RYERSON UNIVERSITY

Faculty of Engineering, Architecture and Science

Department of Electrical and Computer Engineering

Course Number	892
Course Title	Distributed and Cloud Computing
Semester/Year	W2024

Instructor	Dr. Muhammad Jaseemuddin

Lab No.	2

Lab Title	gRPC

Submission Date	March 10th, 2024
Due Date	March 10th, 2024

Student Name	Student ID	Signature*
Abdulrehman Khan	500968727	A.K.

**By signing above you attest that you have contributed to this written lab report and confirm that all work you have contributed to this lab report is your own work.*

Introduction

The objective of this lab was to implement communication between multiple rovers and ground control using gRPC (Google Remote Procedure Call). The system consists of a gRPC server (ground control) and multiple gRPC clients (rovers). Each rover is responsible for various tasks such as retrieving map data, receiving commands, obtaining mine information, sharing mine PINs, and notifying the server about execution status. The communication between the server and clients is facilitated through protocol buffers.

Implementation Overview:

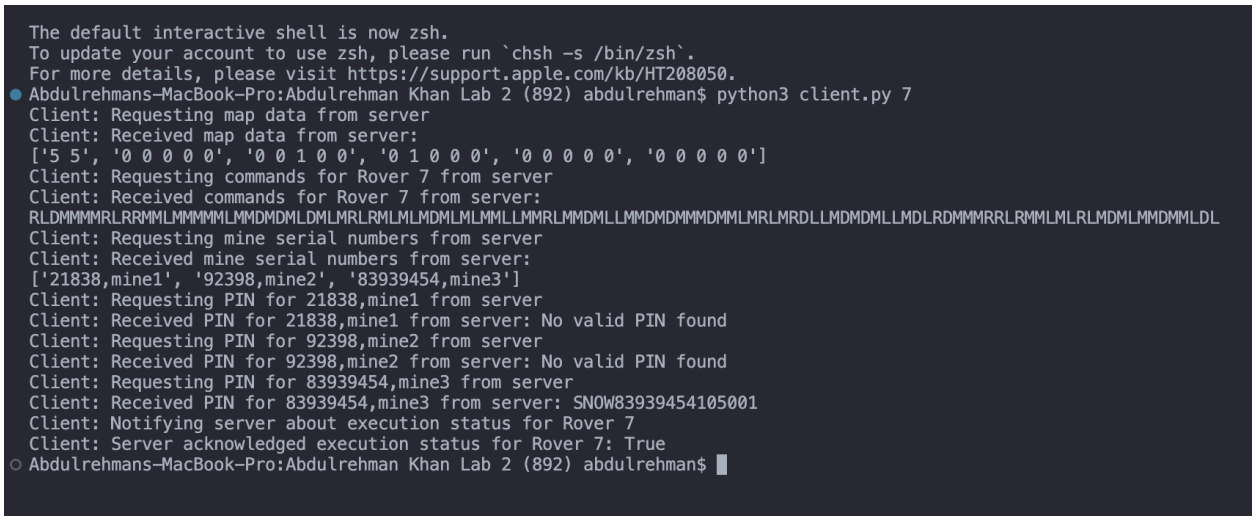
1. **Server Implementation (server.py):**
    - Initialized a gRPC server to handle incoming requests.
    - Implemented methods to handle requests for map data, commands, mine serial numbers, mine PINs, and execution status.
    - Utilized file reading to retrieve map data and mine information.
    - Used hashing to generate mine PINs.
2. **Client Implementation (client.py):**
    - Established connection with the gRPC server.
    - Sent requests for map data, commands, mine serial numbers, mine PINs, and execution status.
    - Printed the received data and server acknowledgments.
3. **Protocol Buffer File (rover.proto):**
    - Defined the RoverControl service with RPC methods.
    - Defined message types for requests and responses exchanged between the server and clients.

Main Data Structures and Functions:

1. **RoverControlServicer:** The main class that implements gRPC service methods to handle client requests.
2. **serve():** Function to start the gRPC server.
3. **run_rover(rover_number):** Function to simulate rover behavior, including requesting data from the server and notifying execution status.
4. **generate_mine_pin(mine_serial_number):** Function to generate a PIN for a given mine serial number using hashing.

Test Run Screenshots:

Below we can see The client successfully interacts with the server, retrieving necessary data and sending execution status notifications. The server responds to client requests and generates PINs as required. This demonstrates the functionality of the rover communication system implemented using gRPC.



Client Output:

1. The client first requests map data from the server and receives a 5x5 grid representing the map.
2. Next, it requests commands for Rover 7 from the server and receives a series of commands.
3. Then, it requests mine serial numbers and receives a list of mine serial numbers along with their names.
4. For each mine, the client requests a PIN from the server and receives the PINs.
5. Finally, the client notifies the server about the execution status for Rover 7, and the server acknowledges it.

```
The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
Abdulrehmans-MacBook-Pro:Abdulrehman Khan Lab 2 (892) abdulrehman$ python3 server.py
Server listening on [::]:00008

(Server) Received request for map, retrieving...

(Server) Received request for commands, retrieving...

(Server) Received request for serial numbers, retrieving...

(Server) Received request for PIN for mine 21838
(Server) Generated PIN for mine 21838: No valid PIN found

(Server) Received request for PIN for mine 92398
(Server) Generated PIN for mine 92398: No valid PIN found

(Server) Received request for PIN for mine 83939454
(Server) Generated PIN for mine 83939454: SNOW83939454105001

(Server) Received execution status for Rover 7
```

Server Output:

1. The server starts listening on port 8.
2. It receives requests for map data, commands, and mine serial numbers from the client.
3. For each mine serial number, the server receives requests for PINs and generates them accordingly.
4. After handling the execution status notification for Rover 7, it prints a message indicating the receipt of execution status for Rover 7.

Conclusion

The implementation successfully achieved the objectives of establishing communication between rovers and ground control using gRPC. Each rover can interact with the server to retrieve necessary data and report execution status. The protocol buffer file defines the service and message types used for communication. This lab provided valuable experience in implementing distributed systems using gRPC and protocol buffers.