KIP-460: Admin Leader Election RPC

Status

Current state: Accepted

Discussion thread: *here*

JIRA:

> Unable to render Jira issues macro, execution error.

PR: https://github.com/apache/kafka/pull/6686

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

Motivation

The admin client allows users to instruct the controller to attempt to elect the preferred replica as leader on a given set of topic partitions. In addition to preferred replica leader elections the controller also supports three other types of elections. One of those other election types is what we call unclean leader election. For the user to enable and trigger unclean leader election they need to either modify the topic configuration or the broker configuration. We believe that this is potentially a dangerous configuration that could lead to data loss. Specially if the user forgets to disable unclean leader election after it completes. Being able to trigger unclean leader election once for a given partition is a much safer mechanism.

In this proposal we modify the “PreferredLeaderElection” RPC to support unclean leader election in addition to preferred leader election. The new RPC definition also makes it possible to easily add new type of elections in the future.

Public Interfaces

Network protocol

Request

Added a top level property called *ElectionType* with supported values of *0* and *1*.

```
{
  "apiKey": 43,
  "type": "request",
  "name": "ElectLeadersRequest",
  "validVersions": "0-1",
  "fields": [
    { "name": "ElectionType", "type": "int8", "versions": "1+",
      "about": "Type of elections to conduct for the partition. A value of '0' elects the preferred replica. A
value of '1' elects the first live replica if there are no in-sync replica." },
    { "name": "TopicPartitions", "type": "[]TopicPartitions", "versions": "0+", "nullableVersions": "0+",
      "about": "The topic partitions to elect leaders.",
      "fields": [
        { "name": "Topic", "type": "string", "versions": "0+",
          "about": "The name of a topic." },
        { "name": "PartitionId", "type": "[]int32", "versions": "0+",
          "about": "The partitions of this topic whose leader should be elected." }
      ]
    },
    { "name": "TimeoutMs", "type": "int32", "versions": "0+", "default": "60000",
      "about": "The time in ms to wait for the election to complete." }
  ]
}
```

Response

Added a top level property called *ErrorCode* for returning errors that apply to all the topic partitions. The current implementation is using this to return cluster authorization errors.

```
{
  "apiKey": 43,
  "type": "response",
  "name": "ElectLeadersResponse",
  "validVersions": "0-1",
  "fields": [
    { "name": "ThrottleTimeMs", "type": "int32", "versions": "0+",
      "about": "The duration in milliseconds for which the request was throttled due to a quota violation, or
zero if the request did not violate any quota." },
    { "name": "ErrorCode", "type": "int16", "versions": "1+", "ignorable": false,
      "about": "The top level response error code." },
    { "name": "ReplicaElectionResults", "type": "[]ReplicaElectionResult", "versions": "0+",
      "about": "The election results, or an empty array if the requester did not have permission and the
request asks for all partitions.", "fields": [
      { "name": "Topic", "type": "string", "versions": "0+", "entityType": "topicName",
        "about": "The topic name" },
      { "name": "PartitionResult", "type": "[]PartitionResult", "versions": "0+",
        "about": "The results for each partition", "fields": [
        { "name": "PartitionId", "type": "int32", "versions": "0+",
          "about": "The partition id" },
        { "name": "ErrorCode", "type": "int16", "versions": "0+",
          "about": "The result error, or zero if there was no error."},
        { "name": "ErrorMessage", "type": "string", "versions": "0+", "nullableVersions": "0+",
          "about": "The result message, or null if there was no error."}
      ]}
    ]}
  ]
}
```

AdminClient Abstract Class

A new method will be added to the AdminClient abstract class to support this new version of the RPC.

When the *partition* parameter is null the controller will attempt the election type on all of partitions that are legible for election. This means that the RPC will only return success or failure on partition that eligible for election as oppose to all of the partitions in the cluster as the previous implemented behaved.

```
package org.apache.kafka.clients.admin;

public abstract class AdminClient ... {
    ...

    /**
     * ...
     * @deprecated Since 2.4.0. Use {@link #electLeaders}.
     */
    @Deprecated
    public abstract ElectPreferredLeadersResult electPreferredLeaders(Collection<TopicPartition> partitions,
                                                                      ElectPreferredLeadersOptions options);


    /**
     * ...
     *
     * @param electionType            The type of election.
     * @param partitions              The partitions for which to conduct elections.
     * @param options                 The options to use when electing the leaders.
     * @return                        The ElectLeadersResult.
     */
    public abstract ElectLeadersResult electLeaders(
            ElectionType electionType,
            Set<TopicPartition> partitions,
            ElectLeadersOptions options);
}
```

ElectLeaderResult Class

```
final public ElectLeadersResult {
    ...

    public KafkaFuture<Map<TopicPartition, Optional<Throwable>>> partitions() {
        ...
    }

    public KafkFuture<Void> all() {
        ...
    }
}
```

ElectionType Enumeration

```
package org.apache.kafka.common;

public enum ElectionType {
    PREFERRED((byte) 0), UNCLEAN((byte) 1);

    public final byte value;

    ElectionType(byte value) {
        this.value = value;
    }
}
```

Admin Command

The command *kafka-preferred-replica-election.{sh,bat}* will be deprecated and the following command will be added.

```
$ bin/kafka-leader-election.sh --help
This tool attempts to elect a new leader for a set of topic partitions. The type of elections supported are
preferred replicas and unclean replica.
Option                                  Description
------                                  -----------

--admin.config <String: config file>    Admin client config properties file to pass to the admin
                                          client when --bootstrap-server is given.
--all-topic-partitions                  Perform election on all of the topic partitions. Not allowed if
                                          --topic, --partition or --path-to-json-file is specified.
--bootstrap-server <String: host:port>  A host name and port for the broker to
                                          connect to, in the form host:port.
                                          Multiple comma-separated URLs can be
                                          given. REQUIRED unless --zookeeper
                                          is given.
--election-type <String: election>      Type of election to attempt. Possible values are "preferred"
                                          for preferred election or "unclean" for unclean election.
                                          REQUIRED.
--topic <String: topic>                 Name of topic for which to perform an election.
                                          REQUIRED if --partition is specified. Not allowed
                                          if --path-to-json-file or --all-topic-partitions is specified.
--partition <Integer: partition id>     Partition id for which to perform an election.
                                          REQUIRED if --topic is specified. Not allowed if
                                          --path-to-json-file or --all-topic-partitions is specified.
--help                                  Print usage information.
--path-to-json-file <String: list of    The JSON file with the list of
  partitions for which replica            partitions for which leader election
  leader election needs to be             should be done. Supported elections
  triggered>                              are 0 for preferred and 1 for unclean.
                                          If an election is not specified,
                                          preferred is the default. This is an
                                          example format.
                                        {"partitions":
                                               [{"topic": "foo", "partition": 1},
                                                {"topic": "foobar", "partition": 2}]
                                        }
                                          Not allowed if --all-topic-partitions, --topic or --partition
                                          flags are specified.
```

Metrics

The following metrics were added:

1. kafka.server:type=DelayedOperationPurgatory,name=NumDelayedOperations,delayedOperation=ElectLeader
2. kafka.server:type=DelayedOperationPurgatory,name=PurgatorySize,delayedOperation=ElectLeader

While the following metrics were removed:

1. kafka.server:type=DelayedOperationPurgatory,name=NumDelayedOperations,delayedOperation=ElectPreferredLeader
2. kafka.server:type=DelayedOperationPurgatory,name=PurgatorySize,delayedOperation=ElectPreferredLeader

Proposed Changes

In addition to the protocol and client changes enumerated above, the Controller will be extended to allow unclean leader election requests to come from the admin client. Currently the controller only supports preferred leader election from the admin client. Unclean leader election can only be enabled through either a topic configuration change or a broker configuration change.

When performing unclean leader election on a topic partition through the admin client the "unclean leader election" topic configuration and broker configuration will be ignored. In the code this will result is code similar to the one below:

```
uncleanElectionFromAdminClient || logConfigs(partition.topic).uncleanLeaderElectionEnable.booleanValue()
```

Compatibility, Deprecation, and Migration Plan

1. The *kafka-preferred-replica-elections.{sh,bat}* scripts will be deprecated.

2. The *electPreferredLeaders* method in *AdminClient* will be deprecated.
3. The *DelayedOperationPurgatory* metrics *ElectPreferredLeader* were replaced with *ElectLeader*

Rejected Alternatives

Not applicable.