Distributed lock with Lamport clocks

Large Scale Distributed Systems

Objectives

Use Maelstrom (`github.com/jepsen-io/maelstrom`) to observe executions of wrong or limited implementations of a distributed lock. Implement a distributed lock using Lamport clocks. Use Maelstrom to help in debugging tentatative implementations towards a correct version.

Software

Have a JVM and node.js installed. Download the version of Maelstrom made available in a self-contained jar, which defines a *workload* named *lock*, used with `-w lock`.

Tasks

1. Study the code of a trivially wrong implementation, in `wrong-lock.mjs`.
2. Run the code in Maelstrom and observe the incorrect behavior, analysing execution histories and message diagrams.
3. Study now the code of a centralized lock in `lock-single-node.mjs`.
4. Run this version in Maelstrom and observe the behavior in the following configurations:
   - a single server node, with option: `--node-count 1`
   - several server nodes, with option: `--node-count 3`
5. Implement a distributed lock based on Lamport clocks, as described in

   `https://lamport.azurewebsites.net/pubs/time-clocks.pdf`

   Make each server capable of handling several clients. Use Maelstrom to help in making the implementation correct, observing histories and message diagrams involving several client processes and server nodes. Sugestion: test in progression 1) several servers with one client per server, 2) several clients and one server, 3) the general case of several servers and several clients per server.