



భారతీయ సాంకేతిక విజ్ఞాన సంస్థ హైదరాబాద్
भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

Programming Assignment #2

WWW

Web Server, Client, Proxy and ExtendedProxy

Course Code & Name:

CS5060 - Advanced Computer Networks

Submitted By:

Kritik Agarwal (CS23MTECH11009)

Malsawmsanga Sailo (CS23MTECH11010)

Bendi Satya Sai Sateesh (SM23MTECH11001)

Course Instructor:

Dr. Bheemarjuna Reddy Tamma

Department Of Computer Science And Engineering,
Indian Institute Of Technology, Hyderabad
2023-2024

Table of Contents

Abstract	3
PART-1: A Simple Client	4
PART-2: A Simple Web Proxy Server	6
PART-3: A Simple Web Server	7
PART-4: (1) Caching at Web Proxy	8
References	10
Anti-Plagiarism Statement	10

ABSTRACT

This assignment encompasses four distinct parts. Part-1 entails the development of a web client capable of connecting directly to web servers or via an intermediary web proxy using TCP connections. The client sends HTTP requests, predominantly utilizing the GET method, and displays server responses, supporting variable command line arguments for specifying connection details. The Part-2 focuses on the creation of a web proxy, which connects to both clients and servers, forwarding HTTP requests and responses. Parts 3 and 4 revolve around the implementation of a multi-threaded web server, equipped to concurrently handle multiple HTTP requests from clients, browsers, and web proxies. The server listens on a fixed port, establishes separate threads for each request/response pair, processes incoming HTTP requests, retrieves requested files, constructs HTTP responses, and dispatches them to clients. In cases of missing files, the server issues an HTTP "404 Not Found" message. Threading is employed for efficient parallel request management across all sections.

PART-1: A Simple Client

Objective:

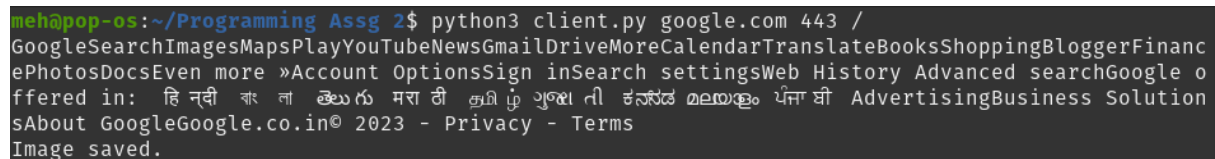
In this part, we develop a simple web client which will connect to the web server directly or through the intermediate web proxy using a TCP connection, send HTTP requests to the server, and display the server responses as output. We sent the HTTP requests using the GET method. The client can take variable number of command line arguments specifying the IP address of the web proxy, host name or IP address of the web server, the port at which the proxy and web server are listening in case of indirect connection through the web proxy or simply IP address and port at which the web server is listening in case of direct connection to the web server and the path at which the requested object is stored at the web server.

If the requested object is the base HTML file, the web client parses it for any reference to other objects and fetches them one after the other by establishing a non-persistent HTTP connection to the web proxy or web server.

Software/System Requirements:

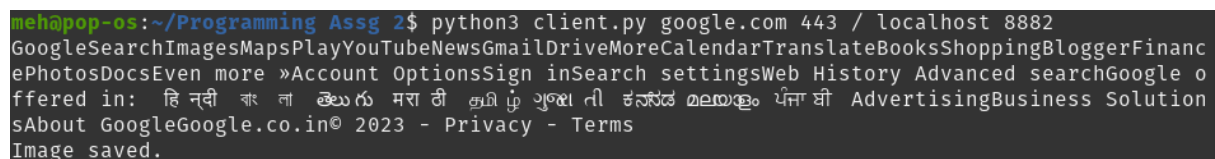
- Language Used: Python
- Operating System: Linux

Output:



```
meh@pop-os:~/Programming Assg 2$ python3 client.py google.com 443 /
GoogleSearchImagesMapsPlayYouTubeNewsGmailDriveMoreCalendarTranslateBooksShoppingBloggerFinanc
ePhotosDocsEven more »Account OptionsSign inSearch settingsWeb History Advanced searchGoogle o
ffered in: हि न्दी बां ला खोलु मरा ठी कुमि प्ठु गुण्ठा नी कनसद मलयल्लु पंजाबी AdvertisingBusiness Solution
sAbout GoogleGoogle.co.in© 2023 - Privacy - Terms
Image saved.
```

The above Screenshot is obtained by connecting the client directly to the server.



```
meh@pop-os:~/Programming Assg 2$ python3 client.py google.com 443 / localhost 8882
GoogleSearchImagesMapsPlayYouTubeNewsGmailDriveMoreCalendarTranslateBooksShoppingBloggerFinanc
ePhotosDocsEven more »Account OptionsSign inSearch settingsWeb History Advanced searchGoogle o
ffered in: हि न्दी बां ला खोलु मरा ठी कुमि प्ठु गुण्ठा नी कनसद मलयल्लु पंजाबी AdvertisingBusiness Solution
sAbout GoogleGoogle.co.in© 2023 - Privacy - Terms
Image saved.
```

The above Screenshot is obtained by connecting the client to the server through proxy.

```
meh@pop-os:~/Programming Assg 2$ python3 client.py localhost 8080 /HelloWorld.html

Welcome to My Website

Hello, World!
Welcome to my website. I'm excited to share some information with you.
Today's date is: November 5, 2023
Your IP address is: 123.456.789.0

No image found!!
No script found!!
```

The above Screenshot is obtained by connecting the client to our simple web server.

```
meh@pop-os:~/Programming Assg 2$ python3 client.py google.com 443 / localhost 8881
GoogleSearchImagesMapsPlayYouTubeNewsGmailDriveMoreCalendarTranslateBooksShoppingBloggerFinanc
ePhotosDocsEven more »Account OptionsSign inSearch settingsWeb History Advanced searchGoogle o
ffered in: हि न्दी বাংলা తెలుగు मराठी தமிழ் ગુજરાતી ಕನ್ನಡ മലയാളം ਪੰਜਾਬੀ AdvertisingBusiness Solution
sAbout GoogleGoogle.co.in© 2023 - Privacy - Terms
Image saved.
```

The above Screenshot is obtained by connecting the client to the server through web caching proxy.

PART-2: A Simple Web Proxy Server

Objective:

In this part, we develop a simple web proxy which will connect to the web client and web server using a TCP connection, send HTTP requests to the server, and send HTTP response back to the web client. We send and receive the HTTP requests using the GET method.

Software/System Requirements:

- Language Used: Python
- Operating System: Linux

Output:

```
me@pop-os:~/Programming Assg 2$ python3 proxy.py
Proxy server listening on :8882
Accepted connection from 127.0.0.1:55064
REQUEST:
b'GET / HTTP/1.0\r\nHost: google.com:443\r\n\r\n'

['Host: google.com:443']
['Host:', 'google.com:443']
Dest Host: google.com
Client Request: ['GET / HTTP/1.0', 'Host: google.com:443', '', '']

Host: google.com                Port: 443

/
Connection CLOSED!!

Accepted connection from 127.0.0.1:48744
REQUEST:
b'GET /images/branding/googlelogo/1x/googlelogo_white_background_color_272x92dp.png HTTP/1.0\r\nHost: google.com:443\r\n\r\n'

['Host: google.com:443']
['Host:', 'google.com:443']
Dest Host: google.com
Client Request: ['GET /images/branding/googlelogo/1x/googlelogo_white_background_color_272x92dp.png HTTP/1.0', 'Host: google.com:443', '', '']

Host: google.com                Port: 443

/images/branding/googlelogo/1x/googlelogo_white_background_color_272x92dp.png
Connection CLOSED!!
```

The above screenshot is obtained by running the client connecting to google.com through the proxy server.

PART-3: A Simple Web Server

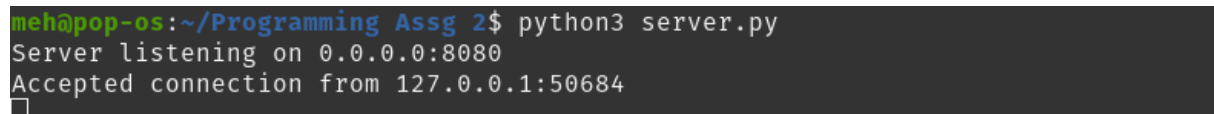
Objective:

In this part, we develop a multi-threaded web server that handles multiple HTTP requests simultaneously from our own web client(s) (PART-1), web browsers and your own proxy server (PART-2). Using threading, first we create a main thread in which we modify the server and listen for clients at a fixed port. When it receives a TCP connection request from a client, it will set up the TCP connection through another port and service the client request in a separate thread. There is a separate TCP connection in a separate thread for each HTTP request/response pair. Each thread at the web server accepts and parses the HTTP request, we get the requested file from the server's file system, and create an HTTP response message consisting of the requested file preceded by header lines, and then send the response directly to the client. If the requested file is not present in the server, it sends an HTTP "404 Not Found" message back to the client.

Software/System Requirements:

- Language Used: Python
- Operating System: Linux

Output:



```
meh@pop-os:~/Programming Assg 2$ python3 server.py
Server listening on 0.0.0.0:8080
Accepted connection from 127.0.0.1:50684
□
```

The above Screenshot is get by connecting our simple web client to our simple web server

PART-4: (1) Caching at web proxy

Objective:

In this part, we develop a multi-threaded web server that handles multiple HTTP requests simultaneously from our own web client(s) (PART-1), web browsers and your own proxy server (PART-2). Using threading, first we create a main thread in which we modify the server and listen for clients at a fixed port. When it receives a TCP connection request from a client, it will set up the TCP connection through another port and service the client request in a separate thread. There is a separate TCP connection in a separate thread for each HTTP request/response pair. Each thread at the web server accepts and parses the HTTP request, we get the requested file from the server's file system, and create an HTTP response message consisting of the requested file preceded by header lines, and then send the response directly to the client. If the requested file is not present in the server, it sends an HTTP "404 Not Found" message back to the client.

Software/System Requirements:

- Language Used: Python
- Operating System: Linux

Output:

```
meh@pop-os:~/Programming Assg 2$ python3 web_cache.py
Proxy server listening on :8881
Accepted connection from 127.0.0.1:58824
REQUEST:
b'GET / HTTP/1.0\r\nHost: google.com:443\r\n\r\n'

SEC: 0
Dest Host: google.com
File Path: cache/google.comindex.html

The file 'google.comindex.html' exists in folder 'cache'.
The file 'cache/google.comindex.html.expire' does not exist in folder 'cache'.
Client Request: ['GET / HTTP/1.0', 'Host: google.com:443', '', '']

Host: google.com                Port: 443

Cached.
Connection CLOSED!!

Accepted connection from 127.0.0.1:58838
REQUEST:
b'GET /images/branding/googlelogo/1x/googlelogo_white_background_color_272x92dp.png HTTP/1.0\r\nHost: google.com:443\r\n\r\n'

SEC: 0
Dest Host: google.com
File Path: cache/google.comgooglelogo_white_background_color_272x92dp.png

The file 'google.comgooglelogo_white_background_color_272x92dp.png' exists in folder 'cache'.
Client Request: ['GET /images/branding/googlelogo/1x/googlelogo_white_background_color_272x92d
p.png HTTP/1.0', 'Host: google.com:443', '', '']

Host: google.com                Port: 443

Cached.
Cached.
Time Saved.
Connection CLOSED!!
```

The above output is obtained by running the client through the web caching proxy

REFERENCES

- [1] <https://www.crummy.com/software/BeautifulSoup/bs4/doc/>
- [2] <https://snyk.io/blog/implementing-tls-ssl-python/>
- [3] <https://docs.python.org/3/library/ssl.html>
- [4] <https://www.internalpointers.com/post/making-http-requests-sockets-python>
- [5] Lecture Notes/Slides provided by Dr. Bheemarjuna Reddy Tamma in Google Classroom.

ANTI-PLAGIARISM STATEMENT

We certify that this assignment/report is our own work, based on our personal study and/or research and that we have acknowledged all material and sources used in its preparation, whether they be books, articles, packages, datasets, reports, lecture notes, and any other kind of document, electronic or personal communication. We also certify that this assignment/report has not previously been submitted for assessment/project in any other course lab, except where specific permission has been granted from all course instructors involved, or at any other time in this course, and that we have not copied in part or whole or otherwise plagiarized the work of other students and/or persons. Additionally, we acknowledge that we may have used AI tools, such as language models (e.g., ChatGPT, Bard), for assistance in generating and refining my assignment, and we have made all reasonable efforts to ensure that such usage complies with the academic integrity policies set for the course. I pledge to uphold the principles of honesty and responsibility at CSE@IITH. In addition, we understand our responsibility to report honor violations by other students if we become aware of it.

Names: Kritik Agarwal (CS23MTECH11009), Malsawmsanga Sailo (CS23MTECH11010),
Bendi Satya Sai Sateesh (SM23MTECH11001)

Date: Nov 5, 2023

Signatures: K.A., M.S., B.S.S.S.