Computer Networks Lab

Week 2

Vishal R

PES1UG19CS571

I Section

February 7, 2021

Understanding Persistent and Non-Persistent HTTP Connections

---

1 Apache Server Configurations

Creation of client-server architecture required to setup two Virtual Machines, were one acts as a client and another acts as a server.

To create the server, `Apache` was installed and configured on one of the Virtual Machines. This Virtual Machine will act as server.

1.1 Installing `Apache`.

`Apache` was installed by typing the command `$ sudo apt-get install apache2`. This will install `Apache` on the Virtual Machine and will automatically start the server for us.

We can check if the server has been successfully started or not by typing the command `$ sudo systemctl status apache2` or `$ sudo service apache2 status` in the terminal.

```
vishalr@ubuntu:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
     Active: active (running) since Sat 2021-02-06 20:59:28 PST; 1h 11min ago
       Docs: https://httpd.apache.org/docs/2.4/
    Process: 25581 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
   Main PID: 25585 (apache2)
      Tasks: 55 (limit: 2882)
     Memory: 5.8M
     CGroup: /system.slice/apache2.service
             ├─25585 /usr/sbin/apache2 -k start
             ├─25586 /usr/sbin/apache2 -k start
             └─25587 /usr/sbin/apache2 -k start

Feb 06 20:59:28 ubuntu systemd[1]: Starting The Apache HTTP Server...
Feb 06 20:59:28 ubuntu systemd[1]: Started The Apache HTTP Server.
vishalr@ubuntu:~$
```

Figure 1: Shows the current status of `apache`.

1.2 Additional `Apache` server configurations.

`apache2.conf` in `/etc/apache2/` was opened using `$ sudo gedit /etc/apache2/apache2.conf`.

Note : This file can also be opened using `$ sudo nano /etc/apache2/apache2.conf`

Following changes were made to setup our server for this lab..

- `KeepAlive` option was set. (i.e value was made **ON**).
- `MaximumKeepAliveRequests` was set to 2.

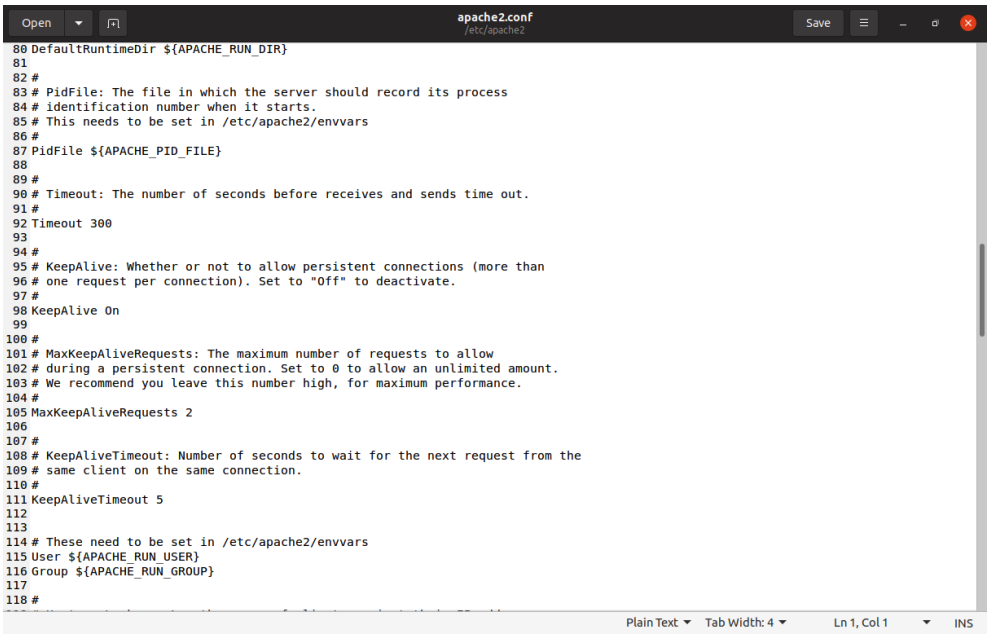

Figure 2: Shows the changes made in `apache2.conf` file

1.3 Setting the IP Addresses of both client and server Virtual Machines.

A new IP Address was set on both client and server Virtual Machines. These IP Addresses were set manually by going into 'edit connections' and manually entering IP Address for client and server. This was done inorder to avoid any fluctuations that occur while assigning IP address.

Note : IP Address can also be set by using the command `$ sudo ip addr add ip-address dev interface-name`.

1.3.1 Assigning IP Address in Server Machine.

IP address assigned to the server is `10.0.9.71/24` as per my SRN.

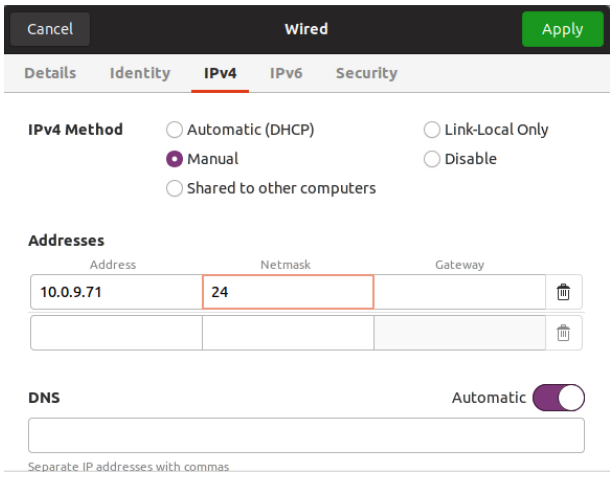

Figure 3: Manually setting IP address on server machine.

We can verify this by typing the command `$ sudo ip addr show` in the terminal.

```
vishalr@ubuntu:~$ ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    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: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:a9:22:d8 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 10.0.9.71/24 brd 10.0.9.255 scope global noprefixroute ens33
       valid_lft forever preferred_lft forever
    inet6 fd15:4ba5:5a2b:1008:661f:b710:2b75:144a/64 scope global temporary dynamic
       valid_lft 592223sec preferred_lft 73777sec
    inet6 fd15:4ba5:5a2b:1008:9c14:3ddf:2966:f3ef/64 scope global dynamic mngtmpaddr noprefixroute
       valid_lft 2591951sec preferred_lft 604751sec
    inet6 fe80::db93:c4b:f387:18f8/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
vishalr@ubuntu:~$
```

Figure 4: Verifying if IP Address has been set successfully on server machine.

1.3.2 Assigning IP Address in Client Machine.

IP address assigned to the server is `10.0.9.38/24` as per my serial number.

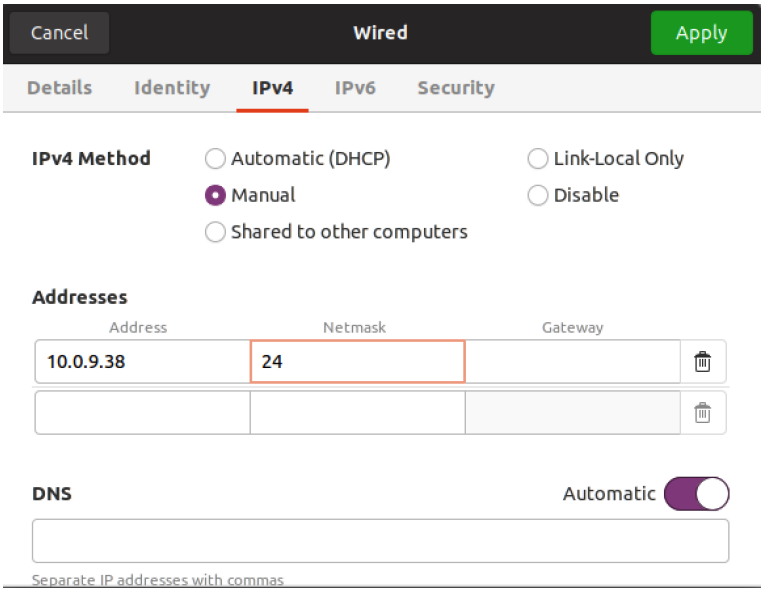

Figure 5: Manually setting IP address on client machine.

We can verify this by typing the command `$ sudo ip addr show` in the terminal.

```
vishal@ubuntu:~$ ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    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: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:8f:95:84 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 10.0.9.38/24 brd 10.0.9.255 scope global noprefixroute ens33
       valid_lft forever preferred_lft forever
    inet6 fd15:4ba5:5a2b:1008:5393:ccdf:7898:73dc/64 scope global temporary dynamic
       valid_lft 592028sec preferred_lft 73230sec
    inet6 fd15:4ba5:5a2b:1008:f2b1:28a:88ef:1731/64 scope global dynamic mngtmpaddr noprefixroute
       valid_lft 2591938sec preferred_lft 604738sec
    inet6 fe80::b2a9:ab36:a543:56a4/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
vishal@ubuntu:~$
```

Figure 6: Verifying if IP Address has been set successfully on client machine.

1.4 Creating a web page in server machine

A new `.html` file was created in the directory `/var/www/html/` and the following contents were added into it.

Open a.html /var/www/html

```
1  <html>
2  <body>
3
4  <img src="img1.jpg" alt="image"/>
5  <img src="img2.jpg" alt="image"/>
6  <img src="img3.jpg" alt="image"/>
7  <img src="img4.jpg" alt="image"/>
8  <img src="img5.jpg" alt="image"/>
9  <img src="img6.jpg" alt="image"/>
10 <img src="img7.jpg" alt="image"/>
11 <img src="img8.jpg" alt="image"/>
12 <img src="img9.jpg" alt="image"/>
13 <img src="img10.jpg" alt="image"/>
14
15 </body>
16 </html>
```

Figure 7: Contents of `a.html`.

1.5 Adding 10 images into the same directory.

10 images were downloaded from internet and were placed in the same directory were `a.html` is present. Images are of sizes 1.0 mb - 1.5 mb .

The images were renamed to (`img1.jpg, img2.jpg, img3.jpg, ..., img10.jpg`).

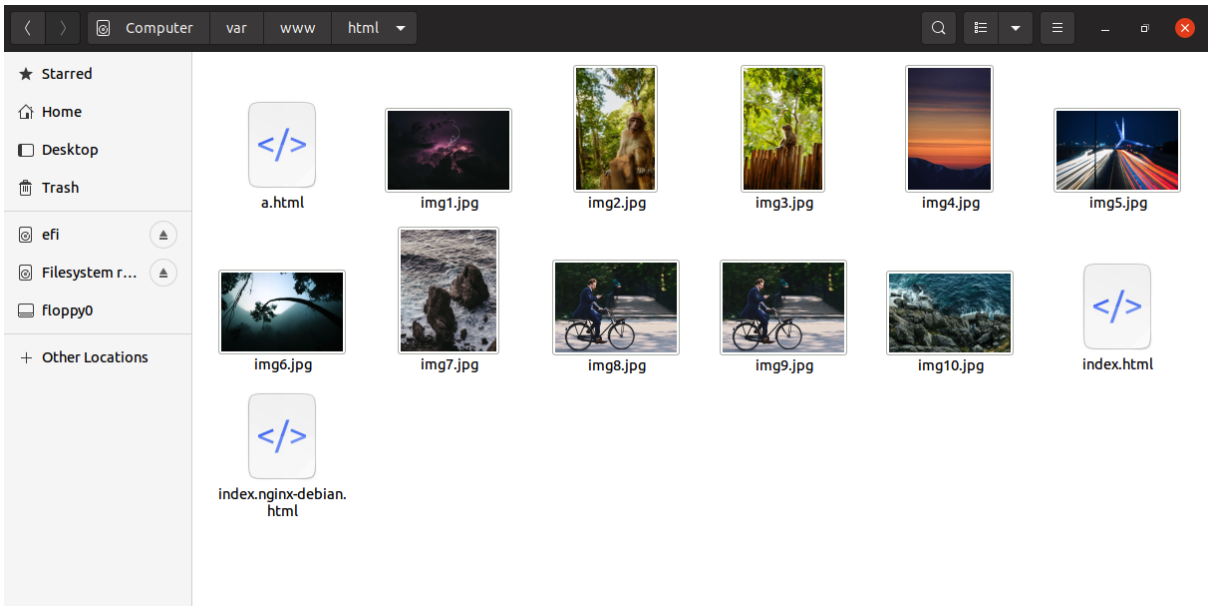

Figure 8: Contents of the folder `/var/www/html/`.

1.6 Testing connectivity between client and server

To test if client and server can communicate with each other, we can ping the server from the client using the command `ping 10.0.9.71`.

```
vishal@ubuntu:~$ ping 10.0.9.71
PING 10.0.9.71 (10.0.9.71) 56(84) bytes of data.
64 bytes from 10.0.9.71: icmp_seq=1 ttl=64 time=2.06 ms
64 bytes from 10.0.9.71: icmp_seq=2 ttl=64 time=1.12 ms
64 bytes from 10.0.9.71: icmp_seq=3 ttl=64 time=1.11 ms
64 bytes from 10.0.9.71: icmp_seq=4 ttl=64 time=1.11 ms
64 bytes from 10.0.9.71: icmp_seq=5 ttl=64 time=1.12 ms
64 bytes from 10.0.9.71: icmp_seq=6 ttl=64 time=1.12 ms
64 bytes from 10.0.9.71: icmp_seq=7 ttl=64 time=1.15 ms
^C
--- 10.0.9.71 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6014ms
rtt min/avg/max/mdev = 1.112/1.255/2.059/0.328 ms
vishal@ubuntu:~$
```

Figure 9: Checking connectivity between client and server using `ping`.

We can see that we are receiving packets from server hence we can conclude that both client and server can communicated with each other.

After all these steps, we are done with configuring the apache server and now we can start analysing persistent and non-persistent connections.

2 Non-Persistent HTTP Connection

A Non-persistent connection is a type of HTTP connection which is closed after the server sends the requested object to client. Hence, if the client request more than one object, the connection is opened and closed for each object that the client requests for.

2.1 Setting up Non-Persistent HTTP Connection

To setup a non-persistent connection, few configurations must be changed in the browser's config files on the client side.

For this experiment, I have used Firefox browser.

In firefox, we will type `about:config` in the search bar.

We will search for 'persistent' in the filter options. We then set the following options in the browser.

- `max-persistent-connections-per-server` was set to `0`.
- `persistent-settings` was set to `false`.

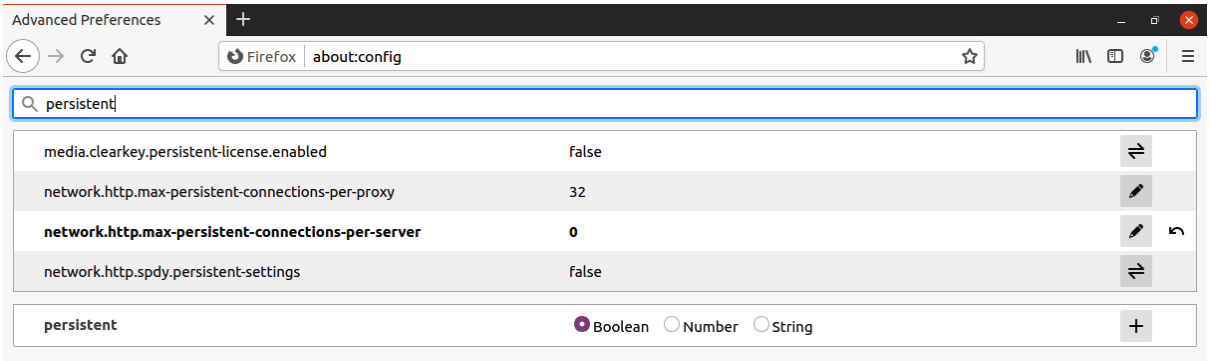

Figure 10: Configuring the browser for non-persistent connections.

Now, we can request for the `html` file we created in server from the client.

2.2 Make `GET` request from the Client to Server

We can request for the `a.html` file we created on server from the client by typing `http://10.0.9.71/a.html` in the browser search bar. This will load the web page from the server. Before making the GET request, we will open wireshark and monitor 'any' interface for HTTP Packets using the `http` filter option.

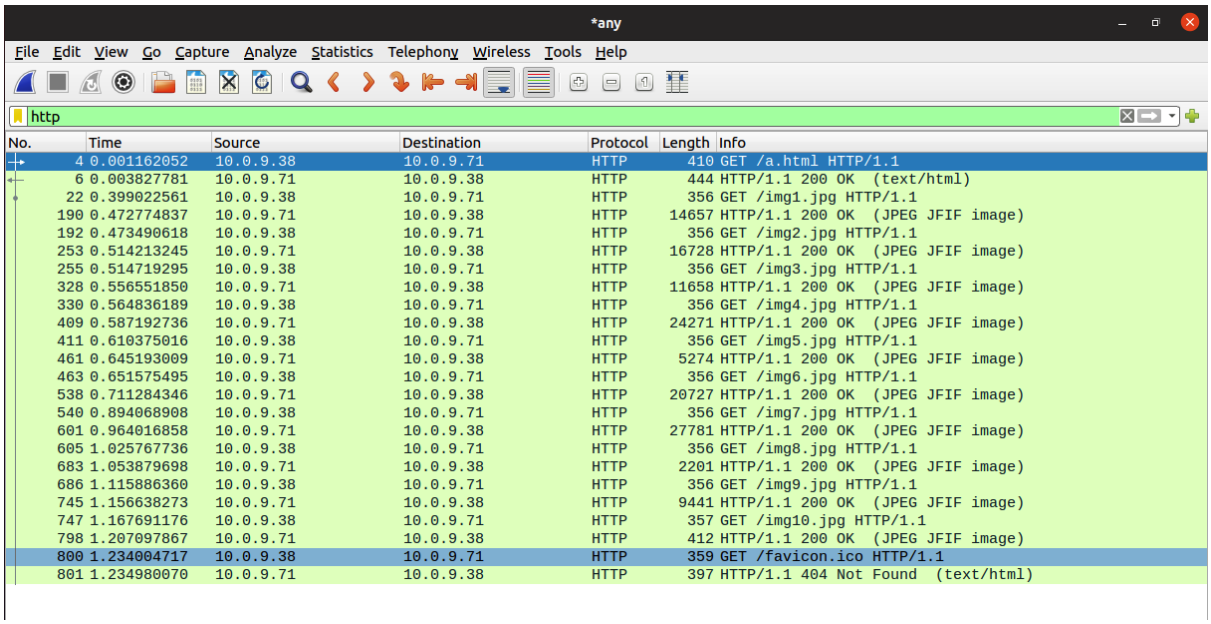

Figure 11: Packet Capture in Wireshark.

We can calculate the total time taken to load all the 10 images by taking the time difference between the two highlighted packets in the figure.

Note : The last packet highlighted is wrong. Please consider packet above it.

`Total Time : 1.207097867 - 0.001162052 = 1.205935815 s.`

3 Persistent HTTP Connections

In non-persistent connections, the server will close the connection after the requested object is sent to client. With Persistent connections, the server leaves the connection open and hence retrieving more objects from server need not require opening another connection.

3.1 Setting up Persistent HTTP Connections

To setup a persistent connection, few configurations must be changed in the browser's config files on the client side.

In firefox, we will type `about:config` in the search bar.

We will search for 'persistent' in the filter options. We then set the following options in the browser.

- `max-persistent-connections-per-server` was set to `<number>`.
- `persistent-settings` was set to `true`.

Here `<number>` can be any integer which denotes the maximum number of persistent connections for a server.

3.2 2 Persistent HTTP Connection

For 2 persistent HTTP connection, we will set `<number>` to 2 as shown below.

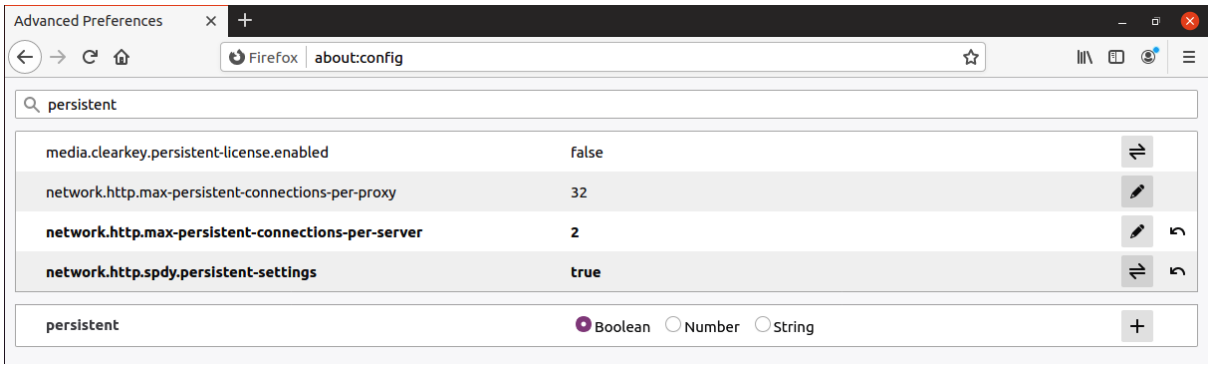

Figure 12: Configuring web browser for 2 persistent connections.

Again can request for the `a.html` file we created on server from the client by typing `http://10.0.9.71/a.html` in the browser search bar. This will load the web page from the server.

Before making the GET request, we will open wireshark and monitor 'any' interface for HTTP Packets using the `http` filter option.

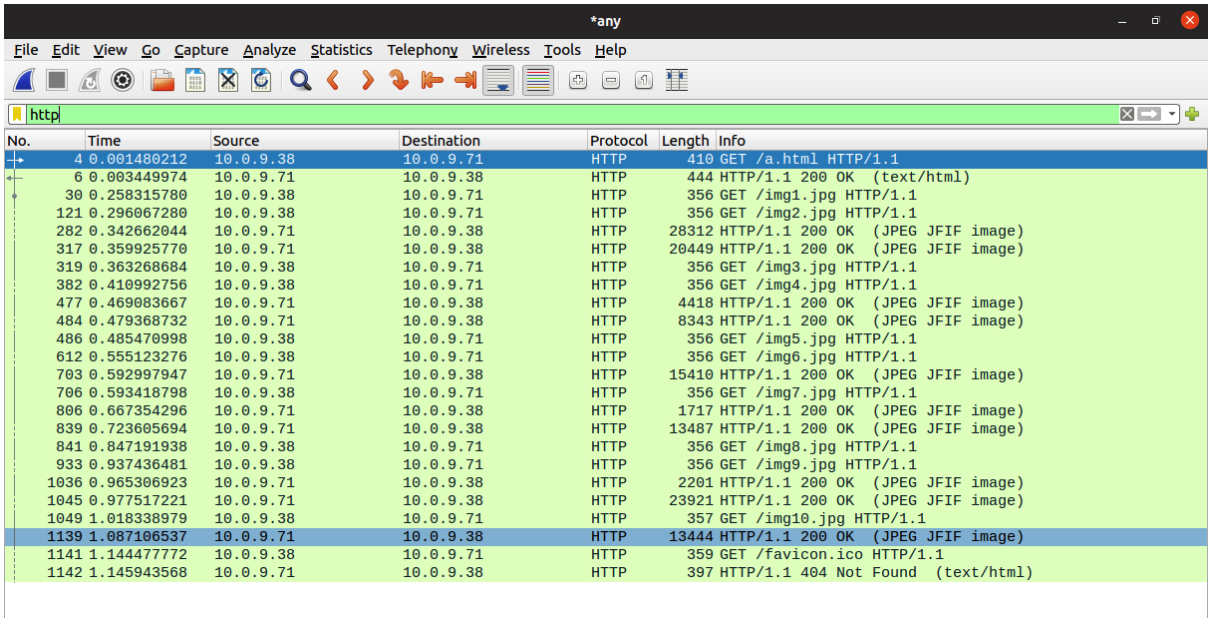

Figure 13: Packet capture in Wireshark.

We can calculate the total time taken to load all the 10 images by taking the time difference between the two highlighted packets in the figure.

`Total Time : 1.087106537 - 0.001480212 = 1.085626325 s.`

3.3 4 Persistent HTTP Connection

For 4 persistent HTTP connection, we will set `<number>` to `4` as shown below.

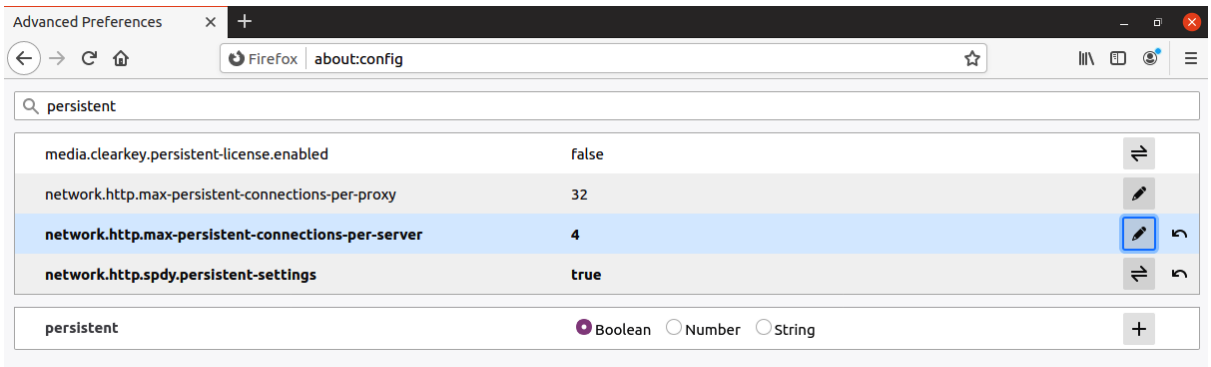

Figure 14: Configuring web browser for 4 persistent connections.

We will make GET request to same url again. Before making the GET request, we will open wireshark and monitor 'any' interface for HTTP Packets using the `http` filter option.

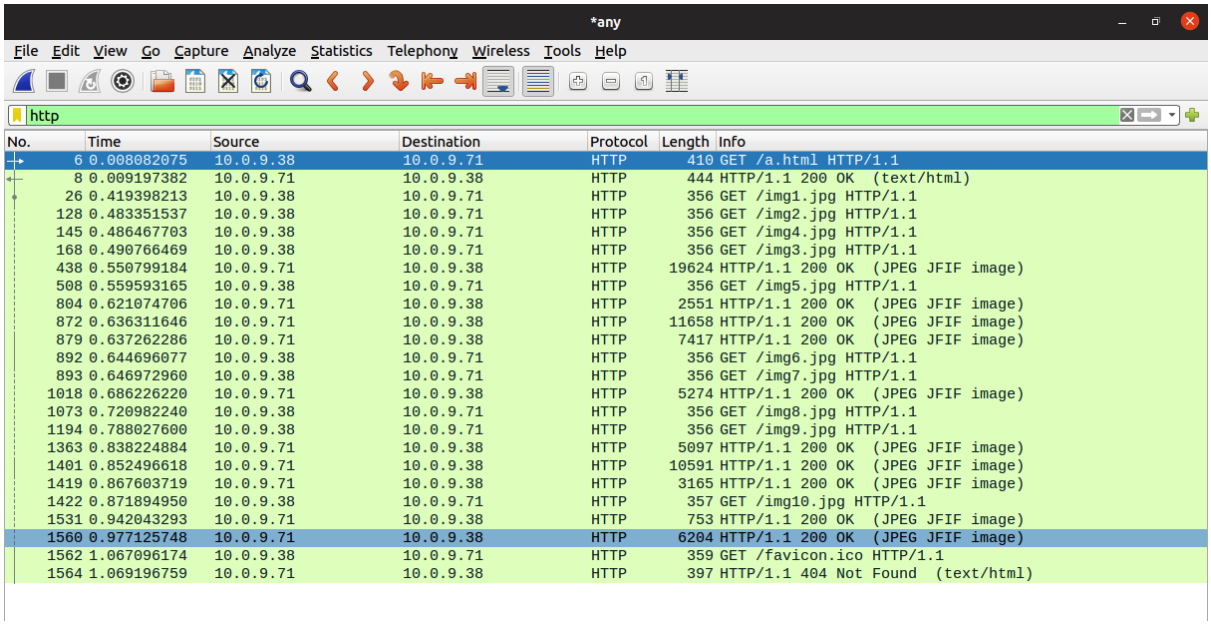

Figure 15: Packet capture in Wireshark.

We can calculate the total time taken to load all the 10 images by taking the time difference between the two highlighted packets in the figure.

`Total Time : 0.977125748 - 0.008082075 = 0.969043673 s.`

3.4 6 Persistent HTTP Connection

For 6 persistent HTTP connection, we will set `<number>` to `6` as shown below.

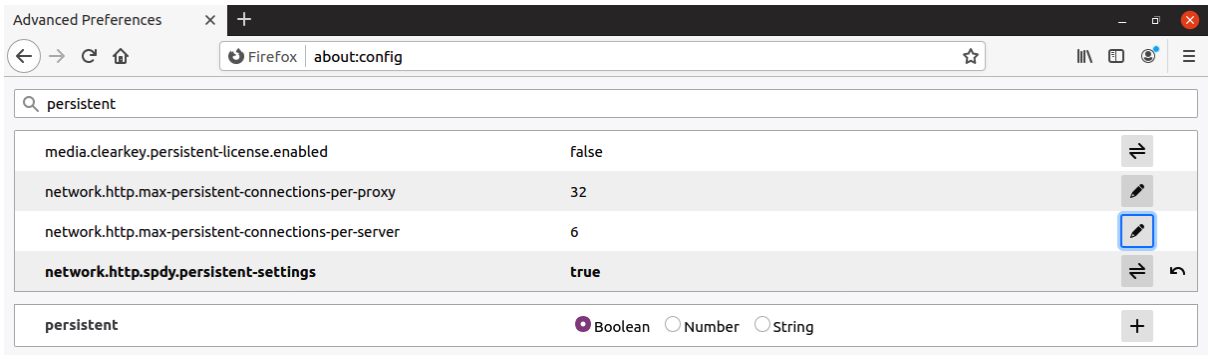

Figure 16: Configuring web browser for 6 persistent connections.

We will make GET request to same url again. Before making the GET request, we will open wireshark and monitor 'any' interface for HTTP Packets using the `http` filter option.

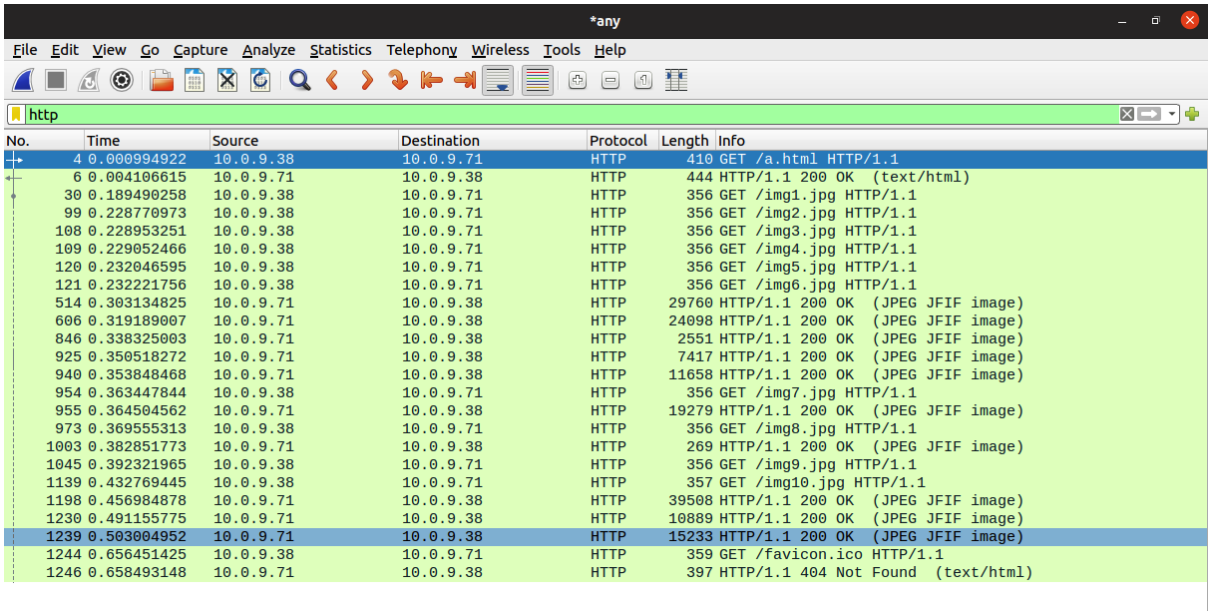

Figure 17: Packet capture in Wireshark.

We can calculate the total time taken to load all the 10 images by taking the time difference between the two highlighted packets in the figure.

`Total Time : 0.503004952 - 0.000994922 = 0.50201003 s.`

3.5 10 Persistent HTTP Connection

For 10 persistent HTTP connection, we will set `<number>` to `10` as shown below.

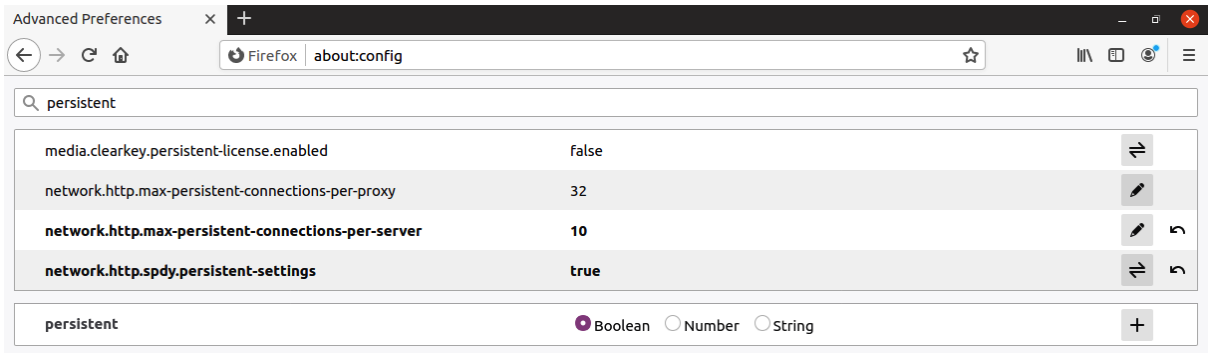

Figure 18: Configuring web browser for 10 persistent connections.

We will make GET request to same url again. Before making the GET request, we will open wireshark and monitor 'any' interface for HTTP Packets using the `http` filter option.

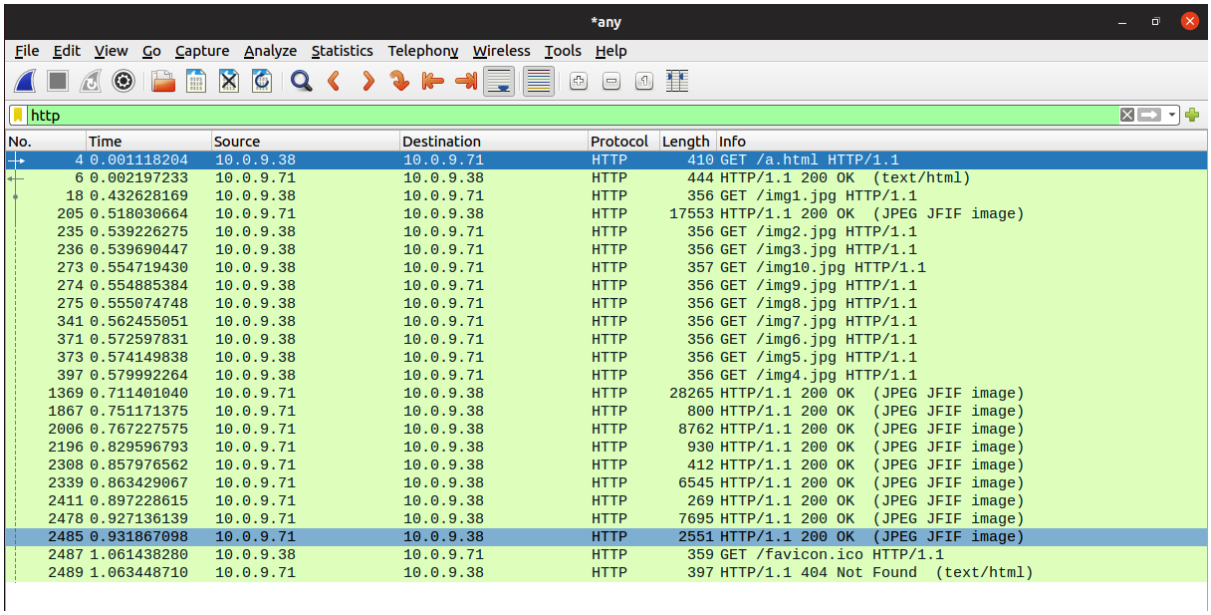

Figure 19: Packet capture in Wireshark.

We can calculate the total time taken to load all the 10 images by taking the time difference between the two highlighted packets in the figure.

`Total Time : 0.931867098 - 0.001118204 = 0.930748894 s.`

4 Observations

Total time calculated is the time difference between the first `GET` request made to server and the last response from the server. By default, time displayed by wireshark is in seconds. Hence we will use seconds itself. Total time here indicates the total load time, basically the amount of time required to load all resources in the web page.

We calculated the total time difference for Non-persistent connection and for 2,4,6,10 persistent connections.

Connection Type	**Total Time**
Non-Persistent	1.205935815 s
2 Persistent	1.085626325 s
4 Persistent	0.969043673 s
6 Persistent	0.50201003 s
10 Persistent	0.930748894 s

We can notice that when using a non-persistent connection, there is a high load time where as if when we switched over to persistent connection, we were able to reduce the load time. The main reason is that in non-persistent connection, the connection gets closed after a response from the server. Thus additional time is required to open a new connection and hence we see high load time in non-persistent connection.

Persistent connection have lower load time because the connection is left open and additional operations like pipelining and parallelism will be performed thus having higher throughput.

From the table above, we can note that the **optimal number of persistent connection is 6** as it has the least load time. When we increased to 10 persistent connections, load time increased and it increases more as we increase the number of persistent connections further. This is because of decrease in throughput of each connection.

By default, most web browsers use 6 persistent connections and increasing it further will not yield better results.