Networking Lab 5 – analysis of normal web browsing

1. Normal Web Browsing analysis

The goal of this laboratory is to analyse traces collected during normal web browsing operations. Starting from simple pages, we understand how DNS and HTTP work first. Next, we try to observe the normal browsing of complicated webpages. This lab is deeply inspired by the laboratory proposed in the Book "Computer Networking: A top-down approach" by Jim Kurose and Keith Ross.

Leave the PCs connected to the normal LAIB network for this experience and configure H1, H2 and H3 so that they automatically get a valid IP address from the LAB Network (i.e., let the Linux Network Configuration manager configure the network setup via DHCP).

Warning: if you cannot connect to the didattica.polito.it portal, it may be because the DNS is not properly configured in your system. Check the file /etc/resolv.conf. If it does not exist, try the following

```
# delete the wrong configuration
sudo rm /etc/resolv.conf
# replace with the one got from the DHCP
sudo ln –s /run/resolvconf/resolv.conf  /etc
```

2. The Basic HTTP GET/Response interaction

Let's begin our exploration of HTTP by downloading a very simple HTML file - one that is very short, and contains no embedded objects. Do the following:

1. Start up your web browser
2. Start up the Wireshark packet sniffer. Enter "http" (just the letters, not the quotation marks) in the display-filter-specification window, so that only captured HTTP messages will be displayed later in the packet-listing window. Try to get rid of other HTTP messages that are not involved in the browsing session
3. Begin Wireshark packet capture.
4. Enter the following to your browser `http://www.tstat.polito.it/wireshark-labs/HTTP-wireshark-file1.html` Your browser should display the very simple, one-line HTML file.
5. Stop Wireshark packet capture.
6. Analyze the packet trace – HTTP analysis
    1. Is your browser running HTTP version 1.0 or 1.1? What version of HTTP is the server running?
    2. What languages (if any) does your browser indicate that it can accept to the server?
    3. What is the IP address of your computer? Of the `www.tstat.polito.it` server?
    4. What is the status code returned from the server to your browser?
    5. When was the HTML file that you are retrieving last modified at the server?
    6. How many bytes of content are being returned to your browser?
    7. Is there a second HTTP request in the trace? What is that?

7. Repeat the request, by forcing the browser to reload the page
    1. Press the reload icon. How is the response modified?
    2. Press `SHIFT` + the reload icon. What is the difference?
8. Use `telnet` to download the page:
    1. Open 4 terminal windows, and in each of them use telnet to connect to the same webserver via
       `telnet www.tstat.polito.it http`
    2. Enter the following lines

WINDOW 1
`GET /wireshark-labs/HTTP-wireshark-file1.html HTTP/1.0`
`[return]`

WINDOW 2
`GET /wireshark-labs/HTTP-wireshark-file1.html HTTP/1.0`
`Host: www.tstat.polito.it`
`[return]`

WINDOW 3
`GET /wireshark-labs/HTTP-wireshark-file1.html HTTP/1.1`
`[return]`

WINDOW 4
`GET /wireshark-labs/HTTP-wireshark-file1.html HTTP/1.1`
`Host: www.tstat.polito.it`
`[return]`

    3. Check the results on each window. What are the differences? Observe the response code from the server. What is the difference between HTTP 1.0 and HTTP 1.1?
    4. Observe the packet trace. How many TCP connections are present? Who initiates the connection tear-down? When? How are PUSH flags being used by the client and the server?
9. Repeat the analysis on the following websites using the browser and pointing to:
    1. http://www.tstat.polito.it/wireshark-labs/HTTP-wireshark-file3.html
    2. http://www.tstat.polito.it/wireshark-labs/HTTP-wireshark-file5.html
    3. http://www.tstat.polito.it/wireshark-labs/protected_pages/HTTP-wireshark-file5.html
       username: wireshark-students ; passwork: network
    4. http://www.tstat.polito.it/wireshark-labs/wireshark-form.php
       compile the form and submit it
10. Observe how the HTTP protocol works, and how HTML pages are processed to retrieve objects referenced in the source of the HTML code.
    1. How many TCP connections are being used? Toward which servers?
    2. How does the response code change?
    3. Why some objects fail to load?
    4. Can you observe the username and password in the trace?
       NOTE: While it may appear that your username and password are encrypted, they are simply encoded in a format known as Base64 format. Username and password are *not* encrypted!

To see this, go to http://www.motobit.com/util/base64-decoder-encoder.asp and enter the string `d2lyZXNoYXJrLXN0dWRlbnRz` and decode. *Voila!*

5. How is timing of the protocol being defined? Use the I/O graph to display over time
    1. DNS requests in red
    2. TCP SYN packets in blue
    3. HTTP requests in green
    4. HTTP response in violet