

Computer Networks **Lab #3**

Name : Laxmikant Bhujang Gurav

Section : K

Roll no : 55

SRN : PES1UG20CS658

Understanding Persistent and Non-persistent HTTP Connections

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

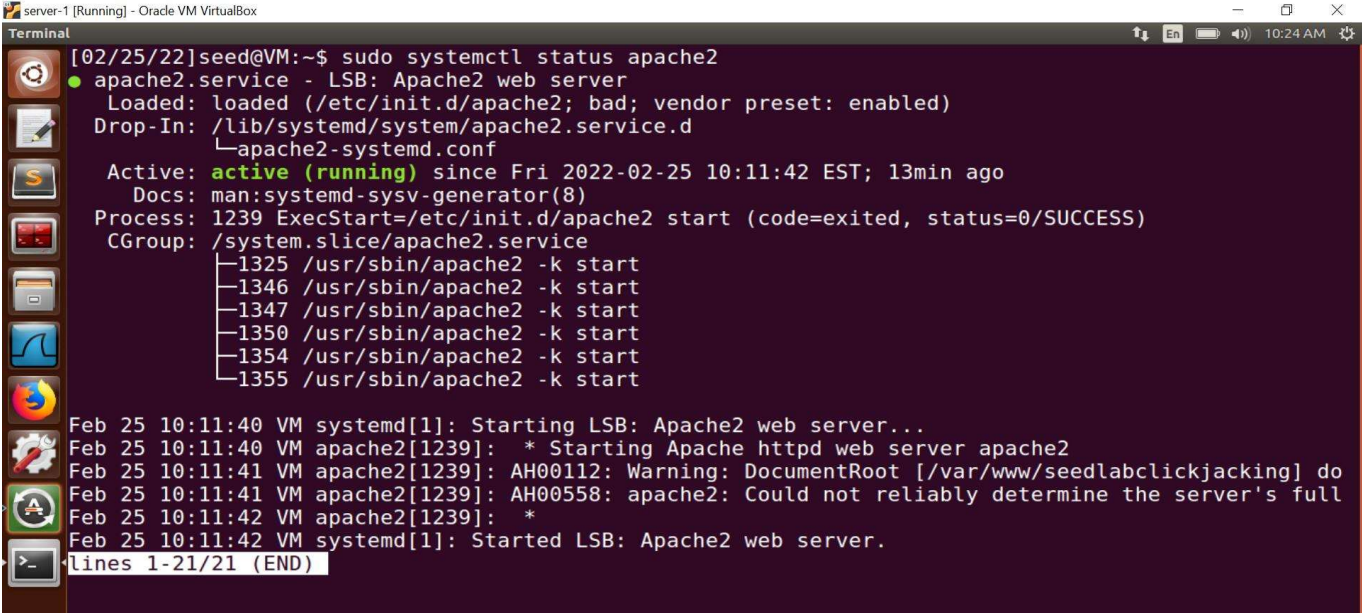
EXECUTION STEPS

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

sudo systemctl status apache2



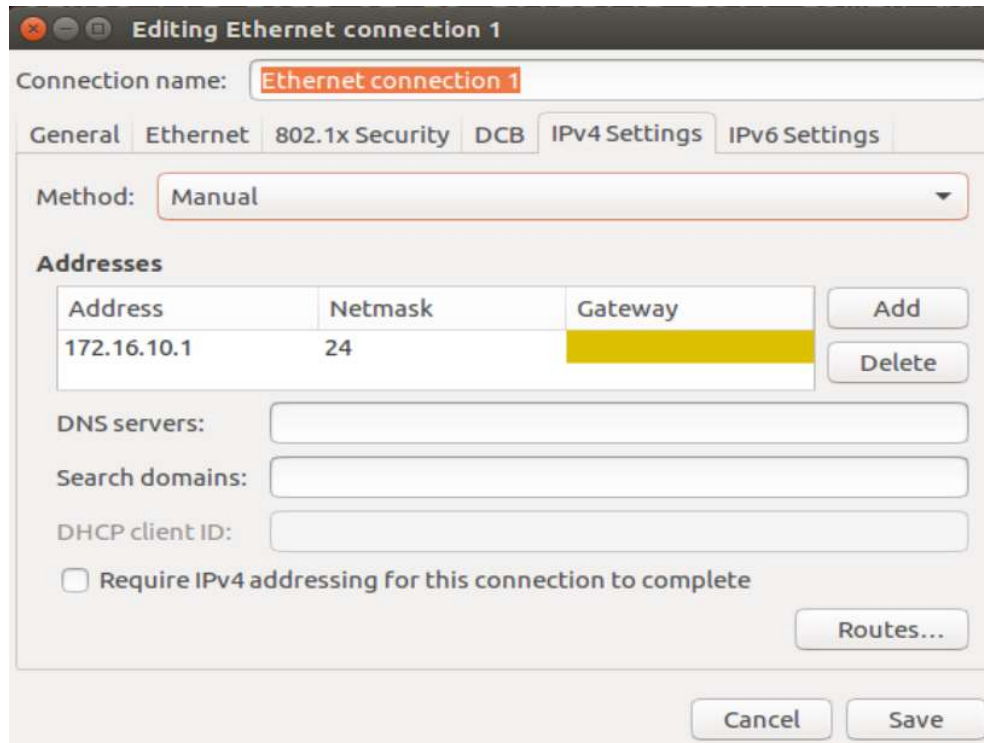
```
[02/25/22]seed@VM:~$ sudo systemctl status apache2
● apache2.service - LSB: Apache2 web server
   Loaded: loaded (/etc/init.d/apache2; bad; vendor preset: enabled)
   Drop-In: /lib/systemd/system/apache2.service.d
            └─apache2-systemd.conf
   Active: active (running) since Fri 2022-02-25 10:11:42 EST; 13min ago
     Docs: man:systemd-sysv-generator(8)
  Process: 1239 ExecStart=/etc/init.d/apache2 start (code=exited, status=0/SUCCESS)
    CGroup: /system.slice/apache2.service
            └─1325 /usr/sbin/apache2 -k start
               1346 /usr/sbin/apache2 -k start
               1347 /usr/sbin/apache2 -k start
               1350 /usr/sbin/apache2 -k start
               1354 /usr/sbin/apache2 -k start
               1355 /usr/sbin/apache2 -k start

Feb 25 10:11:40 VM systemd[1]: Starting LSB: Apache2 web server...
Feb 25 10:11:40 VM apache2[1239]: * Starting Apache httpd web server apache2
Feb 25 10:11:41 VM apache2[1239]: AH00112: Warning: DocumentRoot [/var/www/seedlabclickjacking] do
Feb 25 10:11:41 VM apache2[1239]: AH00558: apache2: Could not reliably determine the server's full
Feb 25 10:11:42 VM apache2[1239]: *
Feb 25 10:11:42 VM systemd[1]: Started LSB: Apache2 web server.
lines 1-21/21 (END)
```

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

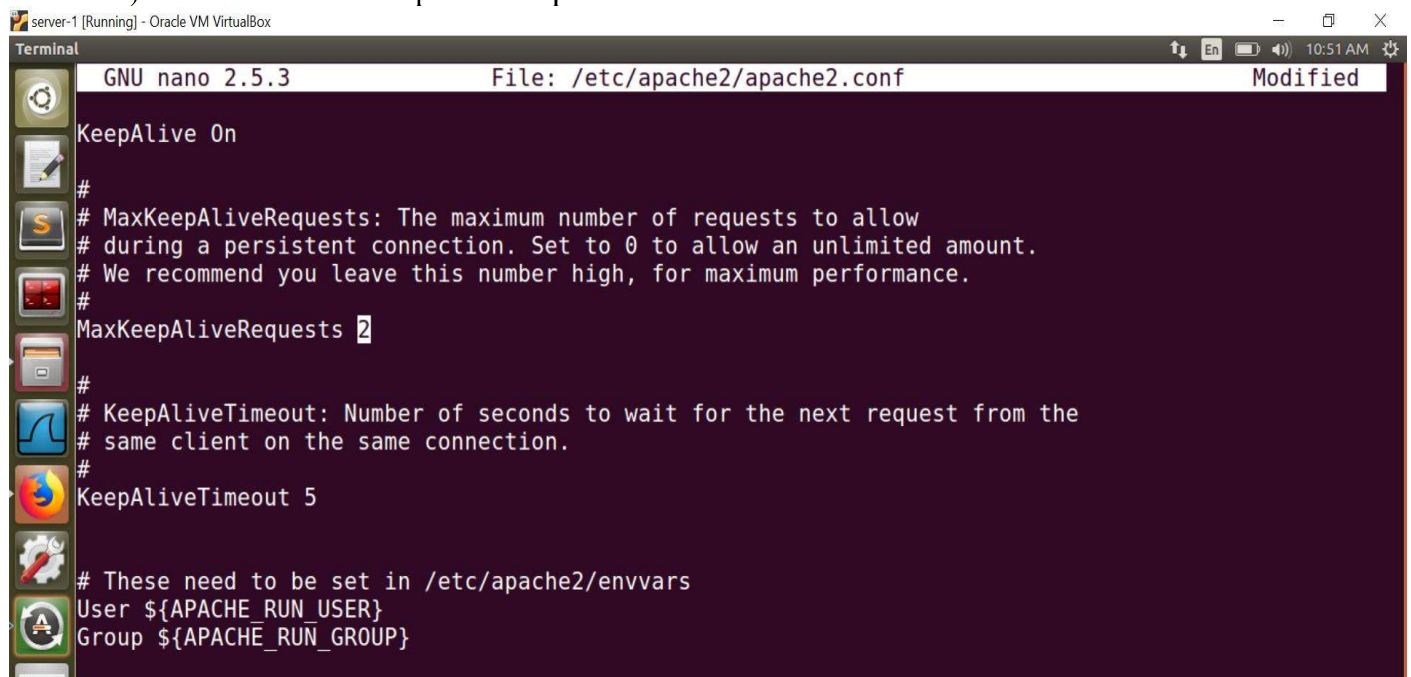
```
$sudo ip addr add 172.16.10.1/24 dev enps0
```

```
$sudo ip addr
```

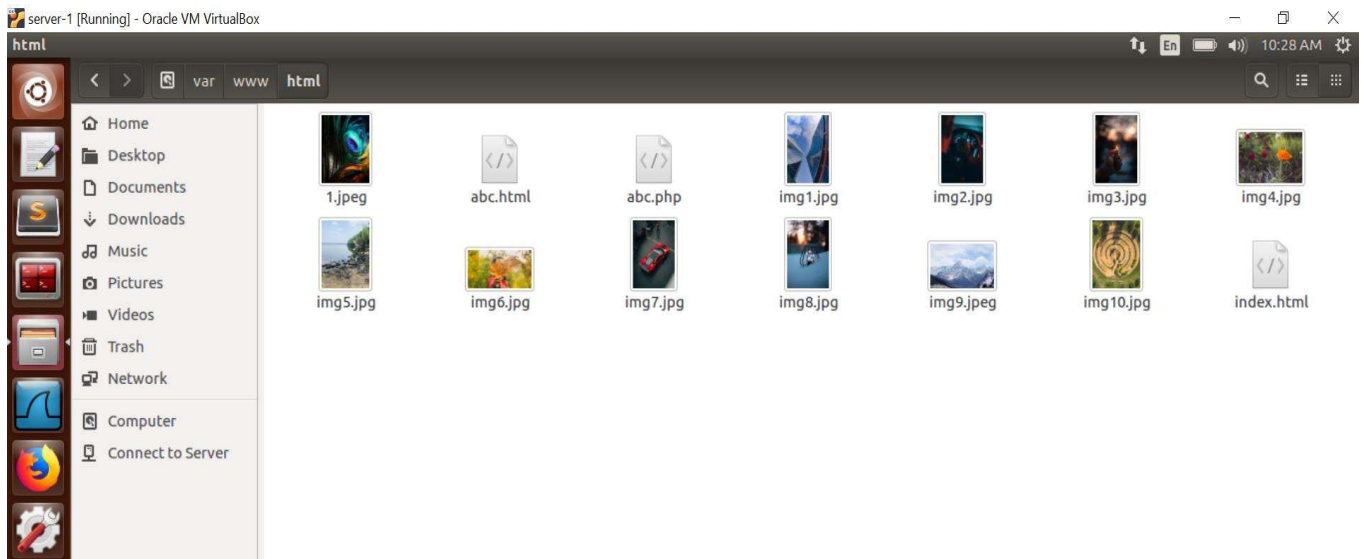


Step 4: The apache2.conf file present in the etc/apache2 directory is modified as:

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



Step 5: Store images in the server path



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)

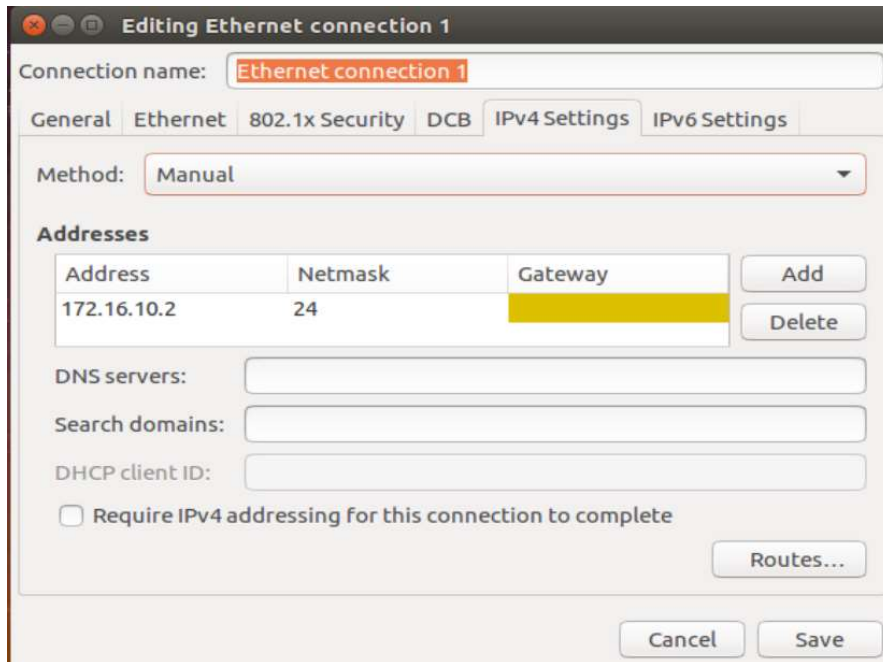


Client side:

Client IP address can be set by the following command.

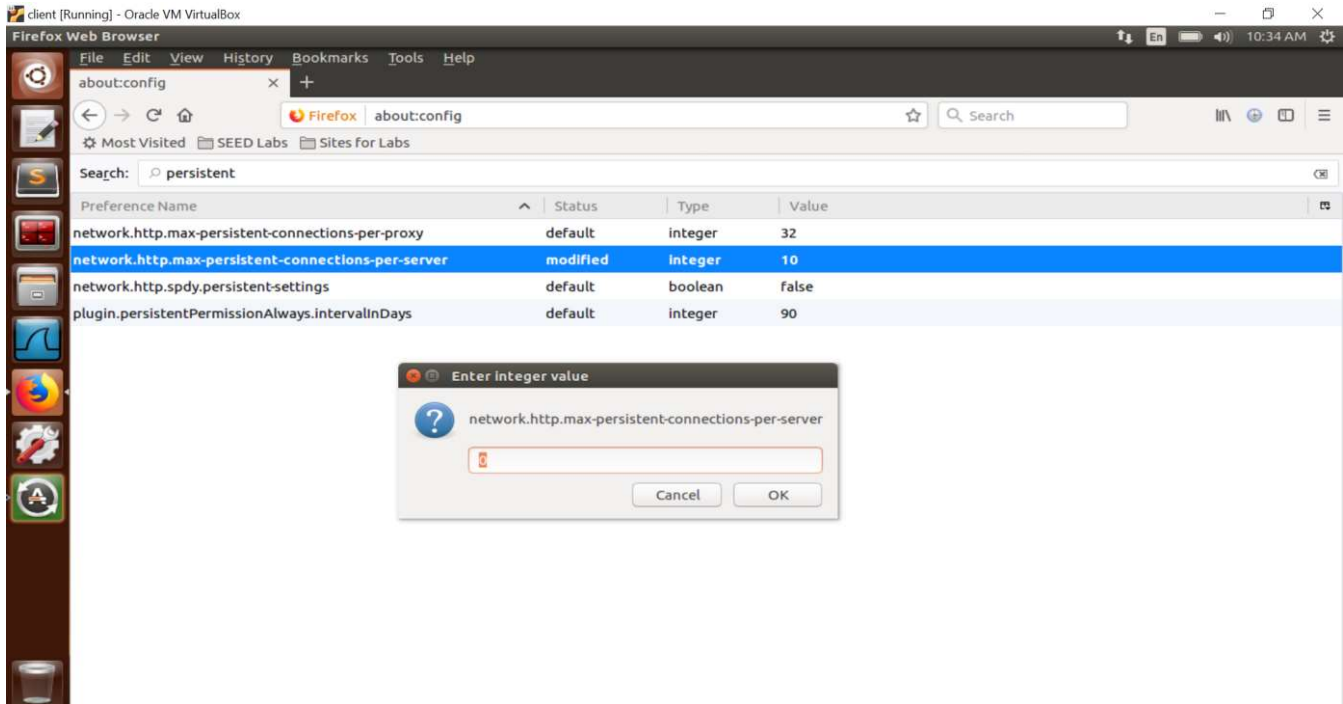
```
$sudo ip addr add 172.16.10.2/24 dev enps0
```

```
$sudo ip addr
```

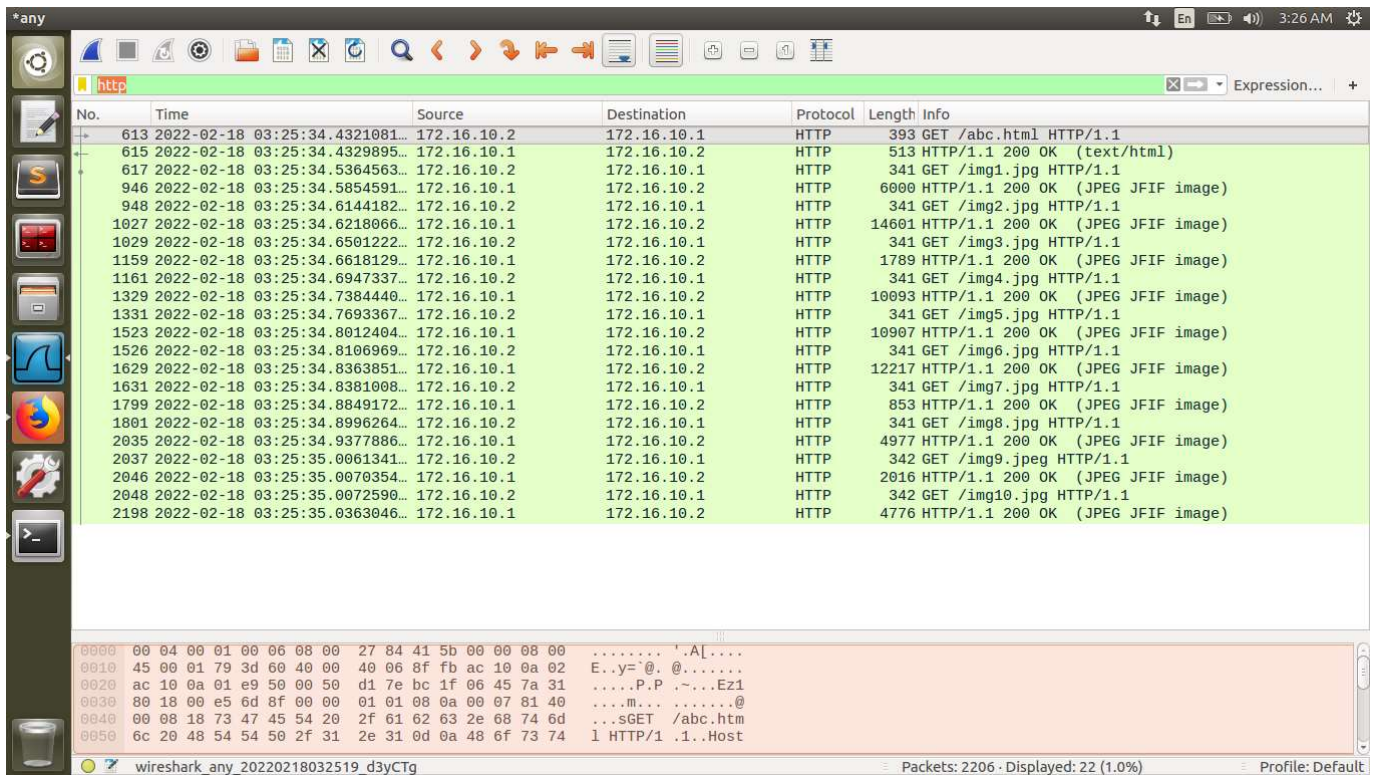


PART 1: NON-PERSISTENT CONNECTION

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)

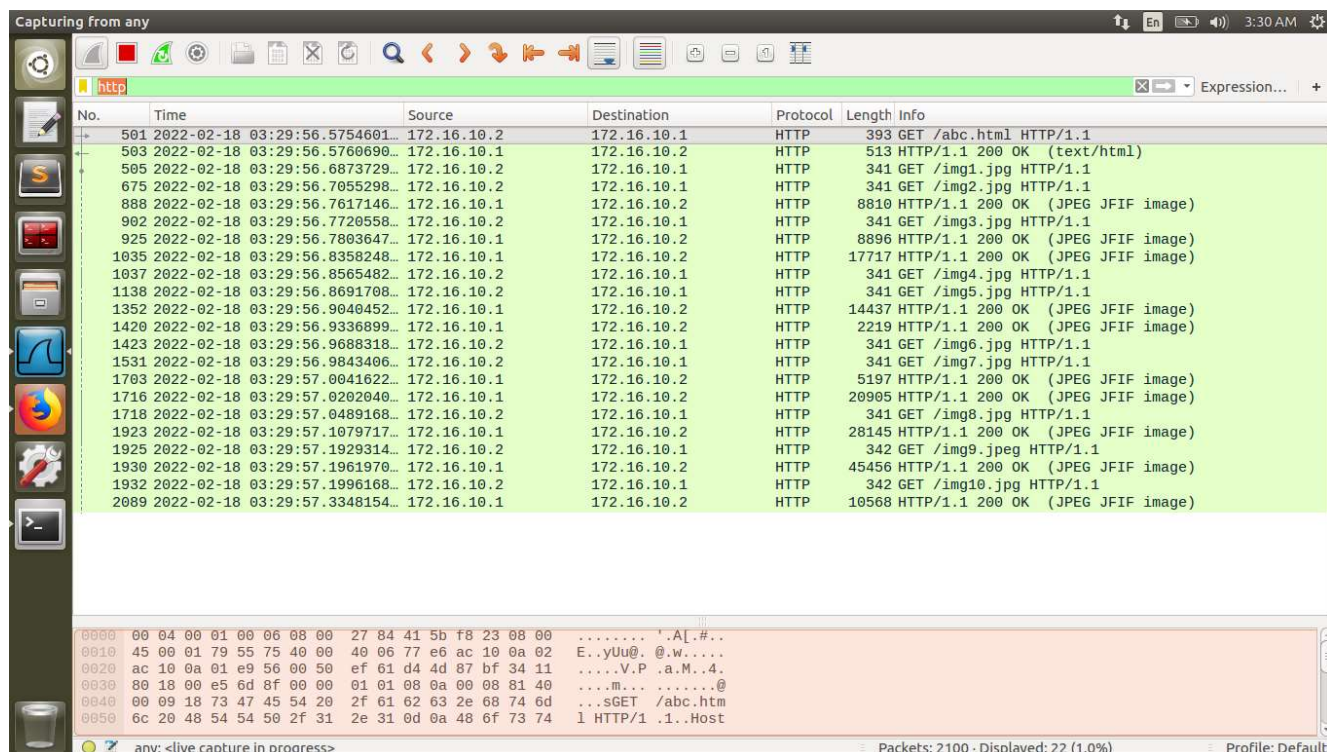


Here Time difference is = 35.0363046 – 34.4321081 = 0.6041965

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.



The image shows a Wireshark packet capture of an HTTP session. The top pane displays a list of 20 packets (No. 501 to 2089) with columns for Time, Source, Destination, Protocol, and Length. The packets are filtered by 'http'. The middle pane shows the details of the selected packet (No. 2089), which is an HTTP GET request for '/abc.html'. The bottom pane shows the raw packet data in hexadecimal and ASCII. The status bar at the bottom indicates 'any: <live capture in progress>', 'Packets: 2100 · Displayed: 22 (1.0%)', and 'Profile: Default'.

No.	Time	Source	Destination	Protocol	Length	Info
501	2022-02-18 03:29:56.5754601...	172.16.10.2	172.16.10.1	HTTP	393	GET /abc.html HTTP/1.1
503	2022-02-18 03:29:56.5760699...	172.16.10.1	172.16.10.2	HTTP	513	HTTP/1.1 200 OK (text/html)
505	2022-02-18 03:29:56.6873729...	172.16.10.2	172.16.10.1	HTTP	341	GET /img1.jpg HTTP/1.1
675	2022-02-18 03:29:56.7055298...	172.16.10.2	172.16.10.1	HTTP	341	GET /img2.jpg HTTP/1.1
888	2022-02-18 03:29:56.7617146...	172.16.10.1	172.16.10.2	HTTP	8810	HTTP/1.1 200 OK (JPEG JFIF image)
902	2022-02-18 03:29:56.7720558...	172.16.10.2	172.16.10.1	HTTP	341	GET /img3.jpg HTTP/1.1
925	2022-02-18 03:29:56.7803647...	172.16.10.1	172.16.10.2	HTTP	8896	HTTP/1.1 200 OK (JPEG JFIF image)
1035	2022-02-18 03:29:56.8358248...	172.16.10.1	172.16.10.2	HTTP	17717	HTTP/1.1 200 OK (JPEG JFIF image)
1037	2022-02-18 03:29:56.8565482...	172.16.10.2	172.16.10.1	HTTP	341	GET /img4.jpg HTTP/1.1
1138	2022-02-18 03:29:56.8691708...	172.16.10.2	172.16.10.1	HTTP	341	GET /img5.jpg HTTP/1.1
1352	2022-02-18 03:29:56.9040452...	172.16.10.1	172.16.10.2	HTTP	14437	HTTP/1.1 200 OK (JPEG JFIF image)
1420	2022-02-18 03:29:56.9336899...	172.16.10.1	172.16.10.2	HTTP	2219	HTTP/1.1 200 OK (JPEG JFIF image)
1423	2022-02-18 03:29:56.9686318...	172.16.10.2	172.16.10.1	HTTP	341	GET /img6.jpg HTTP/1.1
1531	2022-02-18 03:29:56.9843406...	172.16.10.2	172.16.10.1	HTTP	341	GET /img7.jpg HTTP/1.1
1703	2022-02-18 03:29:57.0041622...	172.16.10.1	172.16.10.2	HTTP	5197	HTTP/1.1 200 OK (JPEG JFIF image)
1716	2022-02-18 03:29:57.0202040...	172.16.10.1	172.16.10.2	HTTP	20905	HTTP/1.1 200 OK (JPEG JFIF image)
1718	2022-02-18 03:29:57.0489168...	172.16.10.2	172.16.10.1	HTTP	341	GET /img8.jpg HTTP/1.1
1923	2022-02-18 03:29:57.1079717...	172.16.10.1	172.16.10.2	HTTP	28145	HTTP/1.1 200 OK (JPEG JFIF image)
1925	2022-02-18 03:29:57.1929314...	172.16.10.2	172.16.10.1	HTTP	342	GET /img9.jpeg HTTP/1.1
1930	2022-02-18 03:29:57.1961970...	172.16.10.1	172.16.10.2	HTTP	45456	HTTP/1.1 200 OK (JPEG JFIF image)
1932	2022-02-18 03:29:57.1996168...	172.16.10.2	172.16.10.1	HTTP	342	GET /img10.jpg HTTP/1.1
2089	2022-02-18 03:29:57.3348154...	172.16.10.1	172.16.10.2	HTTP	10568	HTTP/1.1 200 OK (JPEG JFIF image)

Here Time difference = $57.3348154 - 56.5754601 = 0.7593553$

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

Capturing from any

No.	Time	Source	Destination	Protocol	Length	Info
549	2022-02-18 03:38:44.7023291...	172.16.10.2	172.16.10.1	HTTP	393	GET /abc.html HTTP/1.1
551	2022-02-18 03:38:44.7030002...	172.16.10.1	172.16.10.2	HTTP	513	HTTP/1.1 200 OK (text/html)
553	2022-02-18 03:38:44.8262961...	172.16.10.2	172.16.10.1	HTTP	341	GET /img1.jpg HTTP/1.1
764	2022-02-18 03:38:44.8411044...	172.16.10.2	172.16.10.1	HTTP	341	GET /img4.jpg HTTP/1.1
765	2022-02-18 03:38:44.8411436...	172.16.10.2	172.16.10.1	HTTP	341	GET /img3.jpg HTTP/1.1
766	2022-02-18 03:38:44.8411663...	172.16.10.2	172.16.10.1	HTTP	341	GET /img2.jpg HTTP/1.1
1189	2022-02-18 03:38:44.9427942...	172.16.10.1	172.16.10.2	HTTP	15290	HTTP/1.1 200 OK (JPEG JFIF image)
1213	2022-02-18 03:38:44.9640146...	172.16.10.1	172.16.10.2	HTTP	18784	HTTP/1.1 200 OK (JPEG JFIF image)
1402	2022-02-18 03:38:45.0201362...	172.16.10.1	172.16.10.2	HTTP	15886	HTTP/1.1 200 OK (JPEG JFIF image)
1434	2022-02-18 03:38:45.0376004...	172.16.10.1	172.16.10.2	HTTP	13374	HTTP/1.1 200 OK (JPEG JFIF image)
1436	2022-02-18 03:38:45.0412167...	172.16.10.2	172.16.10.1	HTTP	341	GET /img5.jpg HTTP/1.1
1541	2022-02-18 03:38:45.0540407...	172.16.10.2	172.16.10.1	HTTP	341	GET /img6.jpg HTTP/1.1
1640	2022-02-18 03:38:45.0790036...	172.16.10.2	172.16.10.1	HTTP	341	GET /img7.jpg HTTP/1.1
1754	2022-02-18 03:38:45.1124734...	172.16.10.2	172.16.10.1	HTTP	341	GET /img8.jpg HTTP/1.1
1938	2022-02-18 03:38:45.1293418...	172.16.10.1	172.16.10.2	HTTP	16699	HTTP/1.1 200 OK (JPEG JFIF image)
1955	2022-02-18 03:38:45.1597565...	172.16.10.1	172.16.10.2	HTTP	32489	HTTP/1.1 200 OK (JPEG JFIF image)
1990	2022-02-18 03:38:45.1868960...	172.16.10.2	172.16.10.1	HTTP	342	GET /img9.jpeg HTTP/1.1
1998	2022-02-18 03:38:45.1877815...	172.16.10.1	172.16.10.2	HTTP	10704	HTTP/1.1 200 OK (JPEG JFIF image)
2000	2022-02-18 03:38:45.1880263...	172.16.10.2	172.16.10.1	HTTP	342	GET /img10.jpg HTTP/1.1
2163	2022-02-18 03:38:45.2361306...	172.16.10.1	172.16.10.2	HTTP	3749	HTTP/1.1 200 OK (JPEG JFIF image)
2278	2022-02-18 03:38:45.2702418...	172.16.10.1	172.16.10.2	HTTP	6425	HTTP/1.1 200 OK (JPEG JFIF image)
2320	2022-02-18 03:38:45.2976570...	172.16.10.1	172.16.10.2	HTTP	9120	HTTP/1.1 200 OK (JPEG JFIF image)

any: <live capture in progress> Packets: 2689 - Displayed: 22 (0.8%) Profile: Default

Here Time difference = $45.2976570 - 44.7023291 = 0.5953279$

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.

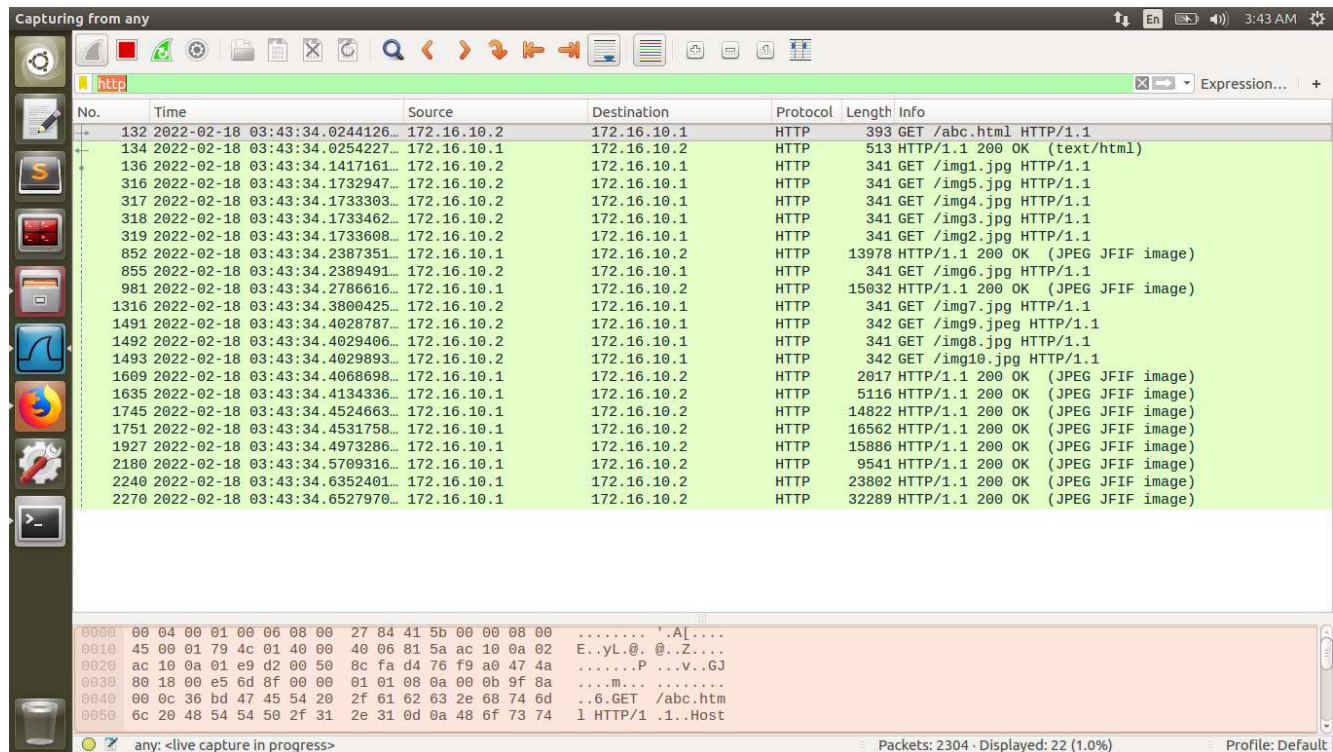
Capturing from any

No.	Time	Source	Destination	Protocol	Length	Info
549	2022-02-18 03:40:45.9774196...	172.16.10.2	172.16.10.1	HTTP	393	GET /abc.html HTTP/1.1
551	2022-02-18 03:40:45.9779539...	172.16.10.1	172.16.10.2	HTTP	513	HTTP/1.1 200 OK (text/html)
553	2022-02-18 03:40:46.0850775...	172.16.10.2	172.16.10.1	HTTP	341	GET /img1.jpg HTTP/1.1
601	2022-02-18 03:40:46.0878571...	172.16.10.2	172.16.10.1	HTTP	341	GET /img2.jpg HTTP/1.1
832	2022-02-18 03:40:46.1198994...	172.16.10.1	172.16.10.2	HTTP	5914	HTTP/1.1 200 OK (JPEG JFIF image)
845	2022-02-18 03:40:46.1241536...	172.16.10.2	172.16.10.1	HTTP	341	GET /img4.jpg HTTP/1.1
846	2022-02-18 03:40:46.1241861...	172.16.10.2	172.16.10.1	HTTP	341	GET /img3.jpg HTTP/1.1
957	2022-02-18 03:40:46.1419585...	172.16.10.2	172.16.10.1	HTTP	341	GET /img6.jpg HTTP/1.1
958	2022-02-18 03:40:46.1421029...	172.16.10.2	172.16.10.1	HTTP	341	GET /img5.jpg HTTP/1.1
1194	2022-02-18 03:40:46.1996490...	172.16.10.1	172.16.10.2	HTTP	341	GET /img7.jpg HTTP/1.1
1765	2022-02-18 03:40:46.3119108...	172.16.10.1	172.16.10.2	HTTP	7582	HTTP/1.1 200 OK (JPEG JFIF image)
1778	2022-02-18 03:40:46.3176391...	172.16.10.1	172.16.10.2	HTTP	3530	HTTP/1.1 200 OK (JPEG JFIF image)
1851	2022-02-18 03:40:46.3397944...	172.16.10.1	172.16.10.2	HTTP	3104	HTTP/1.1 200 OK (JPEG JFIF image)
1871	2022-02-18 03:40:46.3553667...	172.16.10.2	172.16.10.1	HTTP	341	GET /img8.jpg HTTP/1.1
1892	2022-02-18 03:40:46.3574989...	172.16.10.2	172.16.10.1	HTTP	342	GET /img9.jpeg HTTP/1.1
1917	2022-02-18 03:40:46.3586256...	172.16.10.1	172.16.10.2	HTTP	10704	HTTP/1.1 200 OK (JPEG JFIF image)
1954	2022-02-18 03:40:46.3724750...	172.16.10.1	172.16.10.2	HTTP	853	HTTP/1.1 200 OK (JPEG JFIF image)
2009	2022-02-18 03:40:46.3905262...	172.16.10.2	172.16.10.1	HTTP	342	GET /img10.jpg HTTP/1.1
2392	2022-02-18 03:40:46.5338267...	172.16.10.1	172.16.10.2	HTTP	5116	HTTP/1.1 200 OK (JPEG JFIF image)
2555	2022-02-18 03:40:46.6037300...	172.16.10.1	172.16.10.2	HTTP	633	HTTP/1.1 200 OK (JPEG JFIF image)
2564	2022-02-18 03:40:46.6241859...	172.16.10.1	172.16.10.2	HTTP	4302	HTTP/1.1 200 OK (JPEG JFIF image)
2591	2022-02-18 03:40:46.6329216...	172.16.10.1	172.16.10.2	HTTP	4776	HTTP/1.1 200 OK (JPEG JFIF image)

any: <live capture in progress> Packets: 2867 - Displayed: 22 (0.8%) Profile: Default

Here Time difference = $46.6329216 - 45.9774196 = 0.6555020$

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



Here Time difference = 34.6527970– 34.0244126 = 0.6283844

Sl.no	Max-Persistent-Connection-per-server	Load Time
1	0	0.6041965
2	2	0.7593553
3	4	0.5953279
4	6	0.6555020
5	10	0.6283844

Conclusion Remarks : It can be observed the load time is minimum for the 4 -persistent-connection. Hence it can be considered as the optimum persistent connection.

Understand working of HTTP Headers

Authentication: Auth-Basic

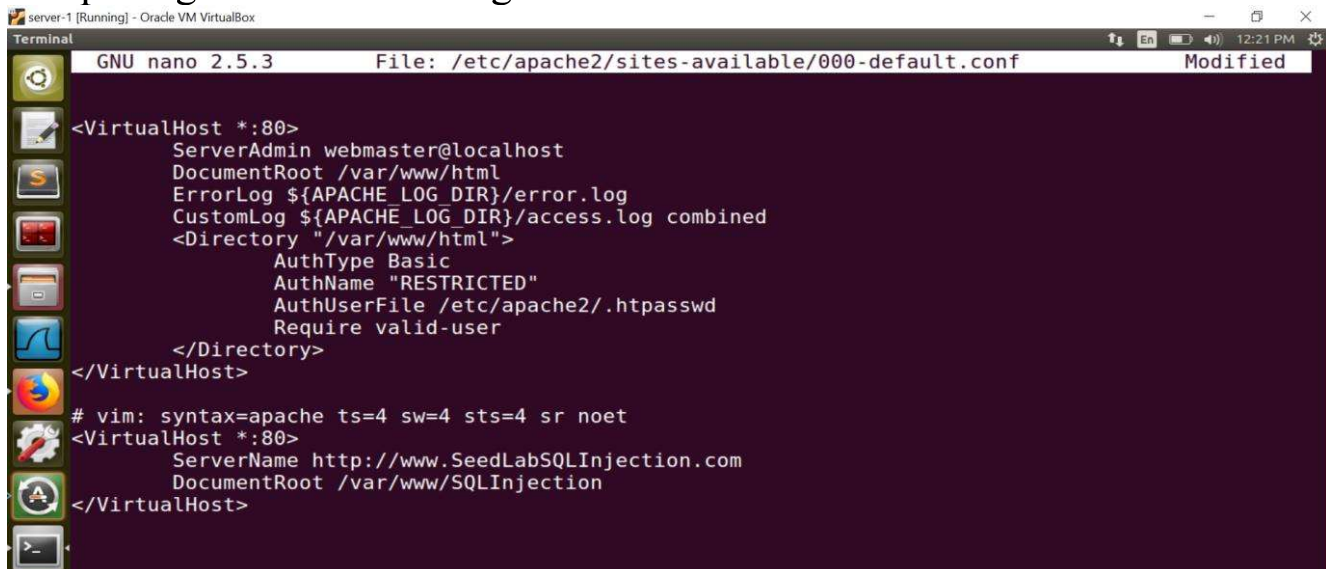
Steps of Execution (for Password Authentication)

1. Executing the below commands on the terminal.

- Provide username and password to set authentication
- View the Authentication

```
laxmikant@MyPC:~$ sudo htpasswd -c /etc/apache2/.htpasswd nobi
New password:
Re-type new password:
Adding password for user nobi
laxmikant@MyPC:~$ sudo cat /etc/apache2/.htpasswd
nob:$apr1$tpqHVSti$Av3SfWF9YCFqmUN08VNkT.
laxmikant@MyPC:~$
```

- Opening the file for setting authentication

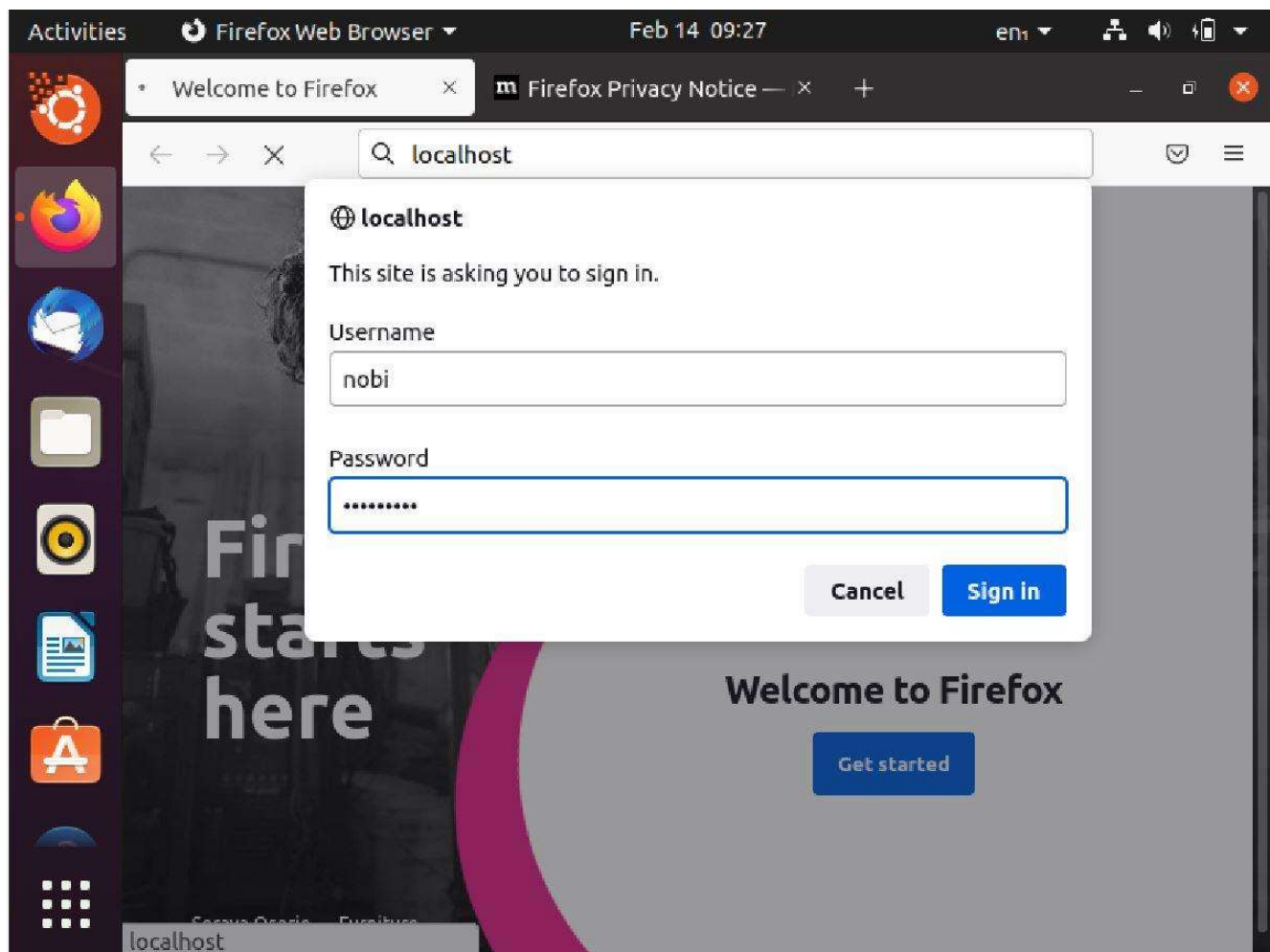


```
server-1 [Running] - Oracle VM VirtualBox
Terminal
GNU nano 2.5.3      File: /etc/apache2/sites-available/000-default.conf      Modified
<VirtualHost *:80>
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
    <Directory "/var/www/html">
        AuthType Basic
        AuthName "RESTRICTED"
        AuthUserFile /etc/apache2/.htpasswd
        Require valid-user
    </Directory>
</VirtualHost>
# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
<VirtualHost *:80>
    ServerName http://www.SeedLabSQLInjection.com
    DocumentRoot /var/www/SQLInjection
</VirtualHost>
```

→Restarting the Server

```
laxmikant@MyPC:~$ sudo service apache2 restart
[sudo] password for laxmikant:
laxmikant@MyPC:~$
```

→Access Localhost using Firefox Browser



→Wireshark Capture

The image shows the Wireshark network traffic capture interface. The top menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, and Help. The toolbar contains various icons for packet capture and analysis. The packet list pane on the left shows a list of captured packets, with packet 762 selected. The packet details pane on the right shows the structure of the selected packet, including Ethernet II, Internet Protocol Version 4, Transmission Control Protocol, and Hypertext Transfer Protocol. The packet bytes pane at the bottom shows the raw data of the selected packet.

No.	Time	Source	Destination	Protocol	Length	Info
757	172.383377880	127.0.0.1	127.0.0.1	TCP	68	80 → 44508 [ACK] Seq=...
758	172.383617455	127.0.0.1	127.0.0.1	HTTP	788	HTTP/1.1 401 Unauthc...
759	172.383665560	127.0.0.1	127.0.0.1	TCP	68	44508 → 80 [ACK] Seq=...
760	177.147326563	127.0.0.1	127.0.0.1	HTTP	533	GET / HTTP/1.1
761	177.147350869	127.0.0.1	127.0.0.1	TCP	68	80 → 44508 [ACK] Seq=...
762	177.148138951	127.0.0.1	127.0.0.1	HTTP	3544	HTTP/1.1 200 OK (te...
763	177.148191937	127.0.0.1	127.0.0.1	TCP	68	44508 → 80 [ACK] Seq=...
778	177.243979059	127.0.0.1	127.0.0.1	HTTP	475	GET /icons/ubuntu-10...
779	177.243992834	127.0.0.1	127.0.0.1	TCP	68	80 → 44508 [ACK] Seq=...

Frame 762: 3544 bytes on wire (28352 bits), 3544 bytes captured (28352 bits) on interface any, id ...
Linux cooked capture
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
Transmission Control Protocol, Src Port: 80, Dst Port: 44508, Seq: 721, Ack: 888, Len: 3476
Hypertext Transfer Protocol
Line-based text data: text/html (375 lines)

0020 7f 00 00 01 00 50 ad dc fe 13 b8 fa b6 35 2f 3eP...5/>
0030 80 18 02 00 0b bd 00 00 01 01 08 0a a3 b4 2a fd*..
0040 a3 b4 2a fc 48 54 54 50 2f 31 2e 31 20 32 30 30 ...*HTTP /1.1 200
0050 20 4f 4b 0d 0a 44 61 74 65 3a 20 4d 6f 6e 2c 20 OK Dat e: Mon,
0060 31 34 20 46 65 62 20 32 30 32 32 20 31 30 3a 31 14 Feb 2 022 10:1

Frame (3544 bytes) Uncompressed entity body (10918 bytes)

Transmission Control Protocol (tcp), 32 byte(s) Packets: 1403 · Displayed: 22 (1.6%) Profile: Default

→Follow TCP Stream using HTTP message

Activities

Wireshark

Feb 14 10:20

en

Wireshark - Follow TCP Stream (tcp.stream eq 19) - any

GET / HTTP/1.1

Host: localhost

User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8

Accept-Language: en-US,en;q=0.5

Accept-Encoding: gzip, deflate

Connection: keep-alive

Upgrade-Insecure-Requests: 1

Sec-Fetch-Dest: document

Sec-Fetch-Mode: navigate

Sec-Fetch-Site: none

Sec-Fetch-User: ?1

Authorization: Basic bm9iaTpsYXhtaWthbnQ=

HTTP/1.1 200 OK

Date: Mon, 14 Feb 2022 10:12:40 GMT

Server: Apache/2.4.41 (Ubuntu)

Last-Modified: Mon, 14 Feb 2022 09:08:34 GMT

ETag: "2aa6-5d7f6c2dba625-gzip"

Accept-Ranges: bytes

Vary: Accept-Encoding

Content-Encoding: gzip

Content-Length: 3138

Keep-Alive: timeout=5, max=99

Connection: Keep-Alive

Content-Type: text/html

.....Z.s.6.....U..\$'....."(&.c....\$......"!.....c....d5...~..H.%..

5...H.....0.....n...../..7...=.....d.....

4 client pkt(s), 4 server pkt(s), 7 turn(s).

Entire conversation (9997 bytes)

Show and save data as ASCII

Stream 19

Find:

Find Next

Filter Out This Stream

Print

Save as...

Back

Close

Help

→Decryption Base64 Encryption

Decode from Base64 format

Simply enter your data then push the decode button.

bm9iaTpsYXhtaWthbnQ=

i For encoded binaries (like images, documents, etc.) use the file upload form a little further down on this page.

UTF-8

▼

Source character set.

☐ Decode each line separately (useful for when you have multiple entries).

☒ Live mode OFF

Decodes in real-time as you type or paste (supports only the UTF-8 character set).

< DECODE >

Decodes your data into the area below.

nobi:laxmikant

Steps of Execution (Cookie Setting)

→PHP File

server-1 [Running] - Oracle VM VirtualBox



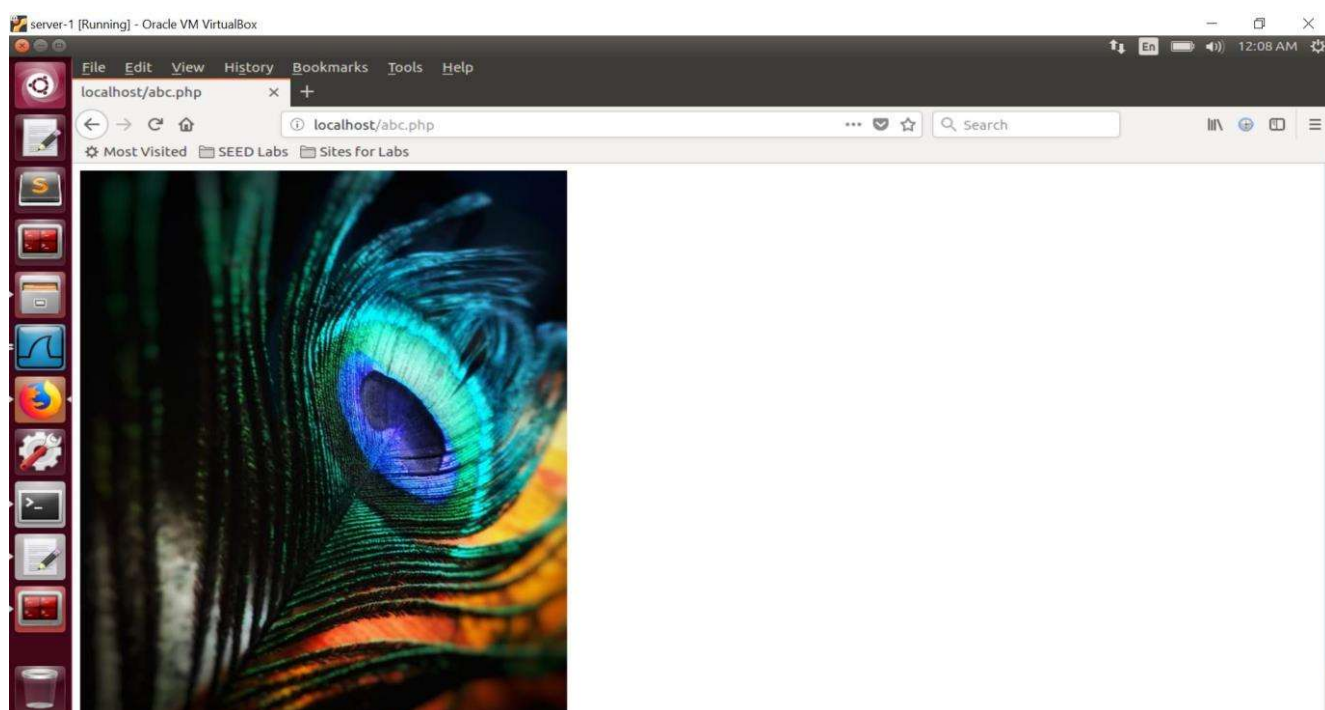
Open ▼

abc.php

```
<html>
<?php
setcookie("laxmikant","nobi",time()+123);
setcookie("PES","UNIVERSity");
?>

</html>
```

→ Access the Php file using Firefox Browser



→ Wireshark Capture And follow TCP Stream

The screenshot shows the Wireshark interface with a packet capture from 'any' on interface 'eth0'. The packet list shows various TCP and HTTP traffic. Packet 854 is selected, showing an HTTP GET request for '/abc.php'. The packet details pane shows the following information:

- Frame 854: 390 bytes on wire (3120 bits), 390 bytes captured (3120 bits) on interface 0
- Linux cooked capture
- Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
- Transmission Control Protocol, Src Port: 39368, Dst Port: 80, Seq: 415351637, Ack: 2005573698, Len: 322
- Hypertext Transfer Protocol

The screenshot shows the 'Follow TCP Stream' window for the selected TCP stream (tcp.stream eq 25). The window displays the raw data of the stream, including the HTTP request and response. The request is a GET request for '/abc.php' with various headers. The response is an HTTP 200 OK with a 'Set-Cookie' header and a 'Content-Type' of 'text/html; charset=UTF-8'. The raw data is shown in ASCII format.

```
GET /abc.php HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1

HTTP/1.1 200 OK
Date: Thu, 17 Feb 2022 05:05:06 GMT
Server: Apache/2.4.18 (Ubuntu)
Set-Cookie: laxmikant=nobi; expires=Thu, 17-Feb-2022 05:07:09 GMT; Max-Age=123
Set-Cookie: PES=UNIVERSiTy
Content-Length: 46
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

<html>

</html>
GET /1.jpeg HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://localhost/abc.php
Cookie: laxmikant=nobi; PES=UNIVERSiTy
Connection: keep-alive

HTTP/1.1 200 OK
Date: Thu, 17 Feb 2022 05:05:06 GMT
```

Packet 855. 5 client pkts, 6 server pkts, 7 turns. Click to select.

Entire conversation (47 kB) Show and save data as ASCII Stream 25

Find: Find Next

Help Filter Out This Stream Print Save as... Back Close

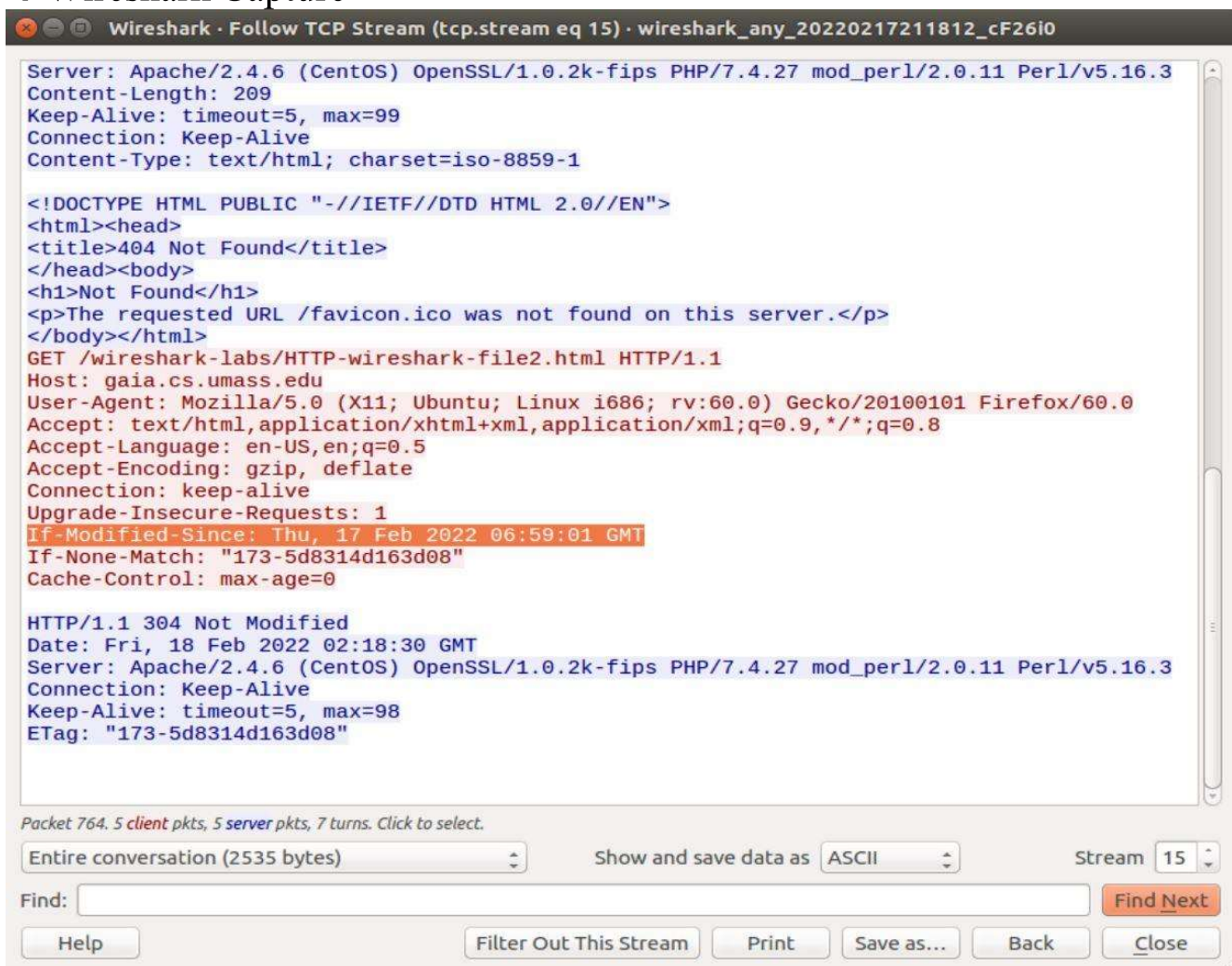
Conditional Get: If-Modified-Since

→ Open the Browser And follow the link

Enter the following URL into your browser <http://gaia.cs.umass.edu/wireshark->

→ Before following the link clear the cache.

→ Wireshark Capture



Question and Answers :

1) 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?

Ans : No, there is no “IF-MODIFIED-SINCE” line in the first HTTP GET.

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

Ans : Yes, the server explicitly returns the contents of the file. we can confirm from header line-based text data.

3) Do you see an “IF-MODIFIED-SINCE:” line in the second HTTP GET request from your browser to the server? If so, what information follows the “IF-MODIFIED-SINCE:” header?

Ans : Yes, there is an “IF-MODIFIED-SINCE:” line in the second HTTP GET request. It contains the time at which 1st response send by a server to the client.

4) 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.

Ans : 304 is the status code and Not modified is the phrase returned from the server in response. The server doesn't explicitly return the contents of the file so we can't find a line-based text data line in the header and the file is loaded from the cache.