KIP-704: Send a hint to the partition leader to recover the partition

- Status
- Motivation
- Proposed Changes
    - Topic Partition Leader
    - Topic Partition Follower
    - Controller
        - ZooKeeper Controller
        - KRaft Controller
- Changed Public Interfaces
    - ZooKeeper State
    - LeaderAndIsr RPC
        - Request Schema
        - Request Handling
        - Response Schema
    - AlterPartition RPC
        - Request Schema
            - Request Handling
        - Response Schema
    - Cluster Metadata Records
        - PartitionRecord
        - PartitionChangeRecord
- Compatibility, Deprecation, and Migration Plan
    - Controller
    - Clients
- Rejected Alternatives
    - Unclean Election Epoch
    - Recovering Election Boolean

Status

Current state: Accepted

Discussion thread: https://lists.apache.org/thread/ld2t2xkby7rpgrggqo1h344goddfdnxb

JIRA: KAFKA-13587

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

Motivation

If none of the in-sync replicas are alive, the controller allows the user to elect a replica that was not a part of the in-sync replica set using the unclean leader election strategy. Since this new leader replica was not a part of the last known in-sync replica set, there is the possibility for data loss by deleting log records committed by the previous leader(s). In addition, the loss of these records can cause some inconsistency with other parts of the system like the transaction coordinators and the group coordinators. If the controller is able to communicate to the new topic partition leader that it was elected using unclean leader election, the new topic partition leader can coordinate this recovery and mitigate these inconsistencies.

Proposed Changes

This KIP proposes extending the communication between the controller and brokers to include enough information so that topic partition leaders know if they need to recover their state. This feature needs to support both the traditional ZooKeeper controller and the more recent KRaft controller. The messages sent by the controller to the broker during leader election will be extended to include if the leader was elected using unclean leader election. It is important to note that when the controller performs unclean leader election the ISR size is 1 and the leader recovery state field will be RECOVERING. In such cases the topic partition leader will perform any necessary recovery steps. Once the leader has recovered it will change the leader recovery state to RECOVERED.

The LeaderRecoveryState field will be an int8 and will have the values of RECOVERED (0) and RECOVERING (1).

Topic Partition Leader

The brokers will communicate to the controller when it has finished recovering from unclean leader election. An important invariant that the partition leader must satisfied is that the ISR size is 1 while the leader recovery state field is RECOVERING. This means that the leader will not allow followers to join the ISR until it has recovered from the unclean leader election. This also means that if the leader changes while a partition is still recovering, the new leader must be elected using the unclean leader election strategy.

While the leader is recovering from an unclean leader election it will return a NOT_LEADER_OR_FOLLOWER error for the FETCH, PRODUCE, LIST_OFFSETS, DELETE_RECORDS and OFFSET_FOR_LEADER_EPOCH requests.

One of the ways Kafka is going to use this feature is to abort all pending transaction when recovering from unclean leader election. See

⚠ Unable to render Jira issues macro, execution error.

for more details.

Topic Partition Follower

With KIP-392, it is possible for follower to receive FETCH request from the consumer. Follower will return a NOT_LEADER_OR_FOLLOWER error while the leader it sill recovering. This means that the follower will return this error when it receives a LEADER_AND_ISR request with RECOVERING set until it receives another request with RECOVERED. The follower will not ignore request that have the same leader epoch.

Controller

ZooKeeper Controller

The controller will use ZooKeeper to persist the decision that it elected a leader through the unclean leader election strategy. This decision will get propagated to all of the affected brokers through the LeaderAndIsr request. Topic partition leaders will inform the controller that they have recovered from the unclean leader election using the AlterPartition (previously named AlterIsr) request.

KRaft Controller

The controller will persist the decision that it elected a leader through the unclean leader election strategy in the cluster metadata internal partition using the PartitionRecord and PartitionChangeRecord. These records will get replicated to all of the brokers using the cluster metadata topic partition. Topic partition leaders will inform the controller that they have recovered from unclean leader election using the AlterPartition request.

Changed Public Interfaces

ZooKeeper State

The state of the partition stored in `/brokers/topics/<topic>/partitions/<partition>/state` as JSON will be extended to add a new property named `leader_recovery_state` to have the following schema:

```
{
  "version": 1,
  "leader": NUMBER,
  "leader_epoch": NUMBER,
  "controller_epoch": NUMBER,
  "isr" ARRAY of NUMBERS,
  "leader_recovery_state": NUMBER // New property
}
```

The property `leader_recovery_state` indicates if the leader was elected uncleanly and if the leader should recover it's state. The default value for this property, in case it is missing, is `RECOVERED` .

LeaderAndIsr RPC

Request Schema

Add a property to indicate to the leader of the topic partition that it was elected using unclean leader election and it must recover the partition.

```
{
  "apiKey": 4,
  "type": "request",
  "listeners": ["zkBroker"],
  "name": "LeaderAndIsrRequest",
  // Version 1 adds IsNew.
  //
  // Version 2 adds broker epoch and reorganizes the partitions by topic.
  //
  // Version 3 adds AddingReplicas and RemovingReplicas.
  //
  // Version 4 is the first flexible version.
  //
  // Version 5 adds Topic ID and Type to the TopicStates, as described in KIP-516.
  //
  // Version 6 adds LeaderRecoveryState as describe in KIP-704.
  "validVersions": "0-6",
  "flexibleVersions": "4+",
  "fields": [
    { "name": "ControllerId", "type": "int32", "versions": "0+", "entityType": "brokerId",
      "about": "The current controller ID." },
    { "name": "ControllerEpoch", "type": "int32", "versions": "0+",
      "about": "The current controller epoch." },
    { "name": "BrokerEpoch", "type": "int64", "versions": "2+", "ignorable": true, "default": "-1",
      "about": "The current broker epoch." },
    { "name": "Type", "type": "int8", "versions": "5+",
      "about": "The type that indicates whether all topics are included in the request"},
    { "name": "UngroupedPartitionStates", "type": "[]LeaderAndIsrPartitionState", "versions": "0-1",
      "about": "The state of each partition, in a v0 or v1 message." },
    // In v0 or v1 requests, each partition is listed alongside its topic name.
    // In v2+ requests, partitions are organized by topic, so that each topic name
    // only needs to be listed once.
    { "name": "TopicStates", "type": "[]LeaderAndIsrTopicState", "versions": "2+",
      "about": "Each topic.", "fields": [
      { "name": "TopicName", "type": "string", "versions": "2+", "entityType": "topicName",
        "about": "The topic name." },
      { "name": "TopicId", "type": "uuid", "versions": "5+", "ignorable": true,
        "about": "The unique topic ID." },
      { "name": "PartitionStates", "type": "[]LeaderAndIsrPartitionState", "versions": "2+",
        "about": "The state of each partition" }
    ]},
    { "name": "LiveLeaders", "type": "[]LeaderAndIsrLiveLeader", "versions": "0+",
      "about": "The current live leaders.", "fields": [
      { "name": "BrokerId", "type": "int32", "versions": "0+", "entityType": "brokerId",
        "about": "The leader's broker ID." },
      { "name": "HostName", "type": "string", "versions": "0+",
        "about": "The leader's hostname." },
      { "name": "Port", "type": "int32", "versions": "0+",
        "about": "The leader's port." }
    ]}
  ],
  "commonStructs": [
    { "name": "LeaderAndIsrPartitionState", "versions": "0+", "fields": [
      { "name": "TopicName", "type": "string", "versions": "0-1", "entityType": "topicName", "ignorable": true,
        "about": "The topic name.  This is only present in v0 or v1." },
      { "name": "PartitionIndex", "type": "int32", "versions": "0+",
        "about": "The partition index." },
      { "name": "ControllerEpoch", "type": "int32", "versions": "0+",
        "about": "The controller epoch." },
      { "name": "Leader", "type": "int32", "versions": "0+", "entityType": "brokerId",
        "about": "The broker ID of the leader." },
      { "name": "LeaderEpoch", "type": "int32", "versions": "0+",
        "about": "The leader epoch." },
      { "name": "Isr", "type": "[]int32", "versions": "0+", "entityType": "brokerId",
        "about": "The in-sync replica IDs." },
      { "name": "ZkVersion", "type": "int32", "versions": "0+",
        "about": "The ZooKeeper version." },
      { "name": "Replicas", "type": "[]int32", "versions": "0+", "entityType": "brokerId",
        "about": "The replica IDs." },
      { "name": "AddingReplicas", "type": "[]int32", "versions": "3+", "ignorable": true, "entityType":
"brokerId",
        "about": "The replica IDs that we are adding this partition to, or null if no replicas are being
added." },
      { "name": "RemovingReplicas", "type": "[]int32", "versions": "3+", "ignorable": true, "entityType":
"brokerId",
        "about": "The replica IDs that we are removing this partition from, or null if no replicas are being
removed." },
      { "name": "IsNew", "type": "bool", "versions": "1+", "default": "false", "ignorable": true,
        "about": "Whether the replica should have existed on the broker or not." },
      // -------- Properties added by this KIP ---------
      { "name": "LeaderRecoveryState", "type": "int8", "versions": "6+", "default": "0",
        "about": "Indicates if the partition is recovering from an election." }
    ]}
  ]
}
```

Request Handling

When the leader receives an LeaderAndIsr request with the LeaderRecoveryState set to RECOVERING it will attempt to recover the partition irrespective of the leader's IBP.

Response Schema

No changes required to the response.

AlterPartition RPC

Request Schema

The name of the request will be changed to AlterPartitionRequest from AlterIsrRequest. The field CurrentIsrVersion will be renamed to PartitionEpoch. The schema will include a new property for the leader to indicate to the controller when it finished recovering the partition from an unclean election.

```
  {
    "apiKey": 56,
    "type": "request",
    "listeners": ["zkBroker", "controller"],
    "name": "AlterPartitionRequest",
    "validVersions": "0-1",
    "flexibleVersions": "0+",
    "fields": [
      { "name": "BrokerId", "type": "int32", "versions": "0+", "entityType": "brokerId",
        "about": "The ID of the requesting broker" },
      { "name": "BrokerEpoch", "type": "int64", "versions": "0+", "default": "-1",
        "about": "The epoch of the requesting broker" },
      { "name": "Topics", "type": "[]TopicData", "versions": "0+", "fields": [
        { "name":  "Name", "type": "string", "versions": "0+", "entityType": "topicName",
          "about": "The name of the topic to alter ISRs for" },
        { "name": "Partitions", "type": "[]PartitionData", "versions": "0+", "fields": [
          { "name": "PartitionIndex", "type": "int32", "versions": "0+",
            "about": "The partition index" },
          { "name": "LeaderEpoch", "type": "int32", "versions": "0+",
            "about": "The leader epoch of this partition" },
          { "name": "NewIsr", "type": "[]int32", "versions": "0+", "entityType": "brokerId",
            "about": "The ISR for this partition"},
          // ----- Start of properties added by this KIP -----
          { "name": "LeaderRecoveryState", "type": "int8", "versions": "1+", "default": "0"
            "about": "Indicates if the partition is recovering from an election." },
          // ----- End of properties added by this KIP -----
          { "name": "PartitionEpoch", "type": "int32", "versions": "0+",
            "about": "The expected epoch of the partition which is being updated"}
        ]}
      ]}
    ]
  }
```

Request Handling

The controller will validate the LeaderRecoveryState field and return an INVALID_REQUEST error when:

1. The size of the ISR is greater than 1 and the LeaderRecoveryState is RECOVERING.
2. The LeaderRecoveryState is changing from RECOVERED to RECOVERING.

If the controllers receives an AlterPartition with a version of 0, it will assume that the leader has recovered from the unclean leader election.

Response Schema

The name of the response will be changed to AlterPartitionResponse from AlterIsrResponse. The field CurrentIsrVersion will be renamed to PartitionEpoch.

Add a property to indicate the leader recovery state after processing the AlterPartition request. The broker does not take the value in this field as a trigger to start recovery. That happens through either the LeaderAndIsr request or the relevant record in Kraft mode.

```
  {
    "apiKey": 56,
    "type": "response",
    "name": "AlterPartitionResponse",
    "validVersions": "0-1",
    "flexibleVersions": "0+",
    "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": "0+",
        "about": "The top level response error code" },
      { "name": "Topics", "type": "[]TopicData", "versions": "0+", "fields": [
        { "name":  "Name", "type": "string", "versions": "0+", "entityType": "topicName",
          "about": "The name of the topic" },
        { "name": "Partitions", "type": "[]PartitionData", "versions": "0+", "fields": [
          { "name": "PartitionIndex", "type": "int32", "versions": "0+",
            "about": "The partition index" },
          { "name": "ErrorCode", "type": "int16", "versions": "0+",
            "about": "The partition level error code" },
          { "name": "LeaderId", "type": "int32", "versions": "0+", "entityType": "brokerId",
            "about": "The broker ID of the leader." },
          { "name": "LeaderEpoch", "type": "int32", "versions": "0+",
            "about": "The leader epoch." },
          { "name": "Isr", "type": "[]int32", "versions": "0+", "entityType": "brokerId",
            "about": "The in-sync replica IDs." },
          // ----- Start of properties added by this KIP -----
          { "name": "LeaderRecoveryState", "type": "int8", "versions": "1+", "default": "0",
            "about": "Indicates if the partition is recovering from an election." },
          // ----- End of properties added by this KIP -----
          { "name": "PartitionEpoch", "type": "int32", "versions": "0+",
            "about": "The current epoch of the partition." }
        ]}
      ]}
    ]
  }
```

Cluster Metadata Records

PartitionRecord

Add a property to indicate to the leader of the topic partition that it must recover the partition.

```
  {
    "apiKey": 3,
    "type": "metadata",
    "name": "PartitionRecord",
    "validVersions": "0",
    "flexibleVersions": "0+",
    "fields": [
      { "name": "PartitionId", "type": "int32", "versions": "0+", "default": "-1",
        "about": "The partition id." },
      { "name": "TopicId", "type": "uuid", "versions": "0+",
        "about": "The unique ID of this topic." },
      { "name": "Replicas", "type":  "[]int32", "versions":  "0+", "entityType": "brokerId",
        "about": "The replicas of this partition, sorted by preferred order." },
      { "name": "Isr", "type":  "[]int32", "versions":  "0+",
        "about": "The in-sync replicas of this partition" },
      { "name": "RemovingReplicas", "type":  "[]int32", "versions":  "0+", "entityType": "brokerId",
        "about": "The replicas that we are in the process of removing." },
      { "name": "AddingReplicas", "type":  "[]int32", "versions":  "0+", "entityType": "brokerId",
        "about": "The replicas that we are in the process of adding." },
      { "name": "Leader", "type": "int32", "versions": "0+", "default": "-1", "entityType": "brokerId",
        "about": "The lead replica, or -1 if there is no leader." },
      // ----- Start of properties added by this KIP -----
      { "name": "LeaderRecoveryState", "type": "int8", "default": "0", "versions": "0+", "taggedVersions":
"0+", "tag": 0,
        "about": "Indicates if the partition is recovering from an election." },
      // ----- End of properties added by this KIP -----
      { "name": "LeaderEpoch", "type": "int32", "versions": "0+", "default": "-1",
        "about": "The epoch of the partition leader." },
      { "name": "PartitionEpoch", "type": "int32", "versions": "0+", "default": "-1",
        "about": "An epoch that gets incremented each time we change anything in the partition." }
    ]
  }
```

PartitionChangeRecord

Add a property to indicate to the leader of the topic partition that it must recover the partition.

```
  {
    "apiKey": 5,
    "type": "metadata",
    "name": "PartitionChangeRecord",
    "validVersions": "0",
    "flexibleVersions": "0+",
    "fields": [
      { "name": "PartitionId", "type": "int32", "versions": "0+", "default": "-1",
        "about": "The partition id." },
      { "name": "TopicId", "type": "uuid", "versions": "0+",
        "about": "The unique ID of this topic." },
      { "name": "Isr", "type":  "[]int32", "default": "null", "entityType": "brokerId",
        "versions": "0+", "nullableVersions": "0+", "taggedVersions": "0+", "tag": 0,
        "about": "null if the ISR didn't change; the new in-sync replicas otherwise." },
      { "name": "Leader", "type": "int32", "default": "-2", "entityType": "brokerId",
        "versions": "0+", "taggedVersions": "0+", "tag": 1,
        "about": "-1 if there is now no leader; -2 if the leader didn't change; the new leader otherwise." },
      { "name": "Replicas", "type": "[]int32", "default": "null", "entityType": "brokerId",
        "versions": "0+", "nullableVersions": "0+", "taggedVersions": "0+", "tag": 2,
        "about": "null if the replicas didn't change; the new replicas otherwise." },
      { "name": "RemovingReplicas", "type": "[]int32", "default": "null", "entityType": "brokerId",
        "versions": "0+", "nullableVersions": "0+", "taggedVersions": "0+", "tag": 3,
        "about": "null if the removing replicas didn't change; the new removing replicas otherwise." },
      { "name": "AddingReplicas", "type": "[]int32", "default": "null", "entityType": "brokerId",
        "versions": "0+", "nullableVersions": "0+", "taggedVersions": "0+", "tag": 4,
        "about": "null if the adding replicas didn't change; the new adding replicas otherwise." },
      // ----- Properties added by this KIP -----
      { "name": "LeaderRecoveryState", "type": "int8", "default": "-1", "versions": "0+", "taggedVersions":
"0+", "tag": 5,
        "about": "-1 if it didn't change; 0 if the leader was elected from the ISR or recovered from an unclean
election; 1 if the leader that was elected using unclean leader election and it is still recovering." }
    ]
  }
```

Compatibility, Deprecation, and Migration Plan

This feature will be guarded by an IBP bump. At a high-level this change is backwards compatible because the default value for all of the leader recovery state field in the protocol is RECOVERED. When thinking about backward compatibility it is important to note that if the leader recovery state field is RECOVERING then the ISR is guarantee to have a size of 1. The topic partition leader will not increase the ISR until it has recovered from the unclean leader election and has set the leader recovery state field to RECOVERED.

Controller

If the ZooKeeper controller supports this KIP and it is enabled because the IBP is large enough, when performing an unclean leader election it will write RECOVERING in ZK for the leader recovery state field and it will set RECOVERING in the LeaderAndIsr request. If the broker doesn't support this feature, the LeaderRecoveryState field in the LeaderAndIsr request will be ignored and behave as it currently does. When the broker sends the AlterPartition request with a version of 0, the controller will assume that the leader has recovered from the unclean leader election.

It is possible that the topic partition leader supports this feature but its IBP hasn't been increased. If the leader receives a LeaderAndIsr request with LeaderRecoveryState field set to RECOVERING, it will perform recovery irrespective of the IBP and send an AlterPartition request that matches the current IBP.

Clients

With this KIP the requests FETCH, PRODUCE, LIST_OFFSETS, DELETE_RECORDS and OFFSET_FOR_LEADER_EPOCH will return a NOT_LEADER_OR_FOLLOWER error for any topic partition for which the leader is recovering. This is backward compatible because the clients will retry this error.

For FETCH requests, the replicas will handle this error by backing off by "replica.fetch.backoff.ms". The consumers will handle this error by queuing a full metadata request for the next metadata request interval.

For PRODUCE requests, the producers will re-queue the request if it is within the retry window.

Rejected Alternatives

Unclean Election Epoch

An alternative solution is to store the leader epoch when the unclean leader election was performed instead of storing the RECOVERED and RECOVERING values in the LeaderRecoveryState field. The topic partition would perform unclean recovery when the unclean leader epoch is equal to the current leader epoch. One issue with this solution is that the controller changes the leader epoch when the leader goes offline. This means that the controller would have to also reset the unclean leader epoch when the leader goes offline.

Recovering Election Boolean

Another alternative is to use a boolean instead of the int8 type and the states RECOVERING and RECOVERED to tracking this information. They are both conceptually similar. The boolean type is represented as an int8 in the Kafka message write protocol.