Item 225
git_comments:

git_commits:

1. **summary:** KAFKA-4443; Controller should send UpdateMetadataRequest prior to LeaderAndIsrRequest during failover
   message: KAFKA-4443; Controller should send UpdateMetadataRequest prior to LeaderAndIsrRequest during failover Author: Dong Lin <lindong28@gmail.com> Reviewers: Jiangjie Qin <becket.qin@gmail.com>, Jun Rao <junrao@gmail.com> Closes #2168 from lindong28/KAFKA-4443

github_issues:

github_issues_comments:

github_pulls:

1. **title:** KAFKA-4443; Controller should send UpdateMetadataRequest prior to LeaderAndIsrRequest during failover
   body:

github_pulls_comments:

1. LGTM. @junrao @jjkoshy Will you also be able to take a look? I took a quick look at KAFKA-901 which introduced sending UpdateMetadataRequest from controller but did not find specific reason on the order. Is there any concern on this change? Thanks.
2. @lindong28 : Thanks for the patch. LGTM. Left a comment in the jira. Perhaps you could clarify there. In KAFKA-3042, there is also discussion on combining UpdateMetadataRequest and LeaderAndIsrRequest together. I am wondering if you have thought about whether that's a better long term solution.
3. **body:** Is it possible to write a test for this (yes, it may sometimes sound like I ask the same question over and over, but it's an important question :)).
   label: test
4. **body:** @ijuma It might be a little tricky to write test in this case. The bug was caught by a race condition that the broker comes up during the controller fail over and before the controller finishing initializing its context. We can write a test but the test might also subject to race condition unless we put special logic in the actual code to guarantee the order, which seems intrusive. Do you have any suggestion on how to test such cases?
   label: test
5. **body:** @becketqin I think for cases like this one would probably write test cases using mocks that validate that things happen in a certain order. I understand that the controller doesn't make this easy and it shouldn't be a blocker if it turns out that it's too hard with the current design. But the default should be to include tests with bug fixes and if we don't, there should be an explanation on the PR.
   label: test

github_pulls_reviews:

1. Should we still send `UpdateMetadataRequest`s here? The requests sent in line 330 were just include the information stored in ZK before the leader elections. So it seems we still need to propagate the state change (if there is any) in `replicaStateMachine.startup()` and `partitionStateMachine.startup()`.
2. @becketqin I think we don't need this. When controller start replica statement and partition and partition state machine, it should send LeanderAndIsrRequest for every partition that will have a leader. `addLeaderAndIsrRequestForBrokers()` will in turn call `addUpdateMetadataRequestForBrokers` to send MetadataUpdateRequest to all live brokers for these partitions. For partition that doesn't have a leader due to NoReplicaOnlineException, MetadataUpdateRequest is not needed for this partition either.
3. Yes, you are right. Then it is fine.
4. **body:** It would probably be useful to add a comment explaining that the order matters.
   label: documentation
5. **body:** Sure. I will add the comment.

label: documentation

jira_issues:

1. **summary:** Controller should send UpdateMetadataRequest prior to LeaderAndIsrRequest during failover
   description: Currently in onControllerFailover(), controller will startup replicaStatemachine and partitionStateMachine before invoking sendUpdateMetadataRequest(controllerContext.liveOrShuttingDownBrokerIds.toSeq). However, if a broker starts right after controller election, the LeaderAndIsrRequest sent to follower partitions on this broker will all be ignored because broker doesn't know the leaders are alive. To fix this problem, in onControllerFailover(), controller should send UpdateMetadataRequest to brokers after initializeControllerContext() but before it starts replicaStatemachine and partitionStateMachine. The first MetadatUpdateRequest will include list of live broker. Although it will not include partition leader information, it is OK because we will always send MetadataUpdateRequest again when we send LeaderAndIsrRequest during replicaStateMachine.startup() and partitionStateMachine.startup().

jira_issues_comments:

1. GitHub user lindong28 opened a pull request: https://github.com/apache/kafka/pull/2168 KAFKA-4443; Controller should send UpdateMetadataRequest prior to LeaderAndIsrRequest during failover You can merge this pull request into a Git repository by running: $ git pull https://github.com/lindong28/kafka KAFKA-4443 Alternatively you can review and apply these changes as the patch at: https://github.com/apache/kafka/pull/2168.patch To close this pull request, make a commit to your master/trunk branch with (at least) the following in the commit message: This closes #2168 ---- commit 4db70c9e9960c688932dda8d942ee2fedc459e1e Author: Dong Lin <lindong28@gmail.com> Date: 2016-11-25T04:21:50Z KAFKA-4443; Controller should send UpdateMetadataRequest prior to LeaderAndIsrRequest during failover ----
2. [~lindong], thanks for filing the jira. A couple of questions. (1) I am not sure that I understand the description in the jira "However, if a broker right after controller election,". Could you clarify that? (2) Is this the same issue as reported in https://issues.apache.org/jira/browse/KAFKA-3042 and is the fix the same? If so, it would be useful to mark the other one as duplicate.
3. Github user asfgit closed the pull request at: https://github.com/apache/kafka/pull/2168
4. The fix version should be 0.10.2.0, right? The PR was only merged to trunk.
5. **body:** [~junrao] Sure. I just updated the description to correct the typo. What I mean is that, if a broker starts right after controller election, the LeaderAndIsrRequest will be ignored because the broker doesn't have the needed information (e.g. port) of live brokers. As for (2), I think this is probably the same issue reported in KAFKA-3042. All phenomena described in KAFKA-3042 can be caused by the bug fixed in this JIRA. Actually, you described exactly the same fix applied in this JIRA 7 months ago, i.e. "... to fix this particular issue, the simplest approach is to send UpdateMetadataRequest first during controller failover". As of current design of controller, I prefer the solution where controller sends MetadataUpdateRequest without LeaderAndIsrRequset. Broker will handle MedataDataUpdateRequest in the following steps: 1) update cache with live broker info extracted from MetadataUpdateRequest, 2) reconstruct LeaderAndIsrRequest from MetadataUpdateRequest and process it, and 3) update cache with partition information extracted from MetadataUpdateRequest. This solution is simple and doesn't require wire protocol change. And it is strictly better than current implementation because we no longer have to send MetadataUpdateRequest before LeaderAndIsrRequest. But I am not 100% sure this is long term solution because it relies on existing implementation detail where controller always send MetadataUpdateRequest after LeaderAndIsrRequest. In theory this may not be the case if controller is re-designed. For example, we may want to send MetadataUpdateRequest only after Controller has received LeaderAndIsrResponse with success. The idea is to expose new external state to user only after internal state change is completed. If we don't adopt the solution above which uses MetadataUpateRequest as combination of LeaderAndIsrRequest + MetadataUpdateRequest, then I think we should include endpoints of all leaders in the LeaderAndIsrRequest so that LeaderAndIsrRequest can provide enough information on its own to switch broker between leader and follower.
   label: documentation
6. GitHub user ijuma opened a pull request: https://github.com/apache/kafka/pull/2194 KAFKA-4443: Minor comment clean-up Removed stale comment left behind, minor fixes (UpdateMetadataRequest instead of MetadataUpdateRequest) and remove redundant comments. You can merge this pull request into a Git repository by running: $ git pull https://github.com/ijuma/kafka kafka-4443-minor-follow-up

Alternatively you can review and apply these changes as the patch at: https://github.com/apache/kafka/pull/2194.patch To close this pull request, make a commit to your master/trunk branch with (at least) the following in the commit message: This closes #2194 ---- commit 4f299c7280fe52fe1ec4bc690f61038f05c5de04 Author: Ismael Juma <ismael@juma.me.uk> Date: 2016-11-30T11:12:13Z KAFKA-4443: Minor comment clean-up Removed stale comment left behind, minor fixes (UpdateMetadataRequest instead of MetadataUpdateRequest) and remove redundant comments. ----

7. Hello, Could this fix be back-ported to Kafka 0.9.0.2 please, or better, as a patch for 0.9.0.1 ? We had repeated occurrences in the past weeks with Kafka 0.9.0.1 Best regards, Alex
8. Github user asfgit closed the pull request at: https://github.com/apache/kafka/pull/2194