KIP-491: Preferred Leader Deprioritized List (Temporary Blacklist)

Status

Current state: *Under Discussion*

Discussion thread: *here*

JIRA: *KAFKA-8638*

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

Motivation

Currently, the kafka preferred leader election will pick the broker_id in the topic/partition replica assignments in a priority order when the broker is in ISR. The preferred leader is the broker id in the first position of replica. There are use-cases that, even the first broker in the replica assignment is in ISR, there is a need for it to be moved to the end of ordering (lowest priority) when deciding leadership during preferred leader election.

Let's use topic/partition replica `(1,2,3)` as an example. 1 is the preferred leader. When preferred leadership is run, it will pick 1 as the leader if it's ISR, if 1 is not online and in ISR, then pick 2, if 2 is not in ISR, then pick 3 as the leader. There are use cases that, even 1 is in ISR, we would like it to be moved to the end of ordering (lowest priority) when deciding leadership during preferred leader election. Below is a list of use cases:

- (If broker_id 1 is a swapped failed host and brought up with last segments or latest offset without historical data (There is another effort on this), it's better for it to not serve leadership till it's caught-up.
- The cross-data center cluster has AWS cloud instances which have less computing power than the on-premises bare metal machines (heterogeneous hardware). We could put the AWS broker_ids in Preferred Leader Blacklist, so on-premises brokers can be elected leaders, without changing the reassignments ordering of the replicas.
- If the broker_id 1 is constantly losing leadership after some time: "Flapping". we would want to exclude 1 to be a leader unless all other brokers of this topic/partition are offline. The "Flapping" effect was seen in the past when 2 or more brokers were bad, when they lost leadership constantly /quickly, the sets of partition replicas they belong to will see leadership constantly changing. The ultimate solution is to swap these bad hosts. But for quick mitigation, we can also put the bad hosts in the Preferred Leader Blacklist to move the priority of its being elected as leaders to the lowest.
- If the controller is busy serving an extra load of metadata requests and other tasks. we would like to put the controller's leaders to other brokers to lower its CPU load. currently bouncing to lose leadership would not work for Controller, because after the bounce, the controller fails over to another broker.
- Avoid bouncing broker in order to lose its leadership: it would be good if we have a way to specify which broker should be excluded from serving traffic/leadership (without changing the replica assignment ordering by reassignments, even though that's quick), and run preferred leader election. A bouncing broker will cause temporary URP, and sometimes other issues. Also a bouncing of broker (e.g. broker_id 1) can temporarily lose all its leadership, but if another broker (e.g. broker_id 2) fails or gets bounced, some of its leaderships will likely failover to broker_id 1 on a replica with 3 brokers. If broker_id 1 is in the blacklist, then in such a scenario even broker_id 2 offline, the 3rd broker can take leadership.

Public Interfaces

Introduce a `preferred_leader_blacklist` dynamic config which by default is empty. It allows a list of broker IDs separated by commas. E.g. below broker ID `1,10,65` are being put into the blacklist.

```
/usr/lib/kafka/bin/kafka-configs.sh --bootstrap-server localhost:9092 --entity-type brokers --entity-default --
alter --add-config preferred_leader_blacklist=1,10,65
```

Since the Kafka dynamic config is already using `--bootstrap-server`, it does not need to manipulate the Zookeeper directly. The downside of this: when adding/removing one broker from the list, instead of doing with one ZK node per broker in another design `/preferred_leader_blacklist /<broker_id>` znode, the dynamic config needs to be updated with a new complete list. e.g. in order to remove broker `10` from the blacklist, update `preferred_leader_blacklist=1,65`

The dynamic config should not trigger any leadership changes automatically for the current design.

Proposed Changes

The following is the requirements this KIP is trying to accomplish:

- The ability to add and remove the preferred leader deprioritized list/blacklist. e.g. new ZK path/node or new dynamic config.
- The logic to determine the priority/order of which broker should be preferred leader should be modified. The broker in the preferred leader blacklist should be moved to the end (lowest priority) when determining leadership.
- The blacklist can be at the broker level. However, there might be use cases where a specific topic should blacklist particular brokers, which would be at the Topic level Config. For this use cases of this KIP, it seems that broker level blacklist would suffice. Topic level preferred leader blacklist might be future enhancement work.

The preferred leader blacklist should only be used for leadership determination when either of the two gets triggered below:

1. Preferred leader election is run.
2. When a broker fails, the controller transfer all of this broker's leaderships to the next one in priority. When determining the priority of the leader, should look up the preferred leader blacklist of the brokers and move it to the lowest.
3. When auto.leader.rebalance.enable is enabled. The broker(s) in the preferred leader "blacklist" should be excluded from being elected leaders.

The current design also does not automatically put a broker in the preferred leader blacklist. E.g. when the controller starts up itself or got controller failover, it will put itself to the blacklist. This may be an enhancement later.

Compatibility, Deprecation, and Migration Plan

- This feature will give the Kafka system administrator/on-call engineers the ability to quickly address some issues with bad hardware which should not be serve leadership traffic. Most of the use-cases for the blacklist is temporary. Though some of the case like heterogeneous hardware might be persistent for a longer time.
- The behavior of preferred leader election and failed brokers new leader election logic will be changed. In any cases, the broker in preferred leader blacklist are not excluded from leadership election per se. It's just moved to the lowest priority when determining leadership for a topic /partition.

Rejected Alternatives

- The current work-around of the above is to change the topic/partition's replica reassignments to move the broker_id 1 from the first position to the last position and run preferred leader election. e.g. `(1, 2, 3) => (2, 3, 1)`. This changes the replica reassignments, and we need to keep track of the original one and restore if things change (e.g. controller fails over to another broker, the swapped empty broker caught up). That's a rather tedious task.
- Rejected the design to put the preferred leader blacklist in the per broker zookeeper node. e.g. /preferred_leader_blacklist/<broker_id>. This will introduce new RPC request/response to manipulate these ZK nodes. Also in the future, there might be a need to make this blacklist at the topic Config level.