Computers Network Laboratory

Name – B.Pravena
SRN – PES2UG19CS076

Week: #3 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.

Week: 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 BasicAuth 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

1) Password Authentication

Steps of Execution

1. Execute the commands on the terminal.

--> To update existing softwares

sudo apt-get update

--> To install the apache utility

sudo apt-get install apache2

```
prav@prav-VirtualBox:~$ sudo apt-get install apache2
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  apache2-bin apache2-data libaprutil1-dbd-sqlite3 libaprutil1-ldap
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data libaprutil1-dbd-sqlite3 libaprutil1-ldap
0 upgraded, 5 newly installed, 0 to remove and 62 not upgraded.
Need to get 1,453 kB of archives.
After this operation, 6,526 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://in.archive.ubuntu.com/ubuntu focal/main amd64 libaprutil1-dbd-sqlite3 amd64 1.6.1-4ubuntu2
[10.5 kB]
Get:2 http://in.archive.ubuntu.com/ubuntu focal/main amd64 libaprutil1-ldap amd64 1.6.1-4ubuntu2 [8,736
B]
Get:3 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 apache2-bin amd64 2.4.41-4ubuntu3.1 [
1,180 kB]
Get:4 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 apache2-data all 2.4.41-4ubuntu3.1 [1
58 kB]
Get:5 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 apache2 amd64 2.4.41-4ubuntu3.1 [95.5
 kB]
Fetched 1,453 kB in 3s (544 kB/s)
Selecting previously unselected package libaprutil1-dbd-sqlite3:amd64.
```

--> Provide username and password to set authentication

-->View the authentication

sudo cat /etc/apache2/.htpasswd

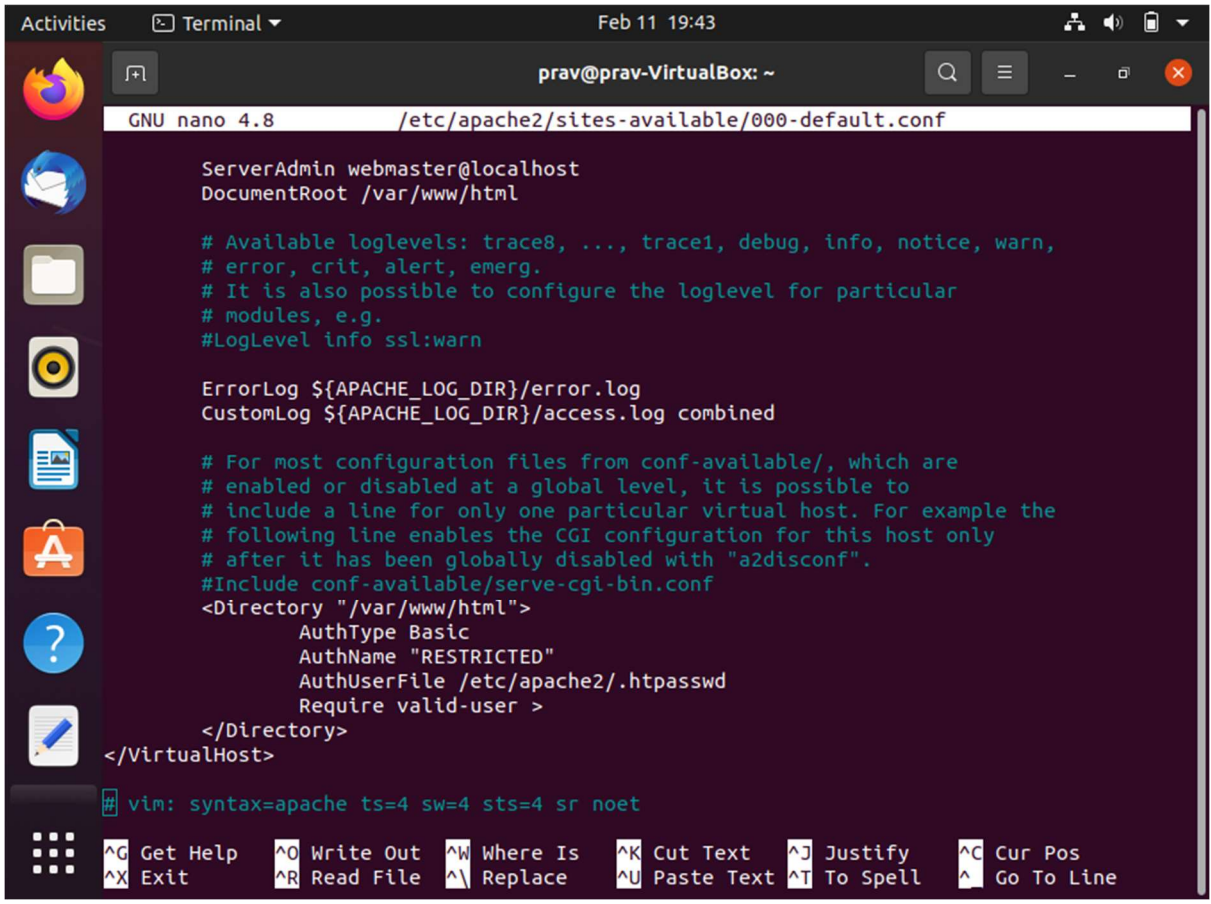
```
prav@prav-VirtualBox:~$ sudo htpasswd -c /etc/apache2/.htpasswd prav
New password:
Re-type new password:
Adding password for user prav
prav@prav-VirtualBox:~$ sudo cat /etc/apache2/.htpasswd
prav:$apr1$eXPMj1QO$t2rwDHJIotMXxnLweFdYY1
prav@prav-VirtualBox:~$
```

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

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>
```

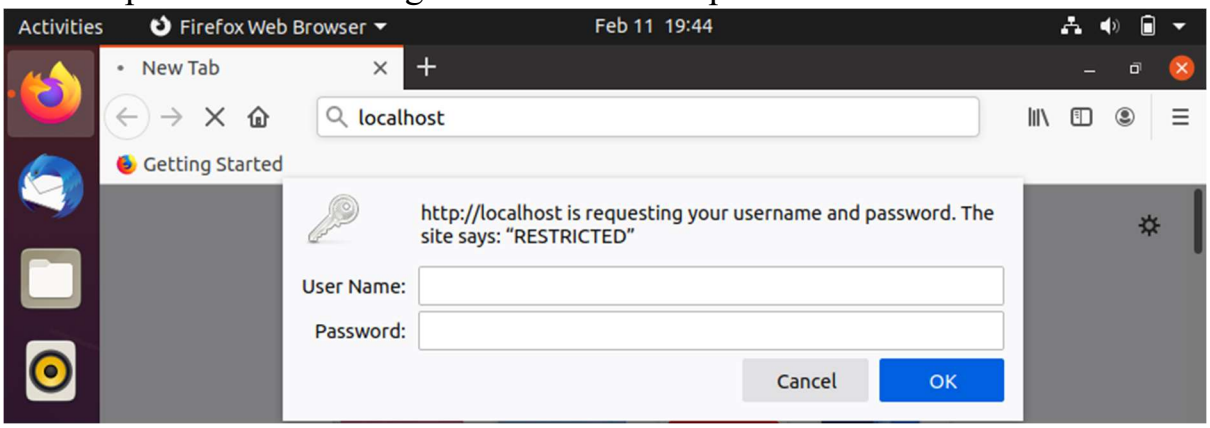


3. Password policy implementation is done by restarting the server as:

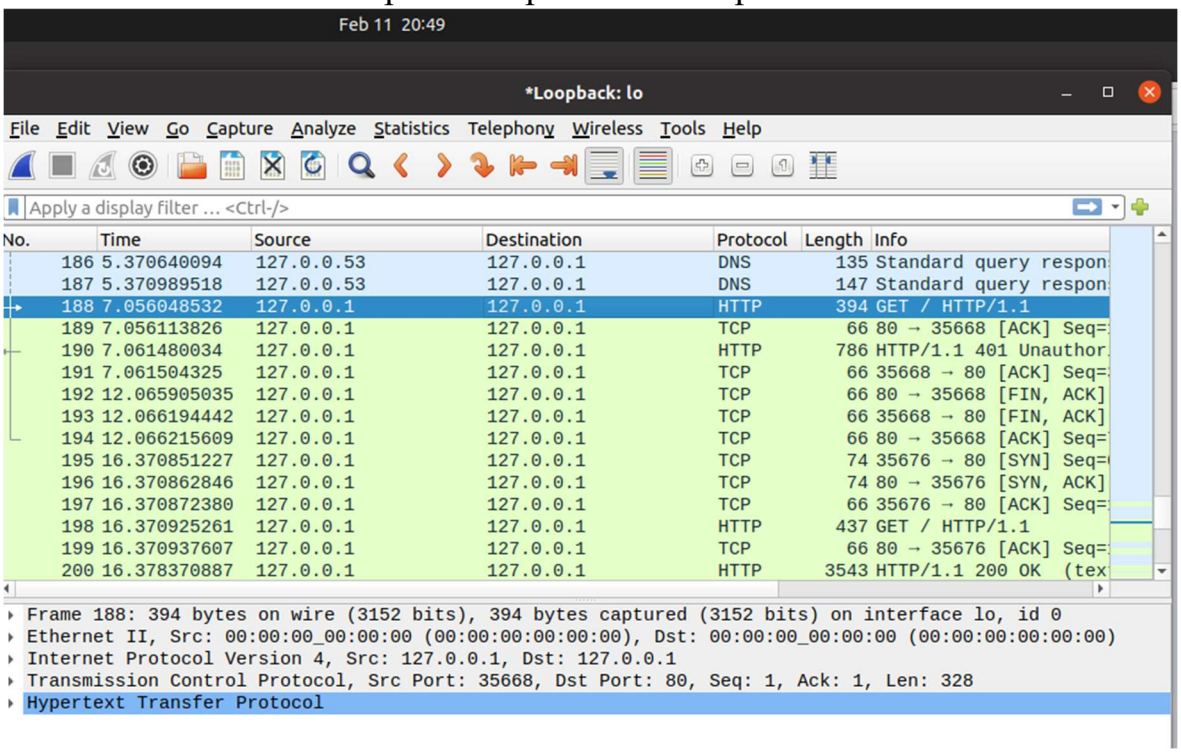
sudo service apache2 restart

```
prav@prav-VirtualBox:~$ sudo service apache2 restart
prav@prav-VirtualBox:~$
```

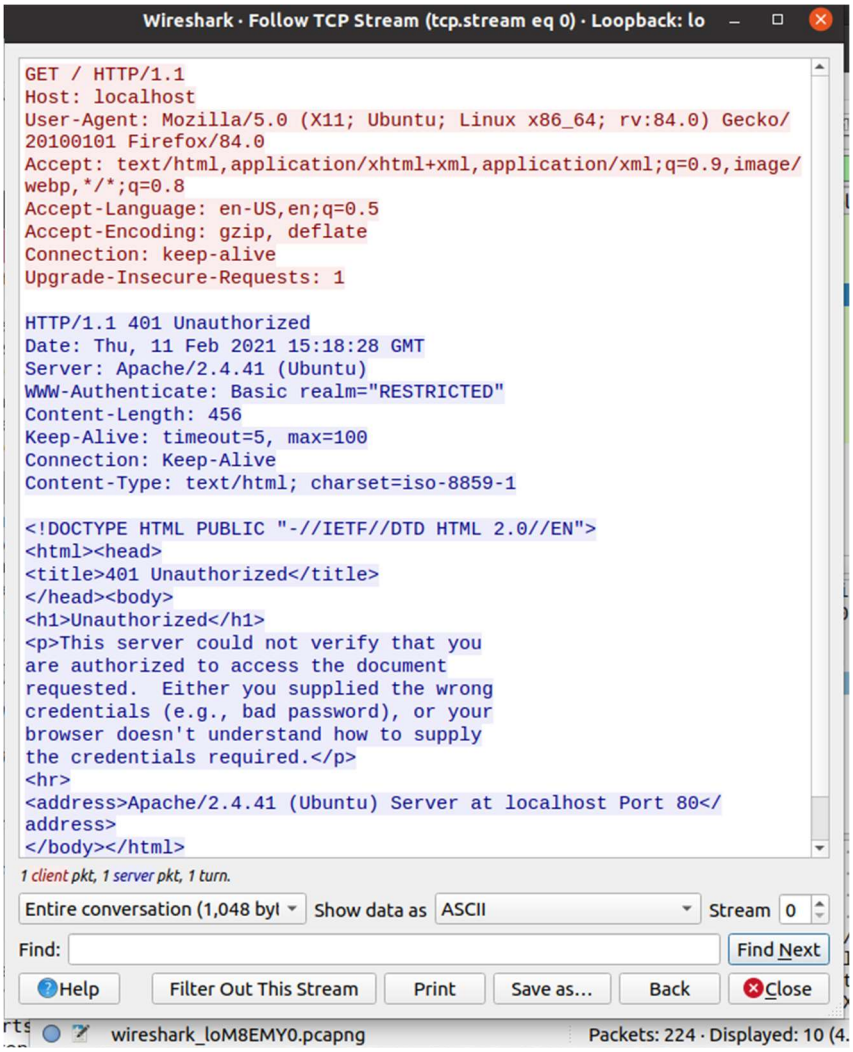
4. The localhost is then accessed using the Firefox browser requiring a username and a password set during the authentication phase.



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.



2) Cookie Setting

Steps of Execution

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()+12
3); setcookie("nickname","work"); ?>

</html>

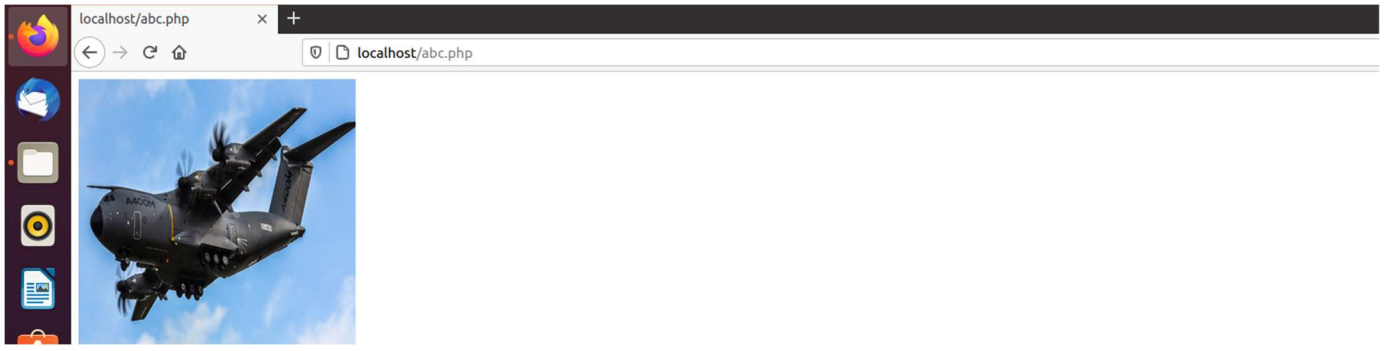
```
 GNU nano 2.5.3                File: abc.php

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

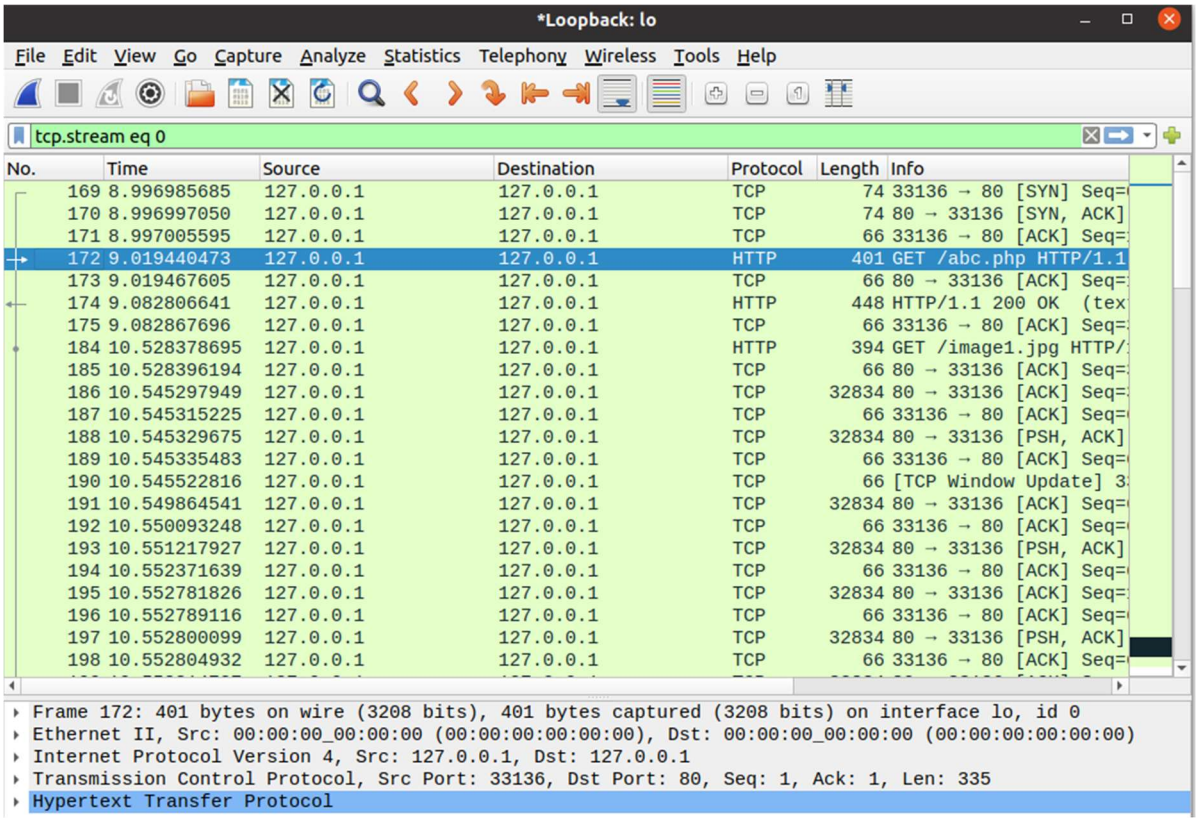
Note: Here you can add any image if required

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.

Observations -:

Working of base 64 algorithm

Base64 encoding is used to convert binary data into a text-like format to be transported across channels that only reliably support text context.

Base64 encoding takes the original binary data and operates on it by dividing it into tokens of 3 bytes. A byte consists of 8 bits, so Base64 takes 24 bits in total. These 3 bytes are then converted into 4 printable characters from the ASCII standard.

Parameters associated with cookie in WireShark capture -:

-Cookie name and value

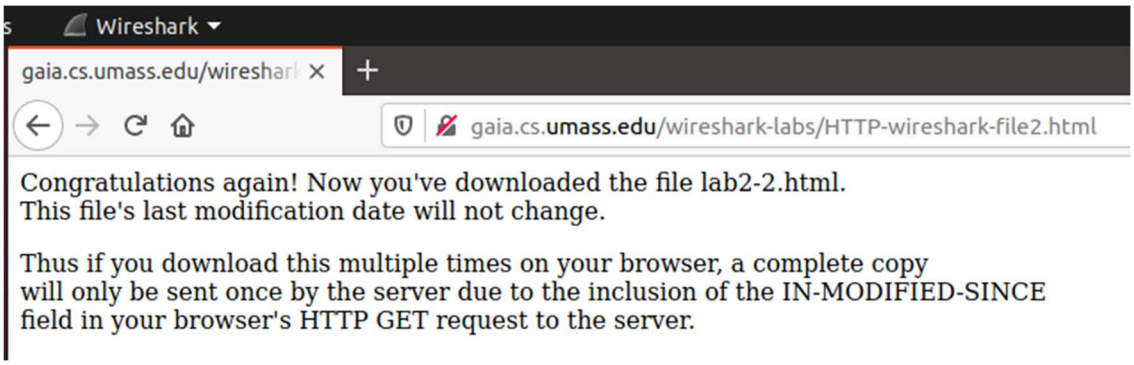
-Expiry time of the cookie

-Max indicated the number of milliseconds the cookie will expire in if its not deleted before.

3) **Conditional Get: If-Modified-Since**

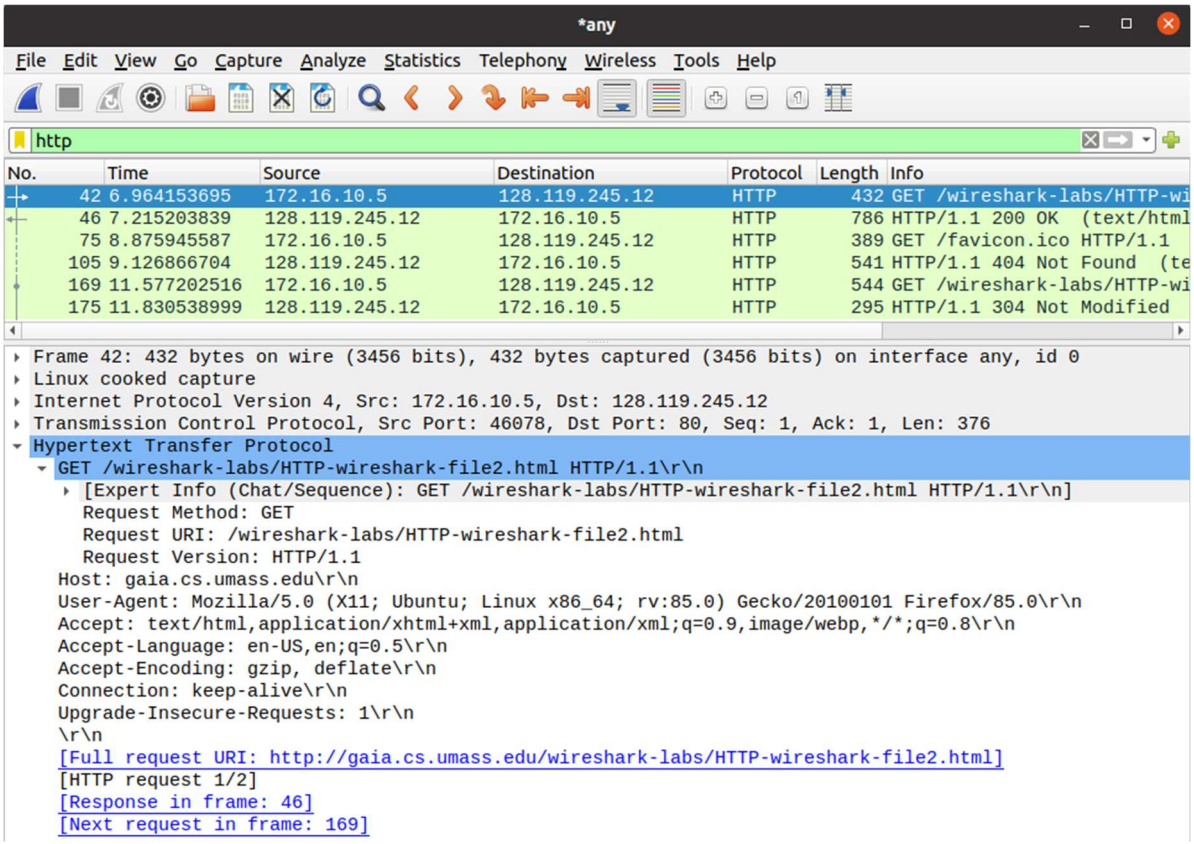
Before performing the steps below, make sure your browser's cache is empty. (To do this under Firefox, select Tools -> Clear Recent History and check the Cache box). Now do the following: ❍ Start up your web browser, and make sure your browser's cache is cleared, as discussed above.

- ❍ Start up the Wireshark packet sniffer.
- ❍ Enter the following URL into your browser http://gaia.cs.umass.edu/wiresharklabs/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 packetlisting window.



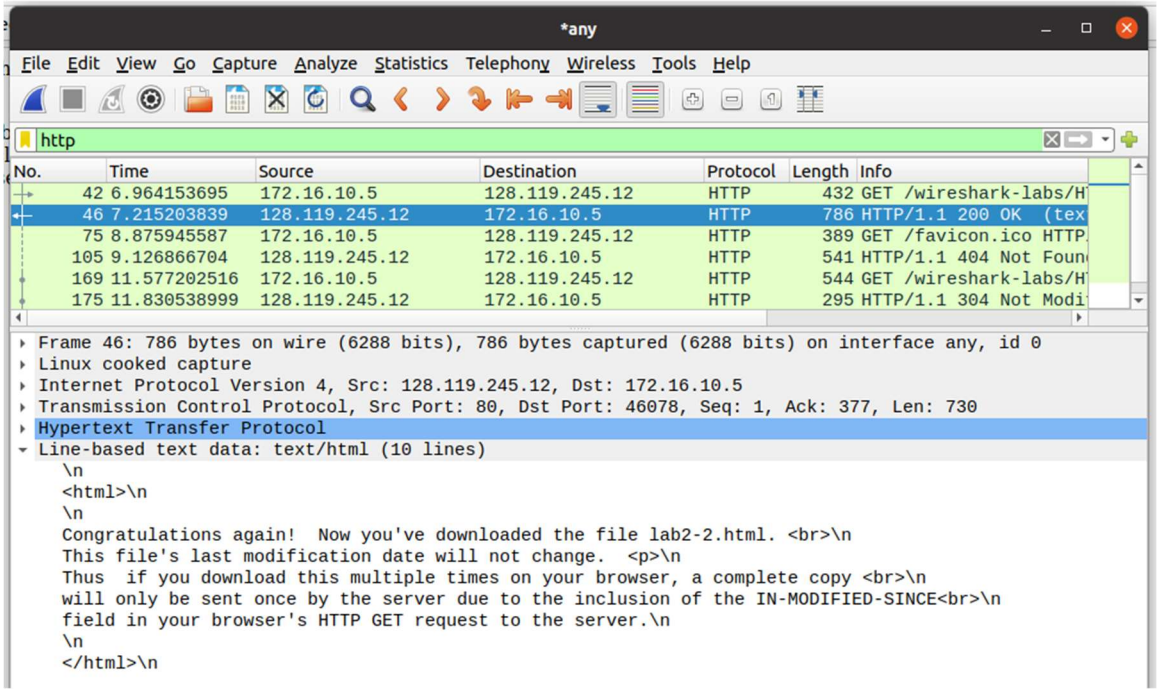
Observations -:

1)First HTTP Get Request -:



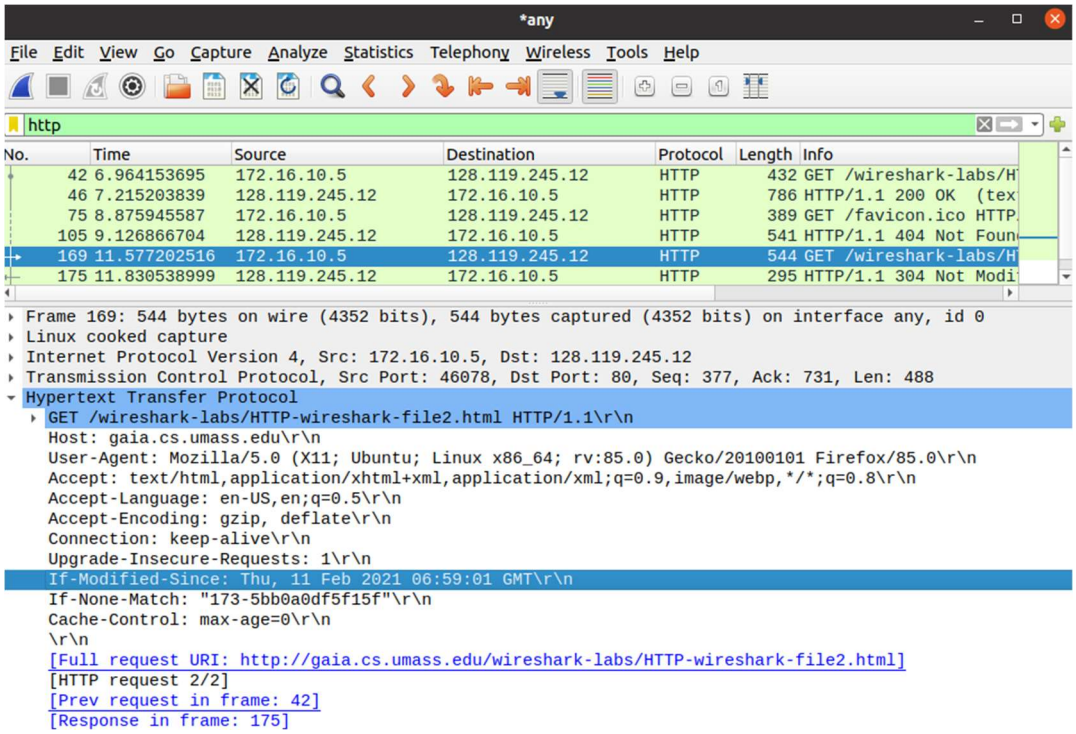
It does not contain an “IF MODIFIED-SINCE LINE”.

2) Contents of the server response -:



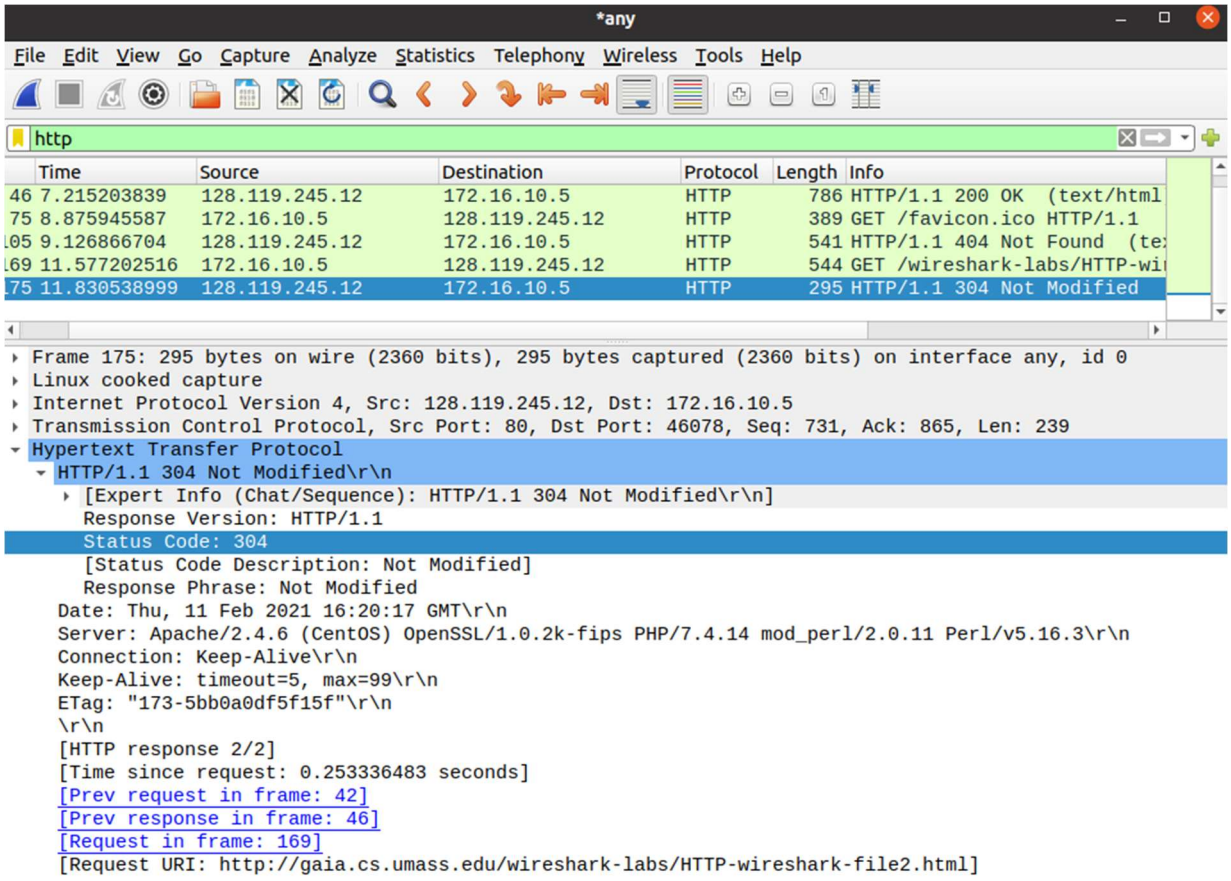
The server explicitly returns the contents of the file since we can see the contents of an HTML document that has been returned.

4) Second HTTP GET Request -:



We see the IF-MODIFIED-SINCE header. It returns the date and time for which a server first downloaded a resource from the server.

4)Second Server Response

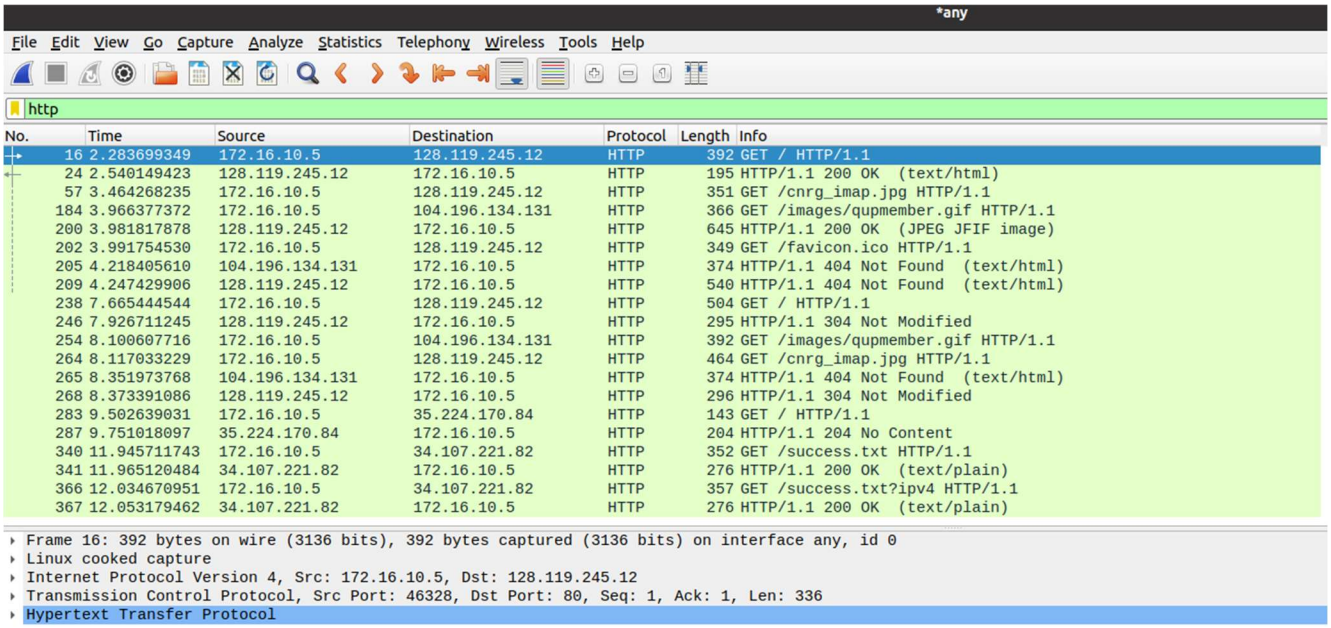


HTTP Status Code 304 and phrase Not Modified returned.

This means that the file has not changed since the last time it was accessed and there is no need to download the file again. Hence the file contents not returned too. This allows both the website owner and the visitor to converse resources as the file does not have to be retrieved each time.

SERVER WITH IMAGES

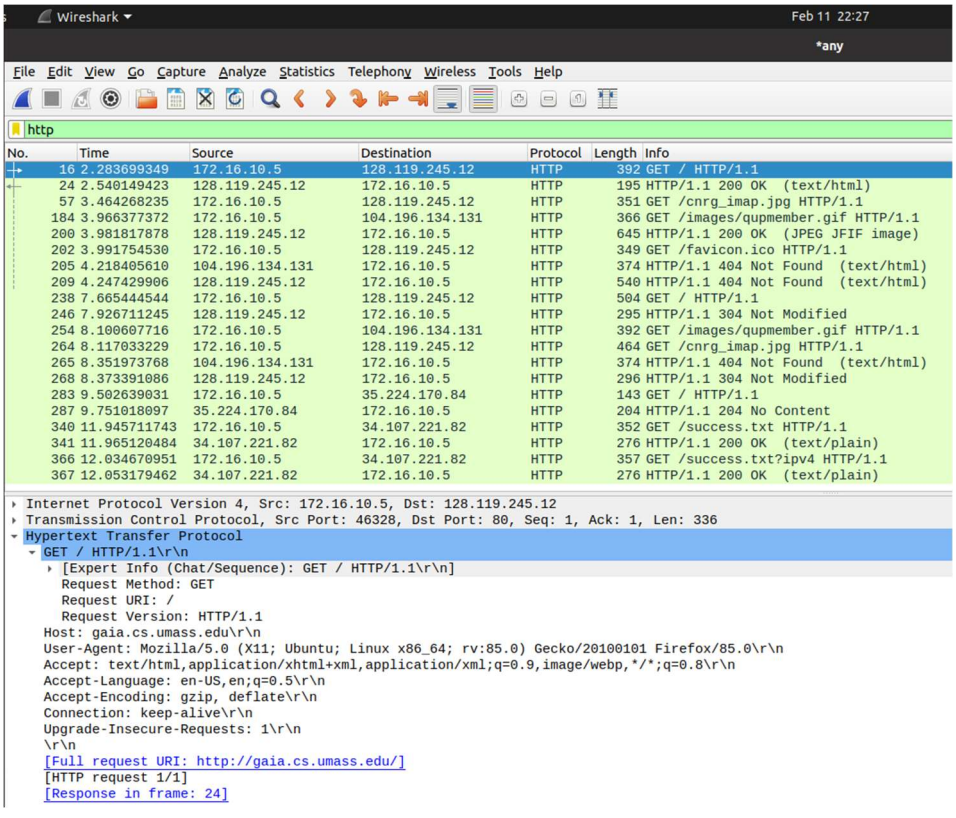
Accessing a website with images -:



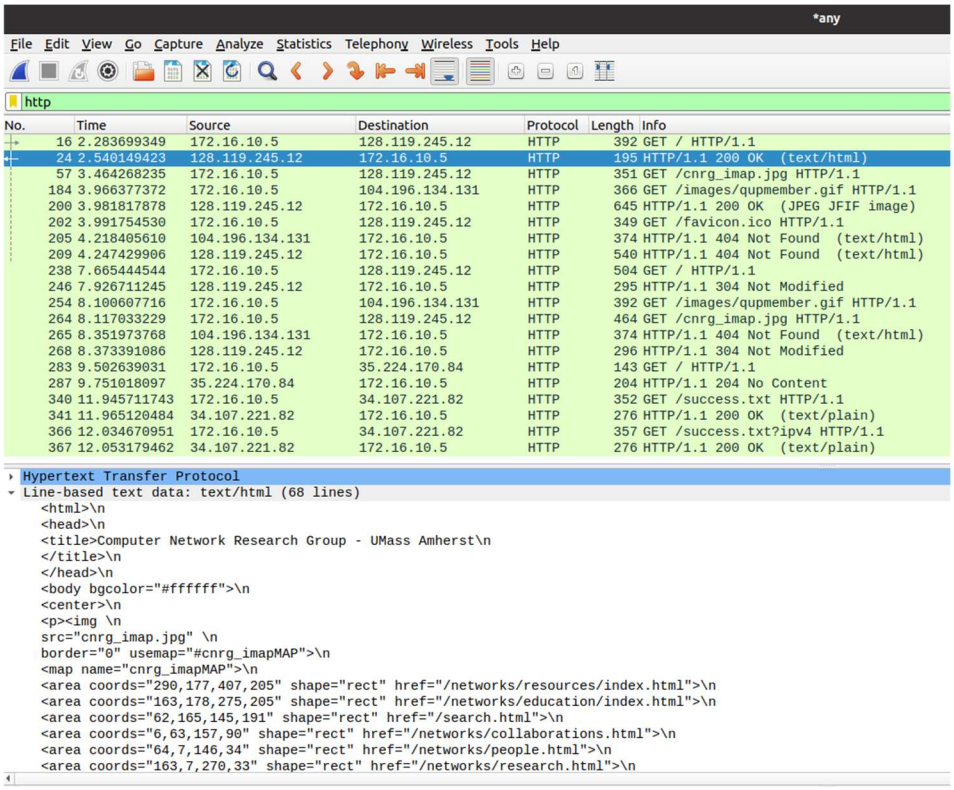
OBSERVATIONS -:

Accessing the website for the first time -:

HTTP GET Request -:

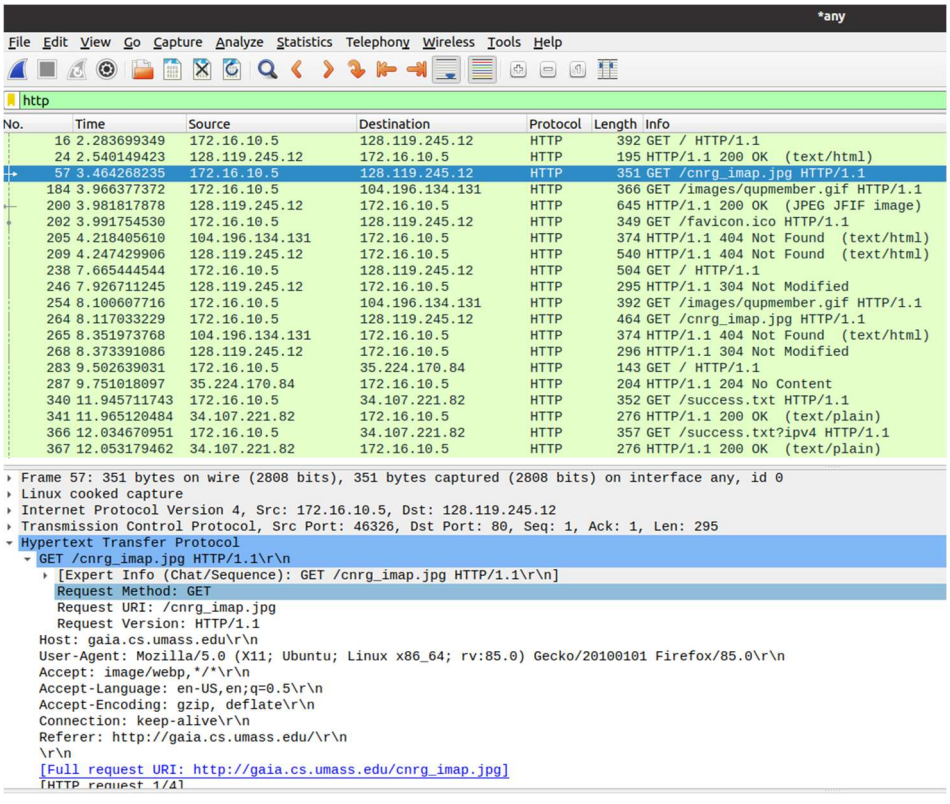


Response -:



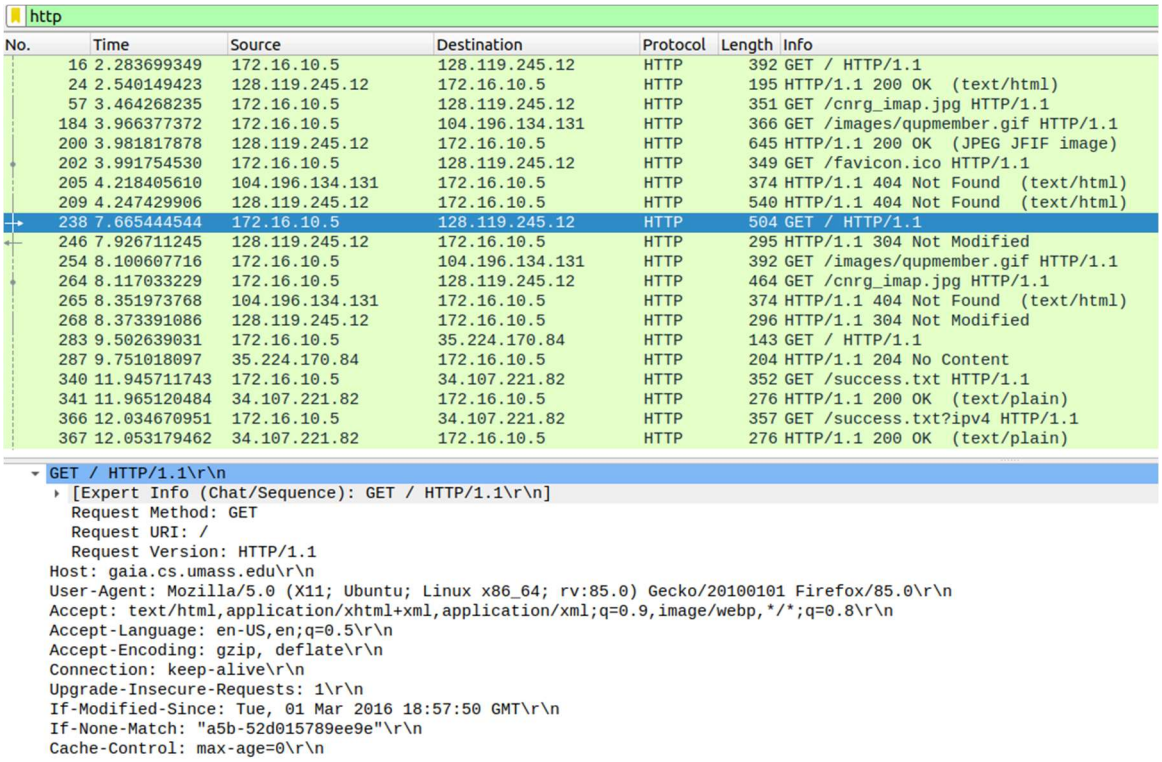
Contents of the HTML document are returned.

Request for an image -:



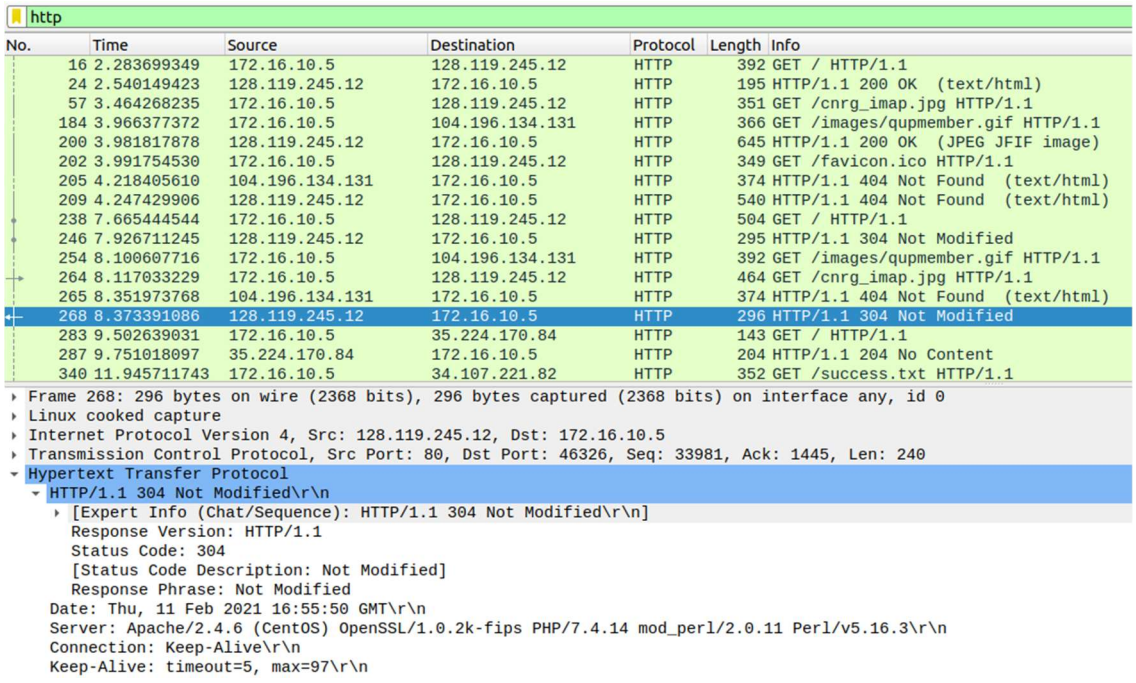
Accessing the website for the second time -:

HTTP Get Request -:



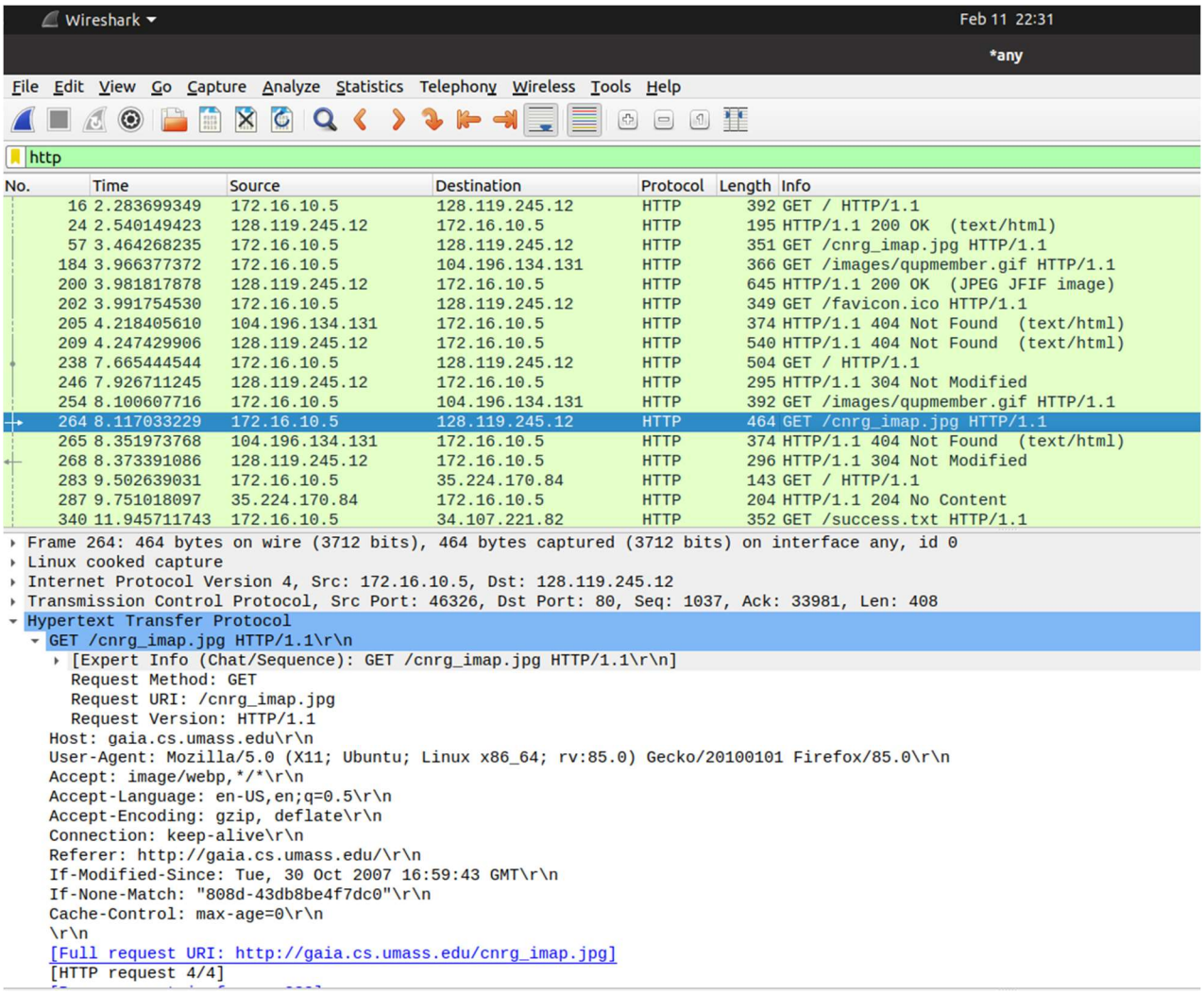
We can see the IF-MODIFIED-SINCE header.

Server Response -:



The status code is 304 (description – Not Modified).

Request for the same image -:



IF-MODIFIED-SINCE header is included here.