

CS536: Networking Lab 1

Name: Mansi Shinde

PUID: **0034784153**

Part A. Sol. Building Client Server over Socket Programming

- Running Client and Server Application on localhost.

Code files to refer - client.c, server.c

Steps to execute the code:

1. Creating executable file using gcc command
2. Running the executable file

Client Side:

1. gcc client.c -o ./client
2. ./client 127.0.0.1 3500 "Testing Message"

Server Side:

1. gcc server.c -o ./server
2. ./server 3500

Client Response:

```
(base) mansishinde@Mansi-MacBook-Air PartA %
(base) mansishinde@Mansi-MacBook-Air PartA % gcc client.c -o ./client
(base) mansishinde@Mansi-MacBook-Air PartA %
(base) mansishinde@Mansi-MacBook-Air PartA %
(base) mansishinde@Mansi-MacBook-Air PartA % ./client 127.0.0.1 3500 "Hello! this is Client talking from localhost"
Connection established with Server

Sending the message to the server.

Modified message received from server:
HELLO! THIS IS CLIENT TALKING FROM LOCALHOST
Closing the connection
(base) mansishinde@Mansi-MacBook-Air PartA %
```

Server response:

```
[(base) mansishinde@Mansi's-MacBook-Air PartA %
[(base) mansishinde@Mansi's-MacBook-Air PartA % gcc server.c -o ./server
[(base) mansishinde@Mansi's-MacBook-Air PartA %
[(base) mansishinde@Mansi's-MacBook-Air PartA %
[(base) mansishinde@Mansi's-MacBook-Air PartA %
[(base) mansishinde@Mansi's-MacBook-Air PartA % ./server 3500
The server is ready to receive

message-from-client : 127.0.0.1 , 64924

Message: Hello! this is Client talking from localhost

Modified Message: HELLO! THIS IS CLIENT TALKING FROM LOCALHOST

close-client : 127.0.0.1 , 64924
```

- Running Client and Server Application on different host machines

Client is running on amber10.cs.purdue.edu whose **IP address is 128.10.112.140** and
Server is running on amber20.cs.purdue.edu whose **IP address is 128.10.112.150**

Code files to refer - client.c, server.c

Steps to execute the code:

1. Creating executable file using gcc command
2. Running the executable file

Client Side:

1. gcc client.c -o ./client
2. ./client 128.10.112.150 9000 "Testing Message"

Server Side:

1. gcc server.c -o ./server
2. ./server 9000

Output:

```

mansi@amber10 ~ % ssh mshinde@amber10.cs.purdue.edu ~ 103x55
[amber10 84 $ gcc client.c -o ./client
[amber10 85 $ ./client 128.10.112.156 9000 "Checking with server on different host"
Connection established with Server
Sending the message to the server.
Modified message received from server:
CHECKING WITH SERVER ON DIFFERENT HOST
[amber10 85 $ ]]

mansi@amber20 ~ % ssh mshinde@amber20.cs.purdue.edu ~ 100x53
[...Documents/ComputerNetworking/lab1/PartC - zsh ... ...loads -- ssh mshinde@amber20.cs.purdue.edu
[amber20 62 $ gcc server.c -o ./server
[amber20 63 $ ./server 9000
The server is ready to receive
message-from-client : 128.10.112.140 , 34916
Message: Checking with server on different host
Modified Message: CHECKING WITH SERVER ON DIFFERENT HOST
[amber20 63 $ ]]

mansi@amber10 ~ % ifconfig
[amber10 53 $ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 128.10.112.148 netmask 255.255.255.0 broadcast 128.10.112.255
        inet6 fe80::a6bb:6dff:fe42:e6ef prefixlen 64 scopeid 0x20<link>
            ether a4:bb:6d:42:e6:ef txqueuelen 1000 (Ethernet)
            RX packets 15886475 bytes 6202465477 (6.2 GB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 57388815 bytes 78055323292 (78.0 GB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
            device interrupt 28 memory 0xd1300000-d1320000
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopeid 0x10<host>
            loop txqueuelen 1000 (Local Loopback)
            RX packets 5349370 bytes 73508571337 (73.5 GB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 5349370 bytes 73508571337 (73.5 GB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
[amber10 54 $ ]]

mansi@amber20 ~ % ifconfig
[amber20 53 $ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 128.10.112.150 netmask 255.255.255.0 broadcast 128.10.112.255
        inet6 fe80::a6bb:6dff:fe46:822b prefixlen 64 scopeid 0x20<link>
            ether a4:bb:6d:46:82:2b txqueuelen 1000 (Ethernet)
            RX packets 604737287 bytes 784567386351 (784.5 GB)
            RX errors 0 dropped 18 overruns 0 frame 0
            TX packets 215956466 bytes 58462838473 (58.4 GB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
            device interrupt 20 memory 0xd1300000-d1320000
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopeid 0x10<host>
            loop txqueuelen 1000 (Local Loopback)
            RX packets 1507128 bytes 797492015 (797.4 MB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 1507128 bytes 797492015 (797.4 MB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
[amber20 54 $ ]

```

Handling Multiple Clients using pthread library

- Running Multiple clients on separate terminals on **localhost** and handling them using **pthread**.

Code files to refer - `client.c`, `serverMul.c`

Steps to execute the code:

1. Creating executable file using `gcc` command
2. Running the executable file. While running the server executable file, we have to pass the argument “`-lpthread`” to notify the server that multiple clients will be connecting during the session.

Client Side:

1. `gcc client.c -o ./client`
2. `./client 127.0.0.1 6000 "Testing Message"`

Server Side:

1. `gcc serverMul.c -lpthread -o serverMul`
2. `./serverMul 6000`

Output on serverMul side:

```
[(base) mansishinde@Mansi-MacBook-Air PartA %
[(base) mansishinde@Mansi-MacBook-Air PartA % ./serverMul 6000
The server is ready to receive

message-from-client : 127.0.0.1 , 64964
Message: Hello! this is Client talking from localhost 1
Modified Message: HELLO! THIS IS CLIENT TALKING FROM LOCALHOST 1

message-from-client : 127.0.0.1 , 64967
Message: Hello! this is Client talking from localhost 2
Modified Message: HELLO! THIS IS CLIENT TALKING FROM LOCALHOST 2

message-from-client : 127.0.0.1 , 64968
Message: Hello! this is Client talking from localhost 3
Modified Message: HELLO! THIS IS CLIENT TALKING FROM LOCALHOST 3

close-client : 127.0.0.1 , 64964

message-from-client : 127.0.0.1 , 64973
Message: Hello! this is Client talking from localhost 4
Modified Message: HELLO! THIS IS CLIENT TALKING FROM LOCALHOST 4

close-client : 127.0.0.1 , 64967

close-client : 127.0.0.1 , 64968
```

Output of client 1:

```
[(base) mansishinde@Mansi-MacBook-Air PartA %
[(base) mansishinde@Mansi-MacBook-Air PartA % ./client 127.0.0.1 6000 "Hello! this is Client talking from localhost 4"
Connection established with Server

Sending the message to the server.

Modified message received from server:
HELLO! THIS IS CLIENT TALKING FROM LOCALHOST 4
```

Output of client 2:

```
[(base) mansishinde@Mansi-MacBook-Air PartA %
[(base) mansishinde@Mansi-MacBook-Air PartA % ./client 127.0.0.1 6000 "Hello! this is Client talking from localhost 3"
Connection established with Server

Sending the message to the server.

Modified message received from server:
HELLO! THIS IS CLIENT TALKING FROM LOCALHOST 3
Closing the connection
(base) mansishinde@Mansi-MacBook-Air PartA %
```

Output of client 3:

```
(base) mansishinde@Mansi-MacBook-Air PartA % ./client 127.0.0.1 6000 "Hello! this is Client talking from localhost 2"
Connection established with Server
Sending the message to the server.

Modified message received from server:
HELLO! THIS IS CLIENT TALKING FROM LOCALHOST 2
Closing the connection
(base) mansishinde@Mansi-MacBook-Air PartA %
```

- Running Multiple clients on different host machines and handling them using pthread.
Here, the server IP address taken is 128.10.112.132

Code files to refer - client.c, serverMul.c

Steps to execute the code:

1. Creating executable file using gcc command
2. Running the executable file. While running the server executable file, we have to pass the argument “lpthread” to notify the server that multiple clients will be connecting during the session.

Client Side:

1. gcc client.c -o ./client
2. ./client 128.10.112.132 4700 “Testing Message”

Server Side:

1. gcc serverMul.c -lpthread -o serverMul
2. ./serverMul 4700

Output of Client on amber06 machine:

```
amber06 55 $
amber06 55 $ ./client 128.10.112.132 4700 "This is the testing message from amber06 machine"
Connection established with Server
Sending the messageTo the server.

Modified sentence received from server:
THIS IS THE TESTING MESSAGE FROM AMBER06 MACHINE
```

Output of Client on amber01 machine:

```
amber01 52 $
amber01 52 $ ./client 128.10.112.132 4700 "This is the testing message from amber01 machine"
Connection established with Server
Sending the messageTo the server.

Modified sentence received from server:
THIS IS THE TESTING MESSAGE FROM AMBER01 MACHINE
```

Output of Client on amber04 machine:

```
[amber04 52 $ ./client 128.10.112.132 4700 "This is the testing message from amber04 machine"
Connection established with Server

Sending the messageTo the server.

Modified sentence received from server:
THIS IS THE TESTING MESSAGE FROM AMBER04 MACHINE
Closing the connection
amber04 53 $ ]
```

Output of Client on amber05 machine:

```
[amber05 53 $ ./client 128.10.112.132 4700 "This is the testing message from amber05 machine"
Connection established with Server

Sending the messageTo the server.

Modified sentence received from server:
THIS IS THE TESTING MESSAGE FROM AMBER05 MACHINE

Closing the connection
amber05 54 $ ]
```

Output on Server side:

```
[amber02 57 $ gcc serverMul.c -lpthread -o serverMul
[amber02 58 $
[amber02 58 $ ./serverMul 4700
The server is ready to receive

message-from-client : 128.10.112.136 , 40442
Message: This is the testing message from amber06 machine
Modified Message: THIS IS THE TESTING MESSAGE FROM AMBER06 MACHINE

message-from-client : 128.10.112.135 , 58282
Message: This is the testing message from amber05 machine
Modified Message: THIS IS THE TESTING MESSAGE FROM AMBER05 MACHINE

message-from-client : 128.10.112.134 , 51804
Message: This is the testing message from amber04 machine
Modified Message: THIS IS THE TESTING MESSAGE FROM AMBER04 MACHINE

message-from-client : 128.10.112.131 , 50760
Message: This is the testing message from amber01 machine
Modified Message: THIS IS THE TESTING MESSAGE FROM AMBER01 MACHINE

close-client : 128.10.112.136 , 40442

message-from-client : 128.10.112.136 , 40444
Message: This is the testing message from amber06 machine
Modified Message: THIS IS THE TESTING MESSAGE FROM AMBER06 MACHINE

close-client : 128.10.112.135 , 58282
]
```

Part B. Sol. Develop Web Server over HTTP/1.1

- Running text.html, picture.html, bigpicture.html on various browsers (have tested on Chrome, Mozilla Firefox, Safari)

Code files to refer - client1.c, server1.c

Steps to execute the code:

1. Creating server executable file using gcc command
2. Running the server executable file. While running the server executable file, we must pass the argument “-lpthread” to notify the server that multiple clients will be connecting during the session.
3. Run http commands on browser, for example: <http://server-ip-address:server-port-no/filename>. These commands will act as input for server1.c, where server will respond with 200, 400, 404 and 505 status codes.

Server Side:

1. gcc server1.c -lpthread -o server1
2. ./server1 6800

Now, the client will start listening to incoming requests.

On Browser:

1. <http://127.0.0.1:6800/text.html>

Browser executes <http://127.0.0.1:6800/www/text.html>, for which it gets HTTP/1.1 200 OK as a response. This http link has been run on Chrome Web Browser.

Test 1

Hello World!

```
(base) mansishinde@Mansi's-MacBook-Air PartB %
(base) mansishinde@Mansi's-MacBook-Air PartB % gcc server1.c -lpthread -o server1
(base) mansishinde@Mansi's-MacBook-Air PartB %
(base) mansishinde@Mansi's-MacBook-Air PartB % ./server1 6800
The server is ready to receive

message-from-client : 127.0.0.1 , 61917
GET /www/text.html HTTP/1.1

message-to-client: 127.0.0.1 61917
HTTP/1.1 200 OK
```

Browser executes <http://127.0.0.1:6800/picture.html>, for which it gets HTTP/1.1 200 OK as a response. This http link has been run on Safari Web Browser.

Test 2

Sample: A small picture.

```
(base) mansishinde@Mansi's-MacBook-Air PartB %
(base) mansishinde@Mansi's-MacBook-Air PartB % gcc server1.c -lpthread -o server1
(base) mansishinde@Mansi's-MacBook-Air PartB %
(base) mansishinde@Mansi's-MacBook-Air PartB % ./server1 6800
The server is ready to receive

message-from-client : 127.0.0.1 , 61917
GET /www/text.html HTTP/1.1

message-to-client: 127.0.0.1 61917
HTTP/1.1 200 OK

message-from-client : 127.0.0.1 , 62091
GET /picture.html HTTP/1.1

message-to-client: 127.0.0.1 62091
HTTP/1.1 200 OK

message-from-client : 127.0.0.1 , 62091
GET /purdue.jpeg HTTP/1.1

message-to-client: 127.0.0.1 62091
HTTP/1.1 200 OK
```

Name	Domain	Type	Transfer Size	Time
picture.html	127.0.0.1	document	220 B	2.000ms
purdue.jpeg	127.0.0.1	jpg	38.90 KB	5.41ms
				4.000ms
				6.000ms
				8.000ms
				10.000ms
				12.00ms
				14.00ms
				16.00ms

Browser executes <http://127.0.0.1:6800/www/bigpicture.html>, for which it gets HTTP/1.1 200 OK as a response. This http link has been run on Mozilla Firefox Web Browser.

The screenshot shows the Mozilla Firefox browser window with the developer tools open. The Network tab is selected, displaying a list of requests made by the browser. The table shows three entries:

Status	Method	Domain	File	Initiator	Type	Transferred	Size	Time
200	GET	127.0.0.1:6800	bigpicture.html	document	html	222 B	157 B	1 ms
200	GET	127.0.0.1:6800	bigpicture.jpeg	img	jpeg	5.24 MB	5.24 MB	30 ms
404	GET	127.0.0.1:6800	favicon.ico	FaviconLoader.jsm:180 (im...)	x-icon	cached	13 B	0 ms

At the bottom of the Network tab, there is a summary: "3 requests | 5.24 MB / 5.24 MB transferred | Finish: 111 ms | DOMContentLoaded: 67 ms | load: 107 ms".

On the right side of the browser window, the terminal and server logs are visible. The terminal shows the command "curl -v 127.0.0.1:6800/www/bigpicture.html" and the server logs show multiple requests for the file, indicating a persistent connection.

Example of http request and response with status code “**404 Not Found**” as it is unable to find the file hello.html

```
Terminal Shell Edit View Window Help
127.0.0.1:6800/www/hello.html
404 Not Found

PartB — server1 6800 — 98x34
...artC --zsh ... ...artC --zsh ... ...ever1 6800 ... --zsh ... ...ds --zsh ...
message-to-client : 127.0.0.1 , 62091
HTTP/1.1 200 OK
close-client : 127.0.0.1 , 62091

message-from-client : 127.0.0.1 , 62284
GET /www/bigpicture.html HTTP/1.1
message-to-client : 127.0.0.1 , 62284
HTTP/1.1 200 OK
message-from-client : 127.0.0.1 , 62284
GET /www/bigpicture.jpeg HTTP/1.1
message-to-client : 127.0.0.1 , 62284
HTTP/1.1 200 OK
close-client : 127.0.0.1 , 61946

message-from-client : 127.0.0.1 , 61917
GET /www/text HTTP/1.1
message-to-client: 127.0.0.1 61917
HTTP/1.1 404 Not Found
message-from-client : 127.0.0.1 , 62621
GET /www/hello.html HTTP/1.1
message-to-client: 127.0.0.1 62621
HTTP/1.1 404 Not Found
```

Example of http request and response with status code “**505 HTTP Version Not Supported**” as the server is running on HTTP/1.1 version whereas it is getting a response from a version which is not supported by the HTTP.

```
PartA — client 127.0.0.1 6800 GET ./www/picture.html HTTP/4.0 — 153x41
(base) mansishinde@Mansi's-MacBook-Air PartA %
(base) mansishinde@Mansi's-MacBook-Air PartA % ./client 127.0.0.1 6800 "GET ./www/picture.html HTTP/4.0"
Connection established with Server

Sending the message to the server.

Modified message received from server:
HTTP/1.1 505 HTTP Version Not Supported
Connection: close
Content-Length:30

505 HTTP Version Not Supported
```

```
PartB — server1 6800 — 98x34
.../lab/PartC --zsh ... r1B — server1 6800 ... --zsh ... ...wnloads --zsh ...
close-client : 127.0.0.1 , 61946

message-from-client : 127.0.0.1 , 61917
GET /www/text HTTP/1.1
message-to-client: 127.0.0.1 61917
HTTP/1.1 404 Not Found
message-from-client : 127.0.0.1 , 62621
GET /www/hello.html HTTP/1.1
message-to-client: 127.0.0.1 62621
HTTP/1.1 404 Not Found
close-client : 127.0.0.1 , 62284

message-from-client : 127.0.0.1 , 63231
HEY There, is this just random message
message-to-client: 127.0.0.1 63231
HTTP/1.1 400 Bad Request
close-client : 127.0.0.1 , 62682

message-from-client : 127.0.0.1 , 63469
GET ./www/picture.html HTTP/4.0
message-to-client: 127.0.0.1 63469
HTTP/1.1 505 HTTP Version Not Supported
```

Example of http request and response with status code “**400 Bad Request**” as it is unable to get the request Method (like GET, PUT etc.), version (like HTTP/1.1, HTTP/2.0 etc.) and protocol which is required for getting HTTP response.

```
(base) mansishinde@Mansi's-MacBook-Air PartA %
(base) mansishinde@Mansi's-MacBook-Air PartA % ./client 127.0.0.1 6800 "HEY There, is this just random message"
Connection established with Server
Sending the message to the server.

Modified message received from server:
HTTP/1.1 400 Bad Request
Connection: close
Content-Length:15

400 Bad Request
```



```
(base) mansishinde@Mansi's-MacBook-Air PartB %
(base) mansishinde@Mansi's-MacBook-Air PartB % ./server1 6800
Connection established with Client
PartA — client 127.0.0.1 6800 HEY There, is this just random message — 153x41
PartB — server1 6800 — 98x34
.../lab1/PartC -- zsh ...rtB -- server1 6800 ~ -- zsh ... ...wnloads -- zsh ... +
```

```
message-from-client : 127.0.0.1 , 62284
GET /www/bigpicture.jpeg HTTP/1.1

message-to-client: 127.0.0.1 62284
HTTP/1.1 200 OK

close-client : 127.0.0.1 , 61946

message-from-client : 127.0.0.1 , 61917
GET /www/text HTTP/1.1

message-to-client: 127.0.0.1 61917
HTTP/1.1 404 Not Found

message-from-client : 127.0.0.1 , 62621
GET /www/hello.html HTTP/1.1

message-to-client: 127.0.0.1 62621
HTTP/1.1 404 Not Found

close-client : 127.0.0.1 , 62284

message-from-client : 127.0.0.1 , 63231
HEY There, is this just random message

message-to-client: 127.0.0.1 63231
HTTP/1.1 400 Bad Request

close-client : 127.0.0.1 , 62682
```

Example of http request and response with status code “**505 HTTP Version Not Supported**” as the server is running on HTTP/1.1 but it is getting the request from the client which is running on HTTP/2.0.

```
(base) mansishinde@Mansi's-MacBook-Air PartC %
(base) mansishinde@Mansi's-MacBook-Air PartC % ./client2 http://127.0.0.1:6800/picture.html
ip = "127.0.0.1"
port = "6800"
page = "picture.html"
Connection established with Server
HTTP/1.1 505 HTTP Version Not Supported
Connection: close
Content-Length:30

505 HTTP Version Not Supported
Response is:
Print something
(null)
zsh: segmentation fault ./client2 http://127.0.0.1:6800/picture.html
(base) mansishinde@Mansi's-MacBook-Air PartC %
(base) mansishinde@Mansi's-MacBook-Air PartC %
```



```
(base) mansishinde@Mansi's-MacBook-Air PartB %
(base) mansishinde@Mansi's-MacBook-Air PartB % ./server1 6800
Connection established with Client
PartC -- -zsh — 153x41
PartB — server1 6800 — 98x34
.../lab1/PartC -- zsh ...rtB -- server1 6800 ~ -- zsh ... ...wnloads -- zsh ... +
```

```
message-to-client: 127.0.0.1 61917
HTTP/1.1 404 Not Found

message-from-client : 127.0.0.1 , 62621
GET /www/hello.html HTTP/1.1

message-to-client: 127.0.0.1 62621
HTTP/1.1 404 Not Found

close-client : 127.0.0.1 , 62284

message-from-client : 127.0.0.1 , 63231
HEY There, is this just random message

message-to-client: 127.0.0.1 63231
HTTP/1.1 400 Bad Request

close-client : 127.0.0.1 , 62682

message-from-client : 127.0.0.1 , 63469
GET ./www/picture.html HTTP/4.0

message-to-client: 127.0.0.1 63469
HTTP/1.1 505 HTTP Version Not Supported

message-from-client : 127.0.0.1 , 63786
GET /picture.html HTTP/2.0

message-to-client: 127.0.0.1 63786
HTTP/1.1 505 HTTP Version Not Supported
```

Part C. Sol. Develop Web Server over HTTP/2.0

- Executing Server and Client on Localhost over HTTP/1.1

Code files to refer - client2.c, server2.c

Steps to execute the code:

1. Creating client and server executable files using gcc command
2. Running executable files

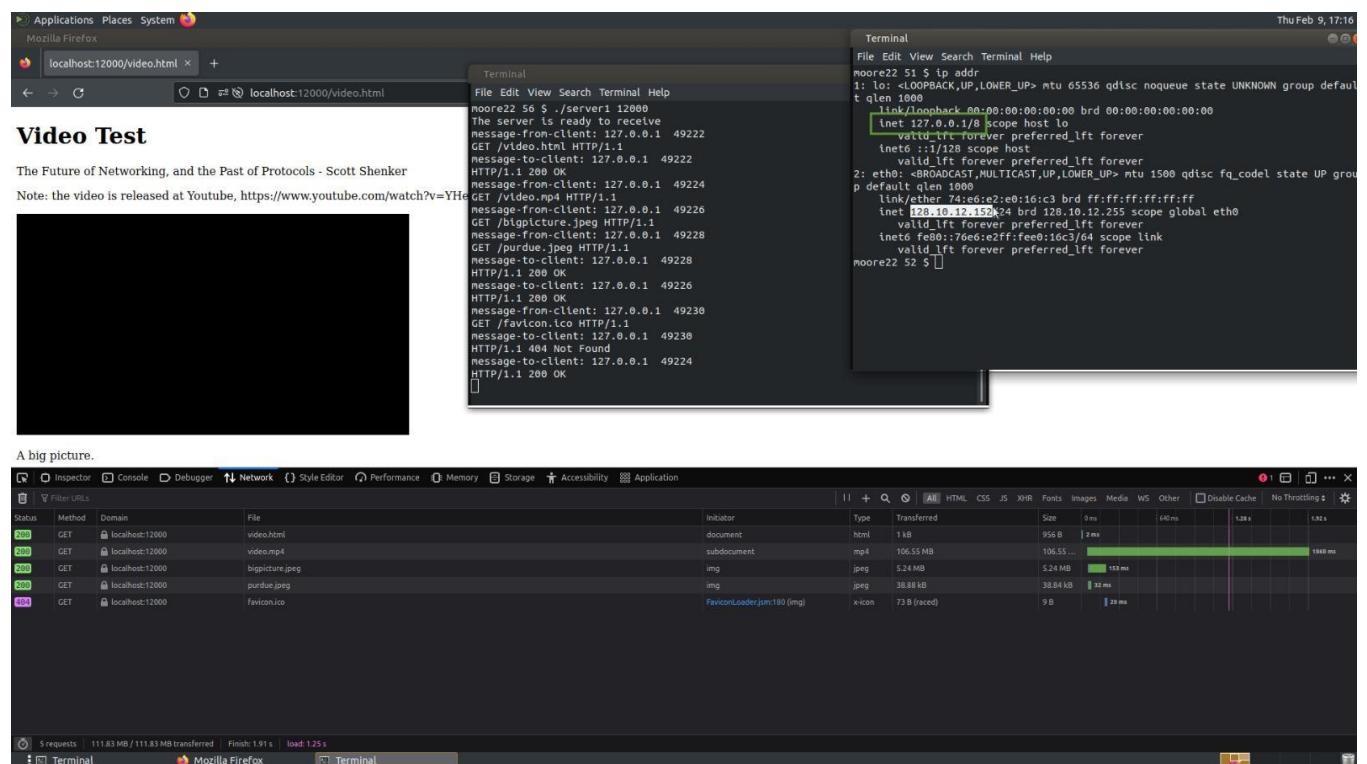
Server Side:

1. gcc server2.c -lpthread -o server2
2. ./server2 12000

Now, the client will start listening to incoming requests.

On Browser:

1. http://127.0.0.1:12000/video.html



Executing web server over HTTP/1.1 took about **2.05 (2 ms + 153 ms+ 32 ms+ 1868 ms) seconds** to render all 4 objects and display on the browser.

- Executing Server and Client on Localhost over HTTP/2.0

Steps to Execute:

Server Side:

1. gcc server2.c -lpthread -o server2
2. ./server2 5000

Now, the client will start listening to incoming requests.

Client Side:

1. gcc client2.c -o ./client2
2. ./client2 http://127.0.0.1:5000/video.html

The image displays four terminal windows side-by-side, each showing network traffic captured by Wireshark. The top row shows traffic for HTTP 1.1, and the bottom row shows traffic for HTTP 2.0. The leftmost window in each row is a command-line interface (CLI) session, while the rightmost window shows the corresponding network traffic.

Top Row (HTTP 1.1):

- Left Window:** CLI session showing a series of requests for frames 1 through 16, followed by a "Send-All-Frames" command and a closing connection message.
- Right Window:** Network traffic showing multiple separate requests for each frame, with responses for each frame being sent sequentially.

```
(base) manshinde@Mansi's-MacBook-Air PartC % ./client2 http://127.0.0.1:5000/video.html
Connection established with Server
request send
GET /video.mp4 HTTP/2.0

request send
GET /bigpicture.jpeg HTTP/2.0

request send
GET /purdue.jpeg HTTP/2.0

HTTP/2.0 200 OK
Content-Type: video/mp4
Content-Length: 106552622

HTTP/2.0 200 OK
Content-Type: image/jpeg
Content-Length: 5241296

HTTP/2.0 200 OK
Content-Type: image/jpeg
Content-Length: 38836

Object-Frame:[Object 1] Frame 1
Object-Frame:[Object 2] Frame 1
Object-Frame:[Object 3] Frame 1
Object-Frame:[Object 1] Frame 2
Object-Frame:[Object 2] Frame 2
Object-Frame:[Object 1] Frame 3
Object-Frame:[Object 2] Frame 3
Object-Frame:[Object 1] Frame 4
Object-Frame:[Object 2] Frame 4
Object-Frame:[Object 1] Frame 5
Object-Frame:[Object 2] Frame 5
Object-Frame:[Object 1] Frame 6
Object-Frame:[Object 2] Frame 6
Object-Frame:[Object 1] Frame 7
Object-Frame:[Object 2] Frame 7
Object-Frame:[Object 1] Frame 8
Object-Frame:[Object 2] Frame 8
Object-Frame:[Object 1] Frame 9
Object-Frame:[Object 2] Frame 9
Object-Frame:[Object 1] Frame 10
Object-Frame:[Object 2] Frame 10
Object-Frame:[Object 1] Frame 11
Object-Frame:[Object 2] Frame 11
Object-Frame:[Object 1] Frame 12
Object-Frame:[Object 2] Frame 12
Object-Frame:[Object 1] Frame 13
Object-Frame:[Object 2] Frame 13
Object-Frame:[Object 1] Frame 14
Object-Frame:[Object 2] Frame 14
Object-Frame:[Object 1] Frame 15
Object-Frame:[Object 2] Frame 15
Object-Frame:[Object 1] Frame 16
```

Bottom Row (HTTP 2.0):

- Left Window:** CLI session showing a series of requests for frames 1 through 16, followed by a "Send-All-Frames" command and a closing connection message.
- Right Window:** Network traffic showing a single multiplexed stream where all frames are sent together, with responses for each frame being sent sequentially.

```
-/Documents/ComputerNetworking/lab1/PartC -- zsh
...loads ssh mshinde@amber10.cs.purdue.edu

message-from-client : 127.0.0.1 , 64692
GET /video.html HTTP/2.0

message-to-client: 127.0.0.1 64692
HTTP/2.0 200 OK
requestNo:0
file len:106552622

message-from-client : 127.0.0.1 , 64692
GET /bigpicture.jpeg HTTP/2.0

message-to-client: 127.0.0.1 64692
HTTP/2.0 200 OK
requestNo:1
file len:5241296

message-from-client : 127.0.0.1 , 64692
GET /purdue.jpeg HTTP/2.0

message-to-client: 127.0.0.1 64692
HTTP/2.0 200 OK
requestNo:2
file len:38836
End of file reached
initial total data size:111832754
sending :Object-Frame:[Object 1] Frame 1
sending :Object-Frame:[Object 2] Frame 1
sending :Object-Frame:[Object 3] Frame 1
sending :Object-Frame:[Object 1] Frame 2
sending :Object-Frame:[Object 2] Frame 2
sending :Object-Frame:[Object 1] Frame 3
sending :Object-Frame:[Object 2] Frame 3
sending :Object-Frame:[Object 1] Frame 4
sending :Object-Frame:[Object 2] Frame 4
sending :Object-Frame:[Object 1] Frame 5
sending :Object-Frame:[Object 2] Frame 5
sending :Object-Frame:[Object 1] Frame 6
sending :Object-Frame:[Object 2] Frame 6
sending :Object-Frame:[Object 1] Frame 7
sending :Object-Frame:[Object 2] Frame 7
sending :Object-Frame:[Object 1] Frame 8
sending :Object-Frame:[Object 2] Frame 8
sending :Object-Frame:[Object 1] Frame 9
sending :Object-Frame:[Object 2] Frame 9
sending :Object-Frame:[Object 1] Frame 10
sending :Object-Frame:[Object 2] Frame 10

-/Documents/ComputerNetworking/lab1/PartC -- zsh
...loads ssh mshinde@amber10.cs.purdue.edu

Object-Frame:[Object 1] Frame 2552
Object-Frame:[Object 1] Frame 2553
Object-Frame:[Object 1] Frame 2554
Object-Frame:[Object 1] Frame 2555
Object-Frame:[Object 1] Frame 2556
Object-Frame:[Object 1] Frame 2557
Object-Frame:[Object 1] Frame 2558
Object-Frame:[Object 1] Frame 2559
Object-Frame:[Object 1] Frame 2560
Object-Frame:[Object 1] Frame 2561
Object-Frame:[Object 1] Frame 2562
Object-Frame:[Object 1] Frame 2563
Object-Frame:[Object 1] Frame 2564
Object-Frame:[Object 1] Frame 2565
Object-Frame:[Object 1] Frame 2566
Object-Frame:[Object 1] Frame 2567
Object-Frame:[Object 1] Frame 2568
Object-Frame:[Object 1] Frame 2569
Object-Frame:[Object 1] Frame 2570
Object-Frame:[Object 1] Frame 2571
Object-Frame:[Object 1] Frame 2572
Object-Frame:[Object 1] Frame 2573
Object-Frame:[Object 1] Frame 2574
Object-Frame:[Object 1] Frame 2575
Object-Frame:[Object 1] Frame 2576
Object-Frame:[Object 1] Frame 2577
Object-Frame:[Object 1] Frame 2578
Object-Frame:[Object 1] Frame 2579
Object-Frame:[Object 1] Frame 2580
Object-Frame:[Object 1] Frame 2581
Object-Frame:[Object 1] Frame 2582
Object-Frame:[Object 1] Frame 2583
Object-Frame:[Object 1] Frame 2584
Object-Frame:[Object 1] Frame 2585
Object-Frame:[Object 1] Frame 2586
Object-Frame:[Object 1] Frame 2587
Object-Frame:[Object 1] Frame 2588
Object-Frame:[Object 1] Frame 2589
Object-Frame:[Object 1] Frame 2590
Object-Frame:[Object 1] Frame 2591
Object-Frame:[Object 1] Frame 2592
Object-Frame:[Object 1] Frame 2593
Object-Frame:[Object 1] Frame 2594
Object-Frame:[Object 1] Frame 2595
Object-Frame:[Object 1] Frame 2596
Object-Frame:[Object 1] Frame 2597
Object-Frame:[Object 1] Frame 2598
Object-Frame:[Object 1] Frame 2599
Object-Frame:[Object 1] Frame 2600
Object-Frame:[Object 1] Frame 2601
Object-Frame:[Object 1] Frame 2602
Object-Frame:[Object 1] Frame 2603
Send-All-Frames
Execution time:0.00963
Closing the connection
```

message-from-client : 127.0.0.1 , 64692
EndOfMessagejpeg HTTP/2.0

message-to-client: 127.0.0.1 64692
HTTP/2.0 400 Bad Request

Whereas, executing web server over HTTP/2.0 took about **0.009 seconds** to render all 4 objects.

We see a huge difference between the performance of HTTP 1.1 and 2.0 due to the following reason:

HTTP 2.0 supports multiple requests at the same time over a single connection, in our case, multiple GET requests for objects video.mp4, picture.jpeg, bigpicture.jpeg are send to the server at the same time over single connection, whereas in HTTP 1.1 only one request is handled at a time, which leads to better performance.

Therefore, HTTP 2.0 provides several performance improvements over HTTP 1.1, including faster data transfer, reduced latency etc.

- Executing Server and Client on separate hosts over HTTP/1.1

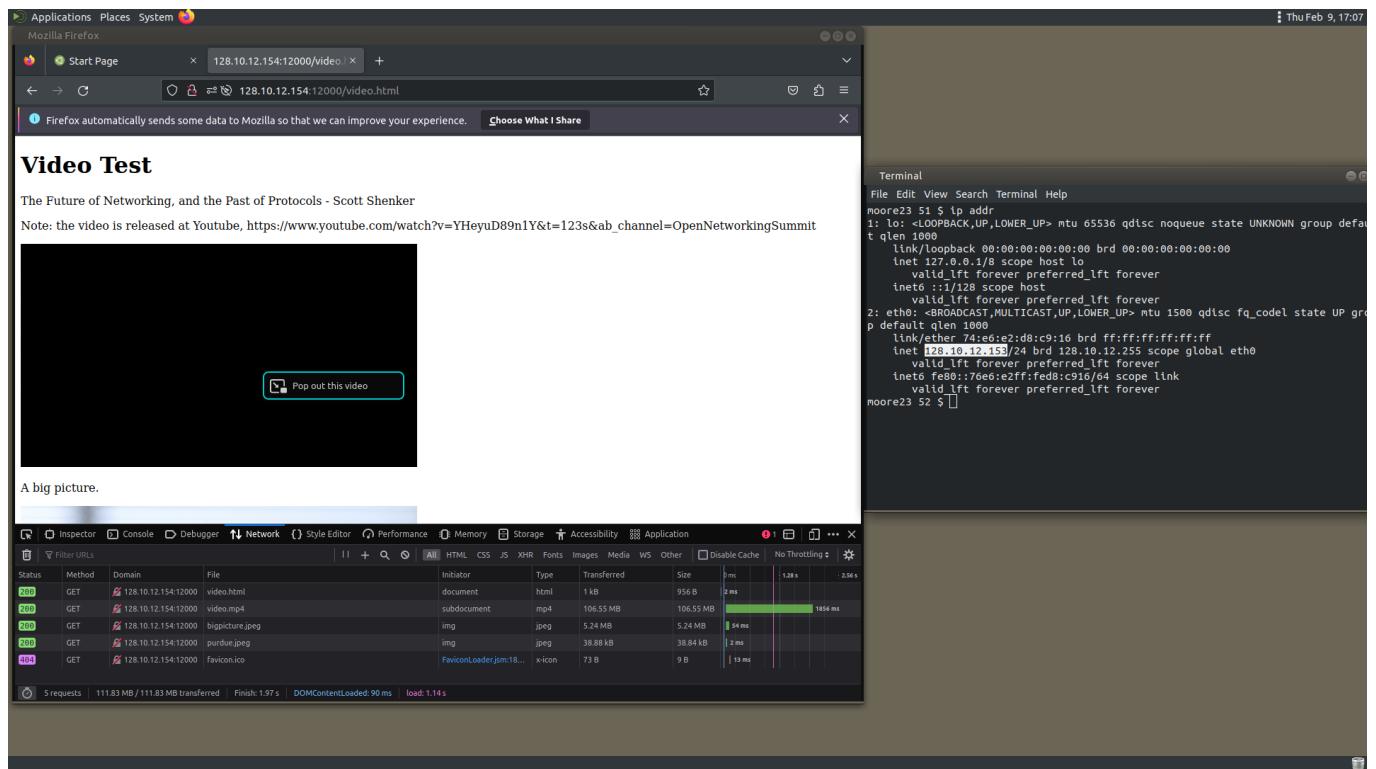
Server Side:

3. gcc server2.c -lpthread -o server2
4. ./server2 12000

Now, the client will start listening to incoming requests.

On Browser:

1. <http://128.10.12.154:12000/video.html>



Executing web server over HTTP/1.1 over different hosts (here server is running on 128.10.12.154 and browser on 128.10.12.153), took about 1.9 (**1856 ms + 2 ms + 2 ms + 54 ms**) seconds to render all 4 objects and display on the browser.

- Executing Server and Client on separate hosts over HTTP/2.0

Steps to Execute:

Server Side:

3. gcc server2.c -lpthread -o server2
4. ./server2 5000

Now, the client will start listening to incoming requests.

Client Side:

3. gcc client2.c -o ./client2
4. ./client2 http://128.10.112.150:5000/video.html

Whereas, executing web server over HTTP/2.0 took about 0.012 seconds to render all 4 objects.

We see that, executing client server on different host machines takes a little more time as compared to running it on localhosts because:

When the client and server are running on the same machine, network latency and overhead are reduced as the data does not have to travel over the network. These results improve performance for both HTTP/1.1 and HTTP/2.0.

When client and server are running on different hosts, network latency and overhead increase as data must travel over the network. In this scenario, performance benefits of HTTP 2.0 as seen in above case can help reduce the impact of latency and improve overall performance compared to HTTP 1.1