TCP File Transfer – Practical Work 1

Group X – Distributed Systems

December 1, 2025

1 Introduction

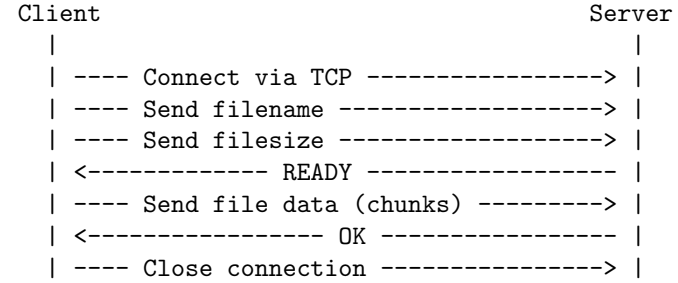
This practical work aims to develop a simple file transfer system using TCP/IP in a command-line interface. The system includes one server and one client using sockets.

2 Protocol Design

Our protocol is based on a simple request–response model:

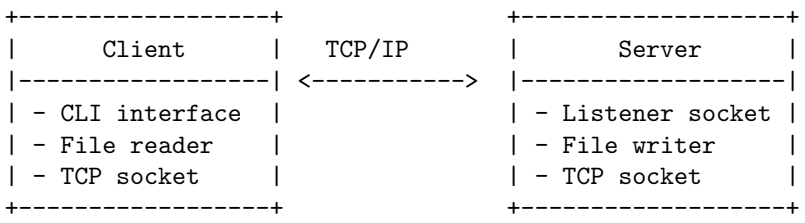
1. Client connects to server.
2. Client sends filename.
3. Client sends file size.
4. Server replies `READY`.
5. Client sends file data in binary chunks.
6. Server stores the file and responds `OK`.

Figure **??** illustrates the protocol flow.



3 System Organization

Figure **??** shows the architecture.



4 Implementation

4.1 Server Code

```
import socket

HOST = "0.0.0.0"
PORT = 5001

server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server.bind((HOST, PORT))
server.listen(1)

conn, addr = server.accept()

filename = conn.recv(1024).decode().strip()
filesize = int(conn.recv(1024).decode().strip())

conn.send(b"READY")

with open(filename, "wb") as f:
    received = 0
    while received < filesize:
        data = conn.recv(4096)
        if not data:
            break
        f.write(data)
        received += len(data)

conn.send(b"OK")
conn.close()
```

4.2 Client Code

```
import socket
import os

HOST = "127.0.0.1"
PORT = 5001

filename = "test.txt"
filesize = os.path.getsize(filename)

client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client.connect((HOST, PORT))

client.send((filename + "\n").encode())
client.send((str(filesize) + "\n").encode())

if client.recv(1024).decode() != "READY":
    client.close()

with open(filename, "rb") as f:
    data = f.read(4096)
    while data:
        client.send(data)
        data = f.read(4096)

print(client.recv(1024).decode())
client.close()
```

5 Work Distribution

- Member A: Protocol design, server implementation
- Member B: Client implementation, testing
- Member C: LaTeX report, diagrams

6 Conclusion

We successfully implemented a TCP file transfer system using one client and one server. The protocol works reliably for text and binary files. The system demonstrates basic distributed communication using sockets.