SHANGHAI JIAO TONG UNIVERSITY

Raft

Consistency across multiple machine

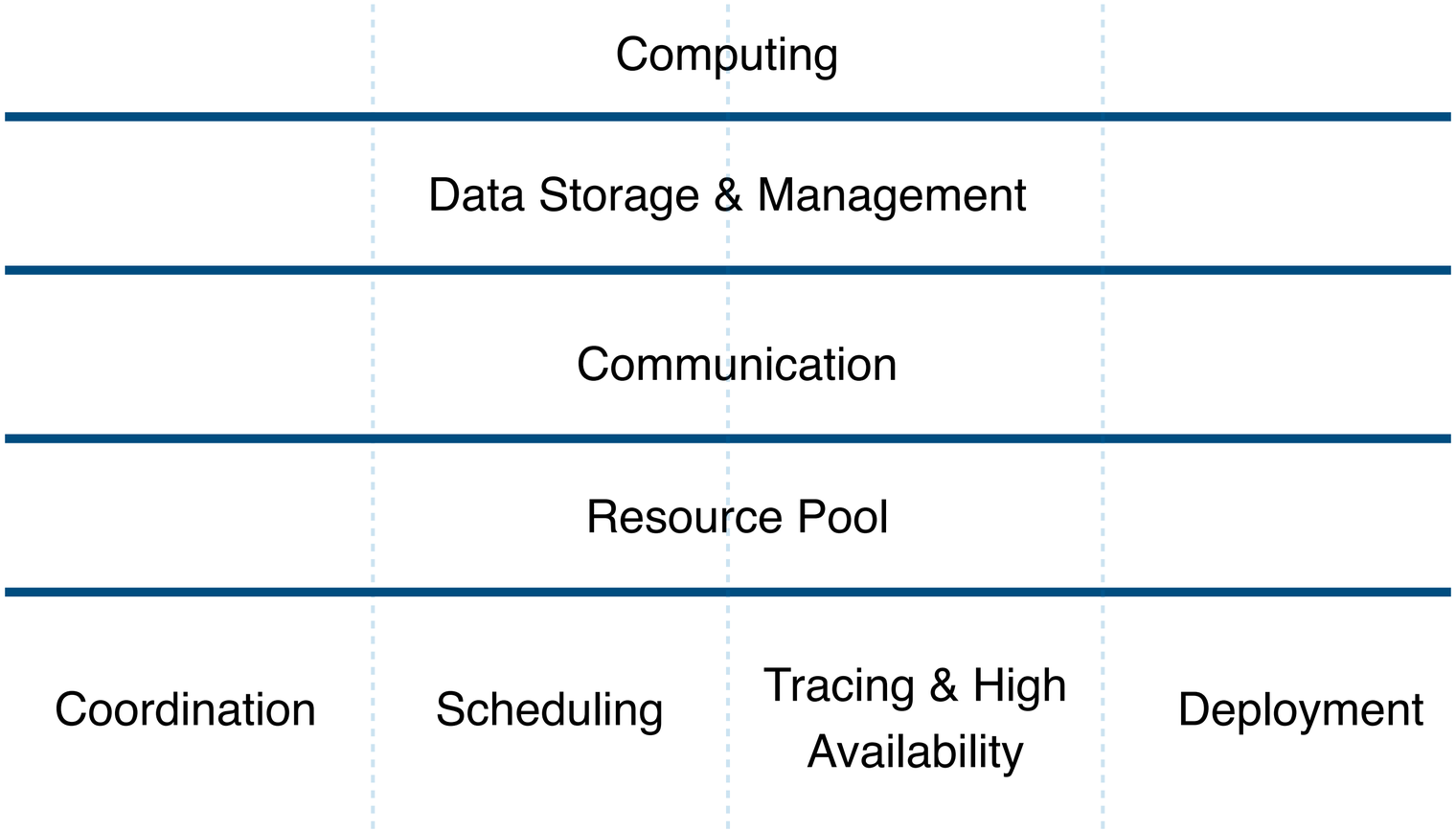
2021-05-24

Alan

SAIL

An Overview of Distributed System

Distributed System



Topics about Distrubuted System

- **Computing**: Actor Model, Pipeline Model, MapReduce, Streaming;
- **Data Storage & Management**: CAP Theorem, Strong/Eventually/Weak Consistency(Quorum Mechanism, N-phase Commit), Distrubuted DB(Spanner, Ceph, GFS);
- **Communication**: RPC, Pub-sub, Message Queue;

Topics about Distrubuted System

- **Coordination:**
  - **Mutual Exclusion:** Centralized Algorithm, Decentralized Algorithm, Token Ring Algorithm;
  - **Election**: Bully, Raft, ZAB(ZooKeeper Atomic Broadcast);
  - **Consensus**: PoW, PoS, DPoS;
  - **Transaction**: 2PC based on XA, 3PC.

- **Scheduling:**
  - **Architecture Perspective:** Centralized(K8s), Decentralized(Akka Cluster);
  - **Scheduling Perspective**: One-level(Borg), Two-level(Mesos), Shared Status(Omega);

- **High Availability & Reliability**:
  - **Load Balance**: Round Robin, Random, Hash;
  - **Traffic Control**: Leaky Bucket, Token Bucket;
  - **Fault Tolerance**.

Raft

Replicated State Machine

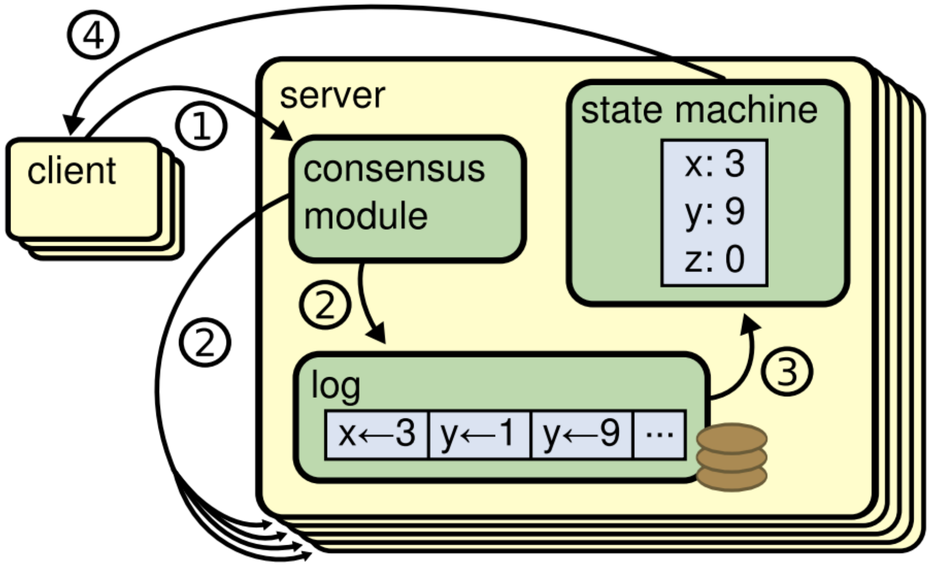


Figure 2.1: Replicated state machine architecture. The consensus algorithm manages a replicated log containing state machine commands from clients. The state machines process identical sequences of commands from the logs, so they produce the same outputs.

Server States

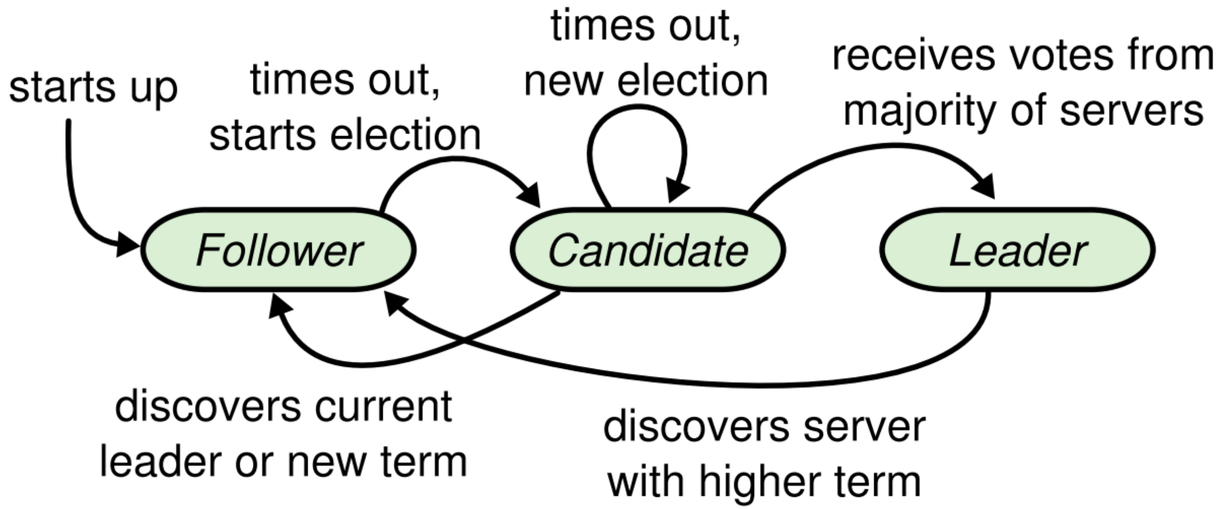


Figure 3.3: Server states. Followers only respond to requests from other servers. If a follower receives no communication, it becomes a candidate and initiates an election. A candidate that receives votes from a majority of the full cluster becomes the new leader. Leaders typically operate until they fail.

Logical Clock

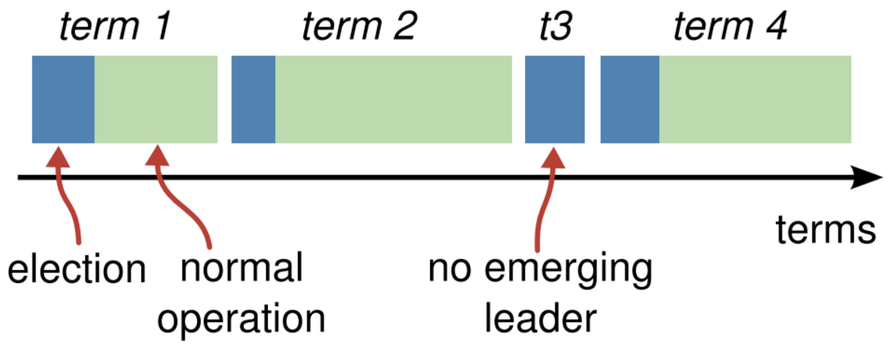


Figure 3.4: Time is divided into terms, and each term begins with an election. After a successful election, a single leader manages the cluster until the end of the term. Some elections fail, in which case the term ends without choosing a leader. The transitions between terms may be observed at different times on different servers.

Server Communication

- **RequestVote RPC**: Invoked by candidates to gather votes

- **AppendEntries RPC**: Invoked by leader to replicate log entries; also used as heartbeat

How to Election?

See this animation.

Log Replication

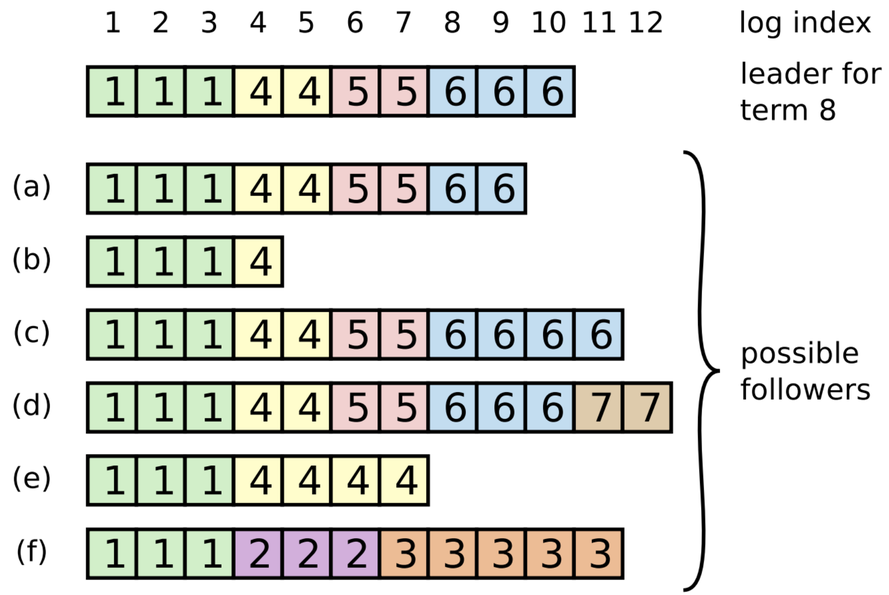


Figure 3.6: When the leader at the top comes to power, it is possible that any of scenarios (a–f) could occur in follower logs. Each box represents one log entry; the number in the box is its term. A follower may be missing entries (a–b), may have extra uncommitted entries (c–d), or both (e–f). For example, scenario (f) could occur if that server was the leader for term 2, added several entries to its log, then crashed before committing any of them; it restarted quickly, became leader for term 3, and added a few more entries to its log; before any of the entries in either term 2 or term 3 were committed, the server crashed again and remained down for several terms.

Log Compression

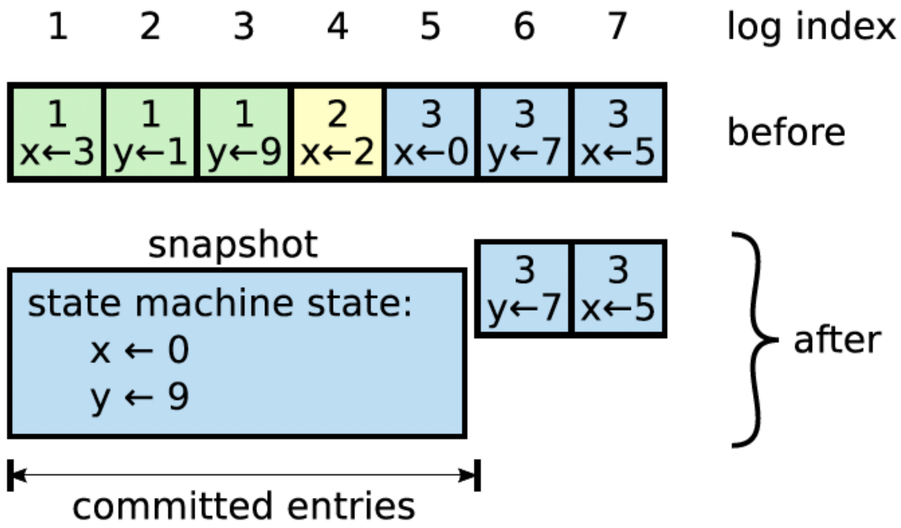


Figure 5.2: A server replaces the committed entries in its log (indexes 1 through 5) with a new snapshot, which stores just the current state (variables x and y in this example). Before discarding entries 1 though 5, Raft saves the snapshot's last included index (5) and term (3) to position the snapshot in the log preceding entry 6.

Thanks