

Design Document MP 2.1

Name: Tanmai H

UIN: 434007349

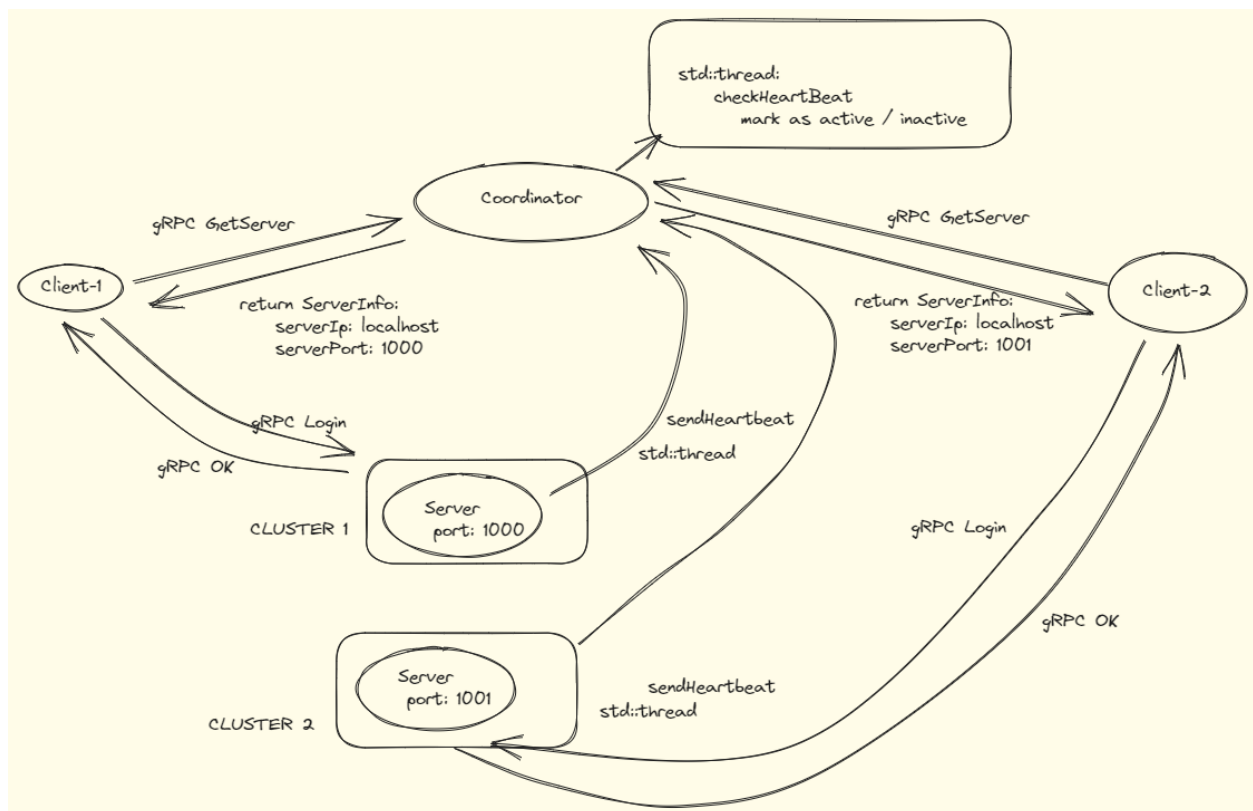
Commands to Start System

make

Now run these commands on separate terminals

1. `./coordinator -p 9090`
2. `./tsd -c 1 -s 1 -h localhost -k 9090 -p 10000`
3. `./tsd -c 2 -s 1 -h localhost -k 9090 -p 10001`
4. `./tsc -h localhost -k 9090 -u 1`
5. `./tsc -h localhost -k 9090 -u 2`

Flow Diagram



RPC Definitions

Coordinator

Heartbeat(ServerInfo serverInfo, Confirmation confirmation):

1. Get cluster ID from serverInfo

Design Document MP 2.1

2. If a cluster doesn't exist in the routing table, return `grpc::StatusCode::NOT_FOUND`
3. If cluster exists, sleep for 5 seconds
4. Get the corresponding node from the cluster (based on `serverId`)
5. Set this `zNode`'s last heartbeat timestamp to the current time
6. Set the confirmation's status as true
7. return `Status::OK`

`GetServer(ID id, ServerInfo serverInfo):`

1. Computer cluster ID for the client's ID. Use $(id-1)\%3 + 1$
2. In routing table check if cluster exists
3. Get the server, check if active, if active set the servers ip and port in the `serverinfo` and return `grpc::Status OK`
4. If not active return `grpc::StatusCode::UNAVAILABLE`

Server

`sendHeartBeat(coordinatorClientStub):`

1. Set server information into `serverInfo` variable
2. Start an infinite loop
3. Use the the stub and the `serverInfo` via gRPC
4. If `grpc` status is OK and confirmation status is true, continue
5. Else break and exit with -1

..
// Code

Main Thread:

1. Server Start
2. `heartbeatThread.join()`
// Code
3. Exit

Client

1. Create coordinator stub
2. Set `clientId` in the ID Protobuf
3. Send the `clientId` via GetServer RPC // `stub_.GetServer(ID id, ServerInfo serverInfo)`
4. If `grpc::Status` is not OK, exit with -1 and error message.

Design Document MP 2.1

5. If `grpc::Status` is OK, get `serverIP` and `serverPort` from the `ServerInfo` response
6. Create server stub based on this data
7. Connect and login to server using `serverStub_.Login(..)` RPC