NAME: VISHWAS M

SRN: PES2UG20CS390

SEC: F

DATE: 17/02/2022

SUB: COMPUTER NETWORKS LAB

Week #4

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)

1. Non-persistent connection
2. 2 persistent connections
3. 4 persistent connections
4. 6 persistent connections
5. 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 Client



172.16.10.2/24

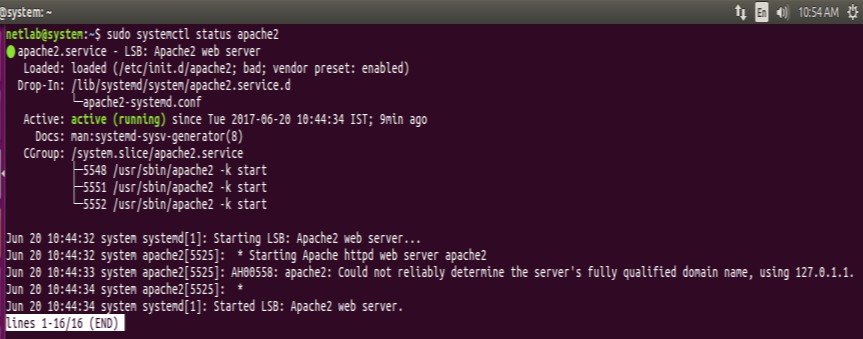
**Server Side:**

172.16.10.1/24

**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**

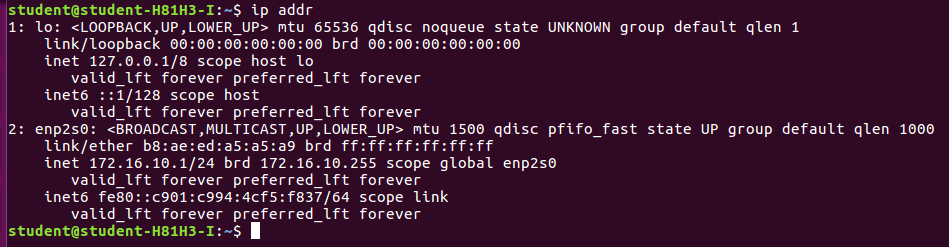
As you can see above, the service appears to have started successfully. However, the best way to test this is to actually request a page from Apache. You can access the default Apache landing page to confirm that the software is running properly. You can access this through your server's domain name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

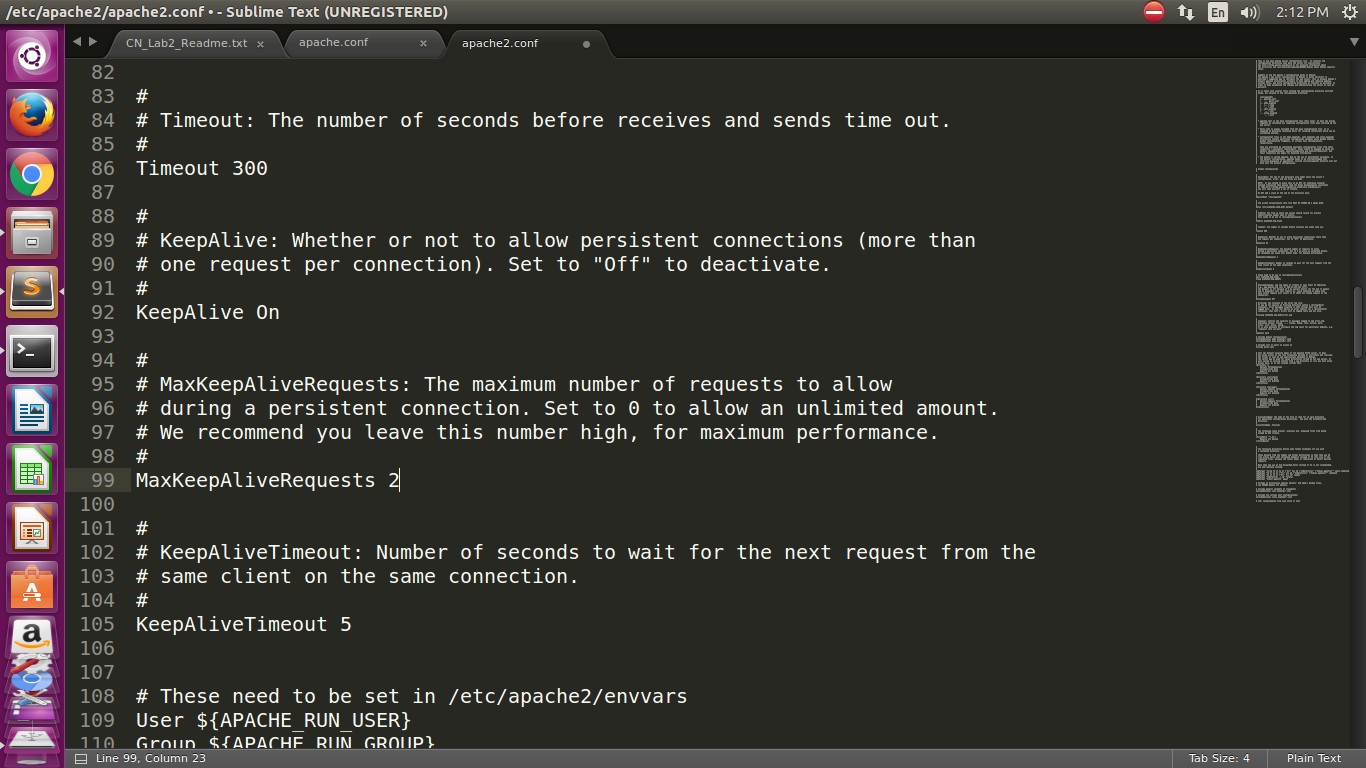
Note: If IP address fluctuates, kindly setup the IP address manually using ‘Edit connections’.



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

1. The **keep-alive** option was set (i.e. value was made **ON**)
2. The **MaximumKeepAliveRequests** were set to **2**

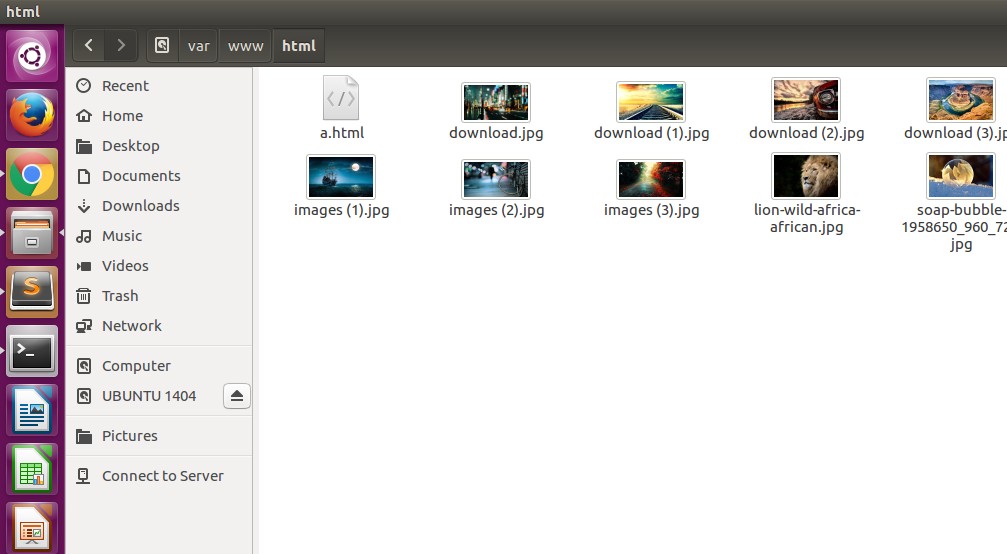
# $sudo nano /etc/apache2/apache2.conf



**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)



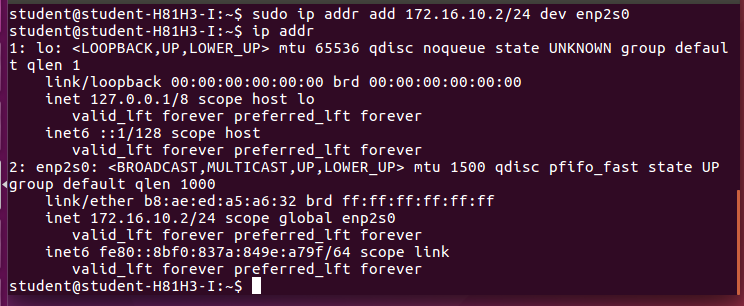
**Client side:**

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’.



There are broadly two parts of execution:

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

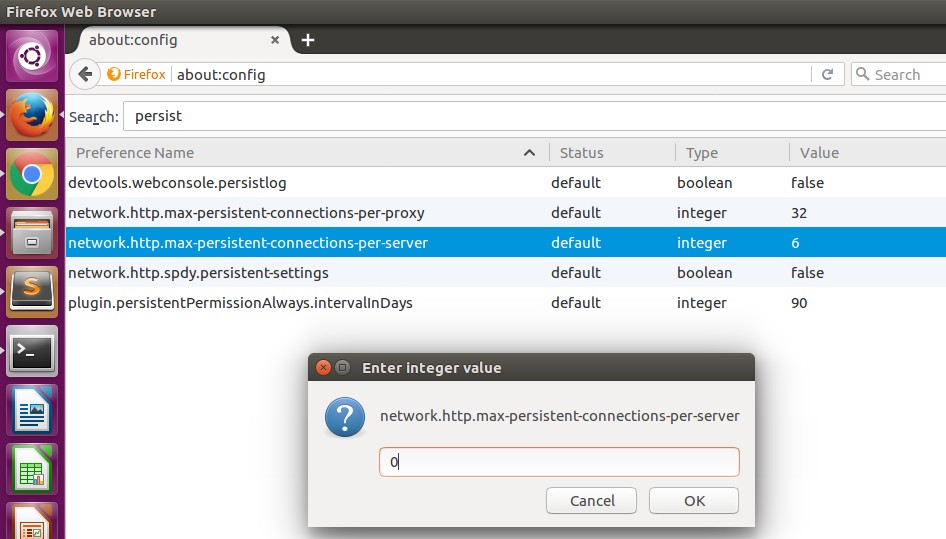
Open Firefox browser to configure for persistent option. Go to browser and type

**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:

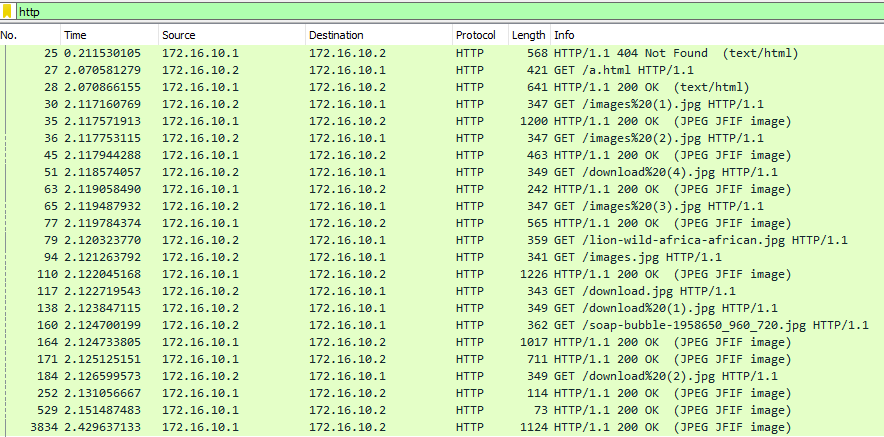
* + - 1. **/file\_name.html** where**--> 172.16.10.1** is Server’s IP

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.

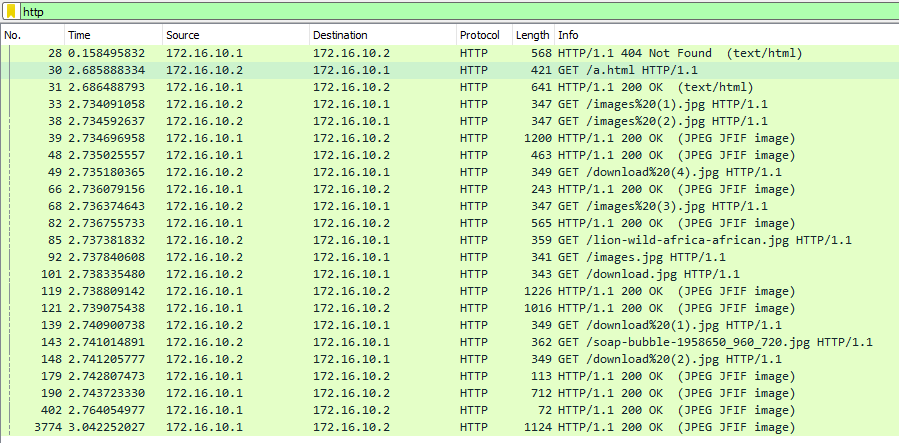


Here it is 2.429637133 - 2.070581279 = 0.359055854

# 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.

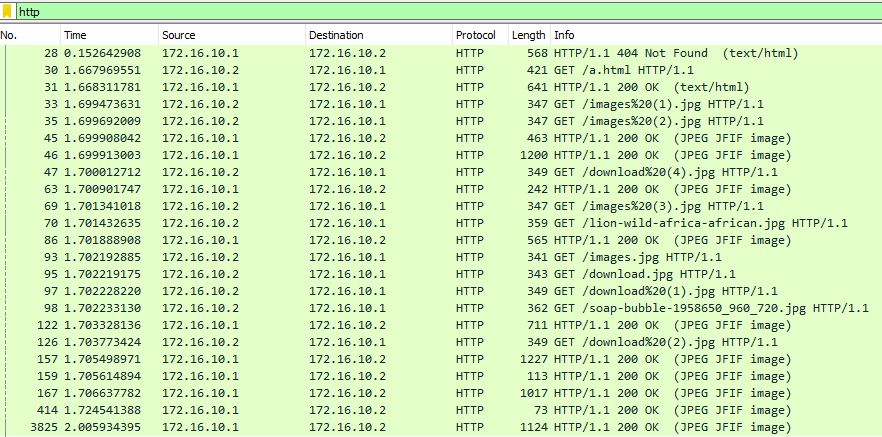
Step 2: Repeat the **steps 1-3** in the previous section.



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.

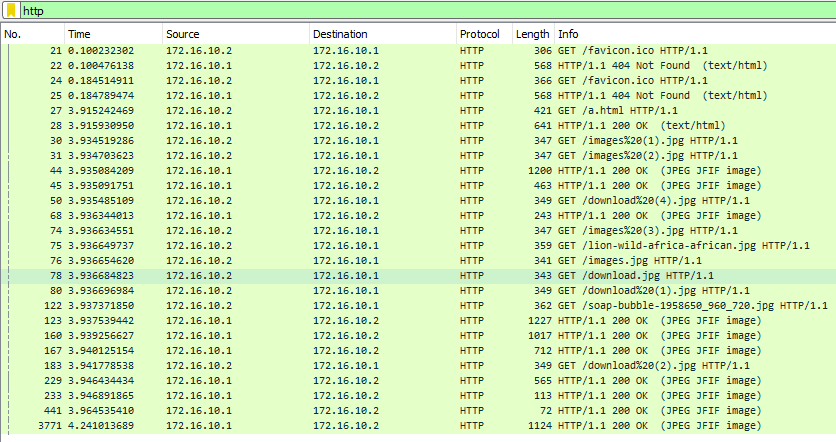
Step 4: Repeat the **steps 1-3** in the previous section.



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.

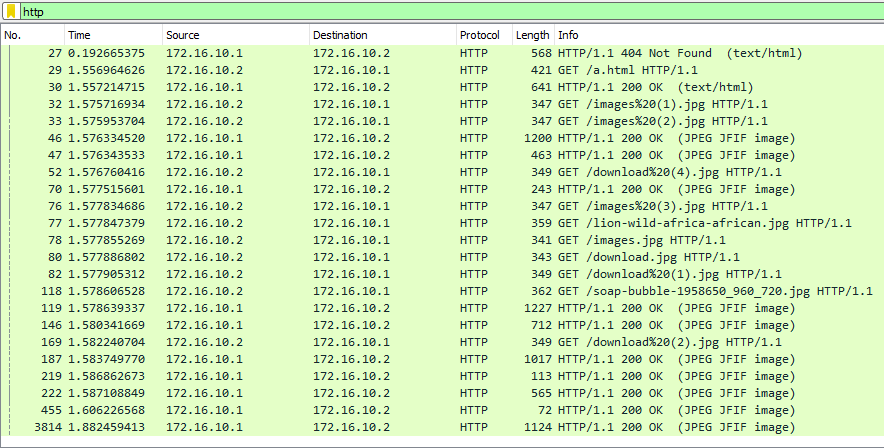
Step 6: Repeat the **steps 1-3** in the previous section.



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.



Here it is 1.882459413-1.556964626=0.325494787

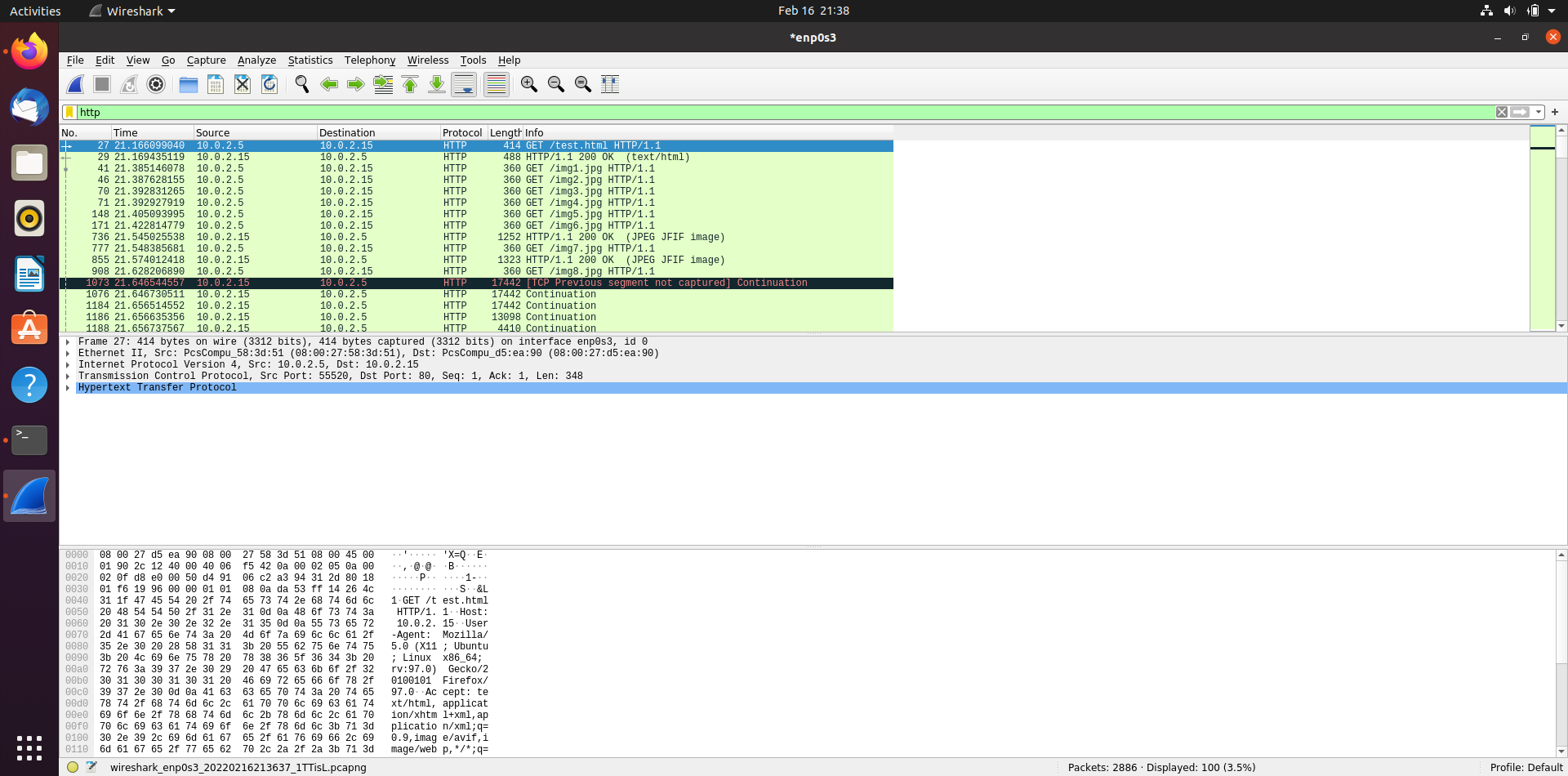
# OBSERVATIONS REQUIRED ON EDMODO:

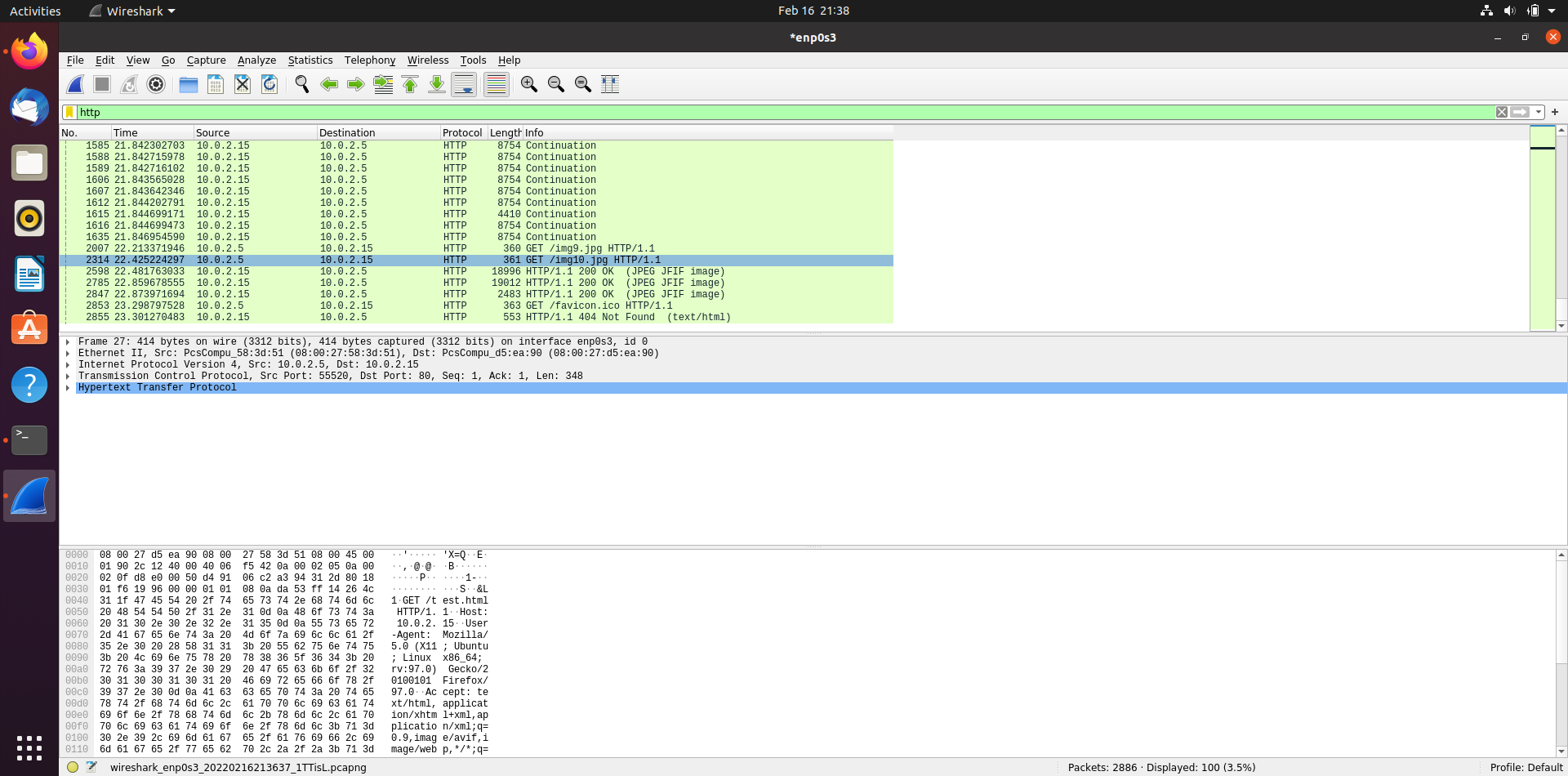
Find out the time taken to load images for 2 4 6 persistent connections is lesser or greater than 10 persistent compared to non-persistent. Why? Find out the optimal persistent connections.

# SCREENSHOTS REQUIRED FOR EDMODO:

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

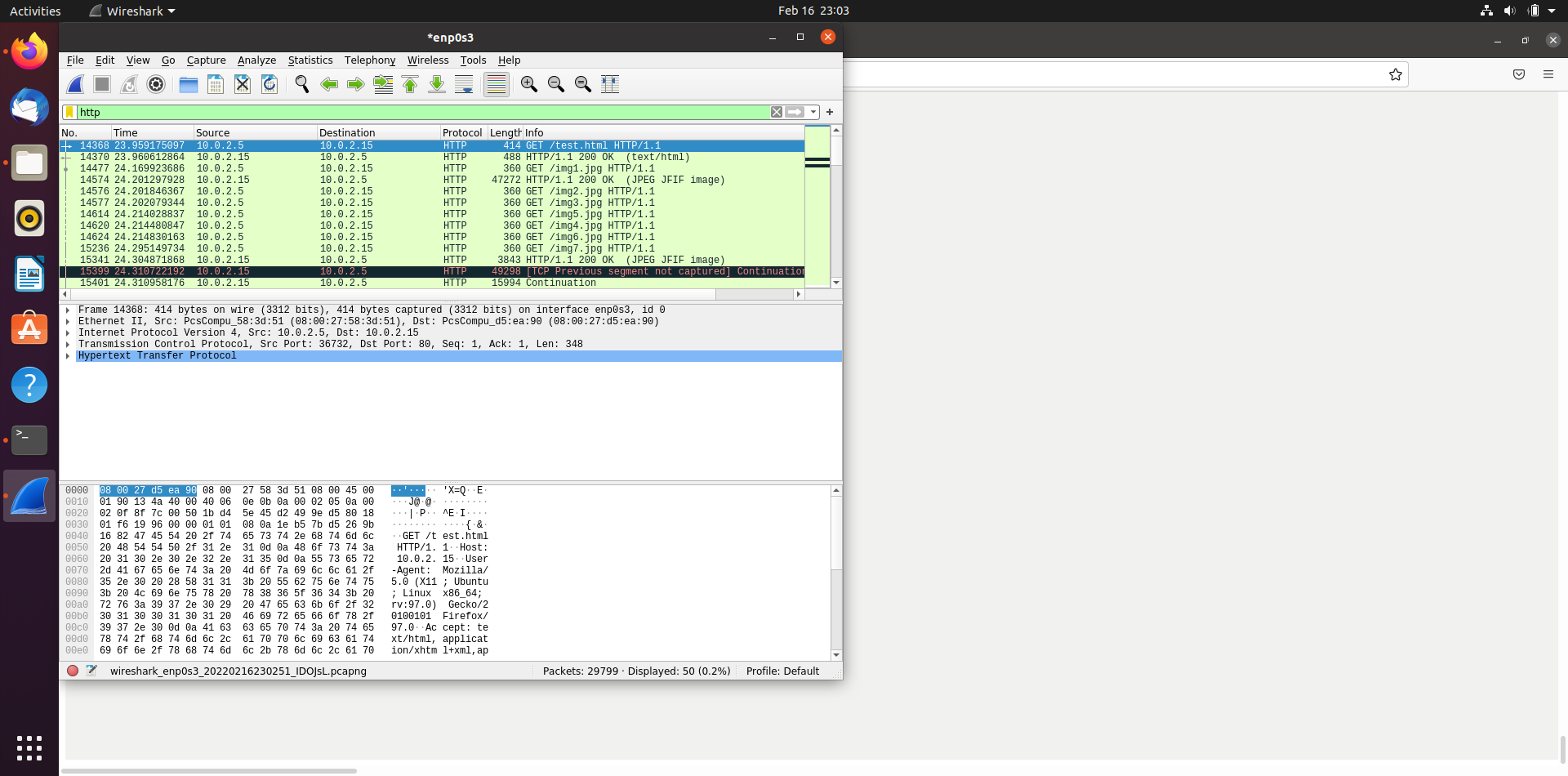
**SCREENSHOTS:**

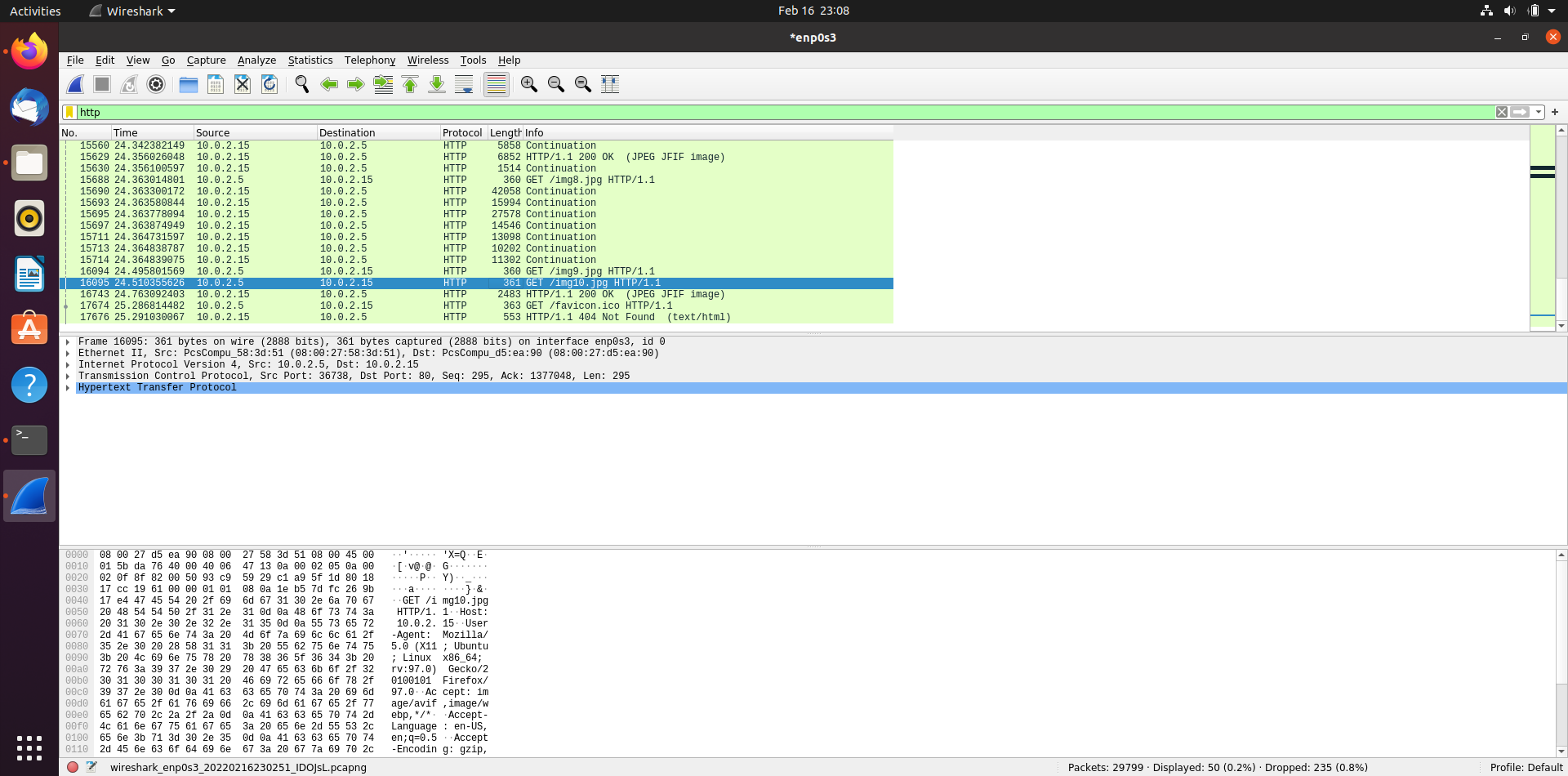
**1)Non-Persistent** :



Here it is 22.425224297-21.385146078=1.040078219

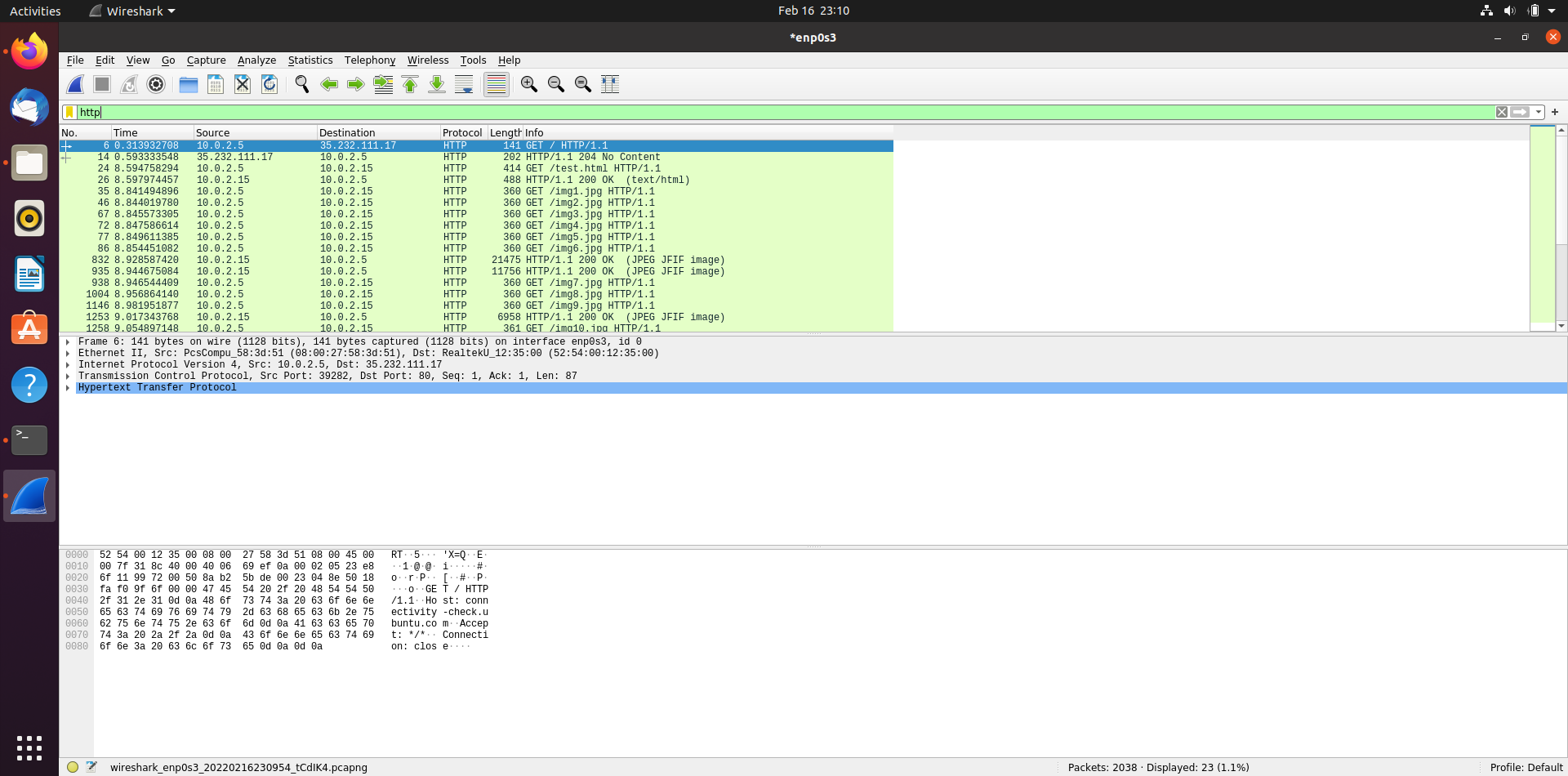
**2)Persistent Connection:**

**a) 2-persistent connection:**



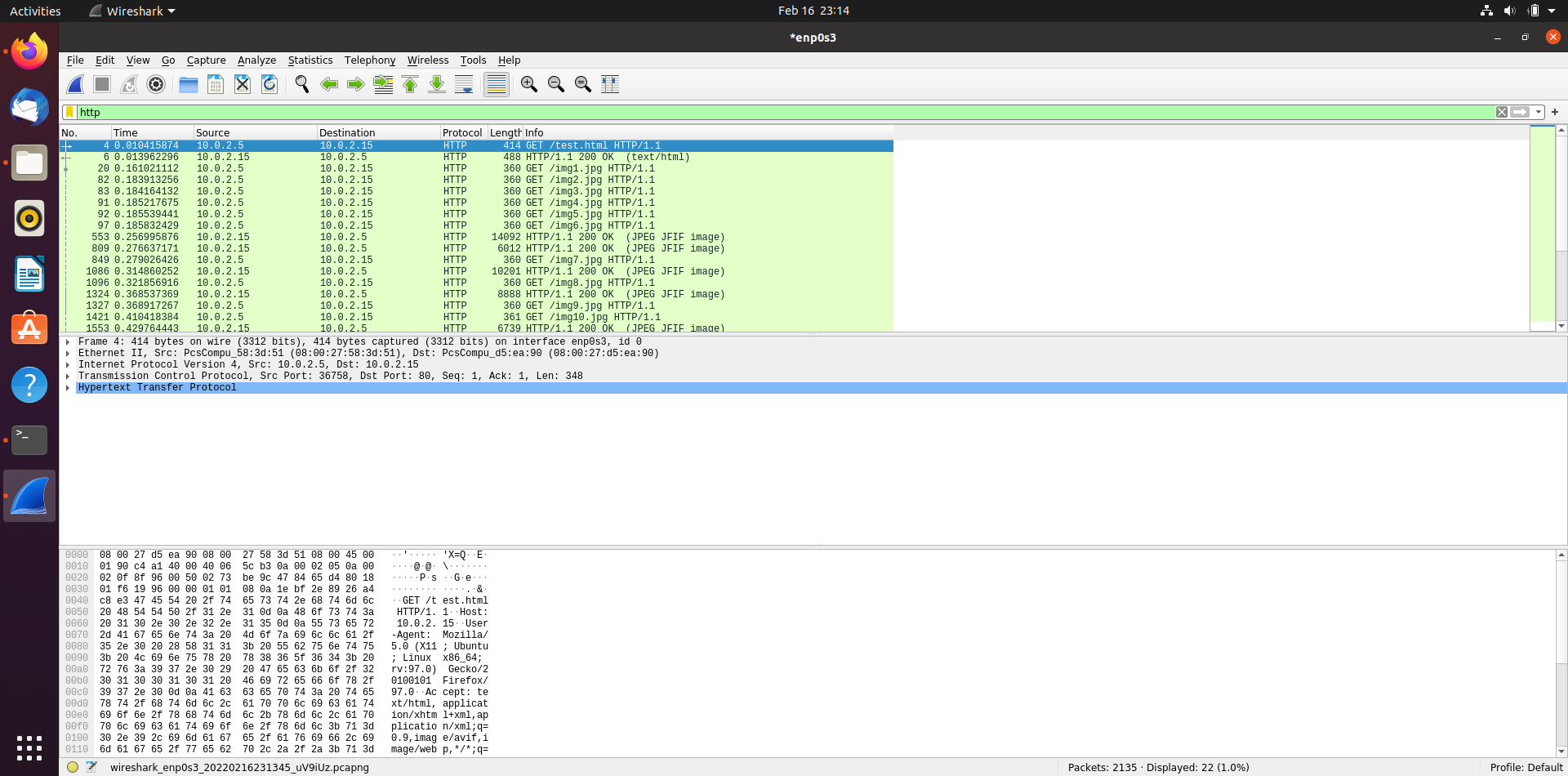
Here it is 24.510355626-23.960612864=0.544226986

**b)4 persistent connection:**

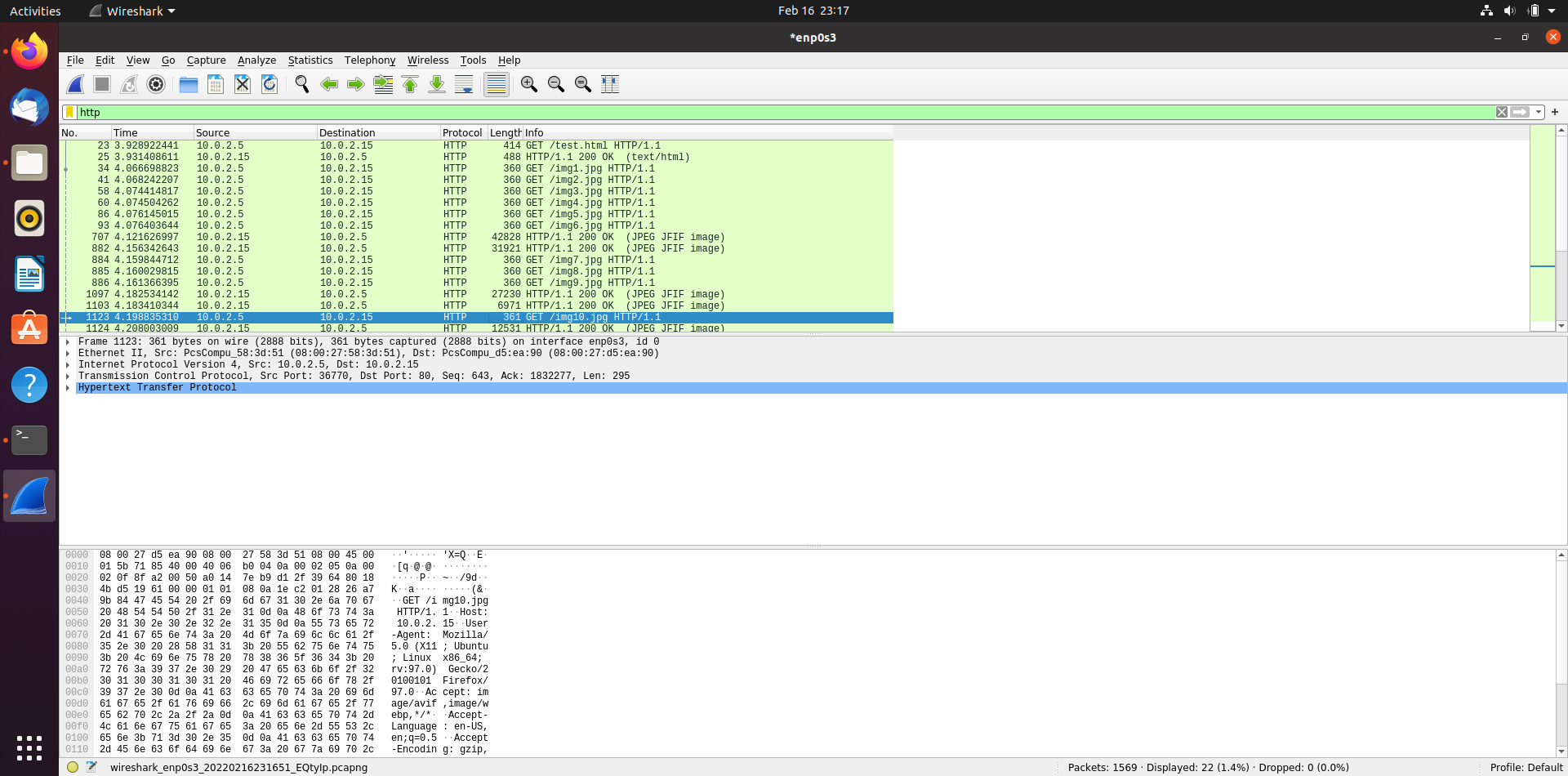


Here it is 9.054897148-8.5947568294=0.4601403186

**c)6 persistent connection:**



Here it is 0.4101418314 – 0.010415874 = 0.40000251

**d)10 persistent connection:**

Here it is 4.198835310 – 3.928922441 = 0.269912869

# TASK 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.

**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)

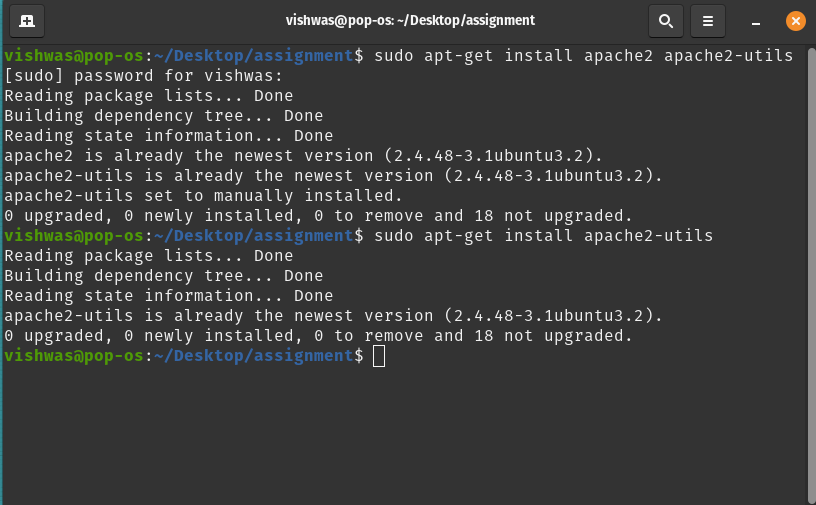
1. Executing the below commands on the terminal.

--> To update and integrate the existing softwares

## sudo apt-get update

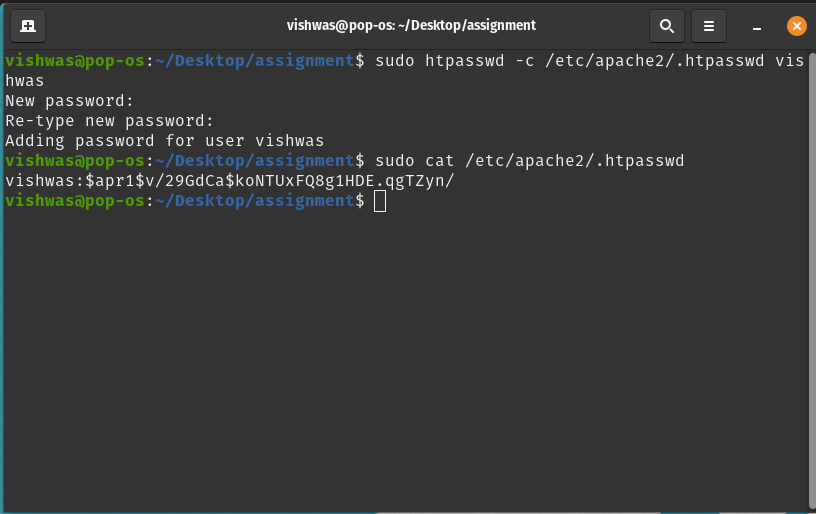
--> To install the apache utility

## sudo apt-get install apache2 apache2-utils



--> Provide username and password to set authentication

## sudo htpasswd -c /etc/apache2/.htpasswd ANY\_USERNAME



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

--> View the authentication

## sudo cat /etc/apache2/.htpasswd

1. 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

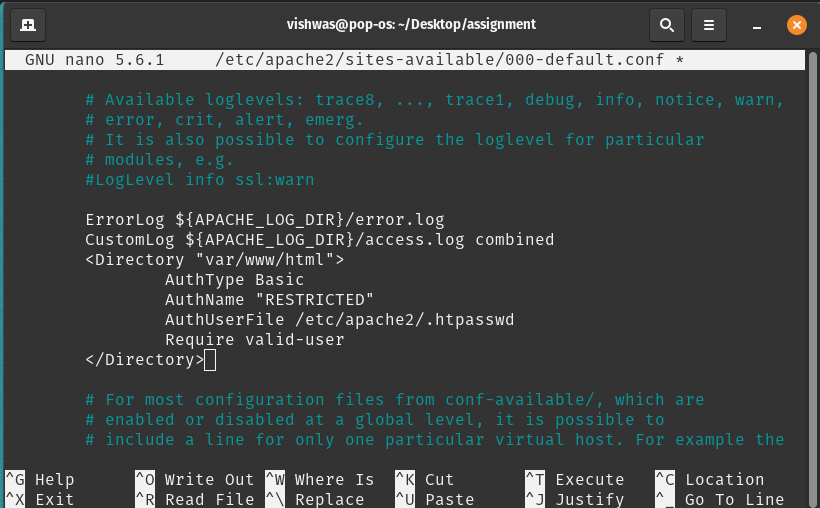
CustomLog ${APACHE\_LOG\_DIR}/access.log combined

<Directory "/var/www/html"> AuthType Basic

AuthName "RESTRICTED" AuthUserFile /etc/apache2/.htpasswd Require valid-user

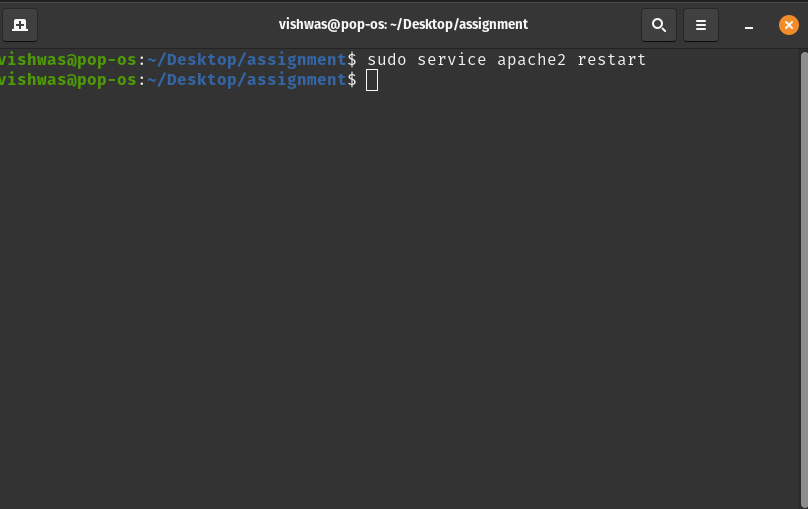
</Directory>

</VirtualHost>

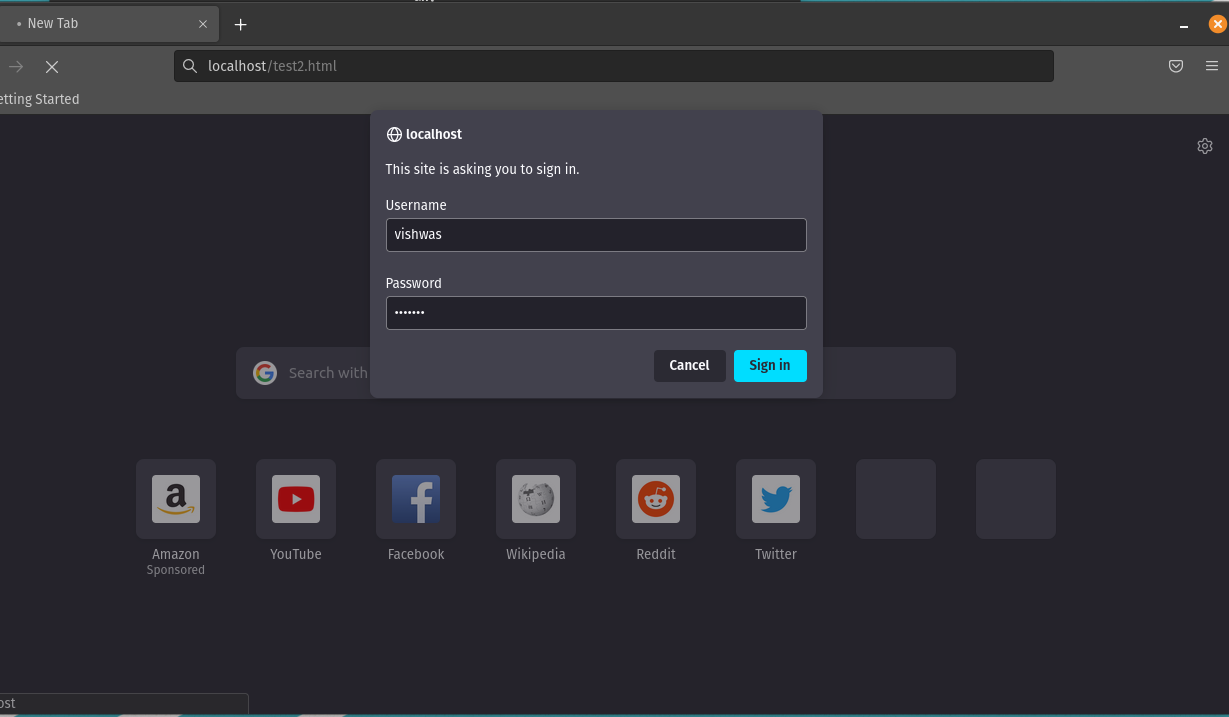


1. Password policy implementation is done by restarting the server as:

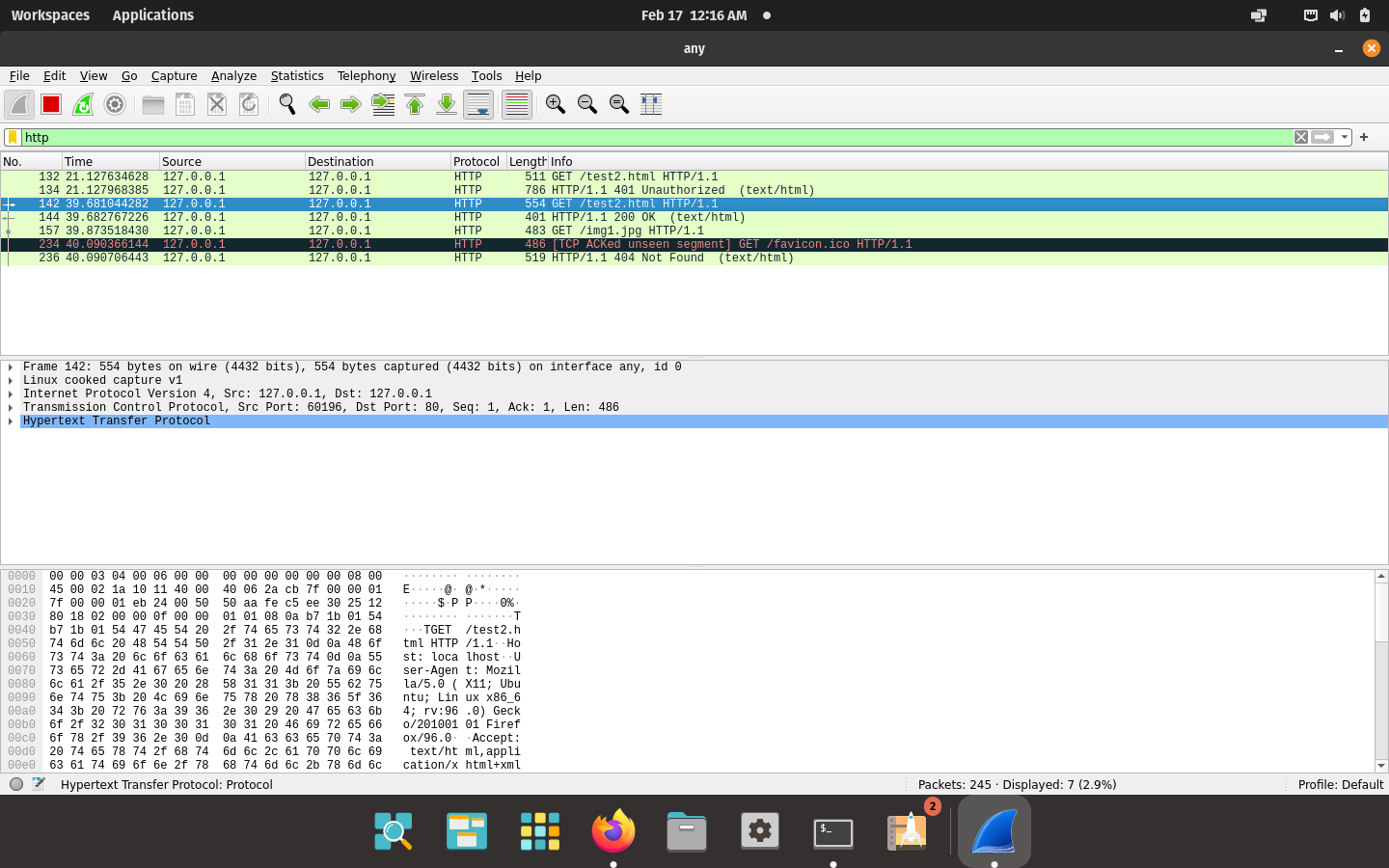
## sudo service apache2 restart



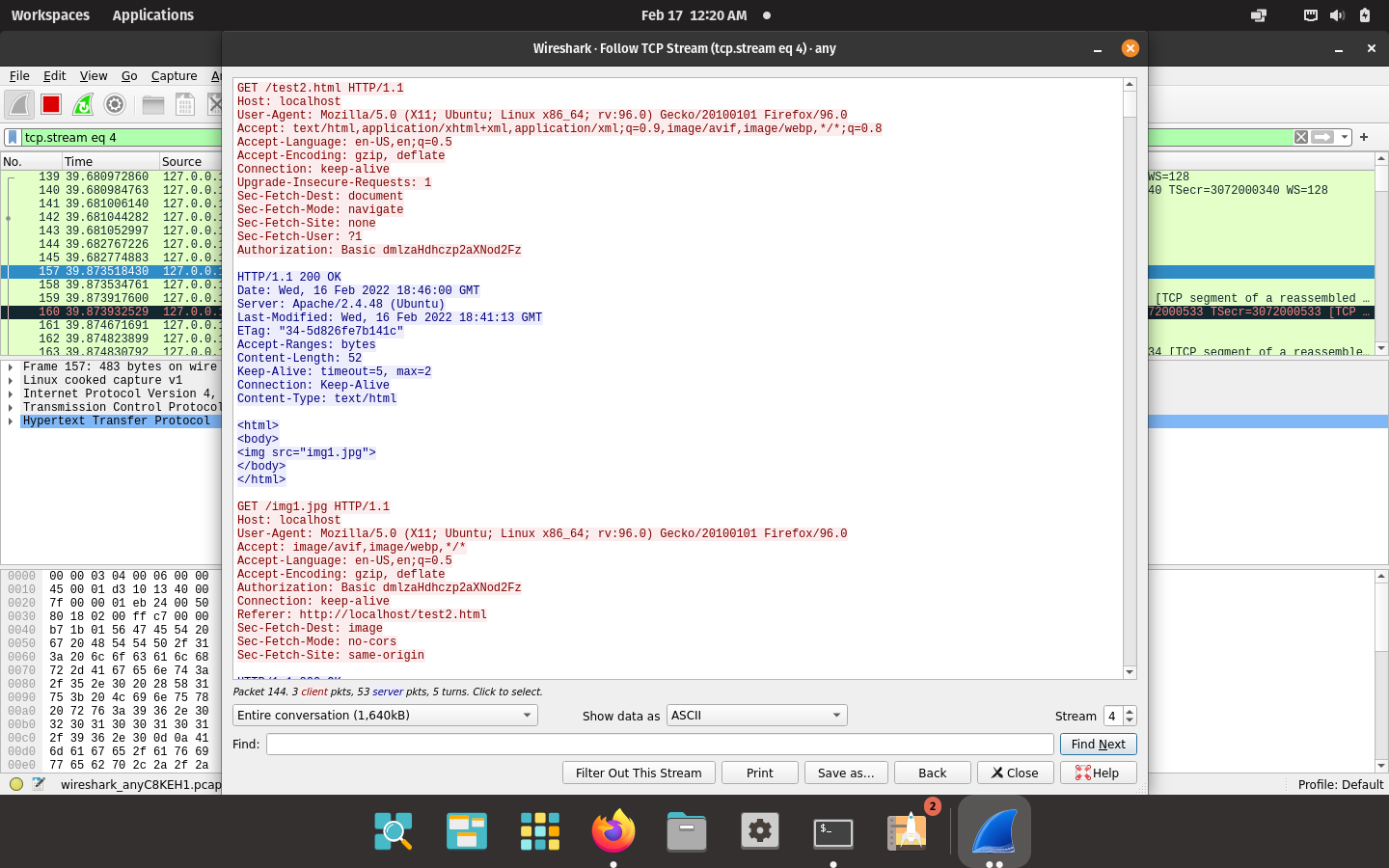
1. The localhost is then accessed using the Firefox browser requiring a username and a password set during the authentication phase.



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



1. 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

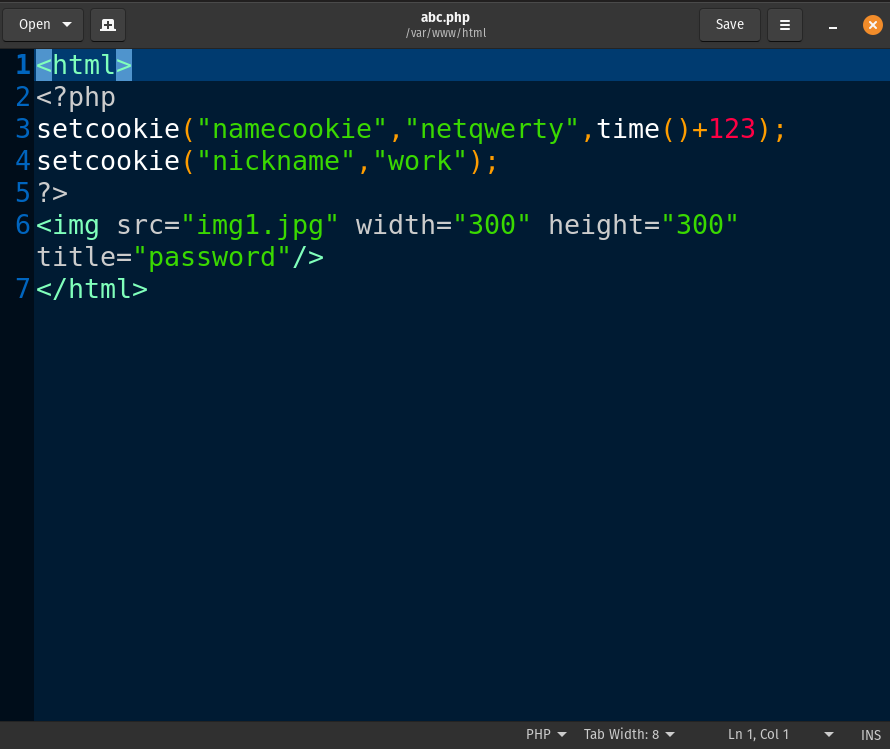
setcookie("namecookie","netqwerty",time()+123);

setcookie("nickname","work");

?>

<img src= “highres.png” width= “300” height= “300” title= “password” />

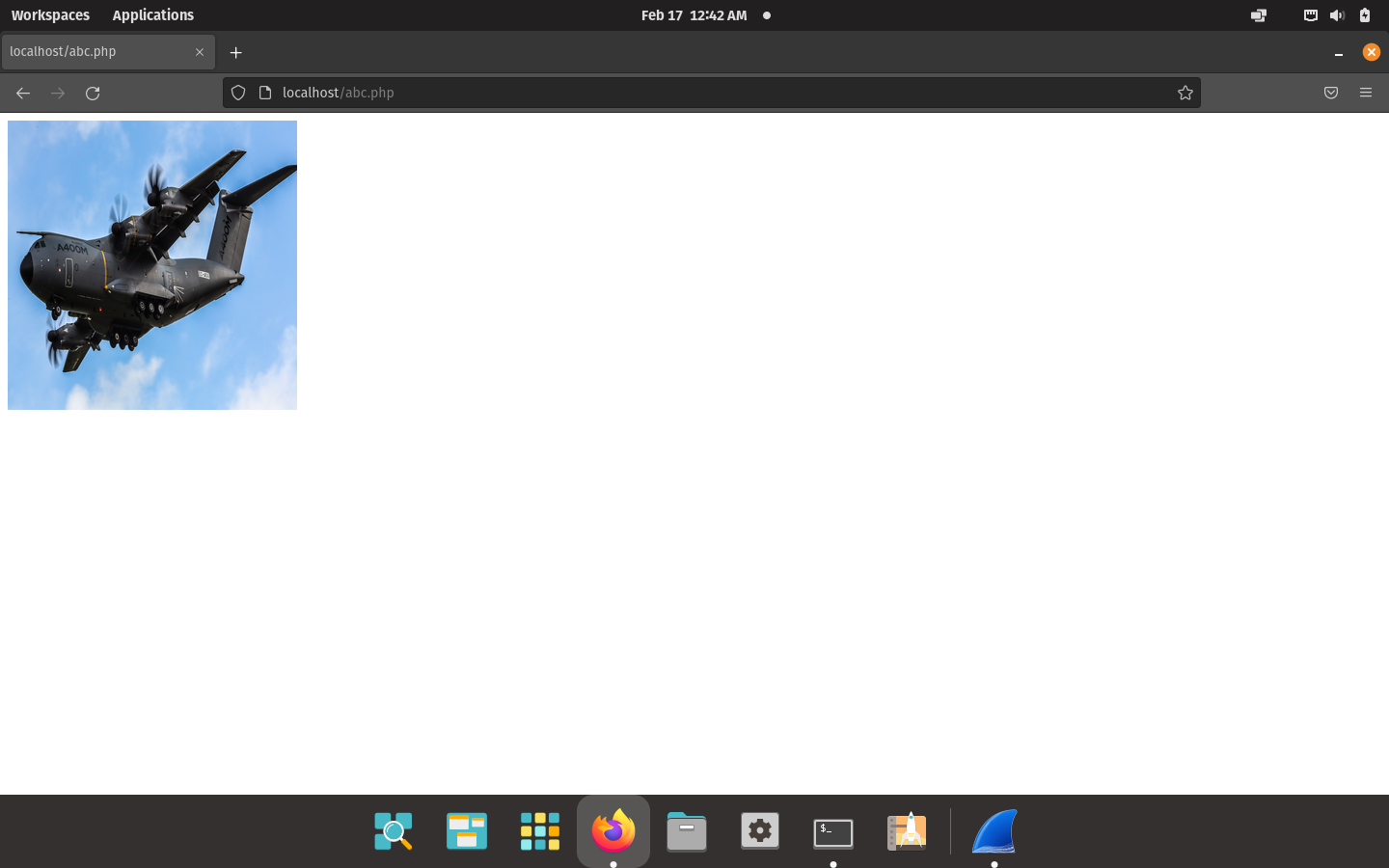
</html>



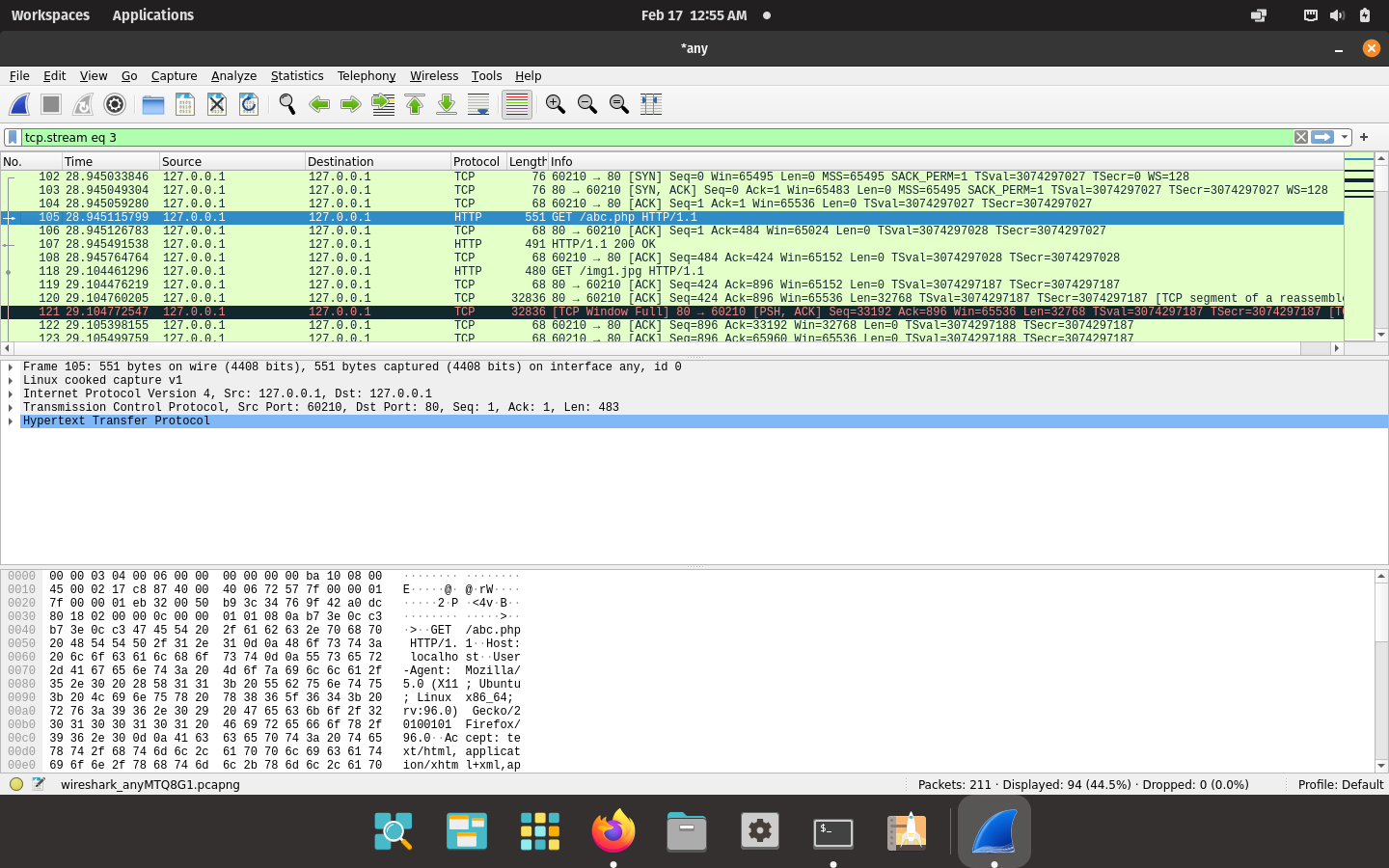
Note: Here you can add any image if required

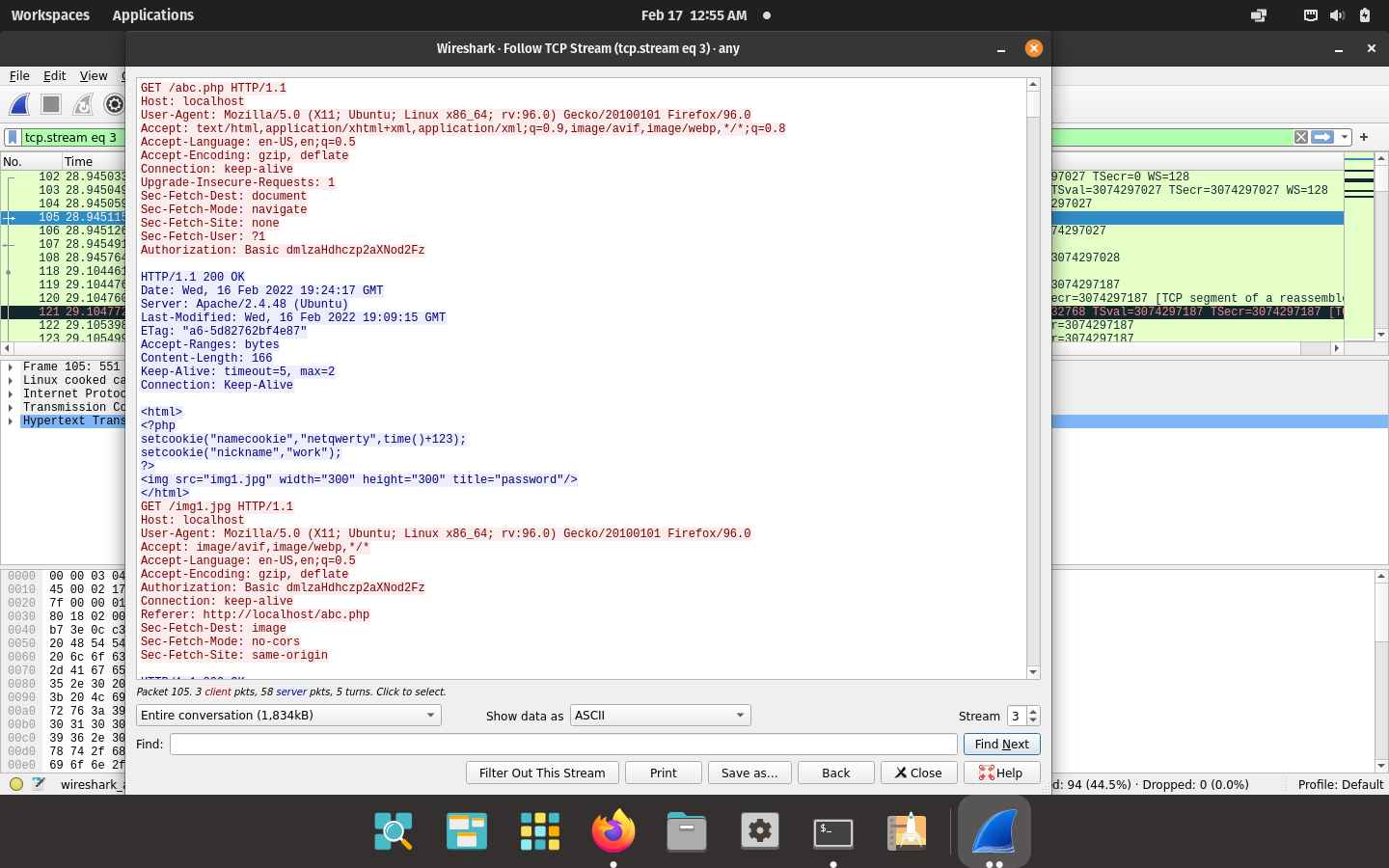
## Note: You can capture Cookies mostly during the first time of web access. Hence keep wireshark capture ready before executing the task for the first time.

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



1. 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.





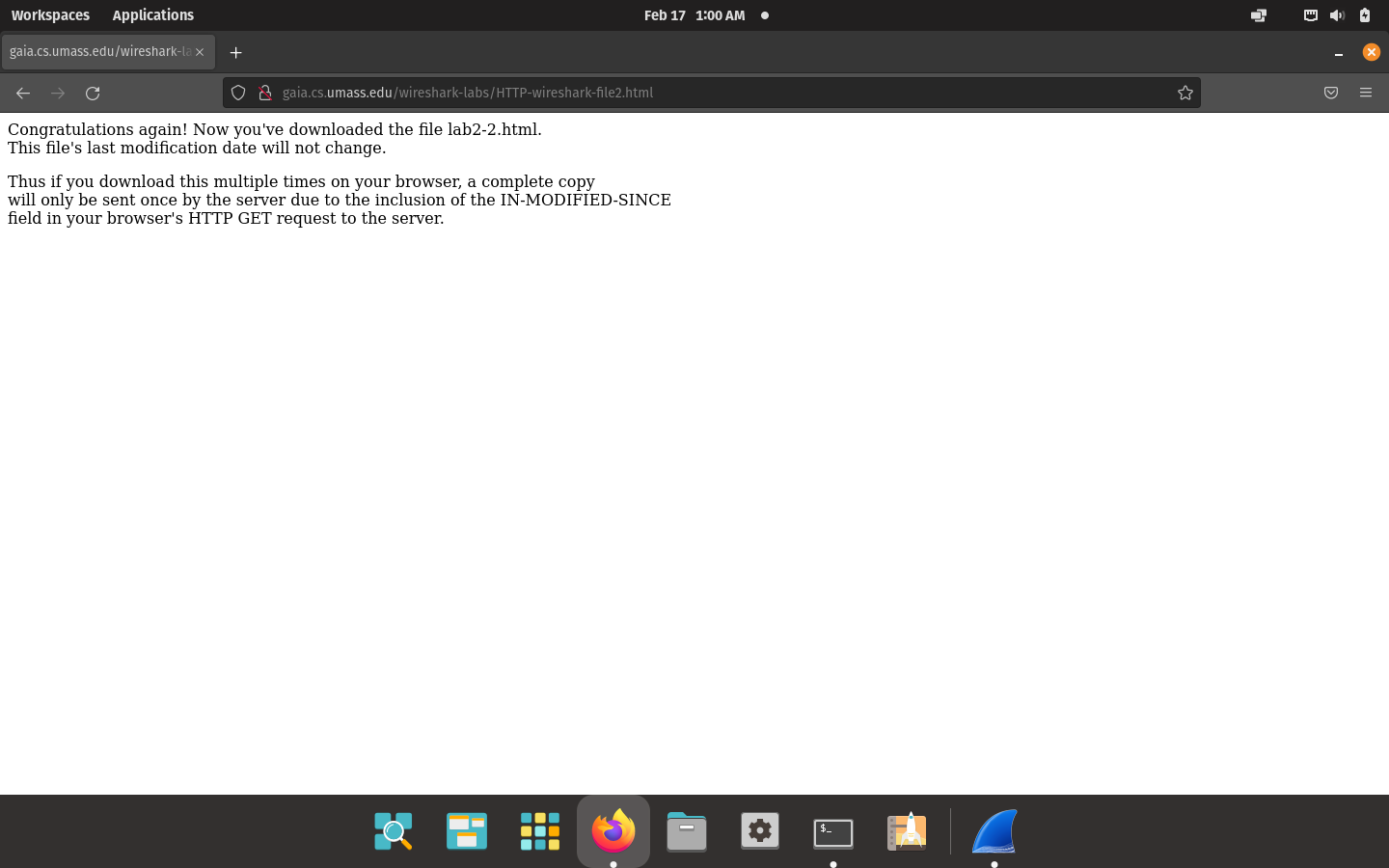
The cookie is set as shown in the above screenshot.

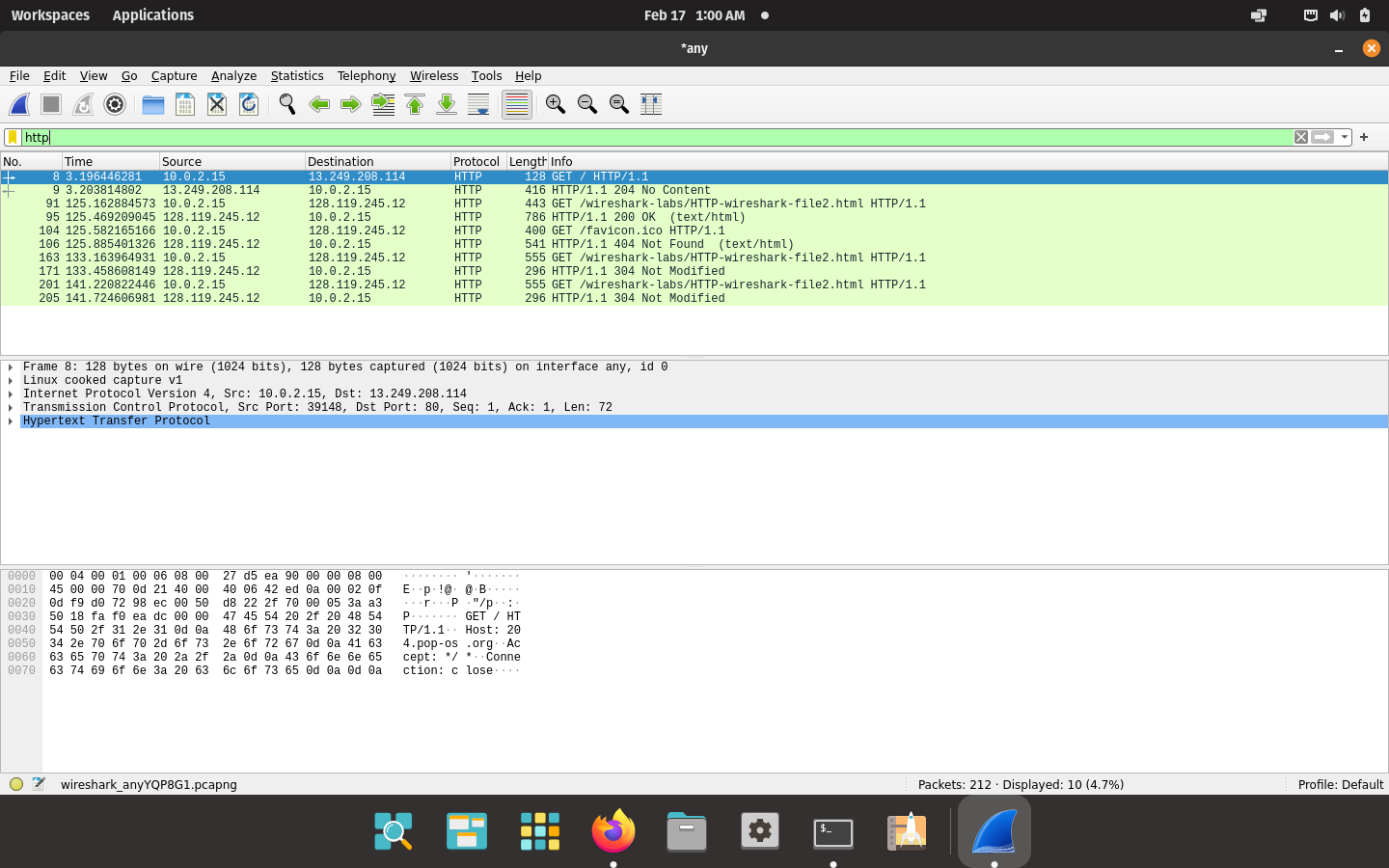
## 

## Here ‘vishwas’ is the admin and ‘vishwas’ is the password.

## 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](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.



**Observation:**

1. IF-Modified-Since line is not seen in the first HTTP get request but seen in the second request followed by the day,date and time of modification in the server.
2. The HTTP Status Code is 304 Not Modified.