

RPC File Transfer Practical Work

Ha Tan Minh

1 Introduction

The purpose of this practical is to implement a file transfer system using Remote Procedure Call (RPC). The system involves an XML-RPC server and client implemented in Python, where the server receives and stores files sent by the client.

2 Protocol Design

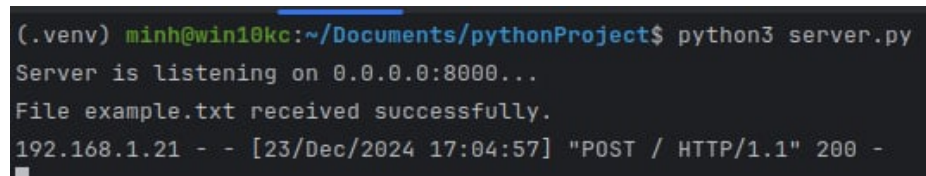
The file transfer protocol using RPC involves:

- Exposing a remote procedure on the server for receiving files.
- The client encoding the file in Base64 and sending it to the server using the remote procedure.
- The server decoding the file and saving it.

3 System Interaction

Laptop 1 (Server)

Laptop 2 (Client)

A terminal window with a dark background and light-colored text. The text shows the execution of a Python script named 'server.py' in a virtual environment. The output indicates the server is listening on port 8000, successfully receives a file named 'example.txt', and logs an incoming POST request from IP 192.168.1.21 on December 23, 2024, at 17:04:57.

```
(.venv) minh@win10kc:~/Documents/pythonProject$ python3 server.py
Server is listening on 0.0.0.0:8000...
File example.txt received successfully.
192.168.1.21 - - [23/Dec/2024 17:04:57] "POST / HTTP/1.1" 200 -
```

Figure 1: Server

```
minh@win10kc:~/PycharmProjects/ds2025_prac/practical2_ds2025$ python3 client.py
File example.txt received successfully.
```

Figure 2: Client

4 Implementation

The implementation involves two Python scripts for the server and client.

4.1 Server Code

The server listens for incoming remote procedure calls and saves the file:

```
from xmlrpc.server import SimpleXMLRPCServer
import base64

def receive_file(filename, filedata):
    # Decode the file data
    file_bytes = base64.b64decode(filedata)
    with open(filename, 'wb') as file:
        file.write(file_bytes)
    print(f"File {filename} received successfully.")
    return f"File {filename} received successfully."

if __name__ == "__main__":
    server_ip = "0.0.0.0"
    server_port = 8000
    server = SimpleXMLRPCServer((server_ip, server_port))
    print(f"Server is listening on {server_ip}:{server_port}...")

    server.register_function(receive_file, "receive_file")
    server.serve_forever()
```

4.2 Client Code

The client connects to the server and sends the file:

```
import xmlrpc.client
import base64

def send_file(server_url, filename):
    with open(filename, 'rb') as file:
        file_bytes = file.read()
        file_data = base64.b64encode(file_bytes).decode('utf-8')

    server = xmlrpc.client.ServerProxy(server_url)
```

```
response = server.receive_file(filename, file_data)
print(response)

if __name__ == "__main__":
    server_url = "http://192.168.1.20:8000/"
    filename = "example.txt"
    send_file(server_url, filename)
```

5 Results

The RPC-based file transfer system was successfully tested. The following results were obtained:

- File transferred: 'example.txt'
- File size: 58 B
- Transfer time: 1 second

6 Roles

Contributed to this project:

- Ha Tan Minh: Developed the server script (laptop 1).
- Ha Tan Minh: Developed the client script (laptop 2).
- Ha Tan Minh: Prepared the report in LaTeX.