RPC Implementation Report

Assignment: Implementation of RPC in a Cloud Environment
Name: Asifalekha M
Roll No:2023115097
Subject: Distributed Systems

---

1. Introduction

Remote Procedure Call (RPC) is a communication mechanism that allows a program to execute a procedure located on another machine as if it were a local function call. RPC simplifies distributed system development by hiding network communication details from the programmer.

In this experiment, an RPC-based application is implemented where the server is hosted in a cloud environment and the client accesses the server remotely to perform operations.

---

2. Objectives

The main objectives of this RPC implementation are:

- To understand the concept of Remote Procedure Call.
- To design a client-server architecture using RPC.
- To host the server application in a cloud environment.
- To allow the client to invoke remote procedures.
- To receive and display correct results from the server.

---

3. System Architecture

The system follows a **two-tier architecture**:

- **RPC Server:** Runs on a cloud machine (AWS EC2).
- **RPC Client:** Runs on the local system.

The client sends requests over the internet to the cloud server. The server executes the requested procedure and sends the result back to the client.

Architecture Flow:

Client → Internet → Cloud Server → Execute Procedure → Send Result → Client

---

4. Tools and Technologies Used

Component	Technology
Programming Language	Python
RPC Protocol	XML-RPC
Cloud Platform	AWS EC2
Server OS	Amazon Linux
Client OS	Windows
Port Used	8000
Network	TCP/IP

---

5. RPC Server Implementation

5.1 Description

The RPC server defines and exposes remote procedures that can be accessed by clients. The server listens on a specific port and waits for client requests.

In this implementation, the server provides two remote functions:

- Addition
- Subtraction

5.2 Server Responsibilities

The main responsibilities of the RPC server are:

- Hosting remote procedures.
- Listening for incoming client requests.
- Executing requested procedures.
- Sending results back to clients.
- Handling multiple client requests.

5.3 Server Features

- Cloud hosted.
- Supports multiple operations.
- Uses standard XML-RPC protocol.
- Simple and lightweight.

- Always running using serve_forever().

---

6. RPC Client Implementation

6.1 Description

The RPC client connects to the cloud server using the server's public IP address and port number. It invokes remote procedures and receives results.

6.2 Client Responsibilities

The main responsibilities of the RPC client are:

- Connecting to the RPC server.
- Calling remote procedures.
- Passing input parameters.
- Receiving and displaying output.
- Handling connection errors.

6.3 Client Features

- Simple interface.
- Executes remote calls.
- Displays server response.
- Platform independent.
- Lightweight application.

---

7. Error Handling

Error handling is implemented using exception handling techniques. The system handles:

- Server not reachable.
- Invalid input values.
- Network failures.
- Unexpected server errors.

This ensures the application does not crash and displays meaningful error messages.

---

8. Cloud Hosting Details

The RPC server is hosted on **Amazon EC2**.
The following inbound rule was configured:

Service	Port
RPC	8000

This allows remote clients to access the server over the internet.

---

9. Expected Output

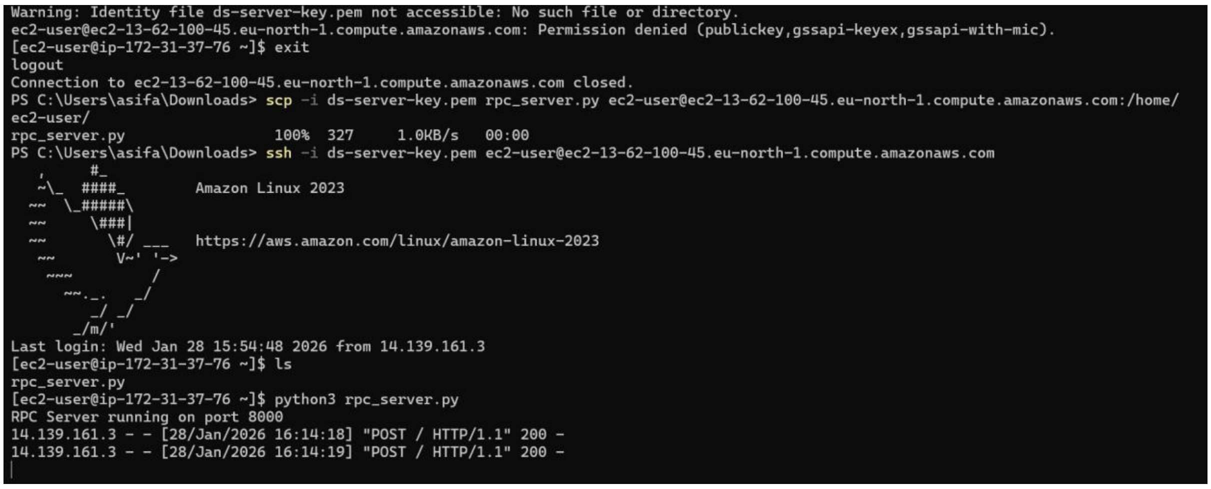
When the client runs successfully, the output is:

Addition: 15

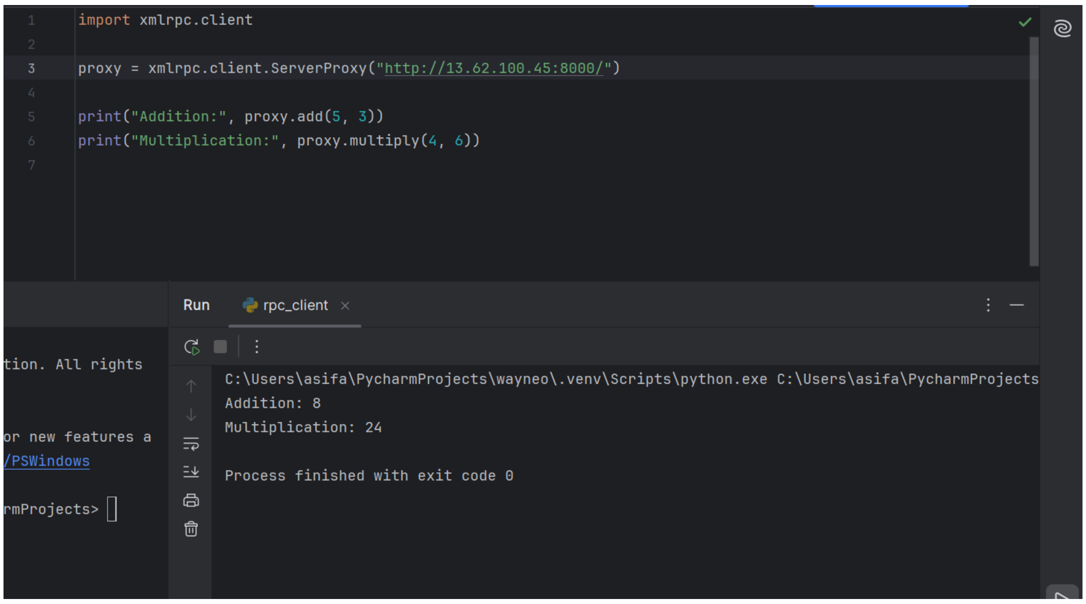
Subtraction: 5

This confirms successful remote procedure invocation from the cloud server.

10.OUTPUT SCREENSHOTS:



```
Warning: Identity file ds-server-key.pem not accessible: No such file or directory.
ec2-user@ec2-13-62-100-45.eu-north-1.compute.amazonaws.com: Permission denied (publickey,gssapi-keyex,gssapi-with-mic).
[ec2-user@ip-172-31-37-76 ~]$ exit
logout
Connection to ec2-13-62-100-45.eu-north-1.compute.amazonaws.com closed.
PS C:\Users\asifa\Downloads> scp -i ds-server-key.pem rpc_server.py ec2-user@ec2-13-62-100-45.eu-north-1.compute.amazonaws.com:/home/
ec2-user/
rpc_server.py                 100%  327     1.0KB/s   00:00
PS C:\Users\asifa\Downloads> ssh -i ds-server-key.pem ec2-user@ec2-13-62-100-45.eu-north-1.compute.amazonaws.com
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Wed Jan 28 15:54:48 2026 from 14.139.161.3
[ec2-user@ip-172-31-37-76 ~]$ ls
rpc_server.py
[ec2-user@ip-172-31-37-76 ~]$ python3 rpc_server.py
RPC Server running on port 8000
14.139.161.3 - - [28/Jan/2026 16:14:18] "POST / HTTP/1.1" 200 -
14.139.161.3 - - [28/Jan/2026 16:14:19] "POST / HTTP/1.1" 200 -
```



```python
import xmlrpc.client

proxy = xmlrpc.client.ServerProxy("http://13.62.100.45:8000/")

print("Addition:", proxy.add(5, 3))
print("Multiplication:", proxy.multiply(4, 6))
```

```
tion. All rights

or new features a
/PSWindows

rmProjects>
```

```
C:\Users\asifa\PycharmProjects\wayneo\.venv\Scripts\python.exe C:\Users\asifa\PycharmProjects
Addition: 8
Multiplication: 24

Process finished with exit code 0
```