This is a ZooKeeper based consensus protocol with the central philosophy of embracing the zoo. It passes all local auto-grader tests. I originally tried using a leader based election approach, but quickly realized that it wasn't feasible since the cassandra instances were also cleared out in addition to the clients. That meant I needed to host all of my data on zookeeper, the only thing I could trust to work as persistent storage. The easiest thing I could think to do was store all of my data (with a persistent global numbering), inside of zookeeper. Then, my program would store the last thing they processed and if they were brutally murdered, they could start over again loading all of the data into the cassandra instance again. I'd like to do some sort of checkpointing in the future, since obviously data sizes would be increased unrealistically high with this system, but it works well for ensuring consistency. I originally used a watcher to scan for new events, but I had trouble with ensuring it got events correctly so I moved over to a schedule based system (I think that both have their pros and cons). Overall that's basically the application, I use literally no messages between servers, as zookeeper handles all the coordination.