Wireshark Lab 1
HTTP

In this lab, we will explore several aspects of the HTTP protocol: the basic GET-response interaction, HTTP message formats, retrieving large HTML files, retrieving HTML files with embedded objects, and HTTP authentication and security.

1 The Basic HTTP GET-response interaction

1. Start up your web browser.
2. Start up the Wireshark packet sniffer. Enter "http" in the display-filter-specification window, so that only captured HTTP messages will be displayed later in the packet-listing window. (We're only interested in the HTTP protocol here, and don't want to see the clutter of all captured packets).
3. Enter the following URL in your web browser:

   `http://www.cas.mcmaster.ca/~soltys/lab/ws-file-1.html`

4. Now stop the Wireshark packet capture.
5. Find the lines in Wireshark that correspond to your browser retrieving the above URL. That is, the "GET message" (from your browser to the web server) and the response message from the server to your browser. The packet-contents window shows details of the selected message (in this case the HTTP OK message, which is highlighted in the packet-listing window). Recall that since the HTTP message was carried inside a TCP segment, which was carried inside an IP datagram, which was carried within an Ethernet frame, Wireshark displays the Frame, Ethernet, IP, and TCP packet information as well.
6. Make sure the boxes at the far left of the Frame, Ethernet, IP and TCP information have a plus sign or a right-pointing triangle (which means there is hidden, undisplayed information), and the HTTP line has a minus sign or a down-pointing triangle (which means that all information about the HTTP message is displayed).
7. By looking at the information in the HTTP GET and response messages, answer the following questions. When you hand in your lab, annotate screen captures so that it is clear where in the output you are getting the information for your answer.
   (a) Is your browser running HTTP version 1.0 or 1.1? What version of HTTP is the server running?
   (b) What languages (if any) does your browser indicate that it can accept to the server?
   (c) What is the IP address of your computer? Of the `www.cas.mcmaster.ca` server?

(d) What is the status code returned from the server to your browser?

(e) When was the HTML file that you are retrieving last modified at the server?

(f) How many bytes of content are being returned to your browser?

(g) By inspecting the raw data in the packet content window, do you see any headers within the data that are not displayed in the packet-listing window? If so, name one.

2 The HTTP CONDITIONAL GET-response interaction

Most web browsers perform object caching and thus perform a conditional GET when retrieving an HTTP object. Before performing the steps below, make sure your browser's cache is empty.

1. Enter the following URL in your web browser:

   `http://www.cas.mcmaster.ca/~soltys/lab/ws-file-2.html`

2. Quickly enter the same URL into your browser again (or simply select the refresh button on your browser).

3. Examine the "http" display-filter-specification in Wireshark, and answer the following question:

   (a) Inspect the contents of the first HTTP GET request from your browser to the server. Do you see an "IF-MODIFIED-SINCE" line in the HTTP GET?

   (b) Inspect the contents of the server response. Did the server explicitly return the contents of the file? How can you tell?

   (c) Now inspect the contents of the second HTTP GET request from your browser to the server. Do you see an "IF-MODIFIED-SINCE:" line in the HTTP GET? If so, what information follows the "IF-MODIFIED-SINCE:" header?

   (d) What is the HTTP status code and phrase returned from the server in response to this second HTTP GET? Did the server explicitly return the contents of the file? Explain.

3 Retrieving Long Documents

In our examples thus far, the documents retrieved have been simple and short HTML files.

Let's next see what happens when we download a long HTML file. Start up your web browser, and make sure your browser's cache is cleared. Start up the Wireshark packet sniffer, and do the following:

1. Enter the following URL in your web browser:

   `http://www.cas.mcmaster.ca/~soltys/lab/ws-file-3.html`

2. Examine the "http" display-filter-specification in Wireshark. In the packet-listing window, you should see your HTTP GET message, followed by a multiple-packet TCP response to your HTTP GET request. HTTP response message consists of a status line, followed by header lines, followed by a blank line, followed by the entity body. In the case of our HTTP GET, the entity body in the response is the entire requested HTML file. In our case here, the HTML file is rather long, and at 3.9Kb it is too large to fit in one TCP packet. The single HTTP response message is thus broken into several pieces by TCP, with each piece being contained within a separate TCP segment.

3. Answer the following questions:

   (a) How many HTTP GET request messages did your browser send? Which packet number in the trace contains the GET message for the McMaster plagiarism policy?

   (b) Which packet number in the trace contains the status code and phrase associated with the response to the HTTP GET request?

   (c) What is the status code and phrase in the response?

   (d) How many data-containing TCP segments were needed to carry the single HTTP response and the text?