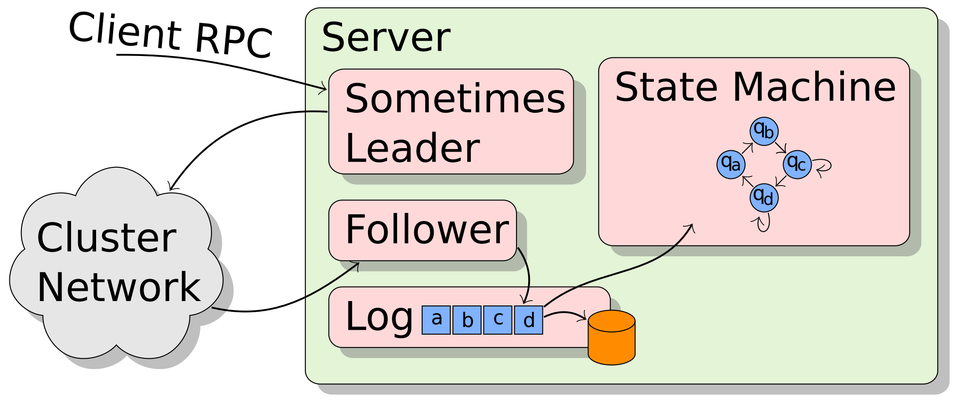
How to apply and flesh out Paxos

Diego Ongaro

Goals and assumptions

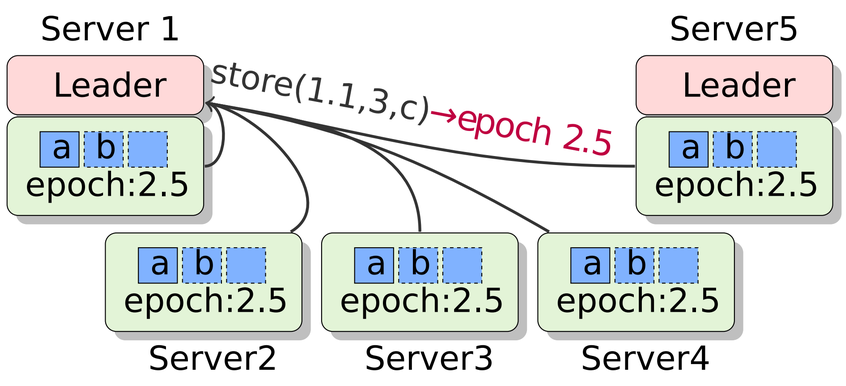
- Framework to build a small, fault-tolerant state machine
- Servers can crash at any time, can later restart
  - Assume non-byzantine failures
- No single point of failure
  - Service should be up if any majority of the cluster is up
- Small cluster sizes, such as 5 servers

Replicating a log of operations



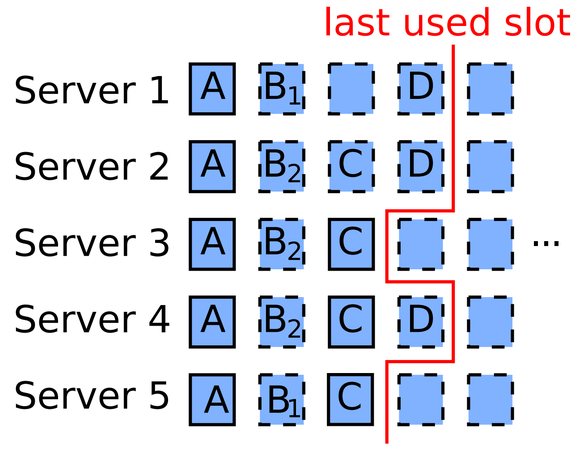
- This framework provides an order log of operations to a state machine
- The state machine can implement a key-value store, a lock server, etc
- If all servers play the same log, their states will be the same

Overthrow stops old leader



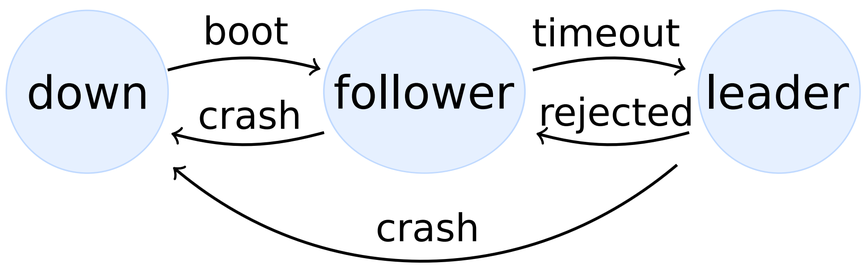
- New leader calls overthrow on a majority of servers
- Then old leader's stores will fail on at least one server

Responsibilities of leadership



- To advance the local state machine, finalize all slots locally up to the last used slot
- To speed up future recoveries, replicate operations and finalized flags widely

Encouraging one leader at a time



- Timeouts make passive servers become leaders
  - Leader issues heartbeats in case of inactivity
  - Timeout period chosen randomly so not all servers wake up at once
- Epoch numbers select arbitrarily between the available leaders
  - If a leader's store is rejected, it becomes passive

RPCs

- overthrow(new epoch) → last used slot | current epoch
  Used by new leader to kill off old leader
- store(epoch, slot, operation) → ok | current epoch
  Used to replicate operations
- finalize(epoch, slot) → ok | missing
  Used to flag slots as immutable
- read(slot) → epoch, operation
  Used to determine previous leaders' operations
- query() → first unfinalized slot
  Used to identify what to replicate to a follower