4. Leader and Follower

Let's learn about the leader and follower patterns and its usage in distributed systems.

We'll cover the following

Background#

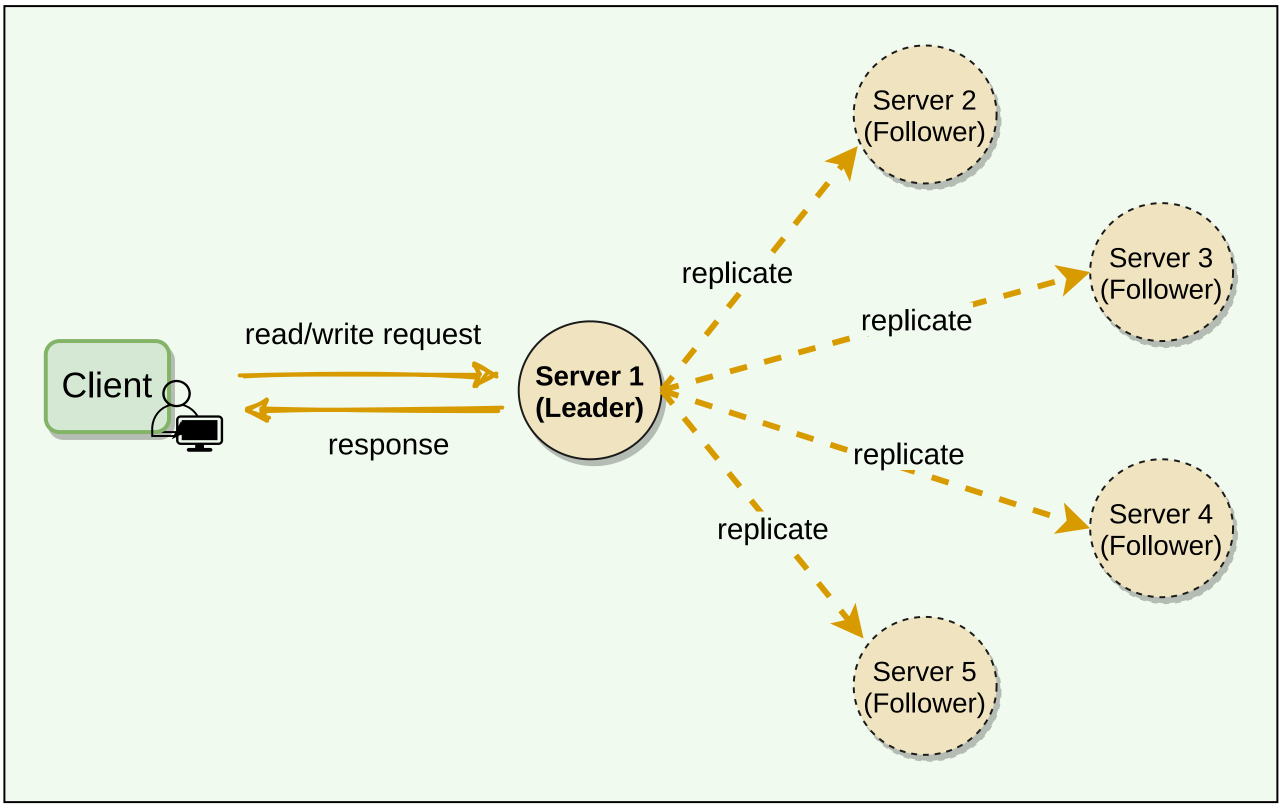
Distributed systems keep multiple copies of data for fault tolerance and higher availability. A system can use quorum to ensure data consistency between replicas, i.e., all reads and writes are not considered successful until a majority of nodes participate in the operation. However, using quorum can lead to another problem, that is, lower availability; at any time, the system needs to ensure that at least a majority of replicas are up and available, otherwise the operation will fail. Quorum is also not sufficient, as in certain failure scenarios, the client can still see inconsistent data.

Definition#

Allow only a single server (called leader) to be responsible for data replication and to coordinate work.

Solution#

At any time, one server is elected as the leader. This leader becomes responsible for data replication and can act as the central point for all coordination. The followers only accept writes from the leader and serve as a backup. In case the leader fails, one of the followers can become the leader. In some cases, the follower can serve read requests for load balancing.



Leader entertains requests from the client and is responsible for replicating and coordinating with followers

Examples#

- In **Kafka**, each partition has a designated leader which is responsible for all reads and writes for that partition. Each follower's responsibility is to replicate the leader's data to serve as a "backup" partition. This provides redundancy of messages in a partition, so that a follower can take over the leadership if the leader goes down.
- Within the **Kafka** cluster, one broker is elected as the **Controller**. This Controller is responsible for admin operations, such as creating/deleting a topic, adding partitions, assigning leaders to partitions, monitoring broker failures, etc. Furthermore, the Controller periodically checks the health of other brokers in the system.
- To ensure strong consistency, **Paxos** (hence **Chubby**) performs leader election at startup. This leader is responsible for data replication and coordination.

← Back

3. Quorum

Next →

5. Write-ahead Log

✓ Mark as Completed

Report an Issue