Kuwait University
College of Engineering and Petroleum Computer Engineering Department

---

CpE-<Course No.0612356>: <Computer Networks>

Semester: <Fall 2021>

Section <No.03AD>

Assignment No. <Phase 2>

Student Name: <Yassine Serrid, Abdulrahman Mohammed >

Student Id: <2181156439, 2181147824>

Instructor Name: <Dr. Fatemah Al-Ali>

TA Name: < Eng. Shahad Alanzi >

Date: <17th Of January 2022>

Table of Contents

Table Of Figures

Table Of Tables

1. Problem Description:

In this project we are required to design our application-layer protocol using a python programming language that can manage our data in a remote server, in this project our protocol supports used to handle directories and files that are stored in a remote server, our application-layer protocol describes different types of messages sent and received, as well as the message format, syntax, and semantics, and the rules for when and how processes send and receive messages. The protocol uses either persistent or non-persistent connections that provide the user with a wider variety of functional scenarios. We are going to use our status codes to replay from the remote server to the client. Our server offers a variety of protocol-layer techniques, our protocol supports the following methods.

- List Files And Directories.
- Download File From Remote Server.
- Upload File From Remote Server.
- Delete A File From A Specific Path.
- Making New Directory In The Remote Server.
- Exit.

2. Application-layer protocol design:

At the very top of the OSI (Open Systems Interconnection) model is the application layer. It's the interface that people use to interact with the system. It helps users by providing services. It's a layer that combines human connection with software and applications to link users all over the world.

2.1. Types of messages exchanged:

Our application-layer protocol handles different kinds of messages exchanged between the client and the remote server as shown in Table 1 below.

Table 1. Types Of The Messages

Types Of The Messages	Definition
Request Messages	Messages sent by the client to initiate an action on the server.
Response Messages	The aim of the response Message is to provide the client with the resource it requested, or inform the client that the action it requested has been carried out; or else to inform the client that an error occurred in processing its request

2.2. Message format and message syntax:

We will deduce the method of designing a protocol from existing protocols such as HTTP or SMTP so we can design the message format and the message syntax. In the message format, the client must make a request that includes the type of method that the application-layer deals within the protocol, packets, and the connection type, whether it is a persistent connection or non-persistent connection. In this section we are going to design request and response messages for our application-layer protocol.

2.2.1. Request Message Format:

Table 2. Request Message Format.

ConnectionType	sp	method	sp	URL	sp	cr	lf
cr			lf				
PacketsLength	sp	PacketsSizeB	sp	cr		lf	
cr			lf				
Data							

Fields semantics:

- ConnectionType: Either the connection is Persistent or Non-Persistent. (required)
- Sp: Space (required)
- Method: List-Download-Upload-Delete-Making New Directory-Exit. (required ,in Non-Persistent Exit method is not required)
- URL: The destination of a file or folder on the remote server. (required)
- Cr: Carriage Return. (required)
- Lf: Line Feedback. (required)
- PacketLength: The length of the data. (not required)
- PacketSize: The length of the data in Bytes. (required)
- Data: Data content sent along with the message. (not required)

2.2.2. Response Message Format:

Table 3. Response Message Format.

ConnectionType	Status Code	sp	Status Code	sp	cr	lf
cr				lf		
PacketsLength		sp	PacketsSizeB	sp	cr	lf
cr				lf		
Data						

Fields Semantics:

- ConnectionType: Either the connection is Persistent or Non-Persistent. (required)
- Sp: Space (required)
- Cr: Carriage Return. (required)
- Lf: Line Feedback. (required)
- PacketLength: The length of the data. (not required)
- PacketSize: The length of the data in Bytes. (required)
- Data: Data content received along with the message. (not required)
- Status Code: Code represents the response status as shown in the table below.

2.3. Status Codes:

Based on our knowledge of different protocols such as HTTP and SMTP, we will design the status code for our protocol that can respond to all methods of the protocol as explained before the status code is carried out in the response message format. In Table 4 below we will explain the status code implemented in our project. The status codes will be stored in the server and will be send to the client to display readable message

Table 4. Status Codes.

Status Code	Replay Message
100	Operation Done Successfully.
210	Persistent Connection Opened Successfully.
220	Non-Persistent Connection Opened Successfully.
401	401 Server Is Not Ready ‘Busy’.
403	ERROR- Wrong Choice.
404	Operation Done Unsuccessfully.
405	Upload Failed.
501	The server directory is empty.
502	File/directory Path not found.
800	Persistent Connection Closed Successfully.
901	Connection Closed Successfully

3. Client's Main Menu And Persistent , Non-Persistent Connections:

From the main menu , the client will choose the type of connection, either Persistent or non- persistent. For persistent connection, this gives the user an opportunity to download more than one file at the same time and upload a file at the same time without closing the server with each process, unlike non persistent, which can download only one file and upload only one file then the connection between the server and client will be closed. In Figure 1 below shows that the server opens a connection and it's waiting for the client to connect.

```
C:\Users\Yessi\PycharmProjects\ClientServer\venv\Scripts
[STARTING] Server is starting
[LISTENING] Server is listening on 172.20.10.11:5000.
```

Figure 1. Opening Server Connection.

So now the server is waiting for the client to connect to it when we run the client we will get this main menu.

```
C:\Users\Yessi\PycharmProjects\ClientServer\ven
Welcome to 'Manage My Files' app!

Do you want the connection to be:
1 - Persistent
2 - Non-Persistent:
1
210 Persistent Connection Opened Successfully

What would you like to do ?
1- list
2- download
3- upload
4- delete
5- make new directory
6- exit
Enter your choice:
```

Figure 2. Manage My Files App Main Menu.

```
C:\Users\Yessi\PycharmProjects\ClientServer\venv\Scrip
[STARTING] Server is starting
[LISTENING] Server is listening on 172.20.10.11:5000.
[ACTIVE CONNECTIONS]
[NEW CONNECTION] ('172.20.10.11', 55201) connected..
```

Figure 3. Server Opened A Connection With A Client.

4. Functionalities Of Each Method:

4.1. List:

List method can be used to show a list of files (and directories) in the current directory or a particular directory. It does not show the files in a directory that is located inside another directory.

4.1.1. Request Message Fields For List Method:

Table 5. Request Message Fields For List Method.

ConnectionType(Persistent/Non-Persistent) (required)	sp	Method: List (required)	URL (not required)	sp	cr	lf
cr				lf		
PacketsLength(not required)	sp	PacketsSizeB(not required)		sp	cr	lf
cr				lf		
Data(not required)						

Example: (Persistent List /n lf /n lf)

4.1.2. Response Message Fields For List Method:

Table 6. Response Message Fields For List Method.

ConnectionType(Persistent/Non-Persistent) (required)	Status Code(required)	sp	Status Code(required)	sp	cr	lf
cr			lf			
PacketsLength(not required)		sp	PacketsSizeB(not required)	sp	cr	lf
cr			lf			
Data(required)						

Example: Listed successfully:

```
Persistent 210 100 /n lf
/n lf
PacketsLength 64 /n lf
/n lf
Data1.txt
Data2.txt
Data3.txt
```

Example: Listed Unsuccessfully:

```
Persistent 210 100 /n lf
/n lf
PacketsLength 0 /n lf
/n lf
```

4.2. Download:

Download method allows the user to download a specific file to a specific directory in the remote server. The required parameter here is the name of the file and the directory after listing the directory using list method.

4.2.1. Request Message Fields For Download Method:

Table 7. Request Message Fields For Download Method.

ConnectionType(Persistent/Non-Persistent) (required)	sp	Method: Download (required)	URL (Optional)	sp	cr	lf
cr			lf			
PacketsLength(not required)	sp	PacketsSizeB(not required)		sp	cr	lf
cr			lf			
Data(not required)						

Example: (Persistent Download /Yessi /n lf /n lf)

4.1.2. Response Message Fields For Download Method:

Table 8. Response Message Fields For Download Method.

ConnectionType(Persistent/Non-Persistent) (required)	Status Code(required)	sp	Status Code(required)	sp	cr	lf
cr			lf			
PacketsLength(Required)	sp	PacketsSizeB(required)		sp	cr	lf
cr			lf			
Data(required)						

Example: Downloaded successfully:

Persistent 210 100 /n lf /n lf PacketsLength 64 /n lf /n lf

Example: Downloaded unsuccessfully:

Persistent 210 404 /n lf /n lf PacketsLength 0 /n lf /n lf	Persistent 210 502 /n lf /n lf PacketsLength 0 /n lf /n lf

4.3. Upload:

In your right pane, we need to browse to the provided directory. This directory serves as the foundation of your website. The content of the uploaded file to the remote server should be identical to the data of the original file in the client data. The required parameter here is the path of the directory inside the remote server. The replay message will be in the status code, and it will be displayed as a readable message for the client.

4.3.1. Request Message Fields For Upload Method:

Table 9. Request Message Fields For Upload Method.

ConnectionType(Persistent/Non-Persistent) (required)	sp	Method: Upload (required)	URL (Optional)	sp	cr	lf
cr			lf			
PacketsLength(required)	sp	PacketsSizeB(required)		sp	cr	lf
cr			lf			
Data(required)						

Example: (Persistent Upload /Yessi/Data1.txt /n lf /n lf PacketsLength 32 \n lf \n lf Networks)

4.3.2. Response Message Fields For Upload Method:

Table 10. Response Message Fields For Upload Method.

ConnectionType(Persistent/Non-Persistent) (required)	Status Code(required)	sp	Status Code(required)	sp	cr	lf
cr			lf			
PacketsLength(not required)		sp	PacketsSizeB(not required)	sp	cr	lf
cr			lf			
Data(required)						

Example: Uploaded Successfully:

Persistent 210 100 /n lf /n lf PacketsLength 0 /n lf /n lf

Example: Uploaded Unsuccessfully:

Persistent 210 404 /n lf /n lf PacketsLength 0 /n lf /n lf	Persistent 210 502 /n lf /n lf PacketsLength 0 /n lf /n lf

4.4. Delete:

The delete method may be used to delete a file from a given path in the remote server. The required parameters here are the path of the file either if it's inside the main server_data or a directory inside server_data.

4.4.1. Request Message Fields For Delete Method:

Table 11. Response Message Fields For Delete Method.

ConnectionType(Persistent/Non-Persistent) (required)	sp	Method: Delete (required)	URL (required)	sp	cr	lf
cr			lf			
PacketsLength(not required)	sp	PacketsSizeB(not required)		sp	cr	lf
cr			lf			
Data(not required)						

Example: (Persistent Delete /Yessi/Data1.txt /n lf /n lf)

4.4.2. Response Message Fields For Delete Method:

Table 12. Response Message Fields For Delete Method.

ConnectionType(Persistent/Non-Persistent) (required)	Status Code(required)	sp	Status Code(required)	sp	cr	lf
cr			lf			
PacketsLength(not required)		sp	PacketsSizeB(not required)	sp	cr	lf
cr			lf			
Data(required)						

Example: Deleted successfully:

Persistent 210 100 /n lf /n lf PacketsLength 0 /n lf /n lf

Example: Deleted Unsuccessfully:

Persistent 210 404 /n lf /n lf PacketsLength 0 /n lf /n lf	Persistent 210 502 /n lf /n lf PacketsLength 0 /n lf /n lf

4.5. Make Directory:

The create directory function allows the user to build a new folder in the remote server. The required parameters here are the name of the file and the destination path.

4.5.1. Request Message Fields For Make Directory Method:

Table 13. Response Message Fields For Make Directory Method.

ConnectionType(Persistent/Non-Persistent) (required)	sp	Method: Makedirectory (required)	URL (required)	sp	cr	lf
cr			lf			
PacketsLength(not required)	sp	PacketsSizeB(not required)		sp	cr	lf
cr			lf			
Data(not required)						

Example: (Persistent Makedirectory /Yessi1 /n lf /n lf)

4.5.2. Response Message Fields For Make Directory Method:

Table 14. Response Message Fields For Make Directory Method.

ConnectionType(Persistent/Non-Persistent) (required)	Status Code(required)	sp	Status Code(required)	sp	cr	lf
cr			lf			
PacketsLength(not required)		sp	PacketsSizeB(not required)	sp	cr	lf
cr			lf			
Data(required)						

Example: Make Directory successfully:

Persistent 210 100 /n lf /n lf PacketsLength 0 /n lf /n lf

Example: Make Directory Unsuccessfully:

Persistent 210 404 /n lf /n lf PacketsLength 0 /n lf /n lf	Persistent 210 502 /n lf /n lf PacketsLength 0 /n lf /n lf

4.6. Exit:

For persistent connection only you can use the exit method. Before ending the connection, the client will wait for the server to send a reply message.

4.5.1. Request Message Fields For Exit Method:

Table 15. Response Message Fields For Exit Method.

ConnectionType(Persistent/Non-Persistent) (required)	sp	Method: upload (required)	URL (Optional)	sp	cr	lf
cr				lf		
PacketsLength(not required)	sp	PacketsSizeB(not required)		sp	cr	lf
cr				lf		
Data(not required)						

Example: (Persistent exit /n lf /n lf)

4.5.2. Response Message Fields For Exit Method:

Table 16. Response Message Fields For Exit Method.

ConnectionType(Persistent/Non-Persistent) (required)	Status Code(required)	sp	Status Code(required)	sp	cr	lf
cr				lf		
PacketsLength(not required)		sp	PacketsSizeB(not required)	sp	cr	lf
cr				lf		
Data(required)						

Example Exiting successfully:

Persistent 210 **901** /n lf /n lf PacketsLength 0 /n lf /n lf	Persistent 210 **800** /n lf /n lf PacketsLength 0 /n lf /n lf

Example Exiting Unsuccessfully:

Persistent 210 100 /n lf /n lf PacketsLength 0 /n lf /n lf

Appendix

Appendix A: Wireshark And Outputs

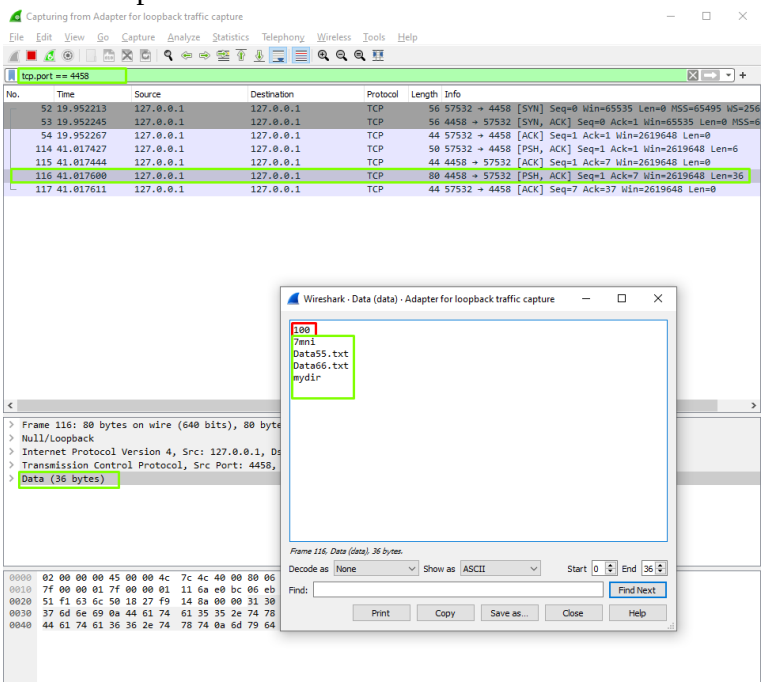

Figure 4. Wireshark Message For Listing.

```
What Would You Like To Do ?
1- List
2- Download
3- Upload
4- Delete
5- Make New Directory
6- exit
Enter Your Choice: 1
Enter the path (/for current directory): /


7mni
Data55.txt
Data66.txt
mydir
```

Figure 5. Listing From The Server.

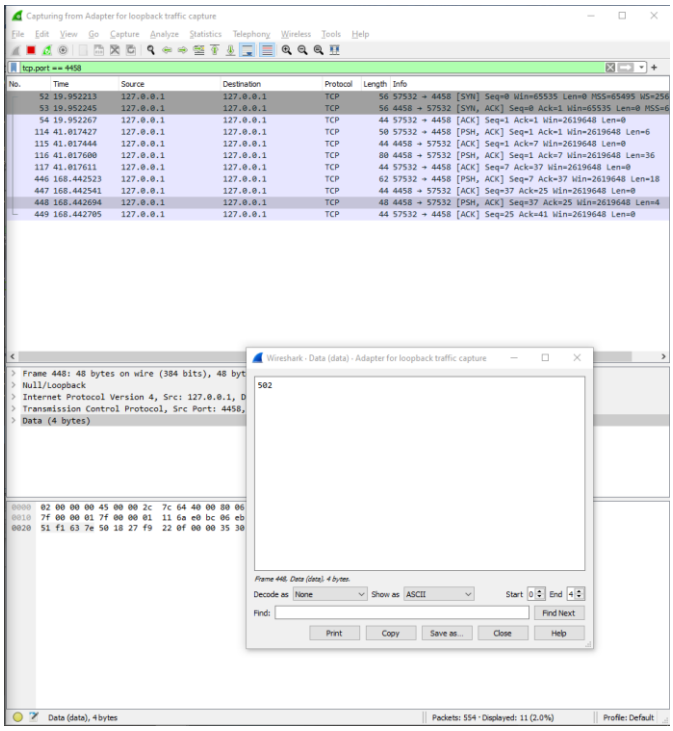

Figure 6. Downloading A File Not In The Directory.

```
What Would You Like To Do ?
1- List
2- Download
3- Upload
4- Delete
5- Make New Directory
6- exit
Enter Your Choice: 2
Enter The Name Of The File: Data2.txt

File/Directory Path Not Found.
```

Figure 7. Downloading The File From The Server.

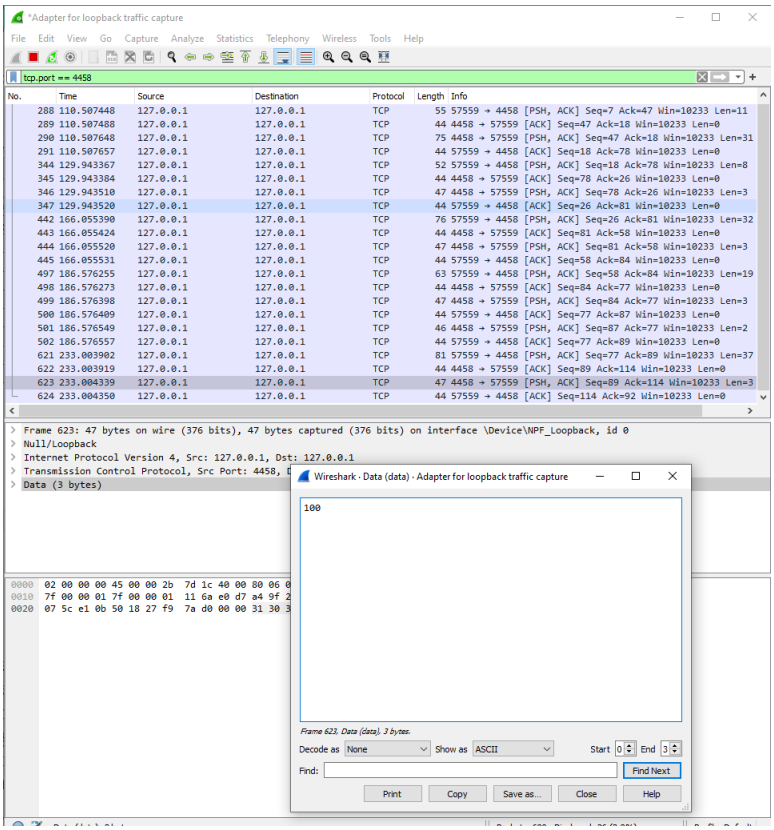

Figure 8. Wireshark For Upload A File From The Client To The Server.

```
What Would You Like To Do ?
1- List
2- Download
3- Upload
4- Delete
5- Make New Directory
6- exit
Enter Your Choice: 3
Enter The Name Of The File: Data66.txt
Enter The Destination Path: /mydir/mydir2

Operation Done Successfully.
```

Figure 9. Upload A File From The Client To The Server.

Appendix B: Meeting Minutes

Meeting Minutes 1

Date & Time	20/12 - 6:00 PM
Meeting Held By	Yassin Serrid – Abdulrahman Mohammed
Meeting Attendance	Yassin Serrid – Abdulrahman Mohammed

1. Meeting Points
   - Understand Project
   - Choose Coding Language
   - Identify Objectives

2. Meeting Actions
   - Researched Definitions
   - Collected Resources
   - Document Plans

3. Next Meeting Actions
   - Start writing the report
   - Start Coding
   - Reading about different protocols like HTTP, UDP.

Meeting Minutes 2

Date & Time	25/12 - 4:00 PM
Meeting Held By	Yassin Serrid – Abdulrahman Mohammed
Meeting Attendance	Yassin Serrid – Abdulrahman Mohammed

1. Meeting Points
   - Start writing the report
   - Start Coding
   - Reading about different protocols like HTTP, UDP

2. Meeting Actions
   - Found a useful resource in GitHub.
   - Start Writing the application-layer protocol design and the status codes we need in our protocol.
   - Read more about functionalities we need in our protocol

3. Next Meeting Actions
   - Start Modifying the resource we found in GitHub.
   - Designing the client and server using python.
   - Collect resources for functionalities for each method

Meeting Minutes 3

Date & Time	30/12 - 5:30 PM - 31/12 - 5:30 PM - 14/1 - 7:30 AM
Meeting Held By	Yassin Serrid – Abdulrahman Mohammed
Meeting Attendance	Yassin Serrid – Abdulrahman Mohammed

1. Meeting Points
   - Start Modifying the resource we found in GitHub.
   - Designing the client and server using python.
   - Collect resources for functionalities for each method.

2. Meeting Actions
   - Implanting the client and server.
   - Finalizing the report
   - Run our code using Wireshark

5. Resources:

1- Gallagher, J. (2021, February 21). Python List Files in a Directory: Step-By-Step Guide. Career Karma.https://careerkarma.com/blog/python-list-files-in-directory/

2- GeeksforGeeks. (2021, June 3). Python - List Files in a Directory. https://www.geeksforgeeks.org/python-list-files-in-a-directory/

3- Take online courses. earn college credit. Research Schools, Degrees & Careers. Study.com | Take OnlineCourses. Earn College Credit. Research Schools, Degrees & Careers. (n.d.). Retrieved December 20, 2021, from https://study.com/academy/lesson/application-layer-protocols-types-uses.html

4- GitHub - nikhilroxtomar/Multithreaded-File-Transfer-using-TCP-Socket-in-Python: A multithreaded file transfer client-server program build using a python programming language. (2022). Retrieved 18 January 2022, from https://github.com/nikhilroxtomar/Multithreaded-File-Transfer-using-TCP-Socket-in-Python