning this lab, you'll probably want to review 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. (Select *Tools->Internet Options->Delete Files)*
• Start up the Wireshark packet sniffer
• Enter the following URL into your browser *http://pages.intnet.mu/rhh/wireshark/file3.html*
• Stop Wireshark packet capture. First, note down the packet numbers (the leftmost column in the upper Wireshark window) of the HTTP GET message that was sent from your computer to *pages.intnet.mu*, as well as the first of the HTTP response message sent back to you. You should see a screen like Fig. 1 (where packet 5 in ***my screenshot*** contains the HTTP GET message).

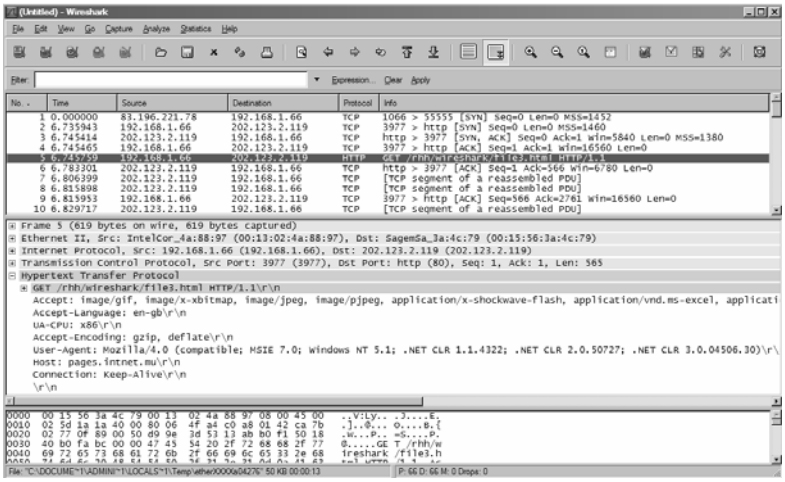

Fig. 1

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 a Wireshark window that looks like this:

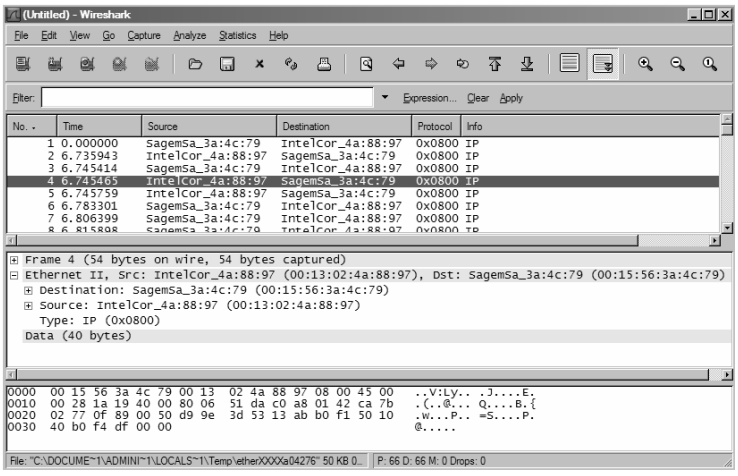

Fig. 2

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). 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 include a screenshot of the packets captured that you used to answer the question asked.

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 *pages.intnet.mu*? (Hint: the answer is *No*). Which device has this as its Ethernet address? [Note: this is an important question and one that students sometimes get wrong]

3. Give the hexadecimal value for the two-byte Frame type field. What does this value mean?

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 *pages.intnet.mu*? (Hint: the answer is *No*). What device has this as its Ethernet address?

6. What is the Ethernet destination address? Is this the Ethernet address of your computer?

7. Give the hexadecimal value for the two-byte Frame type field. What does this mean?

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. I strongly recommend that you refresh yourself on this topic before proceeding.

ARP Caching

Recall that the ARP protocol typically maintains a cache of IP-to-Ethernet address translation pairs on your comnputer 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.** Open a Command Prompt Window by typing *File->Run* and enter *cmd* then at the prompt enter *arp –a* and press Enter.
The *arp –a* command will display the contents of the ARP cache on your computer.

11. 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.

Observing ARP in action

Do the following:
• Clear your ARP cache, as described above.
• Next, make sure your browser's cache is empty. (select *Tools->Internet Options->Delete Files.)*
• Start up the Wireshark packet sniffer
• Enter the following URL into your browser *http://pages.intnet.mu/rhh/wireshark/file3.html*
Your browser should again display the rather lengthy UTM Act 2002.
• 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 IP box and select *OK*. You should now see a Wireshark window that looks like:

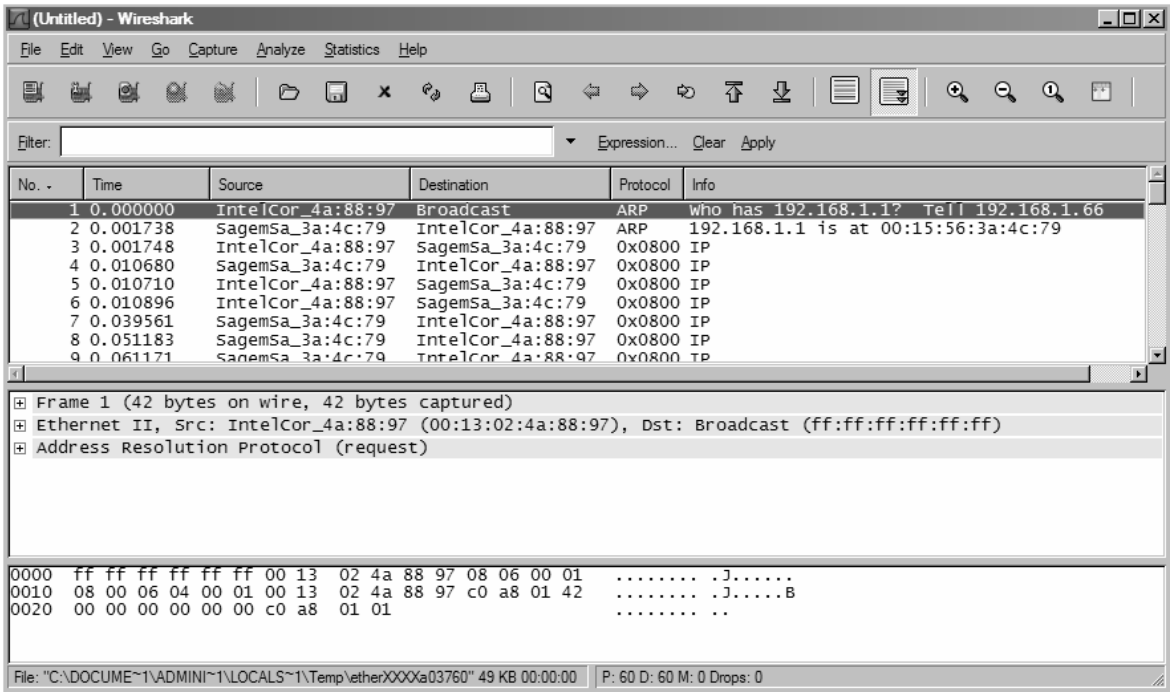

Fig. 3

In the example above, the first two frames in the capture contain ARP messages.

Answer the following questions:

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

13. Give the hexadecimal value for the two-byte Ethernet Frame type field. What does this mean?

14. Download the ARP specification from *http://www.networksorcery.com/enp/protocol/arp.htm*

a) How many bytes from the 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?

15. 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?

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

Extra Credit

EX-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?

EX-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.