ECE419 Milestone 2 Design Document

Members:
- Tzu-An Chen 1000538411
- Dexter Rivera 1002433690
- Vlad Ilgovskiy 1001803195

Consistent Hash Ring

Consistent hash ring is implemented in the ecs/ECSConsistentHash class. This is a TreeMap structure implementation that mocks the functionality of a linked list structure. Each server node will be initialized with its server name and port and inserted into the tree map. Upon insertion, the treemap structure will automatically sort the node inserted based on its hash value, the previous node field of the inserted node will be set using tail map method to get the previous node that will be ranked before the inserted node. The root node's previous node field points to the last node in the tree map. We chose this structure as we have more flexibility with our implementation and its provided interface allows us to remove/ add nodes without too much effort.

External Configuration Service

Our ECS contains the info of every KVServer and their status. ECS also coordinates the status of each active servers by broadcasting/ sending metadata to specific groups of server depending on situation like adding/ removing nodes. It is responsible for adding/ removing nodes, and redistribute the existing KV data between active KV servers. When redistribution of data happens during adding/ deleting nodes, the data that is currently held by nodes that will be deleted will be transferred to nodes that will still be active after operation, or nodes that will be active after adding new nodes. The operation will be handled through ECS and inside KV server by issuing batched put request to other active servers.

Server

To simplify communication between the ECS and Servers, Zookeeper was used. A KVAdminMessage class was created as a message format for admin messages between the servers and ECS. Each KVServer sets a watch on their assigned znode. When the ECS wants to send a command to a KVServer, it will set the data on that server's znode to a serialized KVAdminMessage. The watch is then triggered on the server and the KVAdminMessage is fetched from the znode and parsed for the command the ECS is asking the server to perform. To implement the lockWrite() and unlockWrite() command, a boolean variable lock_writes was used. To implement the start() and stop() commands, a boolean variable stopped was used. The two booleans lock_writes and stopped were checked every time a client tried to perform a PUT or GET.

To ensure each server has an updated copy of the metadata, all the servers set a watch on one znode which holds the metadata of the service. When the ECS updates the metadata, the watch is then triggered on all servers, where the servers update their copy of the metadata.

Client

The client first attempts to connect to a known server. If the server is not responsible, the metadata is cached in the server and the server connects to the correct server. In the case of SERVER_STOPPED and SERVER_LOCK_WRITE, we decide to simply log the message and have the user retry the request again.

Appendix - Documentation of Test Cases

Test Case	Description	Status
ECSTests	A suite of tests to test the logic of the ECS Hash Ring. Tests include adding nodes into the hash ring and serializing the hash ring into json.	Pass
KVServerTests	A suite of tests to test the logic of the KV Server. Tests include whether the server correctly determines if it is responsible for a key, whether or not the server prevents write operations on write locks and whether or not the server successfully prevents operations when it is stopped.	Pass