CSCE 438 : Homework 4
A Scalable and Highly Available SNS Service

Due on 30 April, 2019

Stoleru Spring 2019

M. Hunter Martin, Trenton Smith

Design Document

The task of Homework 4 was to create a system that make homework 3.2 more scalable. This was implemented by load balancing across several servers, rather than having only one server. The client program will query the routing server, and the routing server will send two server IP/ports to the clients. The first is the Master server for the client. The second is a Slave server that will be used if/when the Master dies. Each server also has Slave that will restart its Master if necessary.

I decided to implement the selection of the Master and Slaves to be randomized. While it was easy to make the routing server simply alternate which server it sends traffic to, that idea took into no count how many people were left on the server. Even if we tried to load balance based on how many people were on the server, it was hard to make sure the servers knew when the clients disconnected unceremoniously. My solution was the just randomly assign which server traffic was sent to. Since all of the data is centralized anyway using a shared folder, this solution will be in theory optimal so long as we don't have too many people signing on and staying on one of the servers, while people are jumping off of another.

I apologize for the lateness of this project. I've worked really hard on these assignments, and I've done well in the past. In the past, I've allowed my partner (Trenton) to handle some of the Design Document and I've done all of the coding for the project. This time, Trenton wanted to do the coding and I was ok with him doing that because he hadn't programmed for this course all semester. He had said he'd been working all weekend, but an hour before the project was due he told me he didn't understand the project and couldn't finish it because he hadn't made any progress.

Running the Project

You can use the python file provided, or you can just run the executables.

I couldn't get the bash scripting to work fully. For some reason, the gRPC framework doesn't like strings in streams, or it thinks that strings aren't formatted correctly, and I couldn't work around it. The other commands should still work fine, and you can still enter the `TIMELINE` mode.