Recitation 20: Raft

MIT - 6.033

Spring 2022

Henry Corrigan-Gibbs

Plan

* Recitation Qs

* Recap: Big picture

* Normal operation

* Operation under failure

* Scenarios

Logistics

* MapReduce hands on due today.

* Design project due May 2

* Rest of class is about security...

Recitation Questions

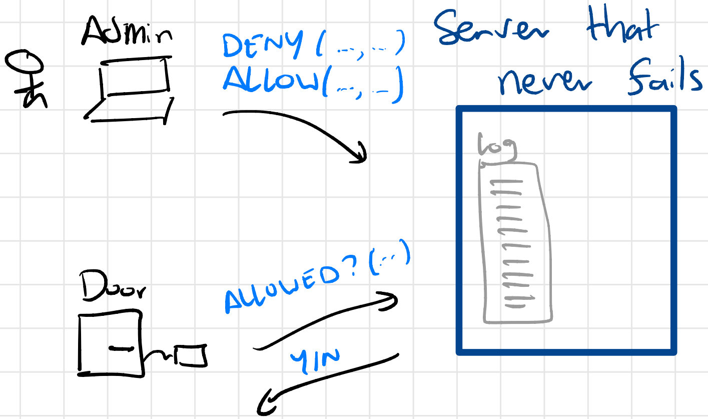
1. The authors of Raft were looking for an "understandable" consensus alg. What does this mean?

2. How understandable is Raft?

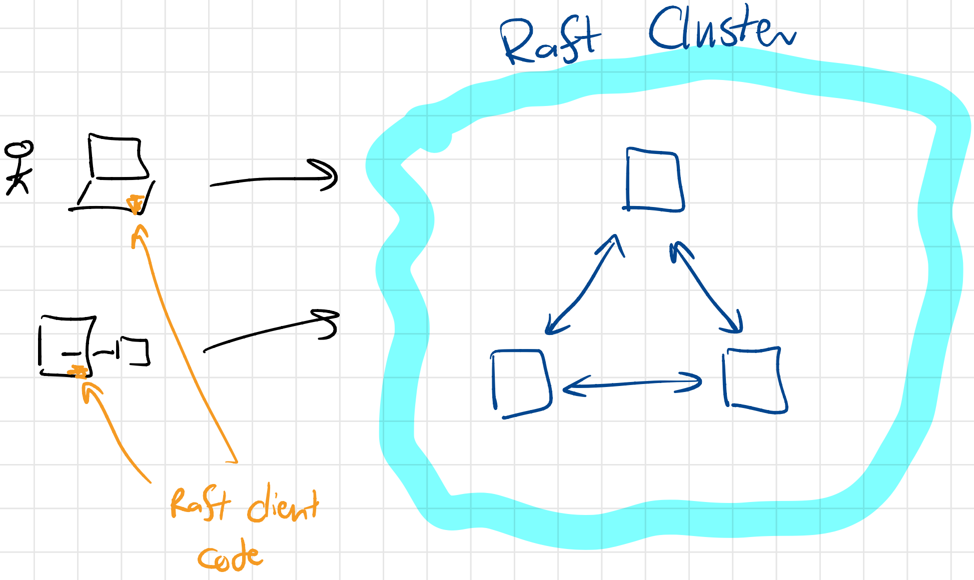
3. Why is understandability important? Is it?

[History of Raft paper...]

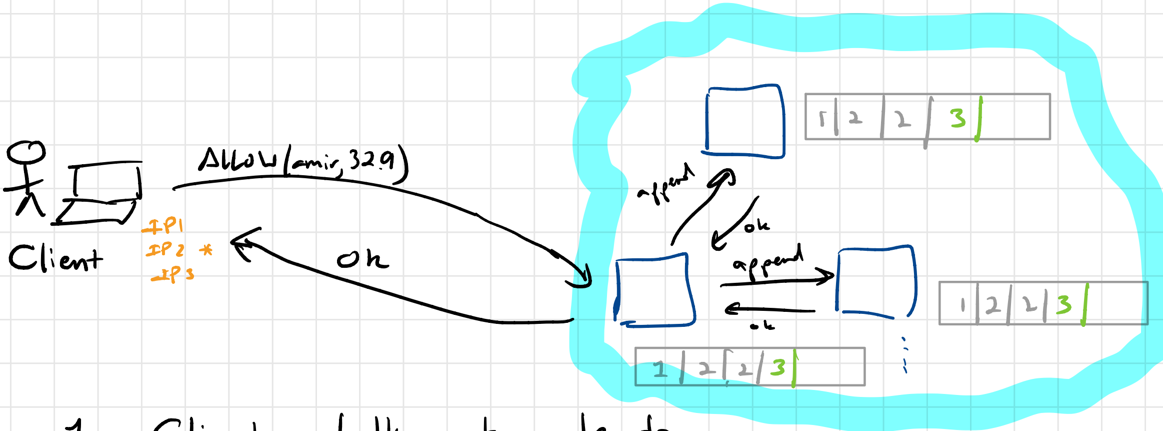
The Big Picture



Goal: Implement a never-fail server using a cluster of some-times-fail servers.
↳ If you have a leader/view server, it also might fail!?



Normal operation in Raft



1. Client talks to leader.

2. Leader pushes update out to all servers.

3. When a majority of servers reply to leader,

   * leader applies change to "state machine"

   * leader replies to client.

   [Remind you of anything?]

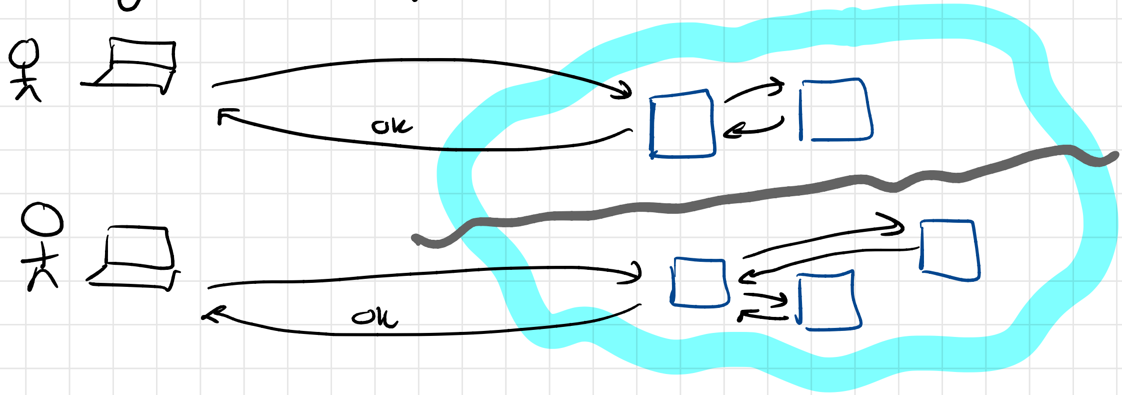
What is this state machine?

* e.g. Collection of (key, value) pairs (DB)

* Log entries are updates to DB (e.g. bld, 32_jack_allowed = "T")

Why a majority of servers?

Imagine a partitioned network



If leader waited for < ½ of nodes to store entry, then you could have a "split brain" situation

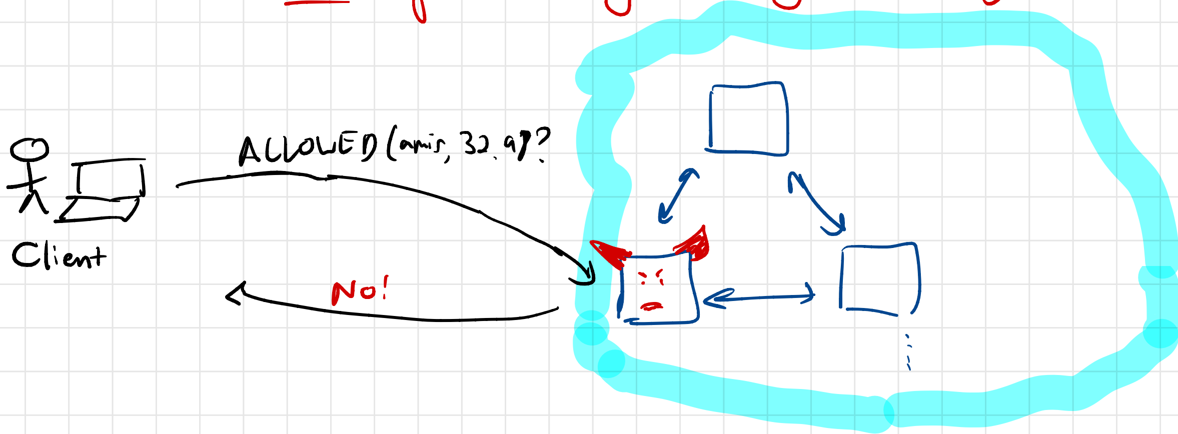
↳ Two inconsistent DBs!?

A big mess!?

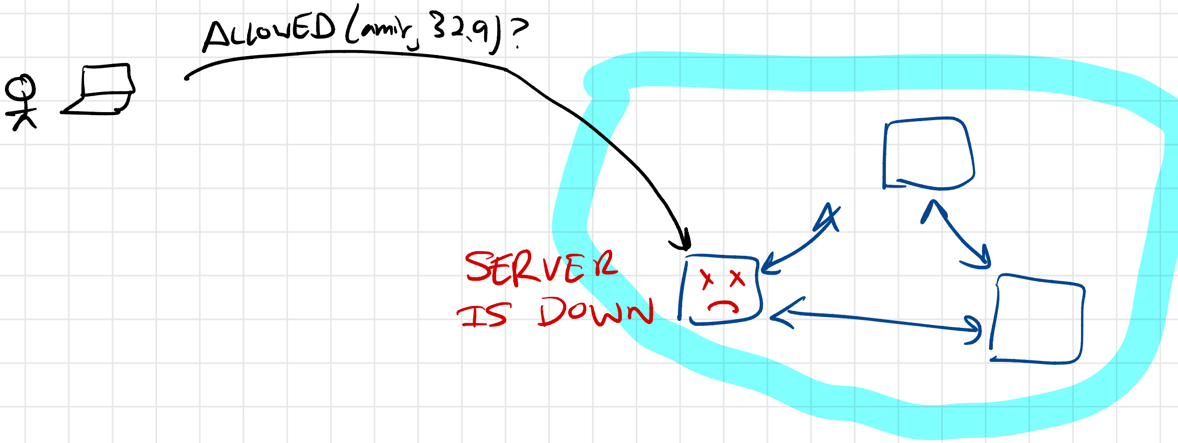
Imagine if this happened with CovidPass.... very bad.

Fail-Stop Failures (vs. "Byzantine" or "malicious")

Raft does NOT protect against "Byzantine" failures.

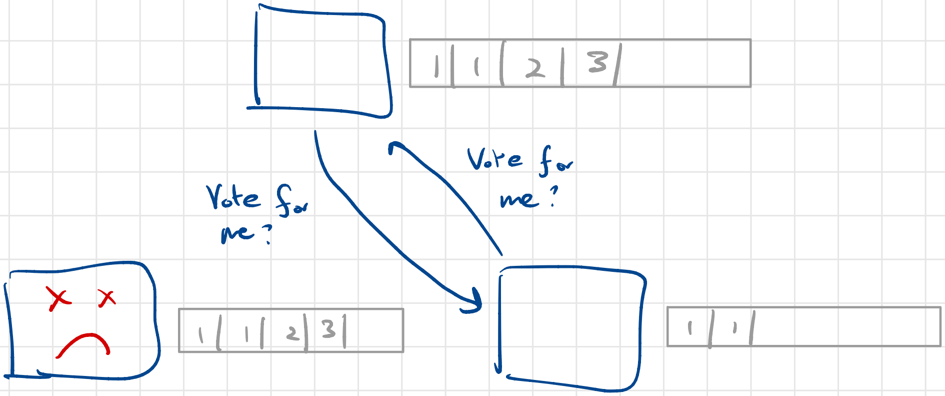


Raft DOES prevent against "fail stop" failures, as long as not "too many" nodes fail. ← How many?



Problem: What happens when leader fail?

Problem w/ all benevolent dictatorships



* When follower doesn't hear from leader after a while → run election

* Node with most up-to-date log wins
  ↳ Later "term" or longest log (if last term equal)

Important: Followers may delete log entries...

The only log entries that are committed are ones that LEADER has decided are committed
↳ after hearing back from maj of nodes.

When a new leader takes over...

→ Its log is authoritative.

→ Committed entries stay committed.

→ Non-leader logs may change.

Followers' logs may be inconsistent with leader's log. Leader takes precedence.

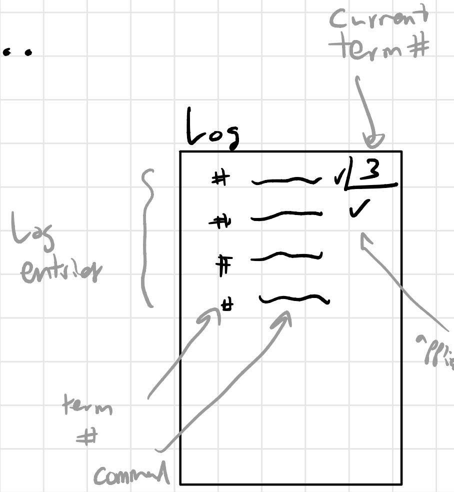
Rules to remember

* In election, server with most upto-date log wins

* Terms increase with each election

* Need a majority of votes to win!

Let us try a simulation...



- Three servers
- One client

Try some scenarios...

1. Normal operation.

   All nodes up... leader election

   - term #
   - last committed idx

   ↳ No commands for a while? Heartbeat

2. One non-leader fail

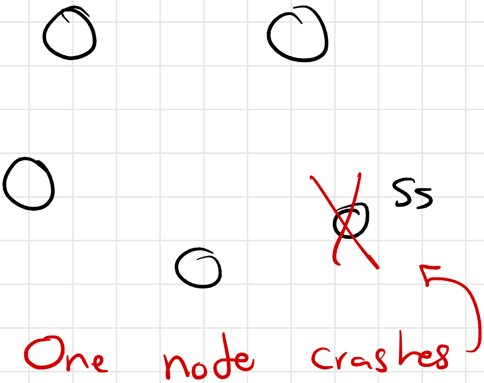
3. Leader fails & stays silent

4. Leader fails & follower fails

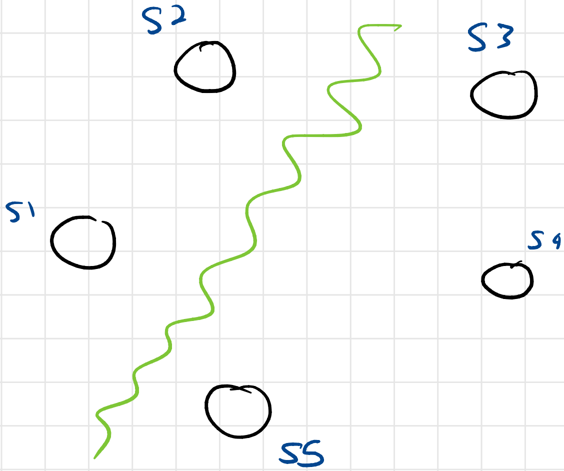
5. Swapping leader failure

Example Scenarios (2019 Final)

1	1	3
1	1	3
1	1	3
1	1	3
1		



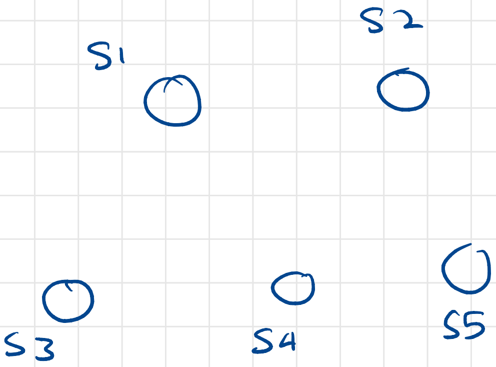
S1	1	2	3
S2	1	2	
S3	1		
S4	1	4	
S5	1	4	5



- S1 crashes & restarts, crashes & restarts
- Network partition
- S3, S4, S5 start making progress

Example scenarios (2019 final)

S1	1	3
S2	1	3
S3	1	2
S4	1	2
S5	1	2



- S1, S2 crash
- S3 is new leader; adds new entry, crashes
- S1 new leader, adds new entry to S2, crashes with S2
- S3 back up, new leader, replicates log to S4, S5

Back to the big picture

The illusion of talking to a never-fail server, but constructed from many some-times-fail servers.

↳ Very powerful idea!!