1. Key Unavailability
To deal with key unavailability, for any simple key, we append a space+nonce to the end to make a compound key. Nonce starts at 0 and every time the compound key becomes unavailable, increment the nonce and use the next key.

2. Leader Election & Broadcast
The key for publishing the active nodes and leader is “leader key”. The leader periodically reads the value at leader key (to see if there’s a broadcast from other potential leader, see below) and broadcasts the following information if it hasn’t been dethroned as the leader (see below):
Era Nonce Version Id Activenode1 Activenode2 Activenode3...

Era is incremented every time a new leader is elected.
Nonce is the nonce (see **1**) at which the node declares itself as the leader.
Version is incremented every time leader published its list of active nodes.

If a leader fails, it will stop broadcasting and the **Version** stays the same. Any follower that sees no changes in the Version after a sufficient period of time will declare itself as a leader (i.e. potential leader). (Era, Nonce) defines a total ordering on the leaders. At any moment, the potential leader who has the highest Era, followed by the lowest Nonce will be elected as the leader. Any other potential leader will drop out of the race and become a follower. Any follower with an older leader will switch to following the newly elected leader. Due to key unavailability, some nodes may follow the wrong leader or think itself as the leader, but if there’s a period with no failure, eventually they will all read a broadcast from the real leader and follow it afterward.

3. Follower Nodes & Joining the Network
The follower will use its id as the key to PING the leader. The value at “join key” is a list of ids separated by space. A follower can join the active nodes by appending its id at “join key”. The leader will read “join key” periodically and add any joining node to the active list and PONG the follower at its id. Then the leader and the follower continues the PING-PONG cycle until one fails. If the follower fails, then it won’t PING back to the leader and the leader will remove the follower from the active nodes. If the leader fails, the follower will know to try to become the new leader.

4. Broadcasting and Retrieving List of Active Nodes
The leader uses a JOIN thread that polls for any joining nodes and send it to the LEADER thread. The LEADER thread adds any joining nodes, deletes disconnected nodes, and broadcasts them (see 3) periodically. The leader also uses separate threads for PONGing any active nodes. The separate threads help reduce latency between the asynchronous rpc calls to ensure the leader doesn’t delay any of its operations for too long. The follower also periodically retrieves the active nodes at leader key.