KIP-39 Pinning controller to broker

Status

Current state:

Discussion thread:

JIRA: KAFKA-1778

Released:

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

Motivation

In this KIP we propose to designate a broker to be preferred controller. This would minimize controller moves during a rolling bounce as well as enable us to choose a lightly loaded broker which is serving fewer partitions

to limit controller functions being affected.

New or Changed Public Interfaces

We will need to change the admin toots to indicate a broker they want to bind the controller to.

Proposed Change

Client will send
a) We send the admin request to any of the broker.
b) It will create a persistent zookeeper node /admin/next_controller with data x.

Handling the admin request in the controller:
c) The controller has a watch on this admin znode getChildren.
d) Controller will add a watch on admin/ready_to_serve ephemeral znode.
e) The broker x if alive will receive the watch on /admin/next_controller and set the admin/ready_to_serve ephemeral node.
f) When controller receives the watch it will resign.
g) At this point elections are triggered.

Changes in the election code:
a) All the brokers will pull from /admin/ready_to_server_as_controller with a watch.
b) If the brokers find that if this znode exists and their broker.id does not match the id specified in this ephemeral node they will simply not participate in the leader election.
c) Broker x will rightfully takes its place as the next controller.

Changes in the broker startup code:
a) Always pull from the /admin/next_controller for data changes as well as new data.
b) If there is any change try to setup the next broker similar to what has been specified in "handling the admin request in the controller"

Possible gotchas:
a) Controller resigns and at the same time broker x also crashes.
b) At this point there are 3 possibilities:
I) None of the brokers knew about the ready_to_serve_as_controller znode and broker x got elected by random chance.
(This can happen today and I believe this would trigger another round of elections.)
II) All of the brokers knew about the ready_to_serve_as_contoller ephemeral znode and hence did not participate in elections. Now if broker x were to crash this will trigger a watch on ready_to_serve ephemeral node and this will be treated as a sign of fresh elections by these nodes.
III) In this case some subset of nodes know about the ready_to_serve_as_controller ephemeral node and some do not. The ones which do not will jump into elections as soon as the controller dies. The ones which do know about it will also jump into elections because of the ready_to_serve ephemeral node watch.

This ensures there would not be a case where we do not have a controller but have non-zero number of brokers up and running.

Migration Plan, Compatibility and Deprecation

We can rollout the new broker bit by bit since it just limits participation of new brokers in election only under some conditions and so if a subset of new brokers has the new code all that will happen is that they might not participate
in the election.

Rejected Alternatives

Use config to whitelist or blacklist a subset of brokers which could be controller. The potential problem with this approach if that if there is a single broker up and if it is not part of the whitelist or is a part of blacklisted brokers then we could be controller-less.