Lab #3

Sriram R SRN : PES1UG20CS435 Section : H

(Note : Answers on pages 10-16 & 25-28)

1. Understanding Persistent and Non-persistent HTTP Connections

To understand persistent and non-persistent HTTP connections and corresponding performance impact.

Create a web page with N (e.g. 10) embedded images. Each image should be of minimum 2 MB size. Configure your browser (Firefox) with following settings (each setting requires repeat of experiment)

- Non persistent connection
- 2 persistent connections
- 4 persistent connections
- 6 persistent connections
- 10 persistent connections.

Observation: Note down the time taken to display the entire page in each of the settings. Ensure that (cache is cleared before starting the web request). Explain the response time differences. What is the optimal number of persistent connections for best performance? Explain your answer.

Introduction

The Apache HTTP server is the most widely used web server in the world. It provides many powerful features including dynamically loadable modules, robust media support, and extensive integration with other popular software.

Objective: Understand persistent and non-persistent HTTP connections and corresponding performance impact.

Experiment: Create a web page with N (e.g. 10) embedded images. Each image should be of minimum 2 MB size. Configure your browser (Firefox) with following settings (each setting requires repeat of experiment)

a) Non-persistent connection
b) 2 persistent connections
c) 4 persistent connections
d) 6 persistent connections
e) 10 persistent connections

Note down the time taken to display the entire page in each of the settings. Ensure that cache is cleared before starting the web request. Explain the response time differences. What is the optimal number of persistent connections for best performance? Explain your answer.

Note: To install Apache server, use the following command,

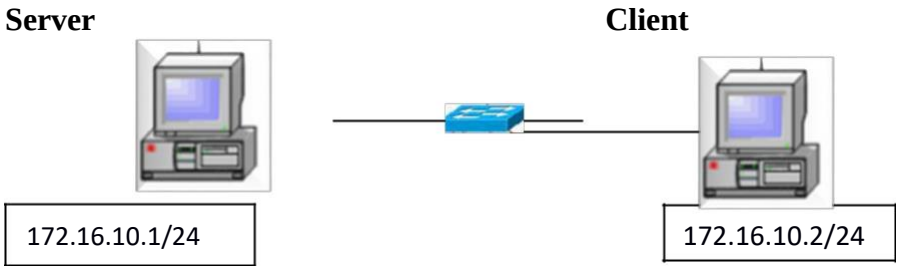
```
sudo apt-get install apache2
```

If there is any error during installation, update the package manager by issuing the command,

```
sudo apt-get update
```

EXECUTION STEPS

Step 1: Connect 2 desktops using switch and cables as shown below. (Use 2 VMs on Virtualbox or VMware instead of physical connections.)



Server Side:

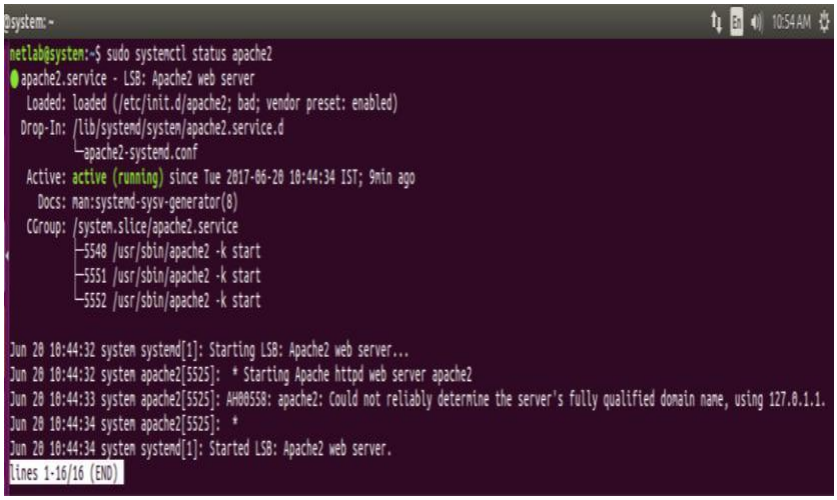
Step 2: Check your Web Server

At the end of the installation process, Ubuntu 16.04 starts Apache. The web server should already be up and running. We can check with the `systemctl` command to make sure the service is running by typing:

sudo systemctl status apache2

or

sudo service apache2 status



As you can see above, the service appears to have started successfully. However, the best way to test this is to actually request a page from Apache. You can access the default Apache landing page to confirm that the software is running properly. You can access this through your server's domain name or IP address.

Step 3: Server IP address can be set by the following command

$sudo ip addr add 172.16.10.1/24 dev enps0
$sudo ip addr

Note: If IP address fluctuates, kindly setup the IP address manually using ‘Edit connections’.

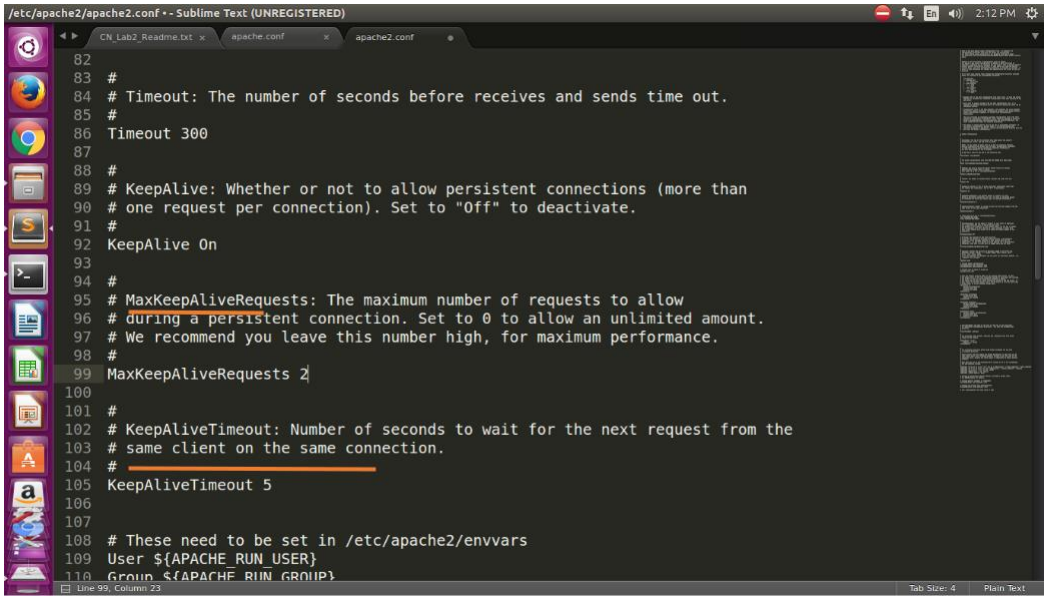
```
student@student-H81H3-I:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp2s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether b8:ae:ed:a5:a5:a9 brd ff:ff:ff:ff:ff:ff
    inet 172.16.10.1/24 brd 172.16.10.255 scope global enp2s0
       valid_lft forever preferred_lft forever
    inet6 fe80::c901:c994:4cf5:f837/64 scope link
       valid_lft forever preferred_lft forever
student@student-H81H3-I:~$
```

Step 4: The **apache2.conf** file present in the **etc/apache2** directory is modified as:

a) The **keep-alive** option was set (i.e. value was made **ON**)

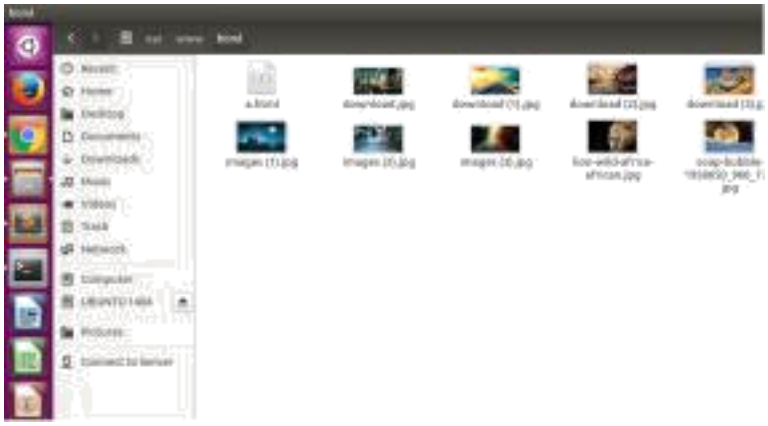
b) The **MaximumKeepAliveRequests** were set to

2 $sudo nano /etc/apache2/apache2.conf



Step 5: Store images in the server path. A html page consisting of 10 images having size > 2MB were placed and accessed by the client. This html page is stored in the location - **/var/www/html/file_name.html**.

Note: Use the images provided by faculty incharges.



Step 6: Prepare a web page as shown below. The html file needs to add 10 images. (Kindly skip the style attribute in the below image)

a.html [Read-Only] (/var/www/html) - gedit

Open

```
<!DOCTYPE html>
<html>
<body>

<h2>Spectacular Mountain</h2>
<img src="images (1).jpg" alt="Mountain View" style="width:304px;height:228px;">
<img src="images (2).jpg" style="width:304px;height:228px;">
<img src="download (4).jpg" alt="Mountain View" style="width:304px;height:228px;">
<img src="images (3).jpg" alt="Mountain View" style="width:304px;height:228px;">
<img src="lion-wild-africa-african.jpg" alt="Mountain View" style="width:304px;height:228px;">
<img src="images.jpg" alt="Mountain View" style="width:304px;height:228px;">
<img src="download.jpg" alt="Mountain View" style="width:304px;height:228px;">
<img src="download (1).jpg" alt="Mountain View" style="width:304px;height:228px;">
<img src="soap-bubble-1958650_960_720.jpg" alt="Mountain View" style="width:304px;height:228px;">
<img src="download (2).jpg" alt="Mountain View" style="width:304px;height:228px;">

</body>
</html>
```

Client side:

Client IP address can be set by the following command.

$sudo ip addr add 172.16.10.2/24 dev enps0

$sudo ip addr

Note: If IP address fluctuates, kindly setup the IP address manually using ‘Edit connections’.

```
student@student-H81H3-I:~$ sudo ip addr add 172.16.10.2/24 dev enp2s0
student@student-H81H3-I:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp2s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
group default qlen 1000
    link/ether b8:ae:ed:a5:a6:32 brd ff:ff:ff:ff:ff:ff
    inet 172.16.10.2/24 scope global enp2s0
       valid_lft forever preferred_lft forever
    inet6 fe80::8bf0:837a:849e:a79f/64 scope link
       valid_lft forever preferred_lft forever
student@student-H81H3-I:~$
```

There are broadly two parts of execution:

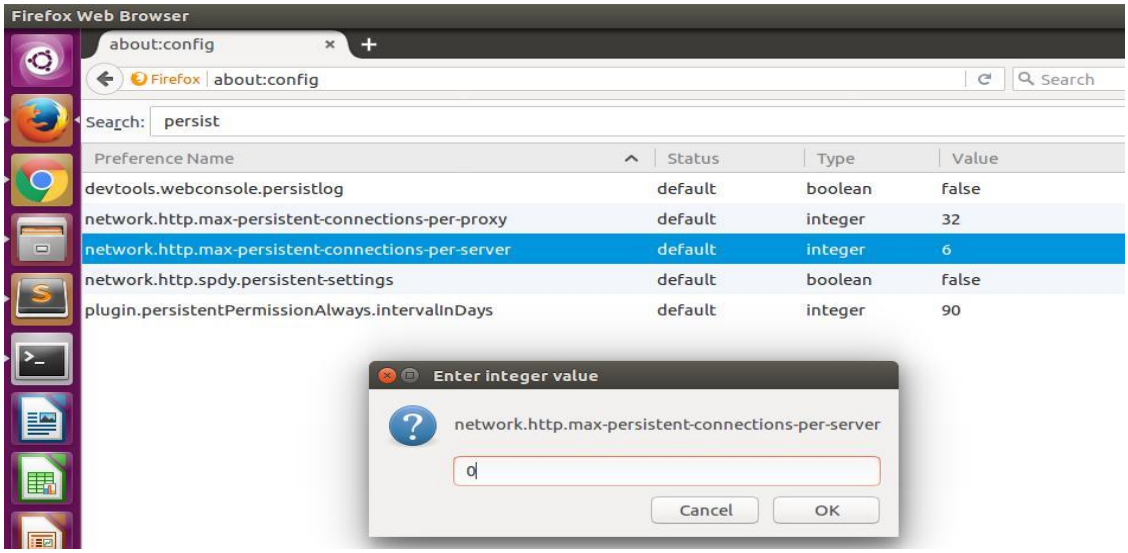
1. Dealing with non-persistent connections
2. Dealing with persistent connections

Open Firefox browser to configure for persistent option. Go to browser and type **about:config** and search for the term **'persistent'**

- While using non-persistent connection experiment, the **max-persistent-connections-per-server** has the value set to **0** and **persistent-settings** value set to false.
- While using persistent connection experiment, the **max-persistent-connections-per-server** should have value greater than 0 (depending on the number of persistent connections needed) and **persistent-settings** value set to true.

PART 1: NON-PERSISTENT CONNECTION

Step 1: This is done by setting the value of max-persistent-connection-per-server to 0 in the client computer.



Step 2: Access web page on client-side browser (Firefox)

The client could access the file as:

172.16.10.1/file_name.html where--> **172.16.10.1** is Server's IP

Here the file name is **a.html** present in server. So, by tying **172.16.10.1/a.html** in client browser, we will be able to open the requested web page.

Note 1: The wireshark should capture the packets between the client and the server while the file is accessed.

Note 2: The images in the HTML page should have all the permissions specified through the server for the proper access.

Step 3: Use wireshark**.** Open wireshark in the server computer while client is trying to access the server's local host webpage. Apply 'http' filter and note the time to capture all the 10 images.

http

No.	Time	Source	Destination	Protocol	Length	Info
25	0.211530105	172.16.10.1	172.16.10.2	HTTP	568	HTTP/1.1 404 Not Found (text/html)
27	2.070581279	172.16.10.2	172.16.10.1	HTTP	421	GET /a.html HTTP/1.1
28	2.070866155	172.16.10.1	172.16.10.2	HTTP	641	HTTP/1.1 200 OK (text/html)
30	2.117160769	172.16.10.2	172.16.10.1	HTTP	347	GET /images%20(1).jpg HTTP/1.1
35	2.117571913	172.16.10.1	172.16.10.2	HTTP	1200	HTTP/1.1 200 OK (JPEG JFIF image)
36	2.117753115	172.16.10.2	172.16.10.1	HTTP	347	GET /images%20(2).jpg HTTP/1.1
45	2.117944288	172.16.10.1	172.16.10.2	HTTP	463	HTTP/1.1 200 OK (JPEG JFIF image)
51	2.118574057	172.16.10.2	172.16.10.1	HTTP	349	GET /download%20(4).jpg HTTP/1.1
63	2.119058490	172.16.10.1	172.16.10.2	HTTP	242	HTTP/1.1 200 OK (JPEG JFIF image)
65	2.119487932	172.16.10.2	172.16.10.1	HTTP	347	GET /images%20(3).jpg HTTP/1.1
77	2.119784374	172.16.10.1	172.16.10.2	HTTP	565	HTTP/1.1 200 OK (JPEG JFIF image)
79	2.120323770	172.16.10.2	172.16.10.1	HTTP	359	GET /lion-wild-africa-african.jpg HTTP/1.1
94	2.121263792	172.16.10.2	172.16.10.1	HTTP	341	GET /images.jpg HTTP/1.1
110	2.122045168	172.16.10.1	172.16.10.2	HTTP	1226	HTTP/1.1 200 OK (JPEG JFIF image)
117	2.122719543	172.16.10.2	172.16.10.1	HTTP	343	GET /download.jpg HTTP/1.1
138	2.123847115	172.16.10.2	172.16.10.1	HTTP	349	GET /download%20(1).jpg HTTP/1.1
160	2.124700199	172.16.10.2	172.16.10.1	HTTP	362	GET /soap-bubble-1958650_960_720.jpg HTTP/1.1
164	2.124733805	172.16.10.1	172.16.10.2	HTTP	1017	HTTP/1.1 200 OK (JPEG JFIF image)
171	2.125125151	172.16.10.1	172.16.10.2	HTTP	711	HTTP/1.1 200 OK (JPEG JFIF image)
184	2.126599573	172.16.10.2	172.16.10.1	HTTP	349	GET /download%20(2).jpg HTTP/1.1
252	2.131056667	172.16.10.1	172.16.10.2	HTTP	114	HTTP/1.1 200 OK (JPEG JFIF image)
529	2.151487483	172.16.10.1	172.16.10.2	HTTP	73	HTTP/1.1 200 OK (JPEG JFIF image)
3834	2.429637133	172.16.10.1	172.16.10.2	HTTP	1124	HTTP/1.1 200 OK (JPEG JFIF image)

Here it is 2.429637133 - 2.070581279 = 0.359055854

PART 2: PERSISTENT CONNECTIONS

Step 1: For 2 persistent connections, set the value of **max-persistent-connection-per-server to 2** in the client computer.

Step 2: Repeat the **steps 1-3** in the previous section.

http

No.	Time	Source	Destination	Protocol	Length	Info
28	0.158495832	172.16.10.1	172.16.10.2	HTTP	568	HTTP/1.1 404 Not Found (text/html)
30	2.685888334	172.16.10.2	172.16.10.1	HTTP	421	GET /a.html HTTP/1.1
31	2.686488793	172.16.10.1	172.16.10.2	HTTP	641	HTTP/1.1 200 OK (text/html)
33	2.734091058	172.16.10.2	172.16.10.1	HTTP	347	GET /images%20(1).jpg HTTP/1.1
38	2.734592637	172.16.10.2	172.16.10.1	HTTP	347	GET /images%20(2).jpg HTTP/1.1
39	2.734696958	172.16.10.1	172.16.10.2	HTTP	1200	HTTP/1.1 200 OK (JPEG JFIF image)
48	2.735025557	172.16.10.1	172.16.10.2	HTTP	463	HTTP/1.1 200 OK (JPEG JFIF image)
49	2.735180365	172.16.10.2	172.16.10.1	HTTP	349	GET /download%20(4).jpg HTTP/1.1
66	2.736079156	172.16.10.1	172.16.10.2	HTTP	243	HTTP/1.1 200 OK (JPEG JFIF image)
68	2.736374643	172.16.10.2	172.16.10.1	HTTP	347	GET /images%20(3).jpg HTTP/1.1
82	2.736755733	172.16.10.1	172.16.10.2	HTTP	565	HTTP/1.1 200 OK (JPEG JFIF image)
85	2.737381832	172.16.10.2	172.16.10.1	HTTP	359	GET /lion-wild-africa-african.jpg HTTP/1.1
92	2.737840608	172.16.10.2	172.16.10.1	HTTP	341	GET /images.jpg HTTP/1.1
101	2.738335480	172.16.10.2	172.16.10.1	HTTP	343	GET /download.jpg HTTP/1.1
119	2.738809142	172.16.10.1	172.16.10.2	HTTP	1226	HTTP/1.1 200 OK (JPEG JFIF image)
121	2.739075438	172.16.10.1	172.16.10.2	HTTP	1016	HTTP/1.1 200 OK (JPEG JFIF image)
139	2.740900738	172.16.10.2	172.16.10.1	HTTP	349	GET /download%20(1).jpg HTTP/1.1
143	2.741014891	172.16.10.2	172.16.10.1	HTTP	362	GET /soap-bubble-1958650_960_720.jpg HTTP/1.1
148	2.741205777	172.16.10.2	172.16.10.1	HTTP	349	GET /download%20(2).jpg HTTP/1.1
179	2.742807473	172.16.10.1	172.16.10.2	HTTP	113	HTTP/1.1 200 OK (JPEG JFIF image)
190	2.743723330	172.16.10.1	172.16.10.2	HTTP	712	HTTP/1.1 200 OK (JPEG JFIF image)
402	2.764054977	172.16.10.1	172.16.10.2	HTTP	72	HTTP/1.1 200 OK (JPEG JFIF image)
3774	3.042252027	172.16.10.1	172.16.10.2	HTTP	1124	HTTP/1.1 200 OK (JPEG JFIF image)

Here it is 3.042252027 - 2.685888334 = 0.356363

Step 3: For 4 persistent connections, Set the value of **max-persistent-connection-per-server** **to** **4** in the client computer.

Step 4: Repeat the **steps 1-3** in the previous section.

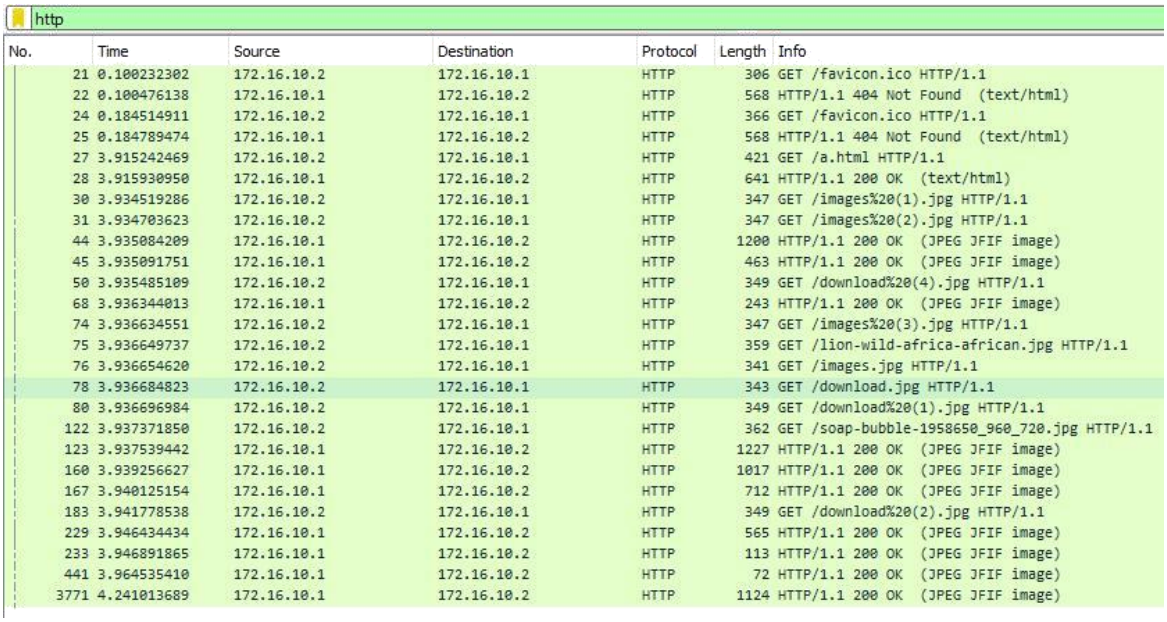
http

No.	Time	Source	Destination	Protocol	Length	Info
28	0.152642908	172.16.10.1	172.16.10.2	HTTP	568	HTTP/1.1 404 Not Found (text/html)
30	1.667969551	172.16.10.2	172.16.10.1	HTTP	421	GET /a.html HTTP/1.1
31	1.668311781	172.16.10.1	172.16.10.2	HTTP	641	HTTP/1.1 200 OK (text/html)
33	1.699473631	172.16.10.2	172.16.10.1	HTTP	347	GET /images%20(1).jpg HTTP/1.1
35	1.699692009	172.16.10.2	172.16.10.1	HTTP	347	GET /images%20(2).jpg HTTP/1.1
45	1.699908042	172.16.10.1	172.16.10.2	HTTP	463	HTTP/1.1 200 OK (JPEG JFIF image)
46	1.699913003	172.16.10.1	172.16.10.2	HTTP	1200	HTTP/1.1 200 OK (JPEG JFIF image)
47	1.700012712	172.16.10.2	172.16.10.1	HTTP	349	GET /download%20(4).jpg HTTP/1.1
63	1.700901747	172.16.10.1	172.16.10.2	HTTP	242	HTTP/1.1 200 OK (JPEG JFIF image)
69	1.701341018	172.16.10.2	172.16.10.1	HTTP	347	GET /images%20(3).jpg HTTP/1.1
70	1.701432635	172.16.10.2	172.16.10.1	HTTP	359	GET /lion-wild-africa-african.jpg HTTP/1.1
86	1.701888908	172.16.10.1	172.16.10.2	HTTP	565	HTTP/1.1 200 OK (JPEG JFIF image)
93	1.702192885	172.16.10.2	172.16.10.1	HTTP	341	GET /images.jpg HTTP/1.1
95	1.702219175	172.16.10.2	172.16.10.1	HTTP	343	GET /download.jpg HTTP/1.1
97	1.702228220	172.16.10.2	172.16.10.1	HTTP	349	GET /download%20(1).jpg HTTP/1.1
98	1.702233130	172.16.10.2	172.16.10.1	HTTP	362	GET /soap-bubble-1958650_960_720.jpg HTTP/1.1
122	1.703328136	172.16.10.1	172.16.10.2	HTTP	711	HTTP/1.1 200 OK (JPEG JFIF image)
126	1.703773424	172.16.10.2	172.16.10.1	HTTP	349	GET /download%20(2).jpg HTTP/1.1
157	1.705498971	172.16.10.1	172.16.10.2	HTTP	1227	HTTP/1.1 200 OK (JPEG JFIF image)
159	1.705614894	172.16.10.1	172.16.10.2	HTTP	113	HTTP/1.1 200 OK (JPEG JFIF image)
167	1.706637782	172.16.10.1	172.16.10.2	HTTP	1017	HTTP/1.1 200 OK (JPEG JFIF image)
414	1.724541388	172.16.10.1	172.16.10.2	HTTP	73	HTTP/1.1 200 OK (JPEG JFIF image)
3825	2.005934395	172.16.10.1	172.16.10.2	HTTP	1124	HTTP/1.1 200 OK (JPEG JFIF image)

Here is it 2.005934395 - 1.667969557 = 0.337964838

Step 5: For 6 persistent connections, set the value of **max-persistent-connection-per-server** **to** **6** in the server computer.

Step 6: Repeat the **steps 1-3** in the previous section.



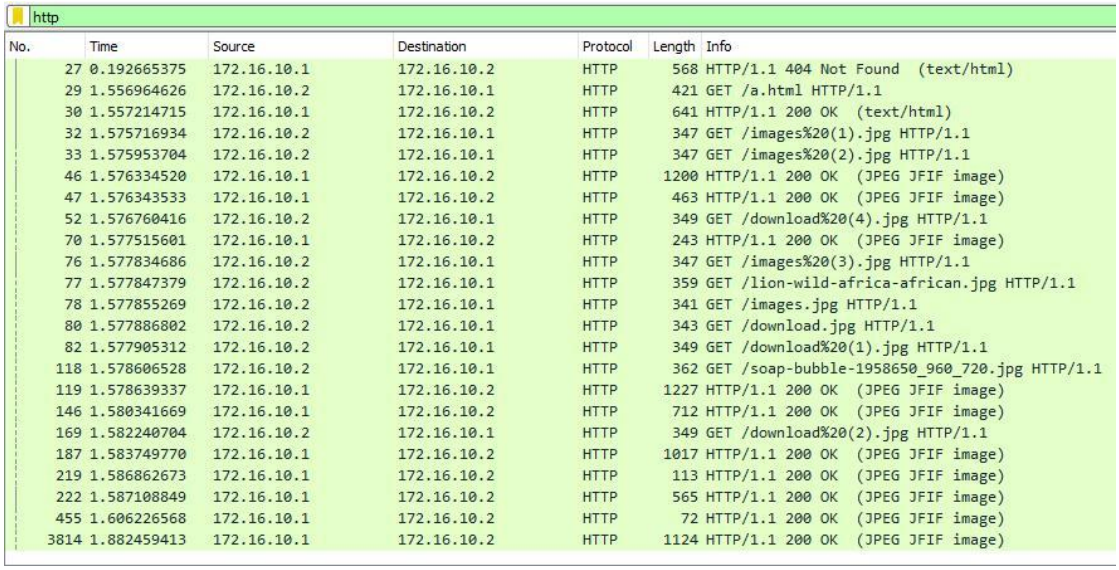
http

No.	Time	Source	Destination	Protocol	Length	Info
21	0.100232302	172.16.10.2	172.16.10.1	HTTP	306	GET /favicon.ico HTTP/1.1
22	0.100476138	172.16.10.1	172.16.10.2	HTTP	568	HTTP/1.1 404 Not Found (text/html)
24	0.184514911	172.16.10.2	172.16.10.1	HTTP	366	GET /favicon.ico HTTP/1.1
25	0.184789474	172.16.10.1	172.16.10.2	HTTP	568	HTTP/1.1 404 Not Found (text/html)
27	3.915242469	172.16.10.2	172.16.10.1	HTTP	421	GET /a.html HTTP/1.1
28	3.915930950	172.16.10.1	172.16.10.2	HTTP	641	HTTP/1.1 200 OK (text/html)
30	3.934519286	172.16.10.2	172.16.10.1	HTTP	347	GET /images%20(1).jpg HTTP/1.1
31	3.934703623	172.16.10.2	172.16.10.1	HTTP	347	GET /images%20(2).jpg HTTP/1.1
44	3.935084209	172.16.10.1	172.16.10.2	HTTP	1200	HTTP/1.1 200 OK (JPEG JFIF image)
45	3.935091751	172.16.10.1	172.16.10.2	HTTP	463	HTTP/1.1 200 OK (JPEG JFIF image)
50	3.935485109	172.16.10.2	172.16.10.1	HTTP	349	GET /download%20(4).jpg HTTP/1.1
68	3.936344013	172.16.10.1	172.16.10.2	HTTP	243	HTTP/1.1 200 OK (JPEG JFIF image)
74	3.936634551	172.16.10.2	172.16.10.1	HTTP	347	GET /images%20(3).jpg HTTP/1.1
75	3.936649737	172.16.10.2	172.16.10.1	HTTP	359	GET /lion-wild-africa-african.jpg HTTP/1.1
76	3.936654620	172.16.10.2	172.16.10.1	HTTP	341	GET /images.jpg HTTP/1.1
78	3.936684823	172.16.10.2	172.16.10.1	HTTP	343	GET /download.jpg HTTP/1.1
80	3.936696984	172.16.10.2	172.16.10.1	HTTP	349	GET /download%20(1).jpg HTTP/1.1
122	3.937371850	172.16.10.2	172.16.10.1	HTTP	362	GET /soap-bubble-1958650_960_720.jpg HTTP/1.1
123	3.937539442	172.16.10.1	172.16.10.2	HTTP	1227	HTTP/1.1 200 OK (JPEG JFIF image)
160	3.939256627	172.16.10.1	172.16.10.2	HTTP	1017	HTTP/1.1 200 OK (JPEG JFIF image)
167	3.940125154	172.16.10.1	172.16.10.2	HTTP	712	HTTP/1.1 200 OK (JPEG JFIF image)
183	3.941778538	172.16.10.2	172.16.10.1	HTTP	349	GET /download%20(2).jpg HTTP/1.1
229	3.946434434	172.16.10.1	172.16.10.2	HTTP	565	HTTP/1.1 200 OK (JPEG JFIF image)
233	3.946891865	172.16.10.1	172.16.10.2	HTTP	113	HTTP/1.1 200 OK (JPEG JFIF image)
441	3.964535410	172.16.10.1	172.16.10.2	HTTP	72	HTTP/1.1 200 OK (JPEG JFIF image)
3771	4.241013689	172.16.10.1	172.16.10.2	HTTP	1124	HTTP/1.1 200 OK (JPEG JFIF image)

Here it is 4.241013689 - 3.915242469 = 0.325771229

Step 7: For 10 persistent connections, set the value of **max-persistent-connection-per-server** **to** **10** in the client computer.

Step 8: Repeat the **steps 1-3** in the previous section.



http

No.	Time	Source	Destination	Protocol	Length	Info
27	0.192665375	172.16.10.1	172.16.10.2	HTTP	568	HTTP/1.1 404 Not Found (text/html)
29	1.556964626	172.16.10.2	172.16.10.1	HTTP	421	GET /a.html HTTP/1.1
30	1.557214715	172.16.10.1	172.16.10.2	HTTP	641	HTTP/1.1 200 OK (text/html)
32	1.575716934	172.16.10.2	172.16.10.1	HTTP	347	GET /images%20(1).jpg HTTP/1.1
33	1.575953704	172.16.10.2	172.16.10.1	HTTP	347	GET /images%20(2).jpg HTTP/1.1
46	1.576334520	172.16.10.1	172.16.10.2	HTTP	1200	HTTP/1.1 200 OK (JPEG JFIF image)
47	1.576343533	172.16.10.1	172.16.10.2	HTTP	463	HTTP/1.1 200 OK (JPEG JFIF image)
52	1.576760416	172.16.10.2	172.16.10.1	HTTP	349	GET /download%20(4).jpg HTTP/1.1
70	1.577515601	172.16.10.1	172.16.10.2	HTTP	243	HTTP/1.1 200 OK (JPEG JFIF image)
76	1.577834686	172.16.10.2	172.16.10.1	HTTP	347	GET /images%20(3).jpg HTTP/1.1
77	1.577847379	172.16.10.2	172.16.10.1	HTTP	359	GET /lion-wild-africa-african.jpg HTTP/1.1
78	1.577855269	172.16.10.2	172.16.10.1	HTTP	341	GET /images.jpg HTTP/1.1
80	1.577886802	172.16.10.2	172.16.10.1	HTTP	343	GET /download.jpg HTTP/1.1
82	1.577905312	172.16.10.2	172.16.10.1	HTTP	349	GET /download%20(1).jpg HTTP/1.1
118	1.578606528	172.16.10.2	172.16.10.1	HTTP	362	GET /soap-bubble-1958650_960_720.jpg HTTP/1.1
119	1.578639337	172.16.10.1	172.16.10.2	HTTP	1227	HTTP/1.1 200 OK (JPEG JFIF image)
146	1.580341669	172.16.10.1	172.16.10.2	HTTP	712	HTTP/1.1 200 OK (JPEG JFIF image)
169	1.582240704	172.16.10.2	172.16.10.1	HTTP	349	GET /download%20(2).jpg HTTP/1.1
187	1.583749770	172.16.10.1	172.16.10.2	HTTP	1017	HTTP/1.1 200 OK (JPEG JFIF image)
219	1.586862673	172.16.10.1	172.16.10.2	HTTP	113	HTTP/1.1 200 OK (JPEG JFIF image)
222	1.587108849	172.16.10.1	172.16.10.2	HTTP	565	HTTP/1.1 200 OK (JPEG JFIF image)
455	1.606226568	172.16.10.1	172.16.10.2	HTTP	72	HTTP/1.1 200 OK (JPEG JFIF image)
3814	1.882459413	172.16.10.1	172.16.10.2	HTTP	1124	HTTP/1.1 200 OK (JPEG JFIF image)

Here it is 1.882459413-1.556964626=0.325494787

OBSERVATIONS REQUIRED ON EDMODO:

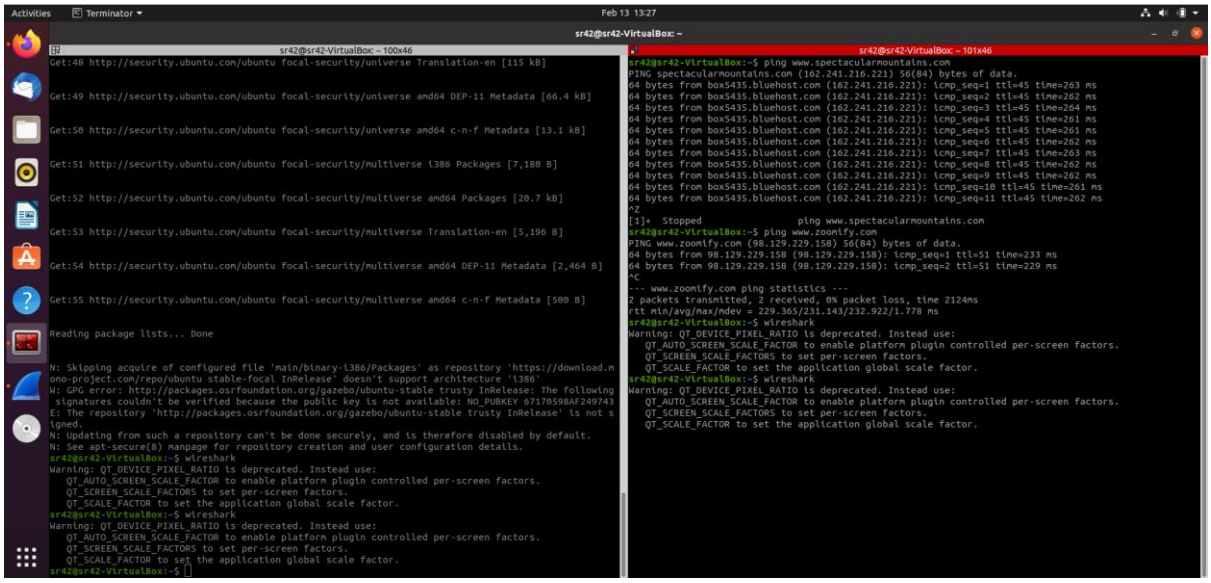
Calculate the time taken to load objects from the server for non-persistent and persistent connections (2, 4, 6, 8 & 10). Find out the optimal number of HTTP persistent connections based on your observations.

SCREENSHOTS REQUIRED FOR EDMODO:

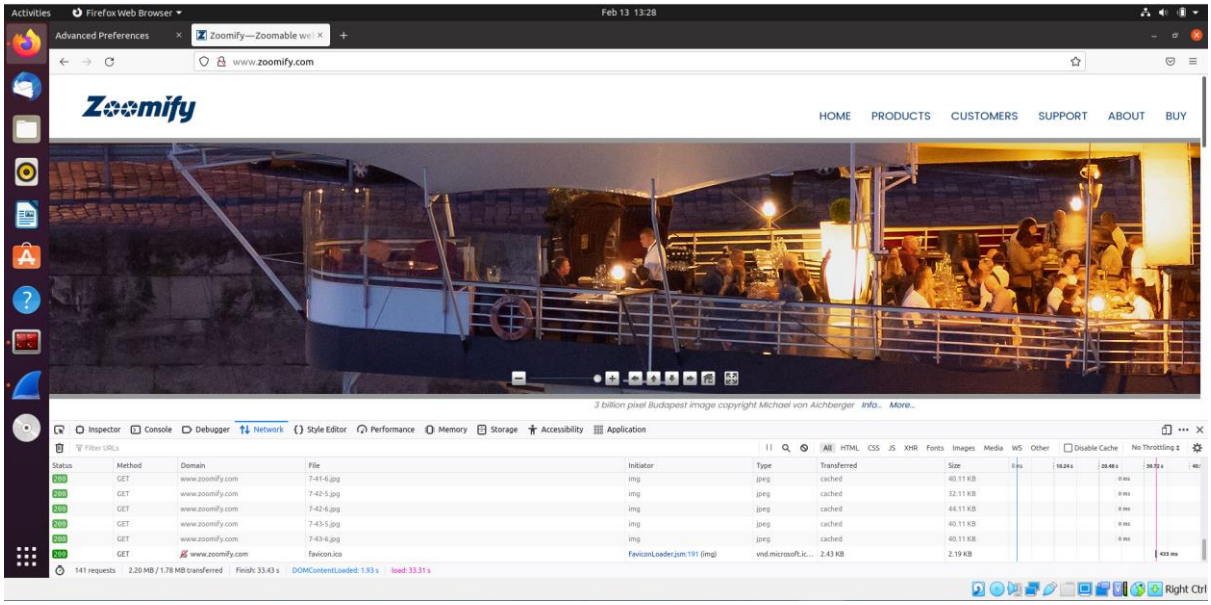
1) Non-persistent connection wireshark capture (should include all 10 images)
2) Persistent connections wireshark capture – 2, 4, 6, 8 & 10 respectively (should include all 10 images).

Initialization :

-

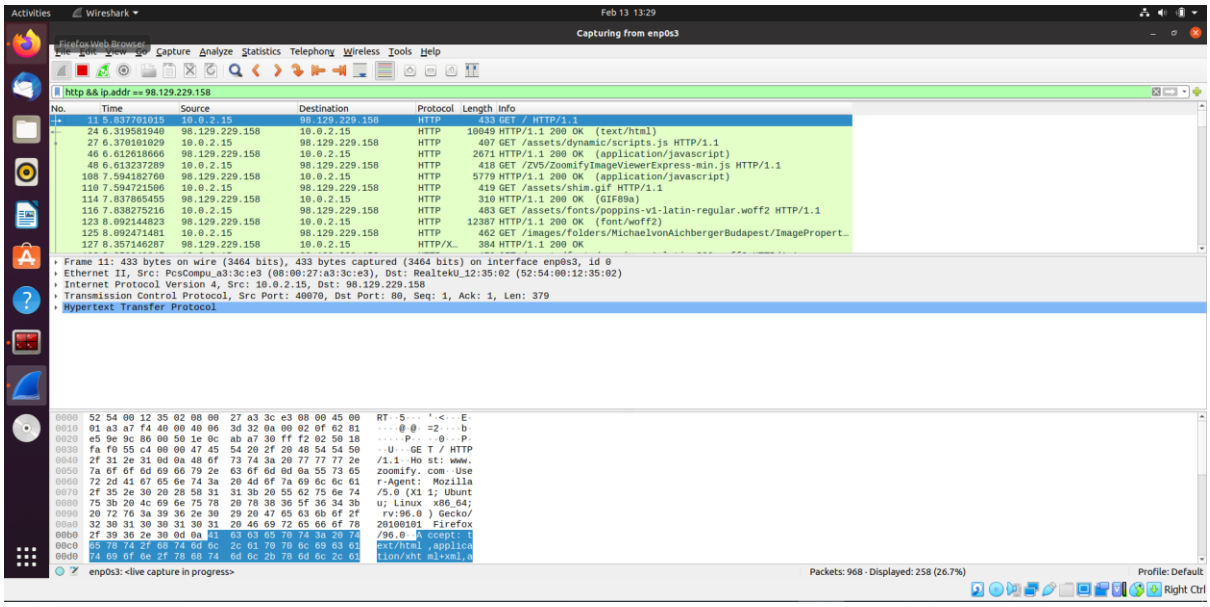


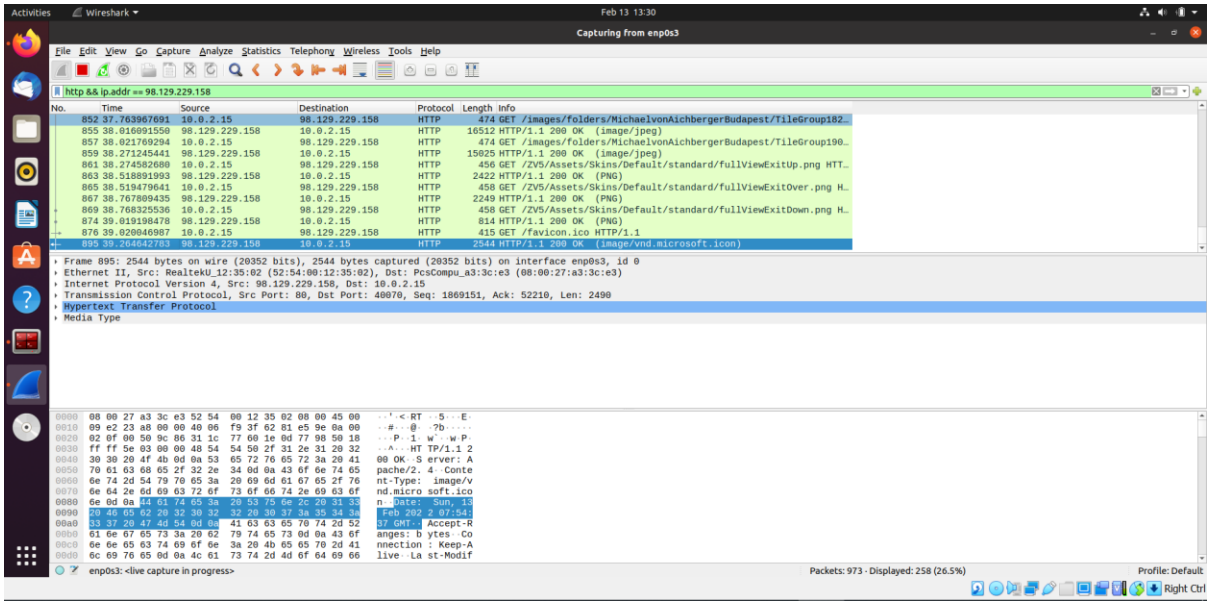
-



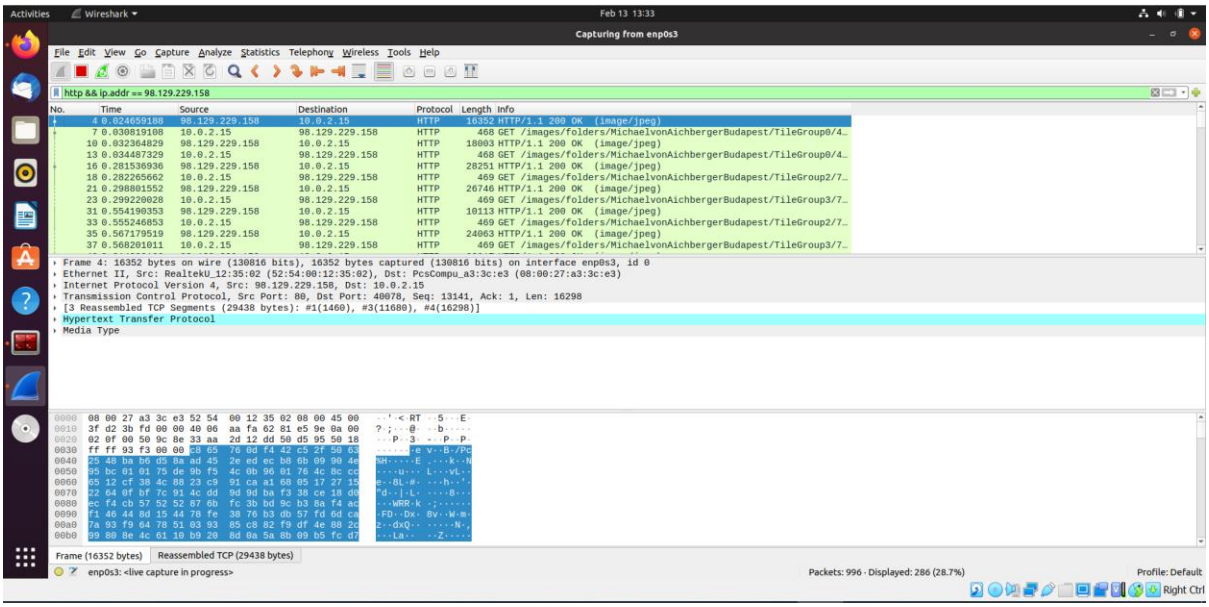
Observation notes :

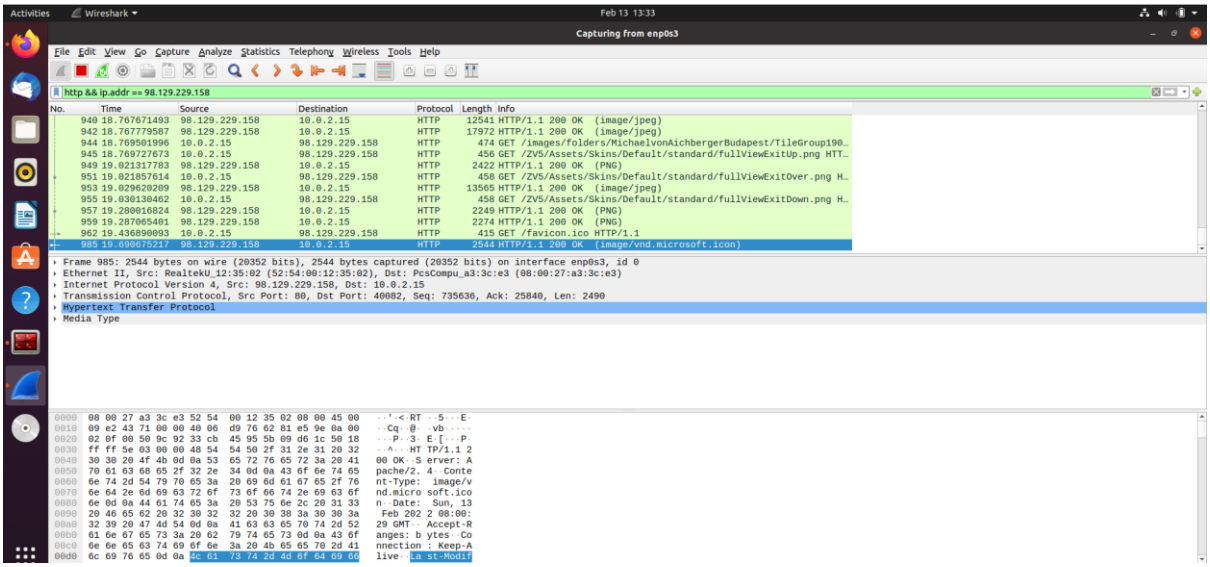
- Non persistent connection time for connection = last HTTP response time - first GET request transmission time = 39.26 - 5.84 = 33.42s



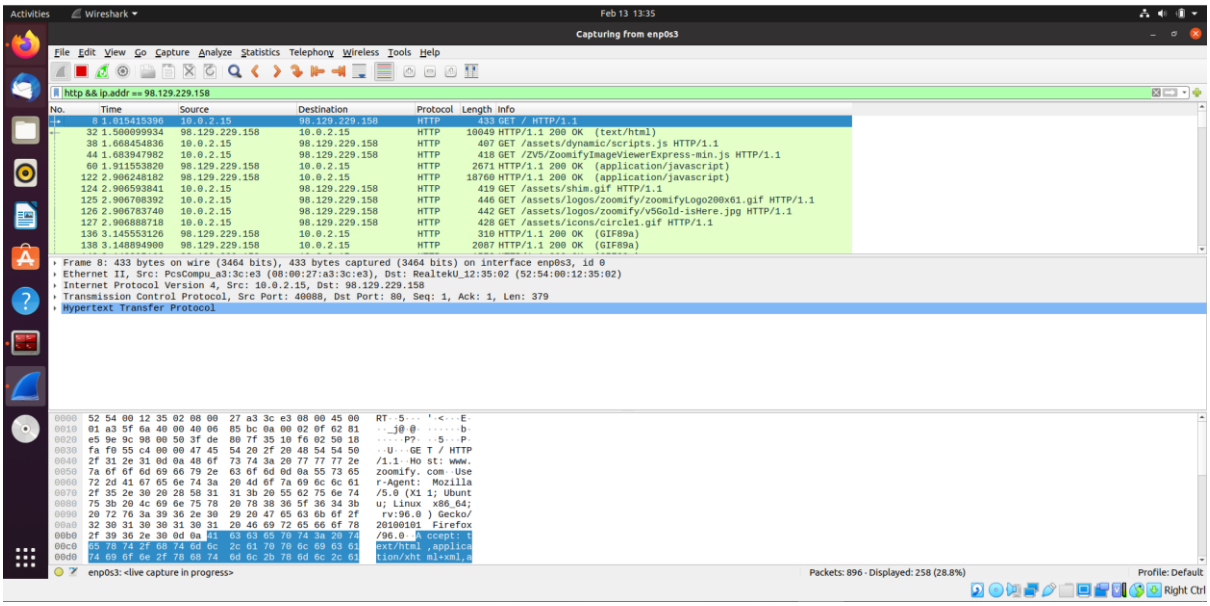


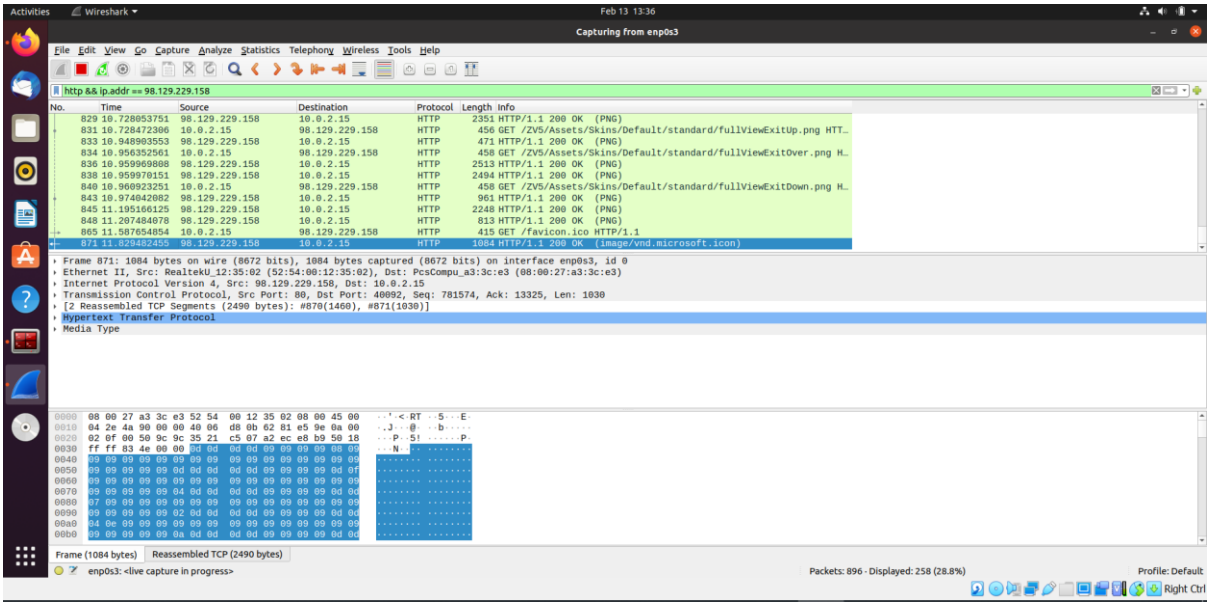
- Persistent connection time (2 connections) for connection = last HTTP response time - first GET request transmission time = 19.69 – 0.02 = 19.67s



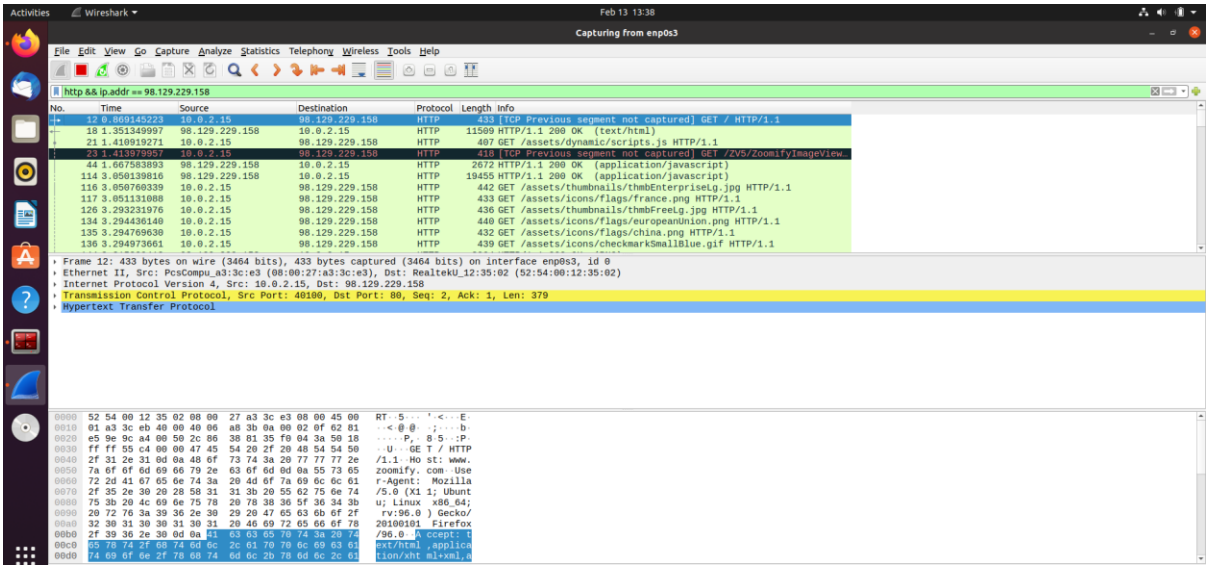


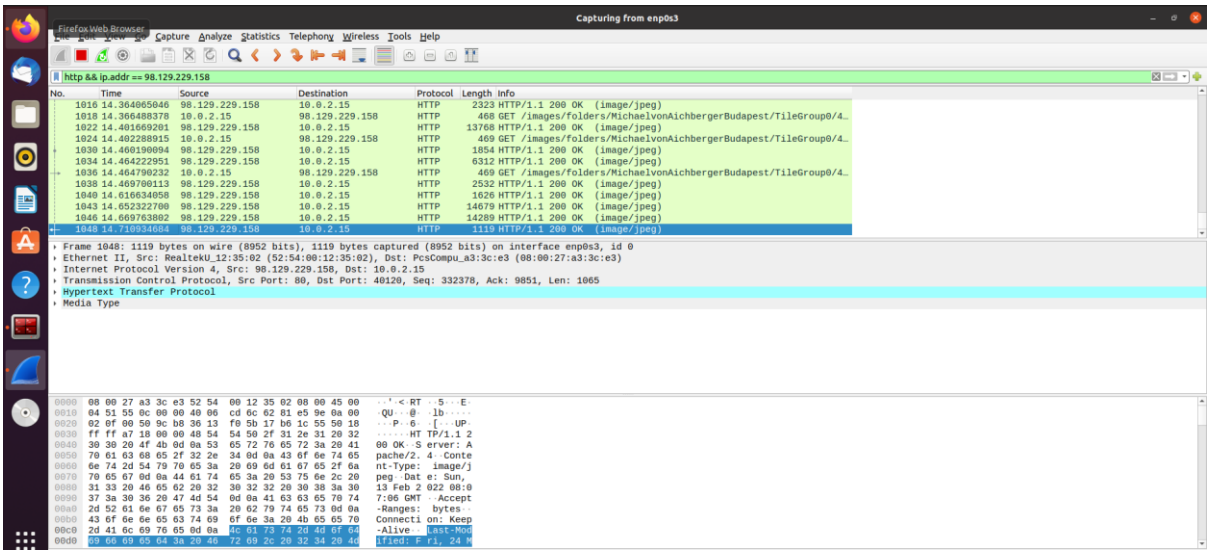
- Persistent connection time (4 connections) for connection = last HTTP response time - first GET request transmission time = 11.83 – 1.01 = 10.82s



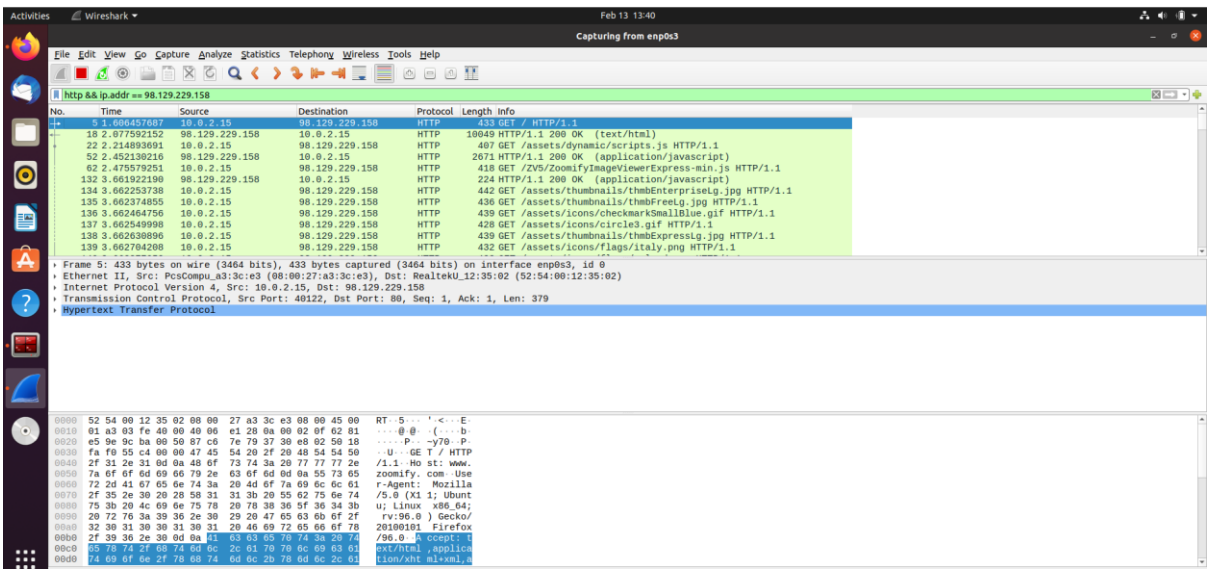


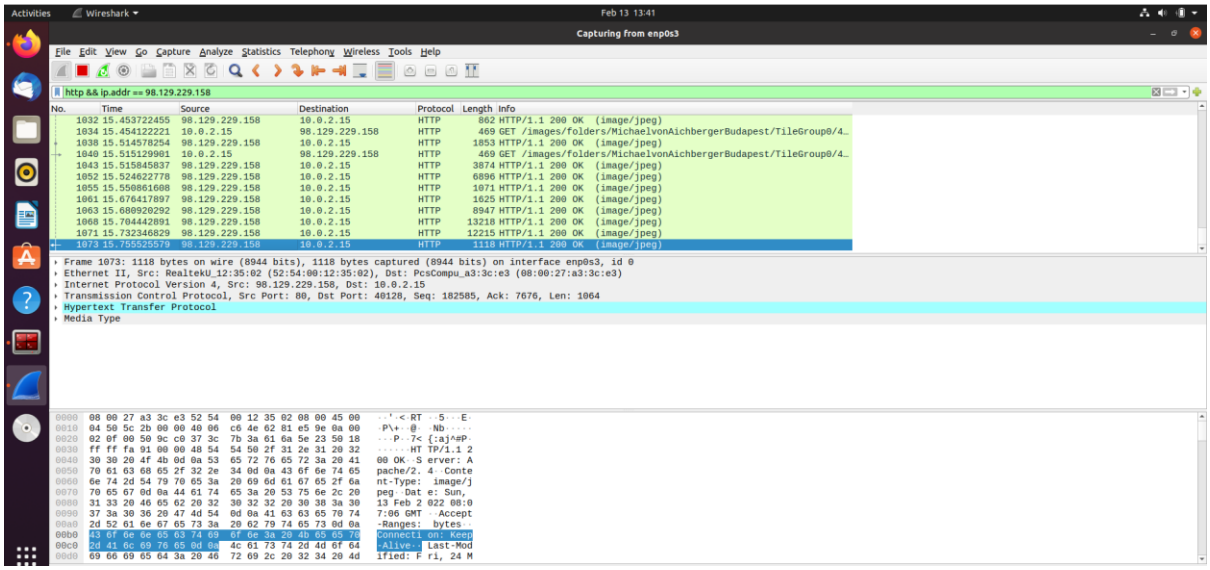
- Persistent connection time (6 connections) for connection = last HTTP response time - first GET request transmission time = 14.71 – 0.87 = 13.84s



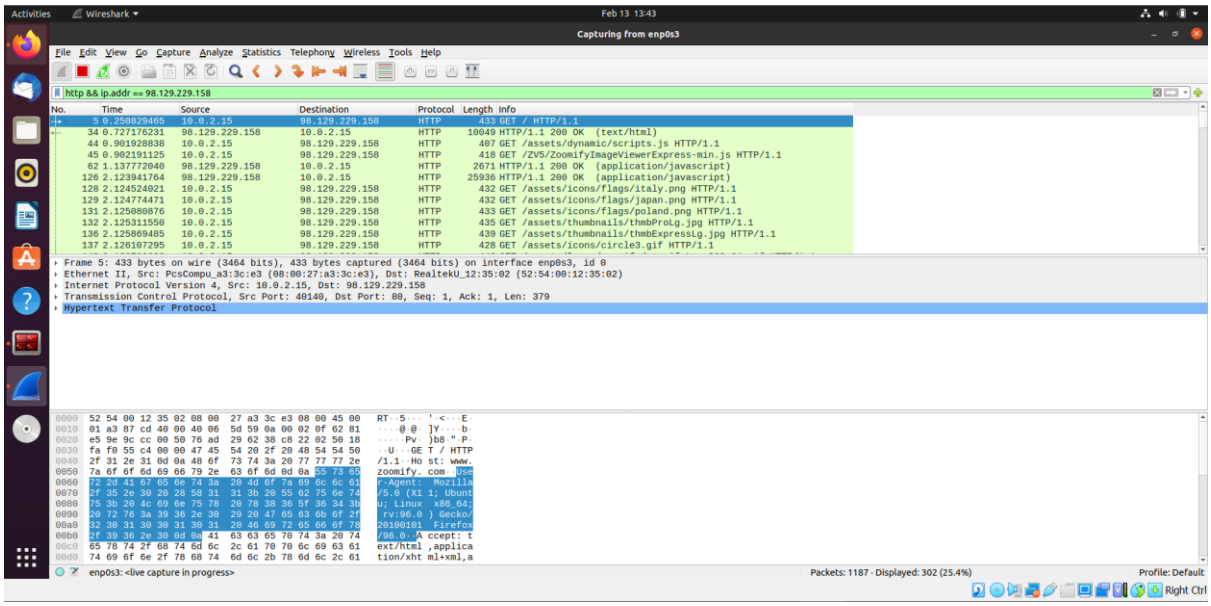


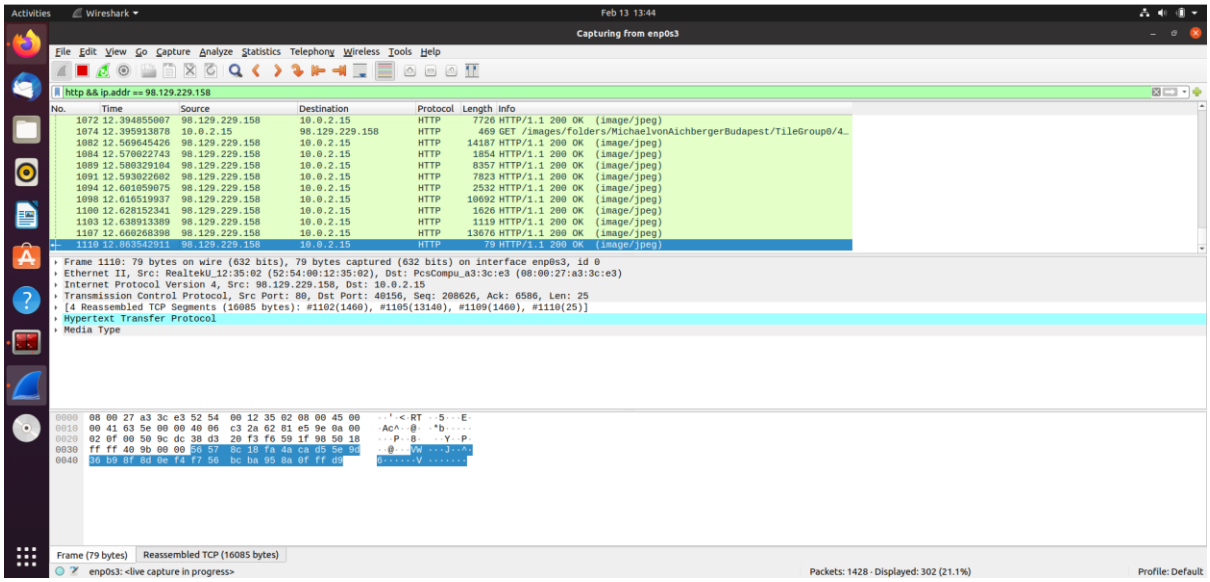
- Persistent connection time (8 connections) for connection = last HTTP response time - first GET request transmission time = 15.75 – 1.61 = 14.14s





- Persistent connection time (10 connections) for connection = last HTTP response time - first GET request transmission time = 12.86 – 0.25 = 12.41s





Conclusion :

Therefore, the best load time for this particular site was obtained with 4 persistent connections.

2. Understand working of HTTP Headers

	Understand working of HTTP headers: Conditional Get: If-Modified-Since HTTP Cookies: Cookie and Set-Cookie Authentication: Auth-Basic Design a web page that has one embedded page (e.g. image) and sets a cookie and enables authentication. You are required to configure the web server (e.g. apache) with authentication mechanism. Show the behavior of conditional get when embedded objects is modified and when it is not (you can just change the create date of the embedded object). Decode the Basic-Auth header using Base64 mechanism as per the password setup. **Observation**: Show the behavior of browser when is cookie is set and when cookie is removed.

Lab: 3

Understanding Working of HTTP Headers

Question: Understand working of HTTP headers

Conditional Get: If-Modified-Since

HTTP Cookies: Cookie and Set-Cookie

Authentication: Auth-Basic

Design a web page that has one embedded page (e.g. image) and sets a cookie and enables authentication. You are required to configure the web server (e.g. apache) with authentication mechanism. Show the behavior of conditional get when embedded objects are modified and when it is not (you can just change the create date of the embedded object). Decode the Basic-Auth header using Base64 mechanism as per the password setup.

Observation: Show the behavior of browser when is cookie is set and when cookie is removed.

Solution: Analyzing Basic Authentication and Cookies

The three parts of experiment are:

1. Password Authentication
2. Cookie Setting
3. Conditional get

Steps of Execution (for Password Authentication)

1. Executing the below commands on the terminal.

--> To update and integrate the existing softwares **sudo apt-get update**

--> To install the apache utility

sudo apt-get install apache2 apache2-utils

```
osboxes@osboxes: ~
osboxes@osboxes:~$ sudo apt-get install apache2-utils
[sudo] password for osboxes:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages will be upgraded:
  apache2-utils
1 upgraded, 0 newly installed, 0 to remove and 247 not upgraded.
Need to get 81.7 kB of archives.
After this operation, 0 B of additional disk space will be used.
Get:1 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 apache2-utils amd64 2.4.18-2ub
untu3.10 [81.7 kB]
Fetched 81.7 kB in 1s (45.5 kB/s)
(Reading database ... 217540 files and directories currently installed.)
Preparing to unpack .../apache2-utils_2.4.18-2ubuntu3.10_amd64.deb ...
Unpacking apache2-utils (2.4.18-2ubuntu3.10) over (2.4.18-2ubuntu3.9) ...
Processing triggers for man-db (2.7.5-1) ...
Setting up apache2-utils (2.4.18-2ubuntu3.10) ...
osboxes@osboxes:~$
```

--> Provide username and password to set authentication

sudo htpasswd -c /etc/apache2/.htpasswd ANY_USERNAME

```
osboxes@osboxes:~$ sudo htpasswd -c /etc/apache2/.htpasswd netwo
New password:
Re-type new password:
Adding password for user netwo
osboxes@osboxes:~$ sudo cat /etc/apache2/.htpasswd
netwo:$apr1$6YdDa0Ti$ELrUaOlQ/jun9TTU1PYKu/
osboxes@osboxes:~$
```

Here “netwo” is the username. Also, password is entered twice.

--> View the authentication

sudo cat /etc/apache2/.htpasswd

2. To setup the authentication phase, execute the following commands. Configuring Access control within the Virtual Host Definition.

--> Opening the file for setting authentication

sudo nano /etc/apache2/sites-available/000-default.conf

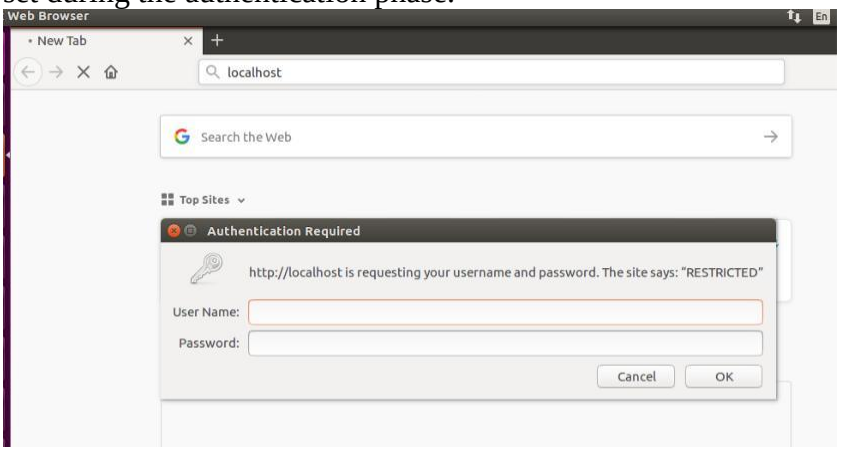
```
<VirtualHost*:80>
        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/html
        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log
        combined <Directory "/var/www/html">
                AuthType Basic
                AuthName "RESTRICTED"
                AuthUserFile /etc/apache2/.htpasswd
                Require valid-user
        </Directory>
</VirtualHost>
```

```
 GNU nano 2.5.3        File: /etc/apache2/sites-available/000-default.conf

<VirtualHost *:80>


        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/html

        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined

                <Directory "/var/www/html">
                        AuthType Basic
                        AuthName "RESTRICTED"
                        AuthUserFile /etc/apache2/.htpasswd
                        Require valid-user >
        </Directory>

</VirtualHost>
```

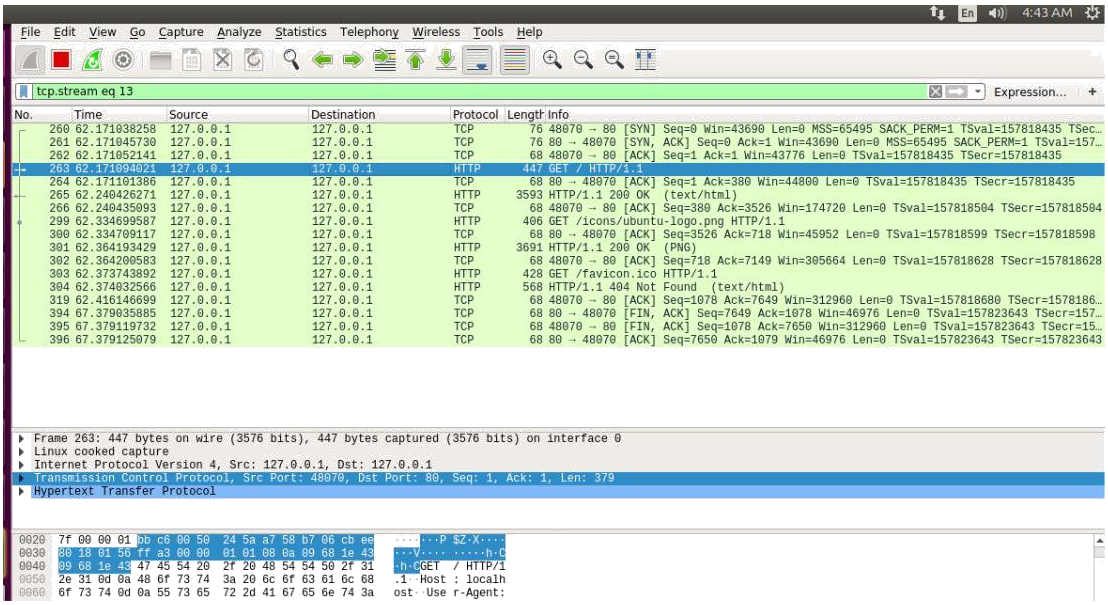
3. Password policy implementation is done by restarting the server as: **sudo service apache2 restart**

```
osboxes@osboxes:~$ sudo service apache2 restart
osboxes@osboxes:~$
```

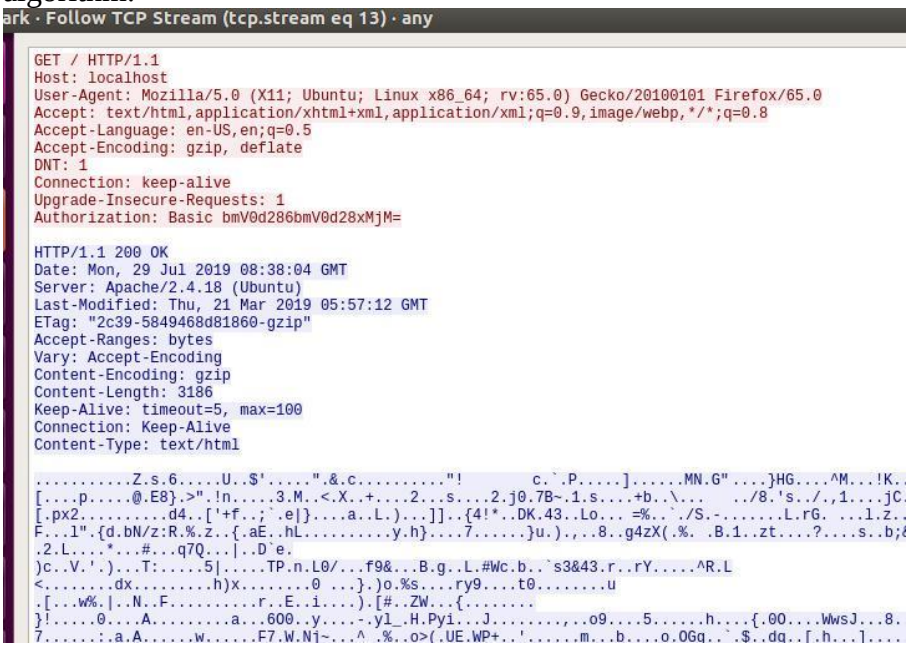
4. The localhost is then accessed using the Firefox browser requiring a username and a password set during the authentication phase.



5. Wireshark is used to capture the packets sent upon the network.



6. Using the “follow TCP stream” on the HTTP message segment the password was retrieved which was encrypted by the base64 algorithm and decryption could be done with same algorithm.



Steps of Execution (Cookie Setting)

1. A PHP file to set the cookie is created which also contains an image in it (placed under the HTML directory) to be accessed once the cookie is set. The following code helped to set the cookie:

<html>
<?php

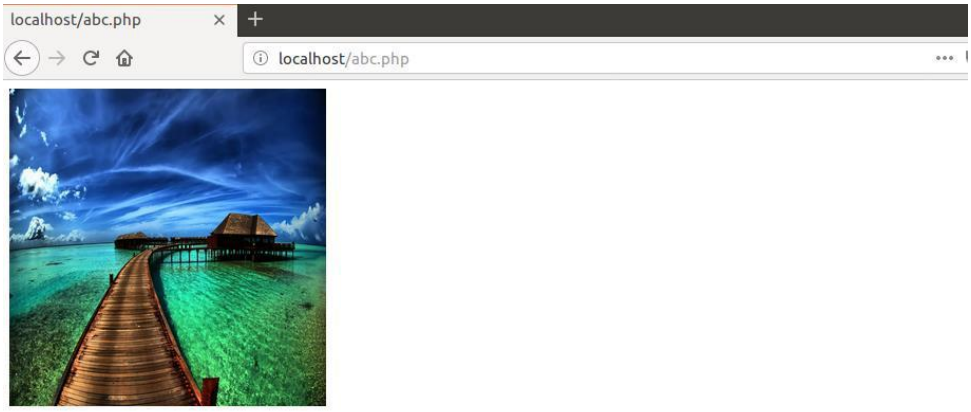
```
        setcookie("namecookie","netqwerty",time()+123);
        setcookie("nickname","work");
?>
<img src= “highres.png” width= “300” height= “300” title= “password” />
</html>
```

```
  GNU nano 2.5.3                File: abc.php

<html>
<?php
setcookie("namecookie","netqwerty",time()+123);
setcookie("nickname","work");
?>
<img src="highres.jpg" width="300" height="300" title="password"/>
</html>
```

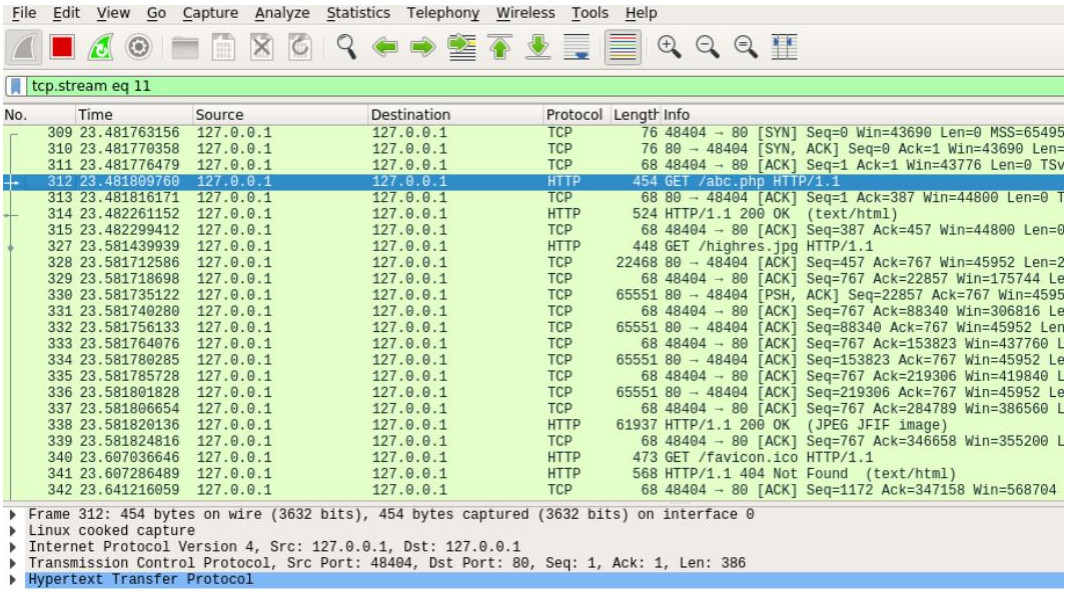
Note: Here you can add any image if required

Note: You can capture Cookies mostly during the first time of web access. Hence keep wireshark capture ready before executing the task for the first time.

2. The combined file saved with a .php extension is placed under **/var/www/html** for accessing.



3. The packets are captured using Wireshark and using the “follow TCP stream” which checks for the set-cookie field whether the cookie is set or not set.



The cookie is set as shown in the above screenshot.

Observation: Understand and work out base 64 algorithm and write in your observation. Observe various parameters associated with Cookie in the wireshark capture.

Conditional Get: If-Modified-Since

Before performing the steps below, make sure your browser's cache is empty. (To do this under Firefox, select Tools -> Clear Recent History and check the Cache box). Now do the following:

- Start up your web browser, and make sure your browser's cache is cleared, as discussed above.
- Start up the Wireshark packet sniffer.
- Enter the following URL into your browser
  http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file2.html
- Your browser should display a very simple five-line HTML file.
- Quickly enter the same URL into your browser again (or simply select the refresh button on your browser)
- Stop Wireshark packet capture, and enter "http" in the display-filter-specification window, so that only captured HTTP messages will be displayed later in the packet-listing window.

Observations:

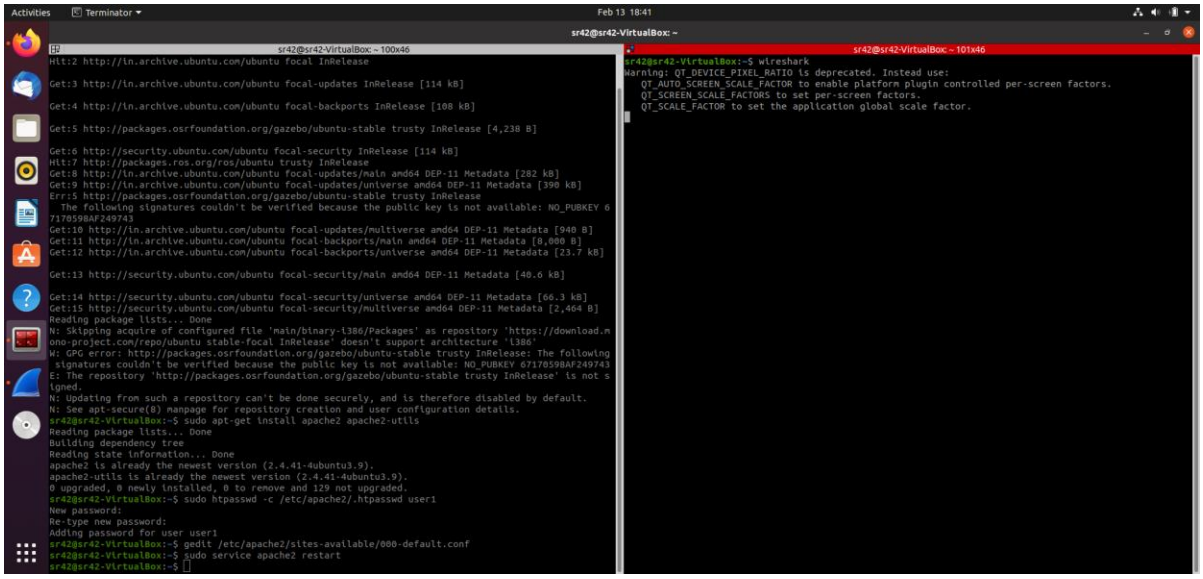
- ✓ Inspect the contents of the first HTTP GET request from your browser to the server. Do you see an "IF-MODIFIED-SINCE" line in the HTTP GET?
- ✓ Inspect the contents of the server response. Did the server explicitly return the contents of the file? How can you tell?
- ✓ Now inspect the contents of 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?
- ✓ 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.

Repeat the above task with some images on the server.

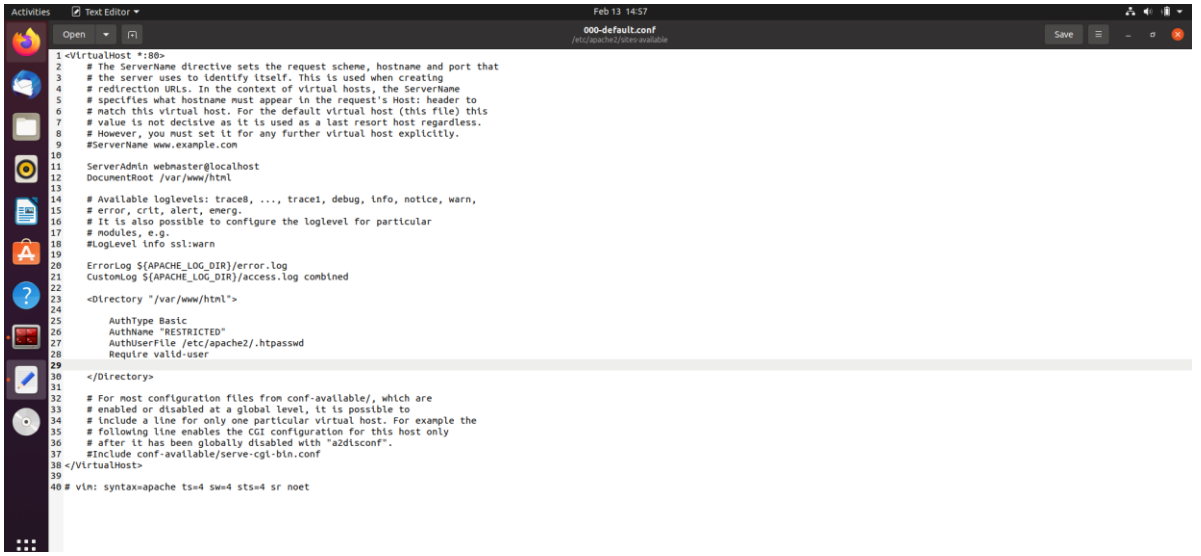
Attach screenshots wherever necessary.

Screenshots & observations :

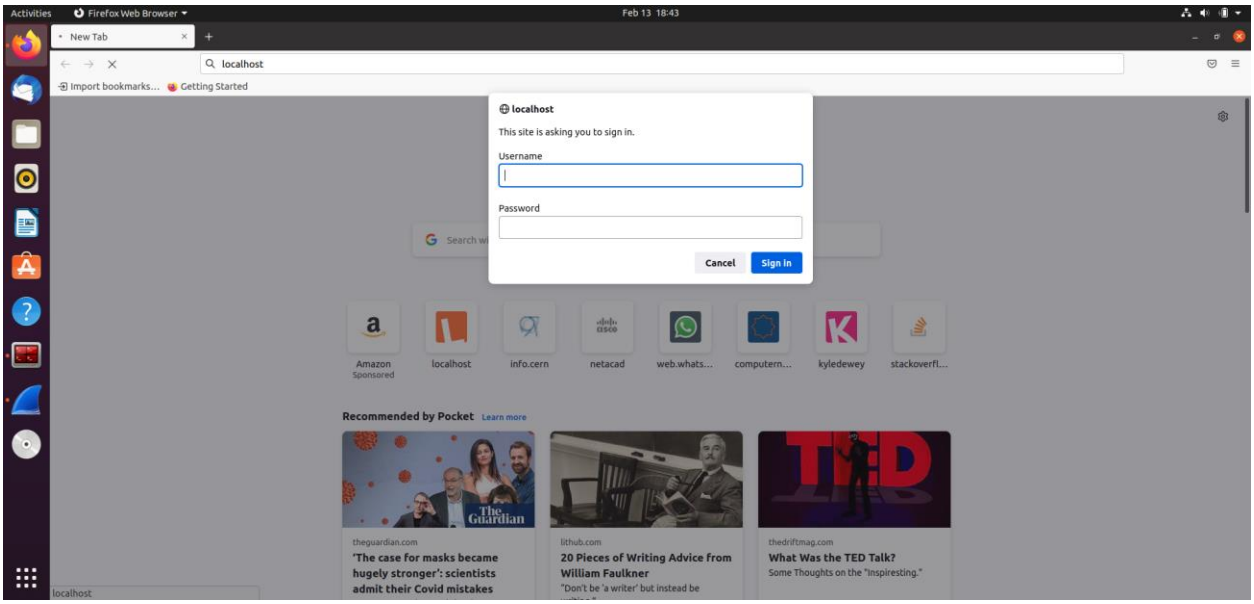
- Password authentication :
    - Terminal initialization :



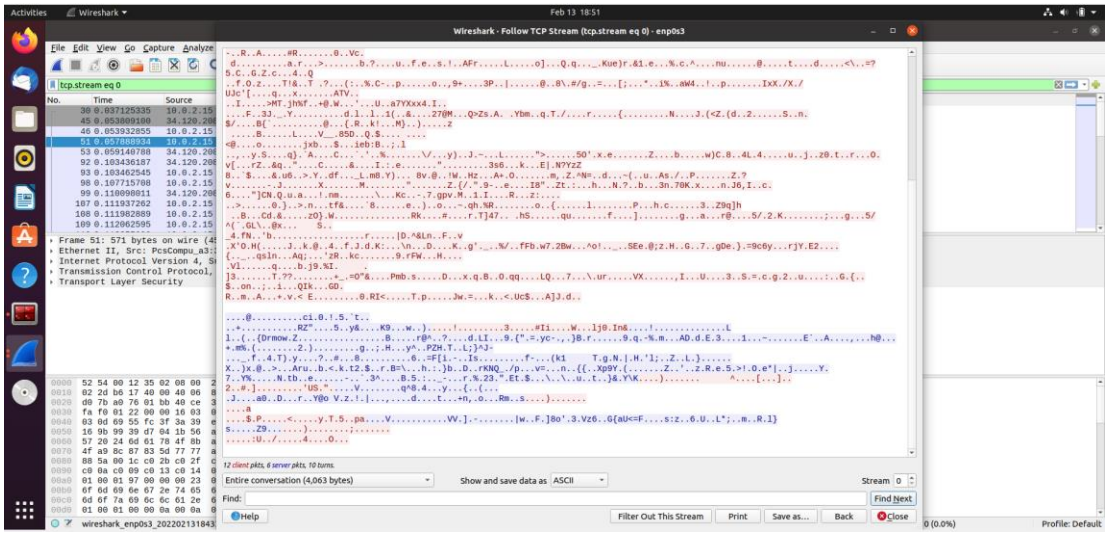
    - config file :



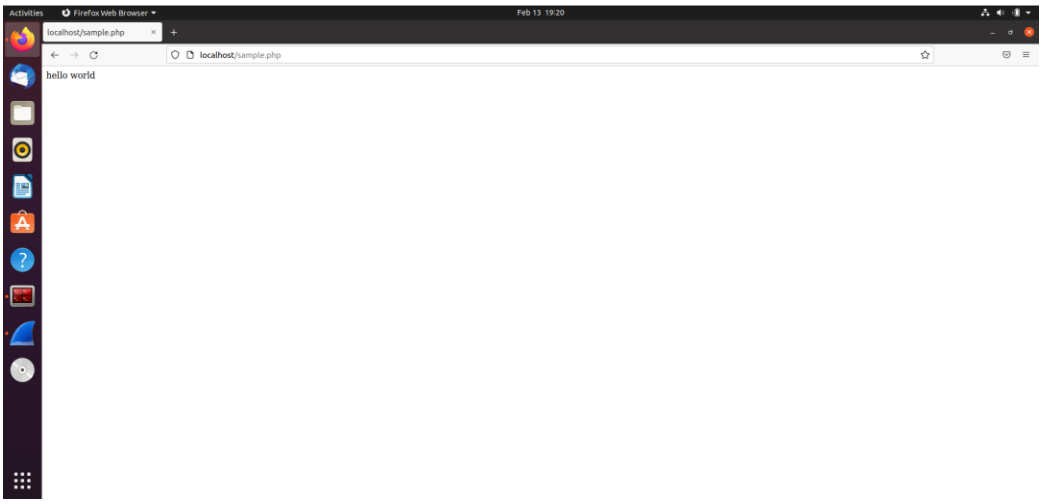
    - Login page :



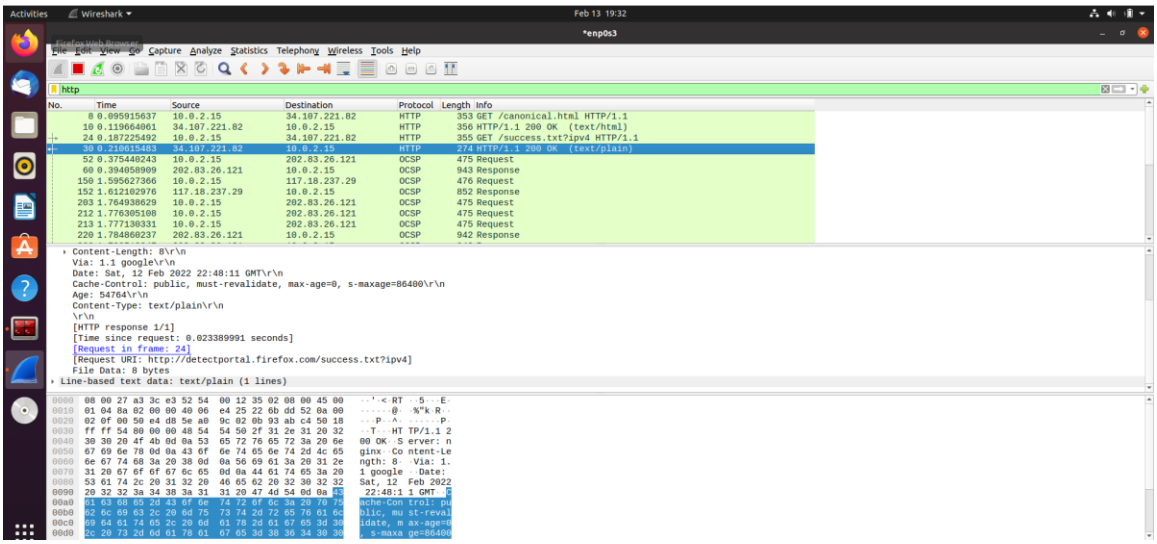
- Wireshark :



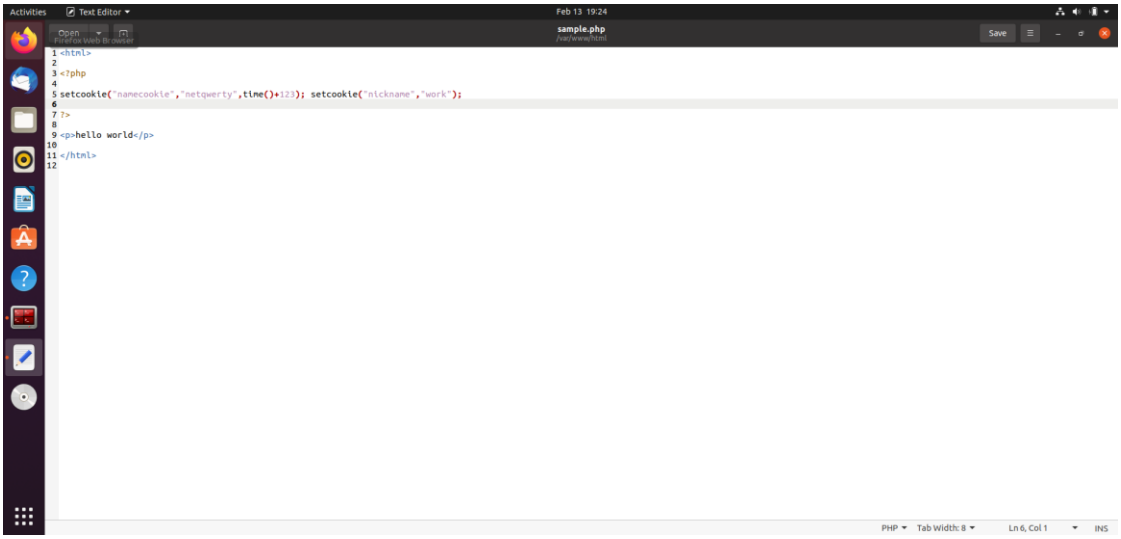
- Cookie setting :
  - Output :



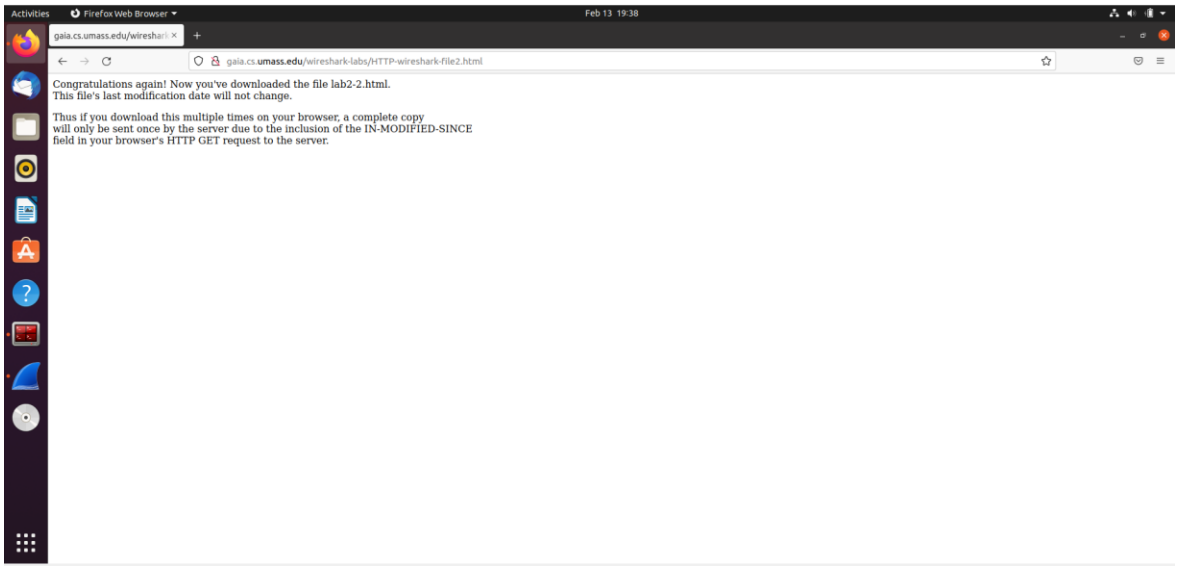
- Wireshark :



- sample.php :



- Conditional get :
    - Output :



    - Wireshark :
        - If-modified-since is a visible field in the get request
        - The server explicitly returned the contents of the file. This was visible by following the TCP stream.
        - The if-modified-since field contains the text ‘Sun, 13 Feb 2022 06:59:02 GMT’
        - HTTP status code in the response is 304 and the response phrase says ‘Not Modified’. The server did not explicitly return the contents of the file in the second response.
        -