Remote Method Invocation (RMI) – Cloud-Based Calculator Application

---

1. Aim

To implement a Remote Method Invocation (RMI) based Calculator Application using Java, where the server is deployed on AWS EC2 and the client remotely invokes arithmetic operations.

---

2. Objective

• To understand object-oriented distributed systems using RMI
• To design remote interfaces and remote objects
• To deploy an RMI server on AWS EC2 cloud
• To invoke remote methods from a client machine
• To perform basic arithmetic operations remotely

---

3. System Requirements

Hardware

• Computer with minimum 4 GB RAM
• Stable Internet connection

Software

• OS: Ubuntu (Server), Windows / Kali Linux (Client)
• Programming Language: Java (OpenJDK)
• Cloud Platform: AWS EC2
• Tools: OpenJDK, RMI Registry

---

4. RMI Architecture

Java RMI follows a client–server architecture:

• Remote Interface defines methods
• Server implements the interface
• RMI Registry binds remote objects
• Client looks up objects using server IP
• Methods are invoked through stubs

5. RMI Implementation Details

5.1 Remote Interface

The remote interface defines the methods that can be invoked by the client from a remote machine. It extends the `Remote` interface provided by Java RMI, and each method throws `RemoteException` to handle network-related errors.

In this application, the remote interface declares the following arithmetic operations:

• `add()` – Performs addition of two numbers
• `sub()` – Performs subtraction of two numbers
• `mul()` – Performs multiplication of two numbers
• `div()` – Performs division of two numbers

These methods are implemented on the server side and accessed remotely by the client.

5.2 Working Principle

The working of the RMI-based calculator application follows these steps:

1. A remote interface is created to declare the calculator methods.

2. The server implements this remote interface and provides concrete definitions for all arithmetic operations.

3. The server creates an object of the implementation class and registers it with the RMI Registry using a unique service name.

4. The client connects to the RMI Registry using the server's public IP address and looks up the registered remote object.

5. Once the object reference is obtained, the client invokes the remote methods (`add`, `sub`, `mul`, `div`) as if they were local methods, and the results are returned from the server.

This process enables transparent remote communication between the client and server using Java RMI.

6. Cloud Deployment

• RMI Server deployed on AWS EC2 Ubuntu instance
• Public IP used for client lookup
• Security Group inbound rules:

– TCP 1099 (RMI Registry)
– TCP 1024–65535 (Dynamic ports)
– SSH 22

Screenshots to Attach:

✔ EC2 running instance
✔ Security group inbound rules
✔ Server execution terminal
✔ Client output

---

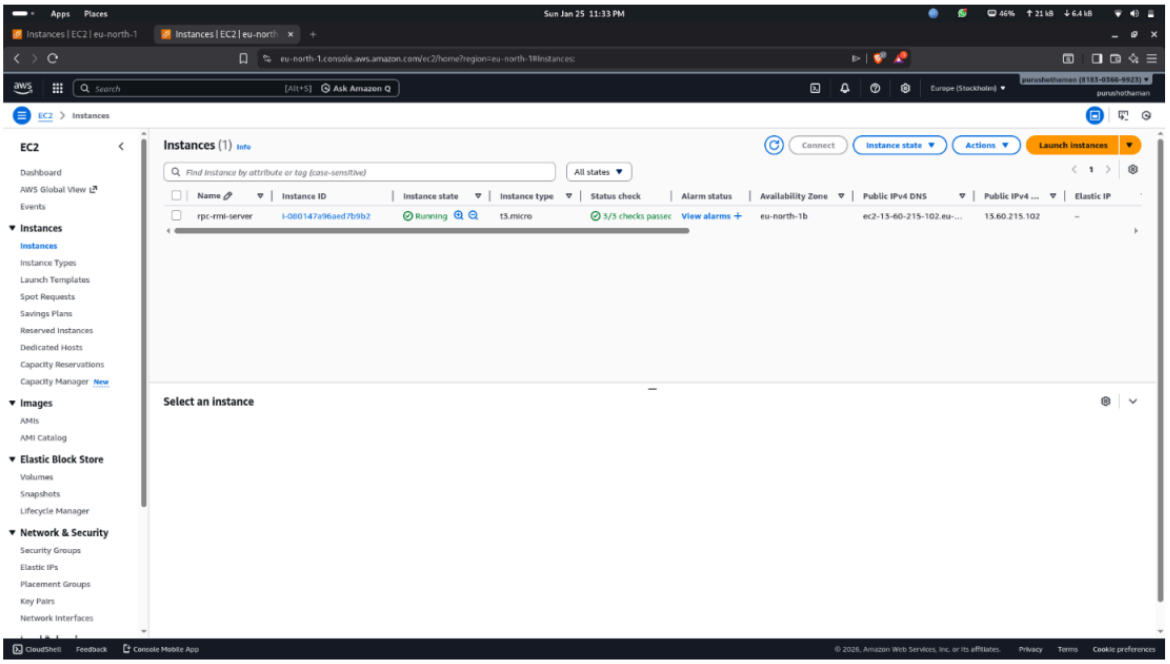
7. Error Handling

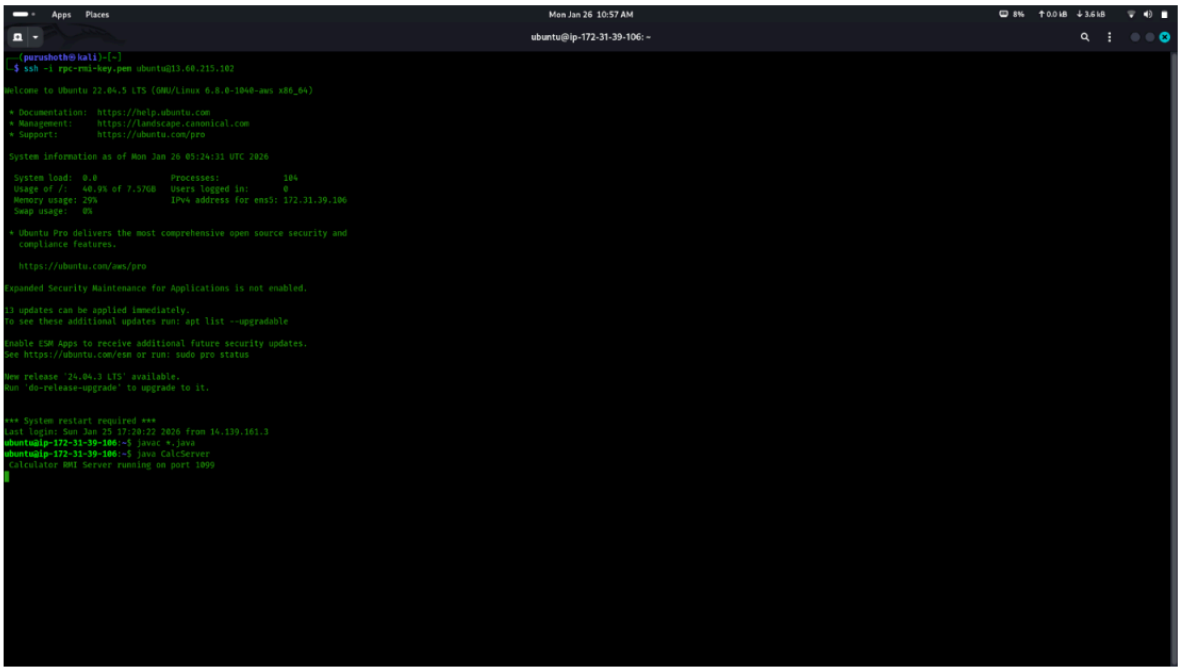
• RemoteException handled using try–catch
• Division by zero handled safely
• Network failures managed

---

8. Output

• Server successfully runs on AWS EC2
• Client remotely invokes methods
• Correct calculator results displayed


Apps Places
Sun Jan 25 11:33 PM
Instances | EC2 | eu-north-1
eu-north-1.console.aws.amazon.com/ec2/home?region=eu-north-1#Instances:
Search
[Alt+S]
Ask Amazon Q
Europe (Stockholm)
purushothaman
EC2 > Instances
EC2
Dashboard
AWS Global View
Events
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations
Capacity Manager New
Images
AMIs
AMI Catalog
Elastic Block Store
Volumes
Snapshots
Lifecycle Manager
Network & Security
Security Groups
Elastic IPs
Placement Groups
Key Pairs
Network Interfaces
Instances (1) Info
Connect
Instance state
Actions
Launch instances
Find instance by attribute or tag (case-sensitive)
All states
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
Availability Zone
Public IPv4 DNS
Public IPv4 ...
Elastic IP
rpc-rmi-server
i-080147a96aed7b9b2
Running
t3.micro
3/3 checks passed
View alarms
eu-north-1b
ec2-13-60-215-102.eu-...
13.60.215.102
Select an instance
CloudShell
Feedback
Console Mobile App
© 2026, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Apps Places
Mon Jan 26 10:57 AM
ubuntu@ip-172-31-39-106: ~
┌──(purushoth㉿kali)-[~]
└─$ ssh -i rpc-rmi-key.pem ubuntu@13.60.215.102
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 6.8.0-1040-aws x86_64)
* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/pro
System information as of Mon Jan 26 05:24:31 UTC 2026
System load: 0.0 Processes: 104
Usage of /: 40.9% of 7.57GB Users logged in: 0
Memory usage: 25% IPv4 address for ens5: 172.31.39.106
Swap usage: 0%
* Ubuntu Pro delivers the most comprehensive open source security and compliance features.
https://ubuntu.com/aws/pro
Expanded Security Maintenance for Applications is not enabled.
13 updates can be applied immediately.
To see these additional updates run: apt list --upgradable
Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status
New release '24.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.
*** System restart required ***
Last login: Sun Jan 25 17:20:22 2026 from 14.139.161.3
ubuntu@ip-172-31-39-106:~$ javac *.java
ubuntu@ip-172-31-39-106:~$ java CalcServer
Calculator RMI Server running on port 1099

Client invoking remote methods

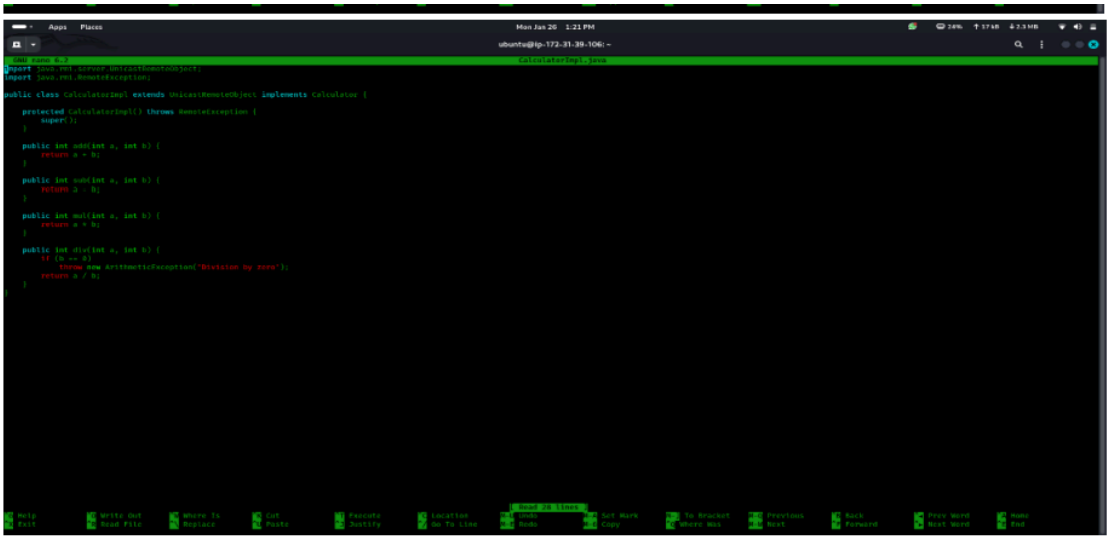
CalcClient.java

```java
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;

public class CalcClient {
    public static void main(String[] args) {
        try {
            String serverIP = "YOUR_EC2_PUBLIC_IP";

            Registry registry = LocateRegistry.getRegistry(serverIP, 1099);
            Calculator calc = (Calculator) registry.lookup("CalcService");

            System.out.println("Add: " + calc.add(10, 5));
            System.out.println("Sub: " + calc.sub(10, 5));
            System.out.println("Mul: " + calc.mul(10, 5));
            System.out.println("Div: " + calc.div(10, 5));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```



9. Result

The RMI-based Calculator Application was successfully implemented and deployed on AWS EC2. The client accessed server-side objects remotely and obtained accurate arithmetic results.

---

10. Conclusion

This experiment demonstrated Java RMI as an effective object-oriented distributed computing model. Hosting the RMI server on AWS EC2 provided practical experience in cloud deployment and remote method invocation.