

# Raft Protocol Summary

## Followers

- Respond to RPCs from candidates and leaders.
- Convert to candidate if election timeout elapses without either:
  - Receiving valid AppendEntries RPC, or
  - Granting vote to candidate

## Candidates

- Increment currentTerm, vote for self
- Reset election timeout
- Send RequestVote RPCs to all other servers, wait for either:
  - Votes received from majority of servers: become leader
  - AppendEntries RPC received from new leader: step down
- Election timeout elapses without election resolution: increment term, start new election
- Discover higher term: step down

## Leaders

- Initialize nextIndex for each to last log index + 1
- Send initial empty AppendEntries RPCs (heartbeat) to each follower; repeat during idle periods to prevent election timeouts
- Accept commands from clients, append new entries to local log
- Whenever last log index  $\geq$  nextIndex for a follower, send AppendEntries RPC with log entries starting at nextIndex, update nextIndex if successful
- If AppendEntries fails because of log inconsistency, decrement nextIndex and retry
- Mark log entries committed if stored on a majority of servers and at least one entry from current term is stored on a majority of servers
- Step down if currentTerm changes

## Persistent State

Each server persists the following to stable storage synchronously before responding to RPCs:

<b>currentTerm</b>	latest term server has seen (initialized to 0 on first boot)
<b>votedFor</b>	candidateId that received vote in current term (or null if none)
<b>log[]</b>	log entries

## Log Entry

<b>term</b>	term when entry was received by leader
<b>index</b>	position of entry in the log
<b>command</b>	command for state machine

## RequestVote RPC

Invoked by candidates to gather votes.

### Arguments:

<b>candidateId</b>	candidate requesting vote
<b>term</b>	candidate's term
<b>lastLogIndex</b>	index of candidate's last log entry
<b>lastLogTerm</b>	term of candidate's last log entry

### Results:

<b>term</b>	currentTerm, for candidate to update itself
<b>voteGranted</b>	true means candidate received vote

### Implementation:

1. If term  $>$  currentTerm, currentTerm  $\leftarrow$  term (step down if leader or candidate)
2. If term  $=$  currentTerm, votedFor is null or candidateId, and candidate's log is at least as complete as local log, grant vote and reset election timeout

## AppendEntries RPC

Invoked by leader to replicate log entries and discover inconsistencies; also used as heartbeat .

### Arguments:

<b>term</b>	leader's term
<b>leaderId</b>	so follower can redirect clients
<b>prevLogIndex</b>	index of log entry immediately preceding new ones
<b>prevLogTerm</b>	term of prevLogIndex entry
<b>entries[]</b>	log entries to store (empty for heartbeat)
<b>commitIndex</b>	last entry known to be committed

### Results:

<b>term</b>	currentTerm, for leader to update itself
<b>success</b>	true if follower contained entry matching prevLogIndex and prevLogTerm

### Implementation:

1. Return if term  $<$  currentTerm
2. If term  $>$  currentTerm, currentTerm  $\leftarrow$  term
3. If candidate or leader, step down
4. Reset election timeout
5. Return failure if log doesn't contain an entry at prevLogIndex whose term matches prevLogTerm
6. If existing entries conflict with new entries, delete all existing entries starting with first conflicting entry
7. Append any new entries not already in the log
8. Advance state machine with newly committed entries