CloudWalk Challenge

Arthur Boschini da Fonseca - arthur.b.dafonseca@gmail.com

October 21th, 2025

Raft-lite Design Document

Consensus via a simplified Raft to keep a replicated key-value store consistent across three nodes.

Overview

The system implements a lightweight variant of the *Raft* consensus algorithm over three nodes. Each node can be a **follower**, **candidate**, or **leader**. The cluster aims for a single leader that orders client writes; followers replicate the leader's log. Once an entry is replicated to a majority, it becomes **committed** and is applied to an in-memory key-value store on each node.

Cluster Size

3 nodes

Quorum

Majority (2 of 3)

Election Timeout

~300–500 ms (randomized)

Heartbeat

~50 ms

Backoff

≤ 1.5 s cap with 200 ms early-exit probe

Election Logic

- Nodes start as **followers** and expect periodic heartbeats from a leader.
- If no heartbeat arrives within a randomized **300–500 ms** window, a follower becomes a **candidate**.

- The candidate increments its term, votes for itself, and sends **vote requests** to peers.
- A peer grants its vote if it hasn't voted in the current term and the candidate's log is at least as up-to-date.
- Upon receiving a **majority of votes**, the node becomes leader and starts sending heartbeats (~50 ms).

Randomized timeouts reduce split votes; recent leader activity suppresses unnecessary elections.

Enhancements: (1) a preflight reachability check avoids term churn when quorum isn't possible; (2) exponential backoff for failed elections is capped at **≤ 1.5 s** and includes a **200 ms early-exit probe** that immediately restarts elections once a majority is reachable; (3) tie-break jitter increased (up to ~300 ms) to reduce split votes.

Log Replication and Persistence

1. Clients send writes to the leader, which appends a **log entry** locally.
2. The leader replicates the entry to followers via an internal append endpoint.
3. When a **majority** acknowledges the entry, it is marked **committed**.
4. Committed entries are applied to each node's in-memory key-value store (state machine).

Each node persists essential data to disk:

- node_X_state.json: current term, votedFor, commit/lastApplied indices.
- node_X_log.json: the replicated log entries (append-only semantics with conflict resolution).

On restart, nodes reload state/log files and rebuild the key-value store by replaying committed entries, ensuring durability across crashes.

Failure Handling

- **Leader failure detection:** Followers monitor heartbeats and, if none are received within ~500 ms, they assume the leader has failed and trigger a new election.
- **Leader re-election:** A new leader is chosen automatically when a majority (2 of 3 nodes) agrees, ensuring cluster availability.
- **Follower or node loss:** The cluster continues to operate as long as a quorum is maintained. Lost nodes automatically synchronize missing log entries after recovery.

- **Network partition:** Isolated nodes cannot form a majority, preventing conflicting leaders. An exponential backoff (capped at ≤ 1.5 s) pauses new elections, while a 200 ms early-exit quorum probe resumes elections immediately once a majority is reachable.
- **Early-exit probe:** During backoff, nodes periodically check peer reachability; on detecting a majority they clear backoff and start an election without waiting for the full backoff window.
- **Data safety:** Only entries replicated to a majority are considered committed, guaranteeing that no confirmed data is ever lost even if one node crashes immediately after commit.

Together, these mechanisms ensure **availability** when at least two nodes are alive and **consistency** across the cluster once all nodes rejoin.

Summary

This Raft-lite design provides: (1) automatic leader election, (2) majority-based commits for consistency, (3) disk-backed persistence for crash recovery, and (4) stable behavior under partial failures and rejoin. Though simplified, it captures the core principles of distributed consensus—**safety**, **availability**, and **fault tolerance**—in a compact three-node setup.