

Table of Contents:

Sl no.	Title	Page No.
1.	Abstract	2
2.	Problem Statement	2
3.	Schematic	2
4.	Software Details	3
5.	Code	3
6.	Screenshots	6
7.	Wireshark Analysis & Results	9
8.	Future Scope	10

Abstract:

The depiction of the working of web caching in the same subnet (using a proxy server).

Problem Statement:

Depiction of web caching between client, proxy server and the main server (i.e. internet) and showing the caching at the proxy server side. If the website is not present in the proxy server's cache, it will request the same from the internet (i.e main server) and return the website to the client.

Upon arrival at the client end, the requested website is opened automatically on the client's default web browser.

Schematic:

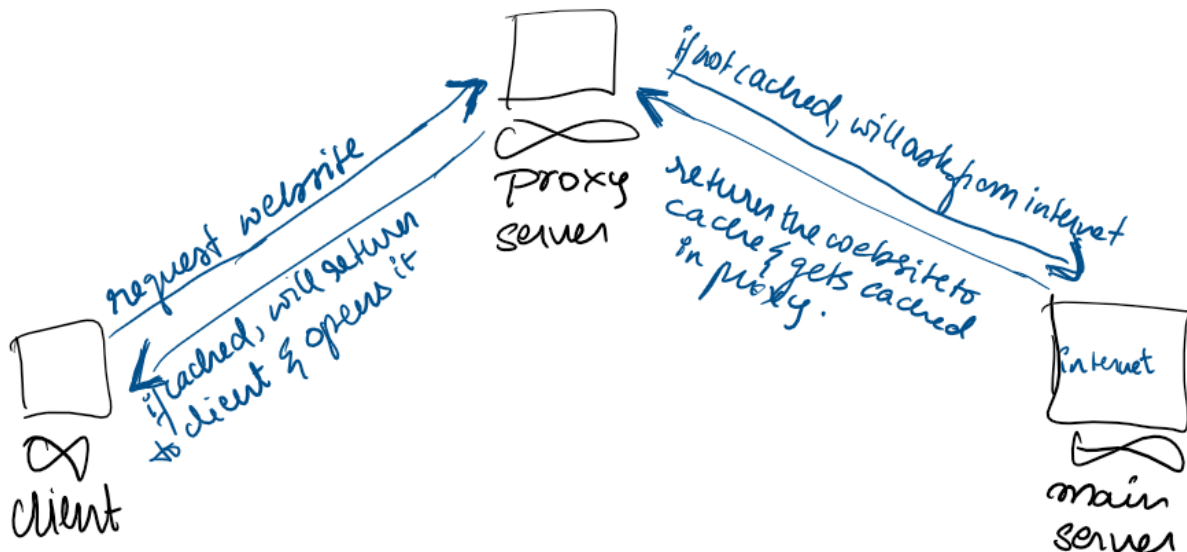


Fig.1 : Rough sketch of the working of web caching

TCP Connection Flow

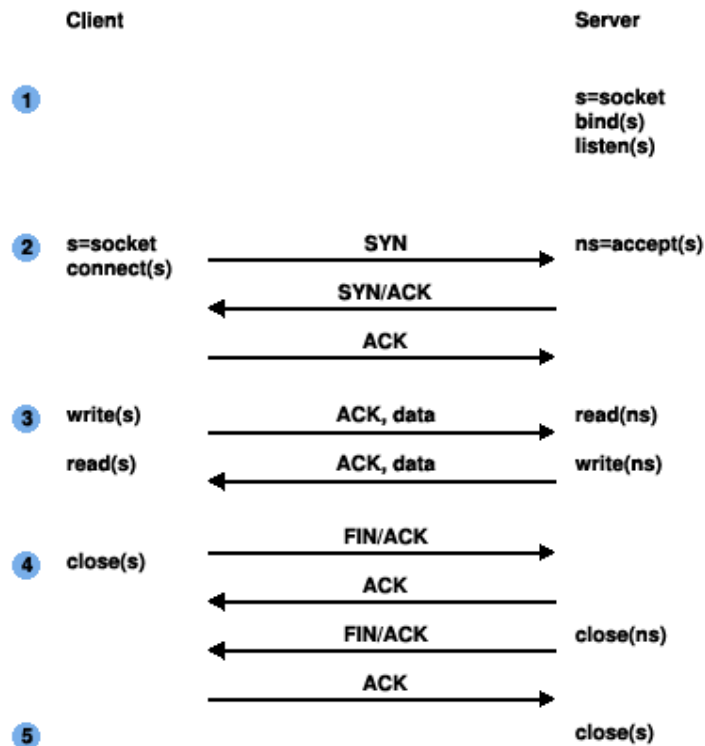


Fig 2: TCP Connection Flow

Software Details:

Socket programming in *Python* and analysis of the packets transmitted using *Wireshark*.

Code:

Proxy Server Code:

```
import socket
import requests

# Proxy server address and port
proxy_host = '192.168.103.239'
proxy_port = 8888

# Create a dictionary to store cached web pages
cache = {}

def proxy_server():
```

```
# Create a socket for the proxy server
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as
proxy_socket:
    proxy_socket.bind((proxy_host, proxy_port))
    proxy_socket.listen()

    print(f"Proxy server listening on {proxy_host}:{proxy_port}")

    while True:
        # Accept client connection
        client_socket, client_address = proxy_socket.accept()
        with client_socket:
            print(f"Connected to client: {client_address}")

            # Receive the requested website URL from the client
            url = client_socket.recv(1024).decode('utf-8')

            if url in cache:
                # Website is in the cache, send the URL to the client
                print(f"Fetching {url} from cache...")
                client_socket.sendall(url.encode('utf-8'))
            else:
                # Website not in cache, fetch from the internet
                print(f"Requesting {url} from the internet...")
                response = requests.get(url)
                web_content = response.text

                # Store the content in the cache
                cache[url] = web_content

                # Send the URL to the client
                client_socket.sendall(url.encode('utf-8'))

            print(f"Sent {url} to the client")

if __name__ == "__main__":
    proxy_server()
```

Client Code:

```
from socket import *
import webbrowser

server_ip = input("Enter the server's IP address: ") #192.168.103.239
server_address = (server_ip, 8888) #8888 is the port number of the proxy
server

# TCP connection
# Client socket
client_socket = socket(AF_INET, SOCK_STREAM)

def client():
    # Create a socket for the client
    client_socket.connect(server_address)

    # Request a website from the proxy server
    website_url = input("Enter the website URL: ")
    client_socket.send(website_url.encode('utf-8'))
    print("Waiting for the URL")

    # Receive the website content from the proxy server
    web_content = client_socket.recv(4096).decode('utf-8')

    # Display the received content
    webbrowser.open(web_content)
    print("Received from Proxy Server:")
    print("Done done")
    print(web_content)

if __name__ == "__main__":
    client()
```

Screenshots:

ipconfig: Displays the IP address of the host computer, among other things

```
C:\Users\cpsin>ipconfig

Windows IP Configuration

Wireless LAN adapter Local Area Connection* 1:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :

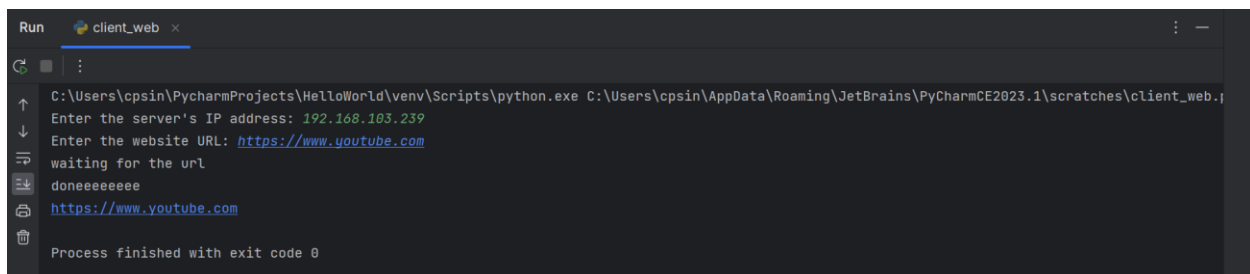
Wireless LAN adapter Local Area Connection* 2:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :

Wireless LAN adapter Wi-Fi:

    Connection-specific DNS Suffix  . :
    IPv6 Address. . . . . : 2401:4900:619e:a332:c74d:452f:3ec4:748
    Temporary IPv6 Address. . . . . : 2401:4900:619e:a332:16:af8:37f:c69d
    Link-local IPv6 Address . . . . . : fe80::c375:7701:c3ed:c252%13
    IPv4 Address. . . . . : 192.168.103.224
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : fe80::2005:d9ff:fe4e:748%13
                                192.168.103.238
```

Fig 3: IP Address of Client 1



```
Run client_web x
C:\Users\cpsin\PycharmProjects\HelloWorld\venv\Scripts\python.exe C:\Users\cpsin\AppData\Roaming\JetBrains\PyCharmCE2023.1\scratches\client_web.j
Enter the server's IP address: 192.168.103.239
Enter the website URL: https://www.youtube.com
waiting for the url
doneeeeeeeee
https://www.youtube.com
Process finished with exit code 0
```

Fig 4: Client 1 requesting for an URL from the Proxy Server

```
C:\Users\Divya Venkat>ipconfig

Windows IP Configuration

Unknown adapter Local Area Connection:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 3:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 12:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :

Wireless LAN adapter Wi-Fi:

    Connection-specific DNS Suffix  . :
    IPv6 Address. . . . . : 2401:4900:619e:a332:f0f:9b3:2ccc:30c1
    Temporary IPv6 Address. . . . . : 2401:4900:619e:a332:30a1:e0d9:8bcc:9f50
    Link-local IPv6 Address . . . . . : fe80::2b7d:d7e8:a654:93f0%10
    IPv4 Address. . . . . : 192.168.103.22
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : fe80::2005:d9ff:fe4e:748%10
                                192.168.103.238
```

Fig 5: IP Address of Client 2

```
PS D:\PESU\SEM 5\CCN> & "C:/Users/Divya Venkat/AppData/Local/Programs/Python/Python311/python.exe" "d:/PESU/SEM 5/CCN/cache_client.py"
"
Enter the server's IP address: 192.168.103.239
Enter the website URL: https://www.pes.edu
Waiting for the URL
Received from Proxy Server:
Done done
https://www.pes.edu
PS D:\PESU\SEM 5\CCN> & "C:/Users/Divya Venkat/AppData/Local/Programs/Python/Python311/python.exe" "d:/PESU/SEM 5/CCN/cache_client.py"
"
Enter the server's IP address: 192.168.103.239
Enter the website URL: https://www.youtube.com
Waiting for the URL
Received from Proxy Server:
Done done
https://www.youtube.com
PS D:\PESU\SEM 5\CCN> & "C:/Users/Divya Venkat/AppData/Local/Programs/Python/Python311/python.exe" "d:/PESU/SEM 5/CCN/cache_client.py"
"
Enter the server's IP address: 192.168.103.239
Enter the website URL: https://open.spotify.com
Waiting for the URL
Received from Proxy Server:
Done done
https://open.spotify.com
```

Fig 6: Client 2 requesting for an URL from the Proxy Server

```
C:\Users\dtsmv>ipconfig

Windows IP Configuration

Ethernet adapter Ethernet:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 1:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 2:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :

Wireless LAN adapter Wi-Fi:

    Connection-specific DNS Suffix  . :
    IPv6 Address. . . . . : 2401:4900:619e:a332:d1a1:d544:f4cb:83f1
    Temporary IPv6 Address. . . . . : 2401:4900:619e:a332:24d7:17e1:48b9:b11c
    Link-local IPv6 Address . . . . . : fe80::a38:1e92:ca3:bcbd%11
    IPv4 Address. . . . . : 192.168.103.239
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : fe80::2005:d9ff:fe4e:748%11
                                192.168.103.238
```

Fig 7: IP Address of the Proxy Server

```
PS C:\Users\dtsmv\Downloads> & C:/Users/dtsmv/anaconda3/python.exe c:/Users/dtsmv/Downloads/ccn_pro.py
Proxy server listening on 192.168.103.239:8888
Connected to client: ('192.168.103.224', 58939)
Requesting https://www.pes.edu from the internet...
Sent https://www.pes.edu to the client
Connected to client: ('192.168.103.22', 52576)
Fetching https://www.pes.edu from cache...
Sent https://www.pes.edu to the client
Connected to client: ('192.168.103.224', 58964)
Requesting https://open.spotify.com from the internet...
Sent https://open.spotify.com to the client
Connected to client: ('192.168.103.22', 52599)
Requesting https://www.youtube.com from the internet...
Sent https://www.youtube.com to the client
Connected to client: ('192.168.103.224', 59000)
Fetching https://www.youtube.com from cache...
Sent https://www.youtube.com to the client
Connected to client: ('192.168.103.22', 52604)
Fetching https://open.spotify.com from cache...
Sent https://open.spotify.com to the client
```

Fig 8: Proxy Server receiving requests from the clients

(The Proxy Server sends the requested URL either directly from its cache (if present) or requests it from the internet and stores it in the cache, which is then sent to the client)

Wireshark Analysis & Results:

Host	IP Address	Port Number
Proxy Server	192.168.103.239	8888
Client 1	192.168.103.224	58570
Client 2	192.168.103.22	52335

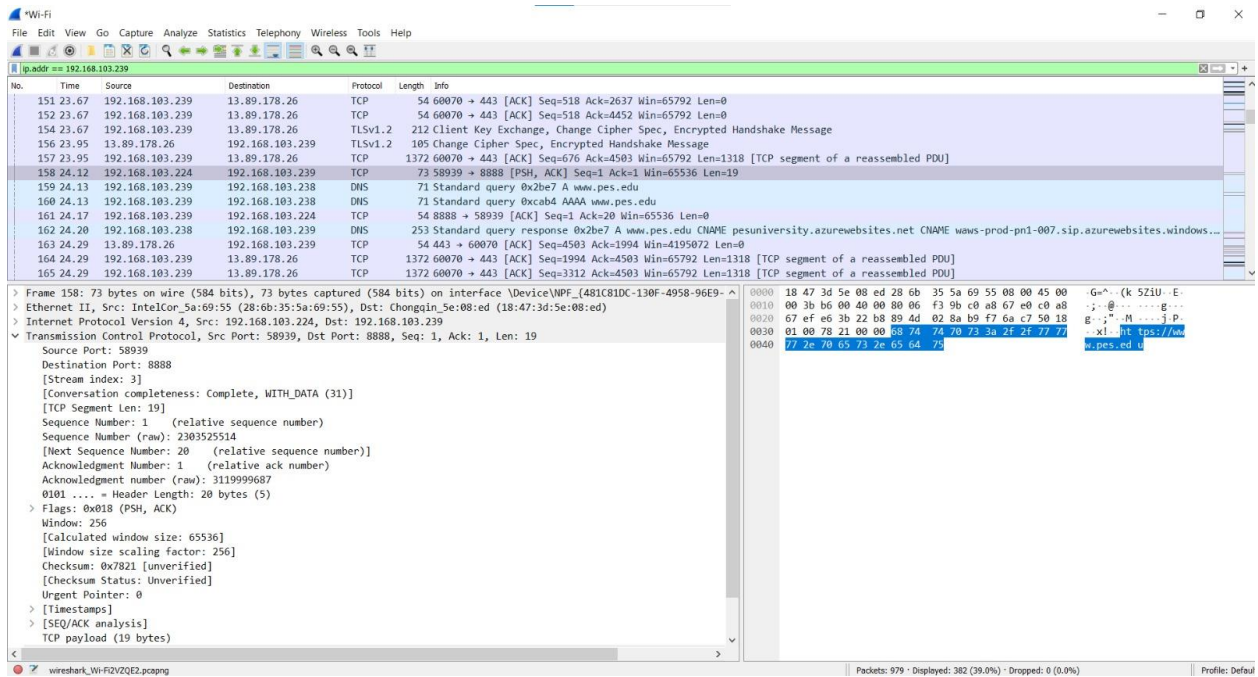


Fig 9: Wireshark capture of the Proxy Server receiving requests from the clients and then sending the requested URL to the respective client

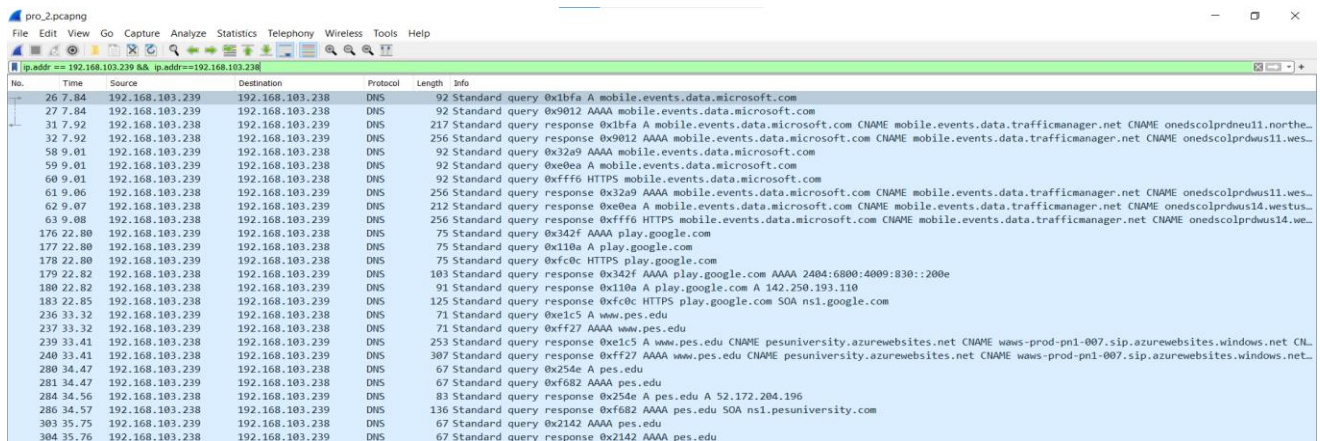


Fig 10: Wireshark packet capture at the Proxy Server side, displaying DNS query

No.	Time	Source	Destination	Protocol	Length	Info
296	12...	192.168.103.224	192.168.103.239	TCP	66	58570 → 8888 [SYN] Seq=2174070432 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
297	12...	192.168.103.239	192.168.103.224	TCP	66	8888 → 58570 [SYN, ACK] Seq=2019483420 Ack=2174070433 Win=65535 Len=0 MSS=1460 WS=256...
298	12...	192.168.103.224	192.168.103.239	TCP	54	58570 → 8888 [ACK] Seq=2174070433 Ack=2019483421 Win=65536 Len=0
348	19...	192.168.103.224	192.168.103.239	TCP	73	58570 → 8888 [PSH, ACK] Seq=2174070433 Ack=2019483421 Win=65536 Len=19
350	19...	192.168.103.239	192.168.103.224	TCP	73	8888 → 58570 [PSH, ACK] Seq=2019483421 Ack=2174070452 Win=65536 Len=19
351	19...	192.168.103.239	192.168.103.224	TCP	54	8888 → 58570 [FIN, ACK] Seq=2019483440 Ack=2174070452 Win=65536 Len=0
352	19...	192.168.103.224	192.168.103.239	TCP	54	58570 → 8888 [ACK] Seq=2174070452 Ack=2019483441 Win=65536 Len=0
354	20...	192.168.103.224	192.168.103.239	TCP	54	58570 → 8888 [FIN, ACK] Seq=2174070452 Ack=2019483441 Win=65536 Len=0
355	20...	192.168.103.239	192.168.103.224	TCP	54	8888 → 58570 [ACK] Seq=2019483441 Ack=2174070453 Win=65536 Len=0

> Frame 348: 73 bytes on wire (584 bits), 73 bytes captured (584 bits) on interface IntelCor_5a:69:55 (28:6b:35:5a:69:55), Dst: Chongqin_5e:00:00:00:00:00	0000 18 47 3d 5e 08 ed 28 6b 35 5a 69 55 08 00 45 00	G=^..(k 5Ziu..E..
> Ethernet II, Src: IntelCor_5a:69:55 (28:6b:35:5a:69:55), Dst: Chongqin_5e:00:00:00:00:00	0010 00 3b b5 f6 40 00 00 00 00 00 c0 a8 67 e0 c0 a8	..@.....g...
> Internet Protocol Version 4, Src: 192.168.103.224, Dst: 192.168.103.239	0020 67 ef e4 ca 22 b8 81 95 ae a1 78 5e df 1d 50 18	g.....x^..P..
> Transmission Control Protocol, Src Port: 58570, Dst Port: 8888, Seq: 2174070452	0030 01 00 51 4e 00 00 68 74 74 70 73 3a 2f 2f 77 77	..QN..ht tps://ww
> Data (19 bytes)	0040 77 2e 70 65 73 2e 65 64 75	w.pes.ed u

Fig 11: Packet capture of Client 1 requesting "www.pes.edu"

-> Packets filtered on the basis of IP Addresses of Client 1 and the Proxy Server

-> The requested URL can be seen in the Packet Content Window

No.	Time	Source	Destination	Protocol	Length	Info
102	3.81	192.168.103.22	192.168.103.239	TCP	66	52335 → 8888 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
106	4.14	192.168.103.239	192.168.103.22	TCP	66	8888 → 52335 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM
107	4.14	192.168.103.22	192.168.103.239	TCP	54	52335 → 8888 [ACK] Seq=1 Ack=1 Win=65536 Len=0
112	17.07	192.168.103.22	192.168.103.239	TCP	77	52335 → 8888 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=23
113	17.13	192.168.103.239	192.168.103.22	TCP	54	8888 → 52335 [ACK] Seq=1 Ack=24 Win=65536 Len=0
121	18.89	192.168.103.239	192.168.103.22	TCP	77	8888 → 52335 [PSH, ACK] Seq=1 Ack=24 Win=65536 Len=23
122	18.89	192.168.103.239	192.168.103.22	TCP	54	8888 → 52335 [FIN, ACK] Seq=24 Ack=24 Win=65536 Len=0
123	18.89	192.168.103.22	192.168.103.239	TCP	54	52335 → 8888 [ACK] Seq=24 Ack=25 Win=65536 Len=0
134	19.28	192.168.103.22	192.168.103.239	TCP	54	52335 → 8888 [FIN, ACK] Seq=24 Ack=25 Win=65536 Len=0
135	19.30	192.168.103.239	192.168.103.22	TCP	54	8888 → 52335 [ACK] Seq=25 Ack=25 Win=65536 Len=0

> Frame 112: 77 bytes on wire (616 bits), 77 bytes captured (616 bits) on interface \Device\NPF{...}	0000 18 47 3d 5e 08 ed 54 8d 5a dc 9a 65 08 00 45 00	G=^..T Z..e..E..
> Ethernet II, Src: IntelCor_dc:9a:65 (54:8d:5a:dc:9a:65), Dst: Chongqin_5e:00:00:00:00:00	0010 00 3f 66 35 40 00 00 00 00 00 c0 a8 67 16 c0 a8	..f5@.....g...
> Internet Protocol Version 4, Src: 192.168.103.22, Dst: 192.168.103.239	0020 67 ef cc 6f 22 b8 cf e0 27 a3 a8 ba 55 aa 50 18	g.....U..P..
> Transmission Control Protocol, Src Port: 52335, Dst Port: 8888, Seq: 1, Ack: 1, Len: 23	0030 01 00 50 88 00 00 68 74 74 70 73 3a 2f 2f 77 77	..P..ht tps://ww
> Data (23 bytes)	0040 77 2e 79 6f 75 74 75 62 65 2e 63 6f 6d	w.youtube e.com

Fig 12 : Packet Capture of Client 2 requesting "www.youtube.com"

-> Packets filtered on the basis of IP Addresses of Client 2 and the Proxy Server

-> The requested URL can be seen in the Packet Content Window

Future Scope:

- Store the Proxy Server's cache in a database
- Host a Proxy Server that can service clients from different subnets as well
- Directing the client's request to another Proxy Server if the cache of the target Proxy Server is full.