KIP-537: Increase default zookeeper session timeout

Status

Current state: Adopted

Discussion thread:

JIRA: Unable to render Jira issues macro, execution error.

Please keep the discussion on the mailing list rather than commenting on the wiki (wiki discussions get unwieldy fast).

Motivation

The current default for `zookeeper.session.timeout.ms` is 6s. This was reasonable for controlled local datacenter environments, but over time, Kafka has increasingly been deployed in more unstable cloud environments. More unstable conditions means more spurious timeouts which can have a bad impact on partition availability. Here we propose to increase the default to 18s.

Public Interfaces

The default values changes in this KIP are listed in the table below.

Configuration	Current Default	New Default
zookeeper.session.timeout.ms	6000	18000
replica.lag.time.max.ms	10000	30000

Proposed Changes

As discussed above, we will increase the default zookeeper session timeout to 18s. The tradeoff with a larger session timeout is that it allows the system to smooth over transient instability at the cost of slower detection of genuine failures. Based on our experience, genuine failures are rare in proportion to the various events that can cause a node to temporarily lose connectivity to the zookeeper quorum.

With the change to `zookeeper.session.timeout.ms`, we will make a similar adjustment `replica.lag.time.max.ms`, which controls how aggressively the leader removes lagging replicas from the ISR. A longer timeout gives replicas more time to catch up before getting kicked from the ISR. The reason to use a value which is larger is to avoid a race condition with the controller in the case of a failure. We want the controller to detect the failure first since it is more efficient for it to remove the node from all ISRs at once.

A quick discussion on replica lag: Intuitively, it might seem reasonable to be more aggressive with ISR eviction. That is, we might consider letting the lag time be *smaller than* the session timeout. The sooner a replica is removed from the ISR, the sooner the partition may be able to accept writes again. However, there are two reasons why this is not so simple. First, when a replica is failing to keep up, it is often not clear whether the problem is on the leader or the follower. It might just be that the leader is failing to work through a backlog of requests quickly enough. We have seen this many times. In this case, shrinking the ISR actually makes recovery more difficult because we are removing a potential leader from the ISR. Secondly, if a follower is genuinely not keeping up, then removing it from the ISR means that the broker gives up its ability to exert back-pressure on the clients through the advancement of the high watermark. If this is a persistent condition, then the lagging broker will fall further and further behind. For these reasons, we think it is smarter to be conservative about shrinking the ISR.

Compatibility, Deprecation, and Migration Plan

The new defaults are more conservative, so we think the impact will be low. Users who have set custom values will not be affected.

Rejected Alternatives

None yet.