Rapoo Oratile RPXORA001

Cebolenkosi Sithole STHCEB002

Comfort Twala TWLCOM001

CSC3002 – Networks Assignment 1

1.Reliability constraints

- The server/client provides feedback to the sender for all the messages received to confirm that the messages were delivered.
- Every request has a response which alerts the client of the process status and all server side errors are reported back to the client
- We included a file verification system that ensures that when uploading/downloading the server/client can verify the files received and it uses a filehash and the byte size of the file.
- The Protocol uses Md5 checksum to verify that the files sent and received are the same for example when uploading files from the client to the server ,the client sends unique filehash that is generated from the file bytes and after the server receives the file it will also generate a filehash from the files received there after compare its filehash with the client filehash and if they are the same the file is reliably received and if not it is damaged
- The protocol uses buffering when receiving large files, were it receives the file bytes as a stream of data byte by byte and it only stops receiving once all the data is sent.
- The protocol ensures that the sender lets the receiver know the file size they are receiving prior to receiving any file, this information is used to verify if the receiver received all the file bytes
- Some reliability is managed by the use of TCP sockets to communicate which provides reliable transport ensuring that every message is received and it also recovers any damaged data

1.3Protocol design & specification

There are 3 types of requests (Download , Upload and Listall) and each have a specific message format and responses

The protocol uses two types of messages Control messages which manage the dialog between client and server. Data transfer messages which transfer data from client to the server

A) DOWNLOAD

message formats/structure

The client sends request to the server to download a file, the server receives the request(checks file access and existence) and then sends the client download header to help the client receive the appropriate file , the client receives the header and prepares to receive the file , the server sends the file and the client receives the file(verify file contents),they both close connection

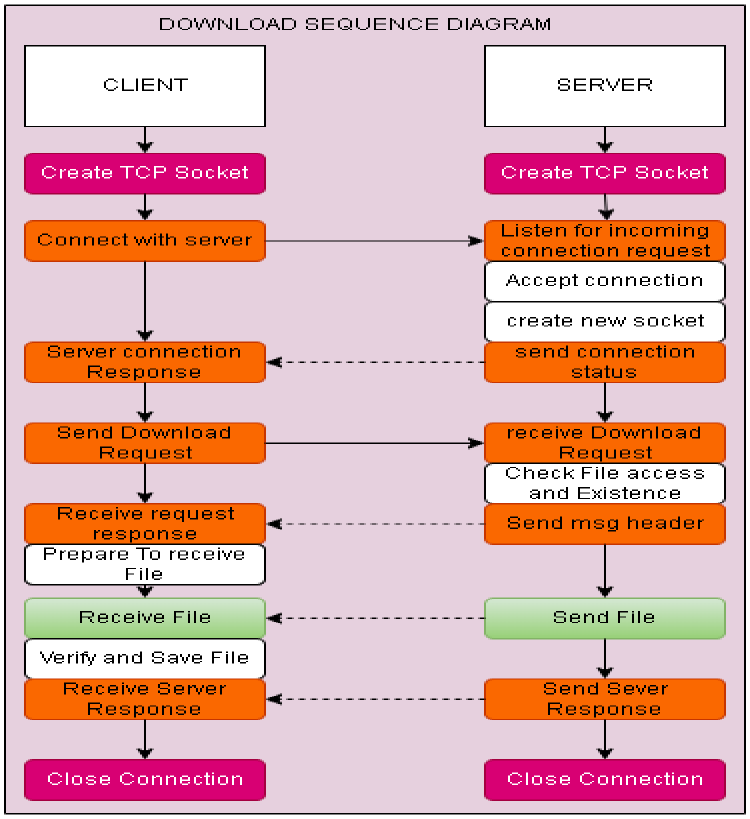
- Download Request Message(from client to server) in form “request,filename,protected,key"→ request is Download, filename is name of file to download, protected is the visibility status of the file (True or False), key is to be used to unlock the file and verify access if file is protected

Request	Filename	Protected	Key

- Download Helper header(from server to client) in form of ”fileSize,Fileverifyhash” → fileSize is used to check if all the file was downloaded or send to client , Fileverifyhash is md5 checksum generated hash and is used to check if the file was not corrupted or if the client has received the same file sent by the server.

Filesize	Fileverifyhash

Sequence diagram and ALL POSSIBLE SCENARIOS when Downloading



1) **Download unprotected file** →the server will send the file and message ” File sent successfully”

2) **Download protected file with wrong key**→ the server will not send any files and send error message “ Access Denied, unauthorized Access wrong key”

3) **Download file that does not exist**→ the server will not send any files and send error message” File does not exist"

4) **Download protected file with right key**→ the server will send the file and message “File sent successfully**”**

B) UPLOAD

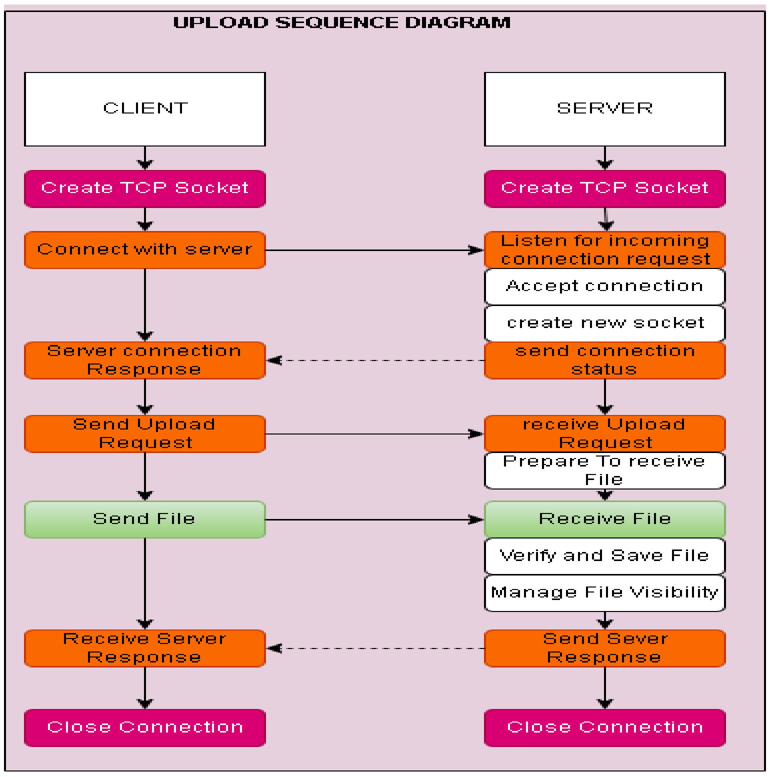
message formats/structure

Request	**Filename**	**FileSize**	**Protected**	**Fileverifyhash**
BODY(file)				

The client sends request to the server to upload a file, the server receives the request(prepares to receive file) , client sends the file to the server and the server receives the file and checks if the file received is the same as the one that was sent by the client and then the server informs the client about the upload process status and they both close connection.

- UPload Request Message(from client to server) in form "request,filename,filesize,protected,fileverifyhash"→ request is UPload, filename is name of file that is going to be uploaded to the server, filesize it is in bytes and will help the server check if all the file bytes were received, protected is the visibility status of the file (True or False) and if true the file is private and requires a key, fileverifyhash is md5 checksum generated hash and is used to check if the file was not corrupted or if the server has received the same file sent by the client.

Sequence diagram and ALL POSSIBLE SCENARIOS when Uploading file to the server



1) Uploading Protected file → the server will receive(save and verify) file and generate key for the file , make the file accessible by the key and send response to client"File uploaded,{File name} , {key}"
2) Uploading unprotected file→server will receive(save and verify) file and send repond to the client "File uploaded,{File name}"
3) Uploading File that got corrupted after sending → the server will verification will be false and it will send error message " File corrupted"

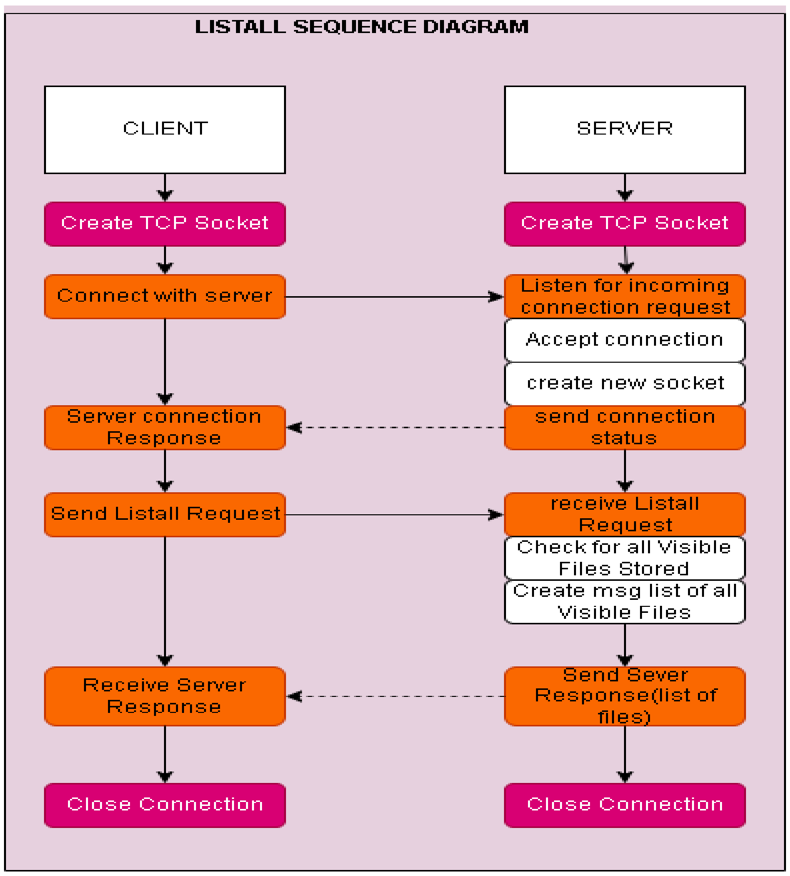
B) Listall

message formats/structure

The client sends request to the server to list all files stored in the server file system, the server receives the request and queries for all visible(unprotected) files in the server file system and sends the list all of all files as message to the client

Request

- Listall Request Message(from client to server) in form "Request"→ Request is LISTALL , this will inform the server that we want to list all files that are stored in the servers file system



2.Description of system functionality and features

The system consists of a client-server architecture that allows clients to upload, download and list files on the server. The server provides a simple file storage system that allows clients to securely upload and download files.

cllient Functionality

• Connect to server using specified hostname and port.

• Upload files to server, with option to compress and protect with key.

• Download files from server, with option to decompress and provide key for protected

files.

• List all files on server.

Server Functionality

• Receive files from client and store them in specified directory.

• Generates unique key for the locked file

• Serve files to client upon request.

• Protect files with a key that the client must provide to access the file.

• Log activity to specified file (openFilesList), creates a record of each new open file.

INDIVIDUAL: Cebolenkosi Sithole STHCEB002

STHEB002_client.py

Additional Features

• Compression of files before upload to server.(STHCEB002_client)

• Server simple file system, All user uploaded data is stored in server/storage to minimize

accidentally reading source code files to client.

INDIVIDUAL: ORATILE RAPOO RPXORA001

RPXORA001_client.py

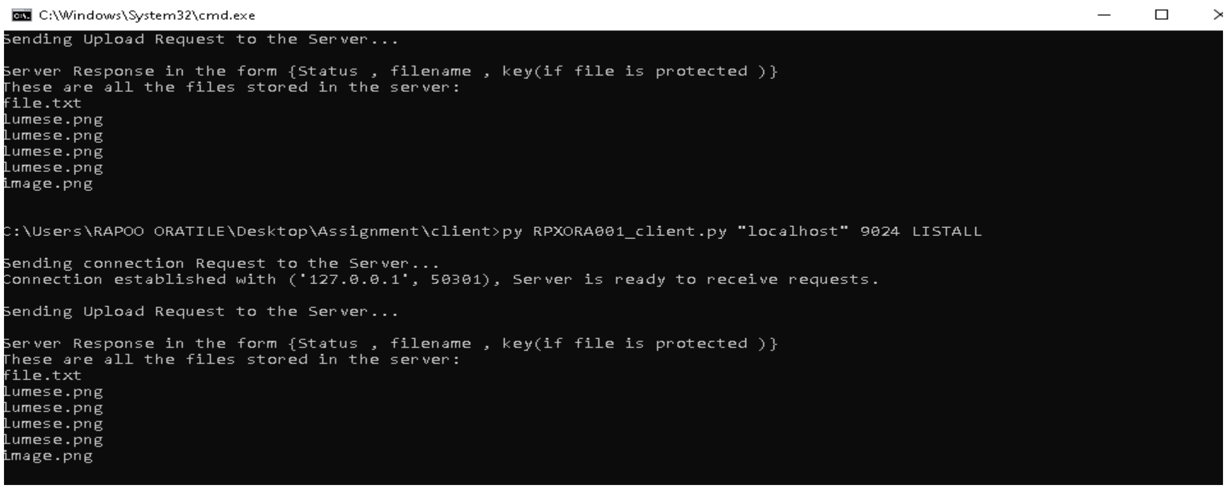
- This program takes command line arguments in the form serverIPAdress, sever portnumber , Request type (where the Request type could be “DOWNLOAD”, “UPLOAD” and “LISTALL”)
- my client allows the user to create request messages to the server using a simple dialog between the client and the user.
- The programs handles error messages from the server and informs the user every time an error occurs
- The client has file validation to verify that the client receives undamaged files

INDIVIDUAL: ComfortTwalaClient

- IT implement the three basic functionalities Download , Upload and ListAll and it also uses a fixed serverIP and Portnumber

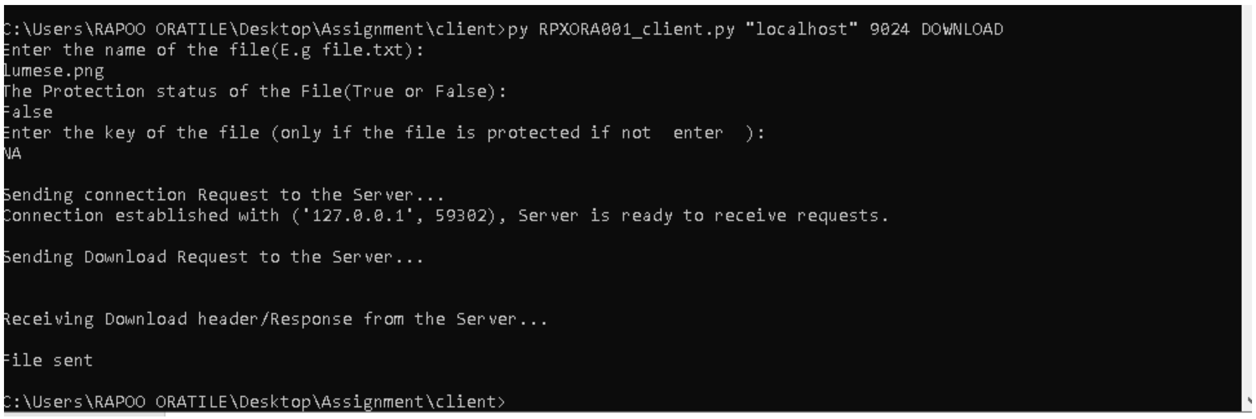
SCREENSHOTS

ListAll



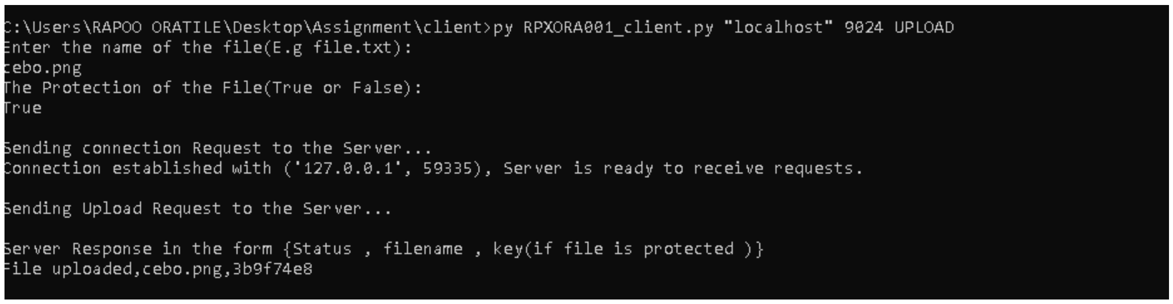
```
C:\Windows\System32\cmd.exe
ending Upload Request to the Server...

erver Response in the form {Status , filename , key(if file is protected )}
hese are all the files stored in the server:
ile.txt
umese.png
umese.png
umese.png
umese.png
mage.png


:\Users\RAPOO ORATILE\Desktop\Assignment\client>py RPXORA001_client.py "localhost" 9024 LISTALL

ending connection Request to the Server...
onnection established with ('127.0.0.1', 50301), Server is ready to receive requests.

ending Upload Request to the Server...

erver Response in the form {Status , filename , key(if file is protected )}
hese are all the files stored in the server:
ile.txt
umese.png
umese.png
umese.png
umese.png
mage.png
```

Download



```
:\Users\RAPOO ORATILE\Desktop\Assignment\client>py RPXORA001_client.py "localhost" 9024 DOWNLOAD
nter the name of the file(E.g file.txt):
umese.png
he Protection status of the File(True or False):
alse
nter the key of the file (only if the file is protected if not  enter  ):
A

ending connection Request to the Server...
onnection established with ('127.0.0.1', 59302), Server is ready to receive requests.

ending Download Request to the Server...


eceiving Download header/Response from the Server...

ile sent

:\Users\RAPOO ORATILE\Desktop\Assignment\client>
```

Upload



```
:\Users\RAPOO ORATILE\Desktop\Assignment\client>py RPXORA001_client.py "localhost" 9024 UPLOAD
nter the name of the file(E.g file.txt):
ebo.png
he Protection of the File(True or False):
rue

ending connection Request to the Server...
onnection established with ('127.0.0.1', 59335), Server is ready to receive requests.

ending Upload Request to the Server...

erver Response in the form {Status , filename , key(if file is protected )}
ile uploaded,cebo.png,3b9f74e8
```