



UNIVERSITY OF DHAKA

Department of Computer Science and Engineering

CSE-3111 : Computer Networking Lab

Lab Report 3: Implementing File transfer using Socket Programming
and HTTP GET/POST requests

Submitted By:

Name: Meherun Farzana

Roll No : 05

Name: Mohd. Jamal Uddin Mallick

Roll No : 07

Submitted On :

February 8, 2024

Submitted To :

Dr. Md. Abdur Razzaque

Dr. Md. Mamun Or Rashid

Dr. Muhammad Ibrahim

Redwan Ahmed Rizvee

1 Introduction

File transfer is a critical aspect of modern networking applications, facilitating the exchange of data between systems. This lab report focuses on implementing file transfer using two essential techniques: socket programming and HTTP GET/POST requests. Socket programming provides a foundational framework for establishing communication channels between client and server applications, while HTTP protocols offer standardized methods for data exchange over the web.

1.1 Objectives

- **Socket Programming:** Develop a file transfer mechanism using socket programming to establish communication between client and server applications.
- **HTTP GET/POST Requests** Implement HTTP GET and POST requests to facilitate file retrieval and submission over the network.

2 Theory

2.1 File Transfer via Socket:

Socket programming is the foundation of network communication, facilitating the establishment of connections between client and server applications for data exchange. File transfer via sockets refers to the process of transferring a file from one system to another over a network connection using socket programming. File transfer via sockets offers more control over the transfer process compared to higher-level protocols like HTTP, but also requires manual handling of error handling, data chunking, and other aspects of the transfer.

2.2 File Transfer via HTTP

HTTP (Hypertext Transfer Protocol) serves as the backbone of web communication, enabling clients to request resources from servers using methods such as GET and POST. HTTP is a standard protocol for transmitting data over the web, and is widely used for web communication between a client (e.g. a web browser) and a server (e.g. a web server). In file transfer via HTTP, the client sends a request to the server using either an HTTP GET or POST request. Overall, file transfer via HTTP is a simple and convenient

way to transfer files over a network, but may not be as fast or efficient as other methods, such as socket programming.

3 Methodology

3.1 Server

The server is initialized on a specific port and it listens for incoming requests. Whenever a client requests to connect, the server accepts the connection and provides necessary services.

3.1.1 File Transfer via Socket

In case of sockets, the server lets the client choose whether they want to get the list of available files, upload or download a file. The server can handle any type of file including text, audio, image, video etc. of all formats.

3.1.2 File Transfer via HTTP

In case of HTTP, the server is an HTTP server, serving GET and POST requests from clients. The client receives a response from the server indicating the success or failure of the transfer.

3.2 Client

3.2.1 File Transfer via Socket

The client side is a program requesting service from the server. It tries to connect to the server address mentioning the particular port on which the server is serving. When the server accepts the connection, it is provided with the options of service the client can avail. The client selects whether they want to see the list of available files, upload or download a file.

3.2.2 File Transfer via HTTP

In case of HTTP, the client can make a *GET* request to the server for a specific resource and avail it. Also it can make a *POST* request for uploading new files into the server.

4 Experimental result

Some Snapshots of the Client Side queries can be seen in the following figures:

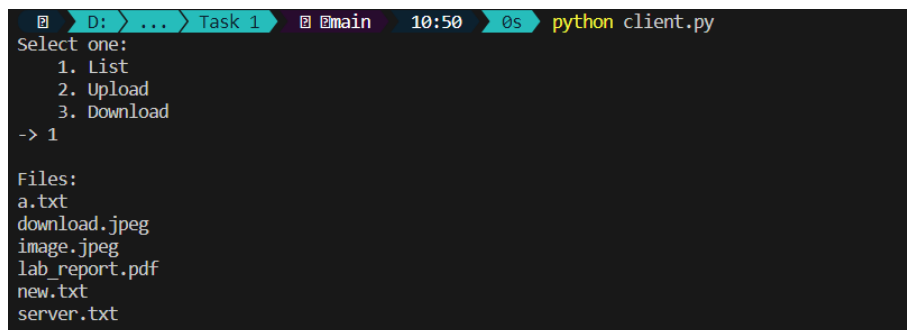
4.1 Task 1: File Transfer via Socket Programming

4.1.1 Server:

Server creates a socket and binds it to one of its ports. And it starts listening to incoming requests. When a client connection arrives, it accepts the connection and serves accordingly.

4.1.2 Client:

The client requests to connect to the server in the mentioned port. After being accepted, the client is asked what request they want to make. The client sends the request using TCP protocol and waits to receive a response.



```
D:\... Task 1 10:50 0s python client.py
Select one:
  1. List
  2. Upload
  3. Download
-> 1

Files:
a.txt
download.jpeg
image.jpeg
lab_report.pdf
new.txt
server.txt
```

Figure 1: Output for Option 1 (Showing all the files available in the directory)

```

Uploaded ./client_files\image.jpeg
Select one:
  1. List
  2. Upload
  3. Download
-> 2
upload
Enter filename: lab_report.pdf
Uploaded ./client_files\lab_report.pdf

```

Figure 2: Output for Option 2 (Client uploading a pdf)

```

Select one:
  1. List
  2. Upload
  3. Download
-> 2
upload
Enter filename: image.jpeg
Uploaded ./client_files\image.jpeg

```

Figure 3: Output for Option 2 (Client uploading a jpeg file)

```

Select one:
  1. List
  2. Upload
  3. Download
-> 3
download
Enter filename: server.txt
100%|████████████████████████████████████████████████████████████████████████████████| 73.0/73.0 [00:00<?, ?B/s]
Downloaded ./client_files\server.txt

```

Figure 4: Output for Option 3 (Client downloading a text file)

4.2 Task 2: File Transfer via HTTP

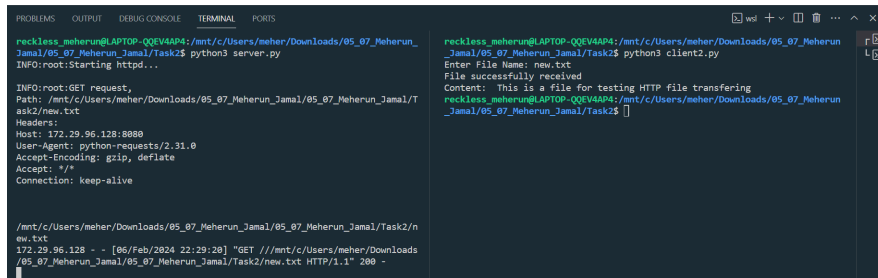
4.2.1 Server:

HTTP file transfer server works by receiving file requests from clients, sending an HTTP response message with the requested file, and providing the necessary information for the client to properly display or handle the file. When a client requests a file from the server, the server sends an HTTP response

message that includes the requested file. The response message has a status code, headers, and an optional message body that contains the file data. The headers contain information about the file such as its size, type, and encoding.

4.2.2 Client:

The client side in an HTTP communication refers to the entity that initiates the request for data from the server. The client can be a web browser, a mobile app, or any other software that needs to retrieve data from an HTTP server. On the client side, an HTTP request message is generated and sent to the server. This message includes information such as the URL of the requested resource, the HTTP method (e.g. GET or POST), and any necessary data or headers. In conclusion, the client side in an HTTP communication is responsible for initiating requests to the server, processing the response from the server, and rendering or further processing the data as necessary.



```
reckless_mehurung@LAPTOP-QQEV4AP4:/mnt/c/Users/mehur/Downloads/05_07_Meherun_Jamal/05_07_Meherun_Jamal/Task2$ python3 server.py
INFO:root:Starting httpd...

INFO:root:GET request,
Path: /mnt/c/Users/mehur/Downloads/05_07_Meherun_Jamal/05_07_Meherun_Jamal/Task2/new.txt
Headers:
Host: 172.29.96.128:8888
User-Agent: python-requests/2.31.0
Accept-encoding: gzip, deflate
Accept: */*
Connection: keep-alive

/mnt/c/Users/mehur/Downloads/05_07_Meherun_Jamal/05_07_Meherun_Jamal/Task2/new.txt
172.29.96.128 - - [06/Feb/2024 22:29:20] "GET //mnt/c/Users/mehur/Downloads/05_07_Meherun_Jamal/05_07_Meherun_Jamal/Task2/new.txt HTTP/1.1" 200 -

reckless_mehurung@LAPTOP-QQEV4AP4:/mnt/c/Users/mehur/Downloads/05_07_Meherun_Jamal/05_07_Meherun_Jamal/Task2$ python3 client2.py
Enter File Name: new.txt
File successfully received
Content: This is a file for testing HTTP file transferring
reckless_mehurung@LAPTOP-QQEV4AP4:/mnt/c/Users/mehur/Downloads/05_07_Meherun_Jamal/05_07_Meherun_Jamal/Task2$
```

Figure 5: Output for Get

Figure 6: Output for Post

5 Experience

1. We had to learn how to enable the server to handle multiple client at once.
2. We had to see some examples of how to use HttpServer package in Python

References

- [1] Multithreading in python. *GeeksforGeeks*, aug 3 2022. [Online; accessed 2023-01-25].
- [2] File transfer using tcp socket in python. *GeeksforGeeks*, apr 17 2023. [Online; accessed 2023-01-25].
- [3] Python Basics. Create a python web server. aug 31 2021.