Practical Work 1: TCP File Transfer

Name: Bui Truong An - 22BA13001

1 Goal

The objective of this practical work is to implement a one-to-one file transfer over TCP/IP using a command-line interface. The system is based on the provided chat architecture and must include:

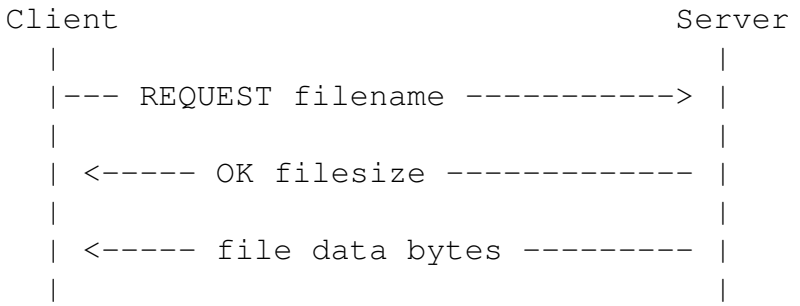
- One server
- One client
- Communication through TCP sockets

2 Protocol Design

We designed a simple application-level protocol for file transfer:

- The client sends a `REQUEST <filename>` command to the server.
- The server checks if the file exists.
- If the file exists, the server responds with `OK <filesize>` and starts sending raw bytes.
- If the file does not exist, the server responds `ERROR`.

Protocol Diagram



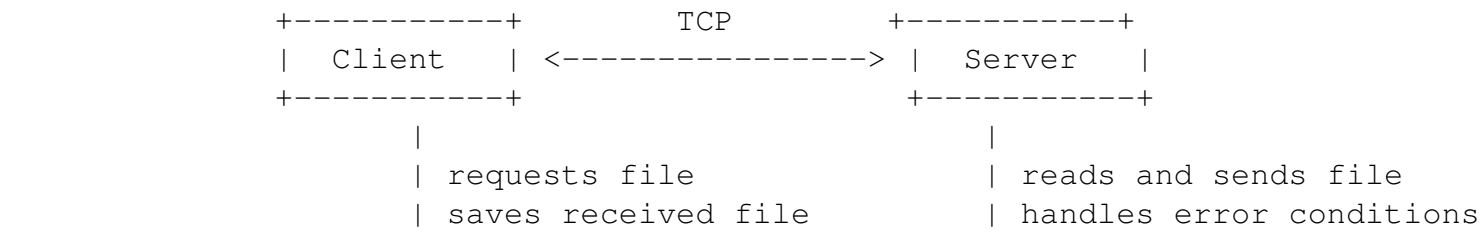
3 System Organization

Architecture Overview

The system consists of two standalone programs:

- **Server**: waits for TCP connections and sends files.
- **Client**: connects to the server and receives files.

System Diagram



4 Implementation

Below are simplified code snippets showing the implementation idea.

4.1 Server Code Snippet (Python Example)

```
import socket, os

s = socket.socket()
s.bind(("0.0.0.0", 9000))
s.listen(1)

conn, addr = s.accept()
request = conn.recv(1024).decode()

cmd, filename = request.split()

if cmd == "REQUEST" and os.path.isfile(filename):
    size = os.path.getsize(filename)
    conn.send(f"OK␣{size}".encode())

    with open(filename, "rb") as f:
        conn.sendall(f.read())
```

```
else:
    conn.send(b"ERROR")

conn.close()
s.close()
```

4.2 Client Code Snippet (Python Example)

```
import socket

filename = "example.txt"
s = socket.socket()
s.connect(("127.0.0.1", 9000))

s.send(f"REQUEST␣{filename}".encode())
response = s.recv(1024).decode()

status, value = response.split()

if status == "OK":
    size = int(value)
    data = b""
    while len(data) < size:
        data += s.recv(4096)

    with open("received_" + filename, "wb") as f:
        f.write(data)

s.close()
```