| Wireshark Lab:  Ethernet and ARP v8.1  Supplement to *Computer Networking: A Top-Down Approach, 8th ed.,* J.F. Kurose and K.W. Ross  *“Tell me and I forget. Show me and I remember. Involve me and I understand.”* Chinese proverb  © 2005-2021, J.F Kurose and K.W. Ross, All Rights Reserved | *A picture containing outdoor, water, bridge, building  Description automatically generated* |
| --- | --- |

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]](#footnote-0). 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.

1. Capturing and analyzing Ethernet frames

Let’s begin by capturing a set of Ethernet frames to study. To do this, of course, you’ll need access to a wired Ethernet connection for your PC or Mac – not necessarily a common scenario these days, given the popularity of wireless WiFi and cellular access. If you’re unable to run Wireshark on a live Ethernet connection, you can download a packet trace that was captured while following the steps below on one of the author’s computers[[2]](#footnote-1). In addition, you may well find it valuable to download this trace even if you’ve captured your own trace and use it, as well as your own trace, when you explore the questions below.

Do the following:

* First, make sure your browser’s cache of previously downloaded documents is empty.
* Start up Wireshark and enter the following URL into your browser: <http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file3.html>. Your browser should display the rather lengthy US Bill of Rights.
* Stop Wireshark packet capture.

First, find the packet number (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 126 in the screen shot below contains the HTTP GET message)

|  |
| --- |
| **Figure 1:** Wireshark display showing HTTP GET message for http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file3.html |

Since this lab is about Ethernet and ARP, we’re not interested in high-level protocols like IP, TCP or HTTP. We’re interested in Ethernet frames and ARP messages!

Let’s start by looking at 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 notion of 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. Your display should look similar to that shown in Figure 2.

|  |
| --- |
| **Figure 2:** Wireshark display showing details of the Ethernet frame containing the HTTP GET request. |

In answering the questions below[[3]](#footnote-2), you can use either your own live trace, or use the Wireshark captured packet file *Ethernet-wireshark-trace1* in <http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces-8.1.zip>

1. **What is the 48-bit Ethernet address of your computer?**

The 48-bit Ethernet address of my computer is e0:be:03:3f:b7:1e.

1. **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 483-484 in the text and make sure you understand the answer here.]**

The 48-bit destination address in the Ethernet frame is 1c:61:b4:47:ef:f0. The destination address in the Ethernet frame which is the path, is TPLink\_47:3f:f0.

1. **What is the hexadecimal value for the two-byte Frame type field in the Ethernet frame carrying the HTTP GET request?  What upper layer protocol does this correspond to?**

The hexadecimal value is (0x0800).

The upper layer protocol this corresponds to IPv4.

1. **How many bytes from the very start of the Ethernet frame does the ASCII “G” in “GET” appear in the Ethernet frame? Do not count any preamble bits in your count, i.e., assume that the Ethernet frame begins with the Ethernet frame's destination address.**

The ASCII “G” in “GET” appear to be 47, but the preamble bits is 54 bytes.

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

1. **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?**

The 48-bit destination address in the Ethernet frame is 1c:61:b4:47:ef:f0.

The device that has its Ethernet address is a router and not the address of my computer, or of gaia.cs.umass.edu

1. **What is the destination address in the Ethernet frame?**

**Is this the Ethernet address of your computer?**

The destination address is e0:cb:4e:03:3f:b7, is the Ethernet address of my computer.

1. **Give the hexadecimal value for the two-byte Frame type field. What upper layer protocol does this correspond to?**

The hexadecimal value is (0x0800) corresponds to IPv4.

1. **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? Do not count any preamble bits in your count, i.e., assume that the Ethernet frame begins with the Ethernet frame's destination address.**

The ASCII "O" in "OK" appears at byte offset 68 in the Ethernet frame and s is counting from the start of the Ethernet frame, otherwise 67 Bytes.

1. **How many Ethernet frames (each containing an IP datagram, each containing a TCP segment) carry data that is part of the complete HTTP “OK 200 ...” reply message?**

* 4 frames which contains #356(1412), #358(1412), #360(1412), #361(625)

1. References to figures and sections are for the 8th edition of our text, *Computer Networks, A Top-down Approach, 8th ed., J.F. Kurose and K.W. Ross, Addison-Wesley/Pearson, 2020.* Our authors’ website for this book is <http://gaia.cs.umass.edu/kurose_ross> You’ll find lots of interesting open material there. [↑](#footnote-ref-0)
2. You can download the zip file <http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces-8.1.zip> and extract the trace file *ethernet-wireshark-trace1*. This trace file can be used to answer this Wireshark lab without actually capturing packets on your own. This trace was made using Wireshark running on one of the author’s computers, while performing the steps indicated in this Wireshark lab. Once you’ve downloaded a trace file, you can load it into Wireshark and view the trace using the *File* pull down menu, choosing *Open*, and then selecting the trace file name. [↑](#footnote-ref-1)
3. For the author’s class, when answering the following questions with hand-in assignments, students sometimes need to print out specific packets (see the introductory Wireshark lab for an explanation of how to do this) and indicate where in the packet they’ve found the information that answers a question. They do this by marking paper copies with a pen or annotating electronic copies with text in a colored font. There are also learning management system (LMS) modules for teachers that allow students to answer these questions online and have answers auto-graded for these Wireshark labs at <http://gaia.cs.umass.edu/kurose_ross/lms.htm> [↑](#footnote-ref-2)