ECE 454 Assignment 1 Design Document

Synchronous vs. Asynchronous RPCs

We chose to use synchronous RPCs throughout our distributed system. We made this decision because it resulted in a much simpler threading model and architecture.

Multi-threading Servers

We chose to use different multi-threading servers depending on the type of node and type of server we were starting. Front-end management servers use the Threaded Selector Server because they need to be able to quickly manage the gossip protocol between other management nodes. The frontend password servers use Thread Pool Server because they need to concurrently handle a large number of requests with low latency. The backend management servers use a Non-blocking Server because they do not receive much traffic and don't need a lot of concurrency. The backend password nodes use a HaHs Server so it does not become overloaded when processing many password hashes and checks at the same time.

Protocols

We chose to use TCompactProtocol for all of our requests because it was the fastest way to send data.

Load Balancing

We use a weighted round robin scheduler for backend nodes. This works by assigning a certain amount of work to a node depending on the number of cores it is allocated. It then rotates through the rest of the nodes.

Crash Failures

If a request fails, we assume the node has crashed. The system then propagates that information to other front end nodes through a gossip protocol.

BE Nodes Joining

To ensure that the nodes are ready to receive requests within 1 second of startup, backend nodes contact seed nodes concurrently on start up. The seed nodes then gossip this information to the rest of the front end nodes in the system. The gossip protocol runs on 100ms intervals so it should propagate all the information within one second.