Item 102
git_comments:

1. *
2. * Licensed to Metamarkets Group Inc. (Metamarkets) under one * or more contributor license agreements. See the NOTICE file * distributed with this work for additional information * regarding copyright ownership. Metamarkets licenses this file * to you under the Apache License, Version 2.0 (the * "License"); you may not use this file except in compliance * with the License. You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, * software distributed under the License is distributed on an * "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY * KIND, either express or implied. See the License for the * specific language governing permissions and limitations * under the License.
3. *
4. * Licensed to Metamarkets Group Inc. (Metamarkets) under one * or more contributor license agreements. See the NOTICE file * distributed with this work for additional information * regarding copyright ownership. Metamarkets licenses this file * to you under the Apache License, Version 2.0 (the * "License"); you may not use this file except in compliance * with the License. You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, * software distributed under the License is distributed on an * "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY * KIND, either express or implied. See the License for the * specific language governing permissions and limitations * under the License.

git_commits:

1. **summary:** druid.coordinator.asOverlord.enabled flag at coordinator to make it an overlord too (#3711)
   message: druid.coordinator.asOverlord.enabled flag at coordinator to make it an overlord too (#3711)

github_issues:

1. **title:** [Proposal] eventual removal of separate overlord node
   body: It has been a desire to merge coordinator and overlord nodes into a single node for simplifying druid cluster deployments. it has been discussed multiple times and agreed upon. it can be achieved in following stages... 1) Add a boolean flag(say, "beOverlordToo", true by default) on coordinator that makes the coordinator process also behave as an overlord process. In this stage, it will look like that there is an overlord running on the same host:port as that of coordinator. Everything else would remain same. At this point, we deprecate the standalone overlord node. 2) Consolidate the code for "similar" things done by coordinator and overlord e.g. just do one leader election instead of two, and maybe some consolidation on background periodical activities.
2. **title:** [Proposal] eventual removal of separate overlord node
   body: It has been a desire to merge coordinator and overlord nodes into a single node for simplifying druid cluster deployments. it has been discussed multiple times and agreed upon. it can be achieved in following stages... 1) Add a boolean flag(say, "beOverlordToo", true by default) on coordinator that makes the coordinator process also behave as an overlord process. In this stage, it will look like that there is an overlord running on the same host:port as that of coordinator. Everything else would remain same. At this point, we deprecate the standalone overlord node. 2) Consolidate the code for "similar" things done by coordinator and overlord e.g. just do one leader election instead of two, and maybe some consolidation on background periodical activities.

github_issues_comments:

1. This issue has been marked as stale due to 280 days of inactivity. It will be closed in 2 weeks if no further activity occurs. If this issue is still relevant, please simply write any comment. Even if closed, you can still revive the issue at any time or discuss it on the dev@druid.apache.org list. Thank you for your contributions.
2. This issue has been closed due to lack of activity. If you think that is incorrect, or the issue requires additional review, you can revive the issue at any time.

github_pulls:

1. **title:** enable coordinator to act as overlord as well
   body: depends on #3700 [MERGED] enables a configuration at coordinator that lets it act as an overlord as well. see doc update to `coordinator.md` for more information. related to #3696

github_pulls_comments:

1. **body:** @himanshug can we update the tutorials? Also all the docs that mention overlord Probably need to update architecture diagrams as well
   label: documentation
2. @fjy i think we should update docs to remove overlord concept sometime after this patch is released in community and its proven in production that mixing coordinator+overlord together in a process presents no issues. Also, around that time we would do stage-2 described in https://github.com/druid-io/druid/issues/3696 as well.
3. @fjy also regarding changes to "overlord" vs "ingestion" etc , let us conclude something in https://groups.google.com/forum/?utm_medium=email&utm_source=footer#!msg/druid-development/6Lz7CGgkgBI/0oBcZHBxDwAJ
4. **body:** I'm 👍 on this as this. There's additional documentation work and whatnot that can be done, but it can wait until some discussion conclude.
   label: documentation
5. :+1:
6. If a user deploys a coordinator with this patch, would this break existing deployments (i.e., could the new hybrid coordinator/overlord take leadership from the existing overlord, with the user having configured their environment to point to only that old overlord?) If so, let's mark this incompatible or set the `beOverlord` to false by default for now
7. **body:** This looks a little messy to me but I cannot think of any other way apart from creating an entirely new node type so 👍
   label: code-design
8. @jon-wei that should not be a problem , it should still work as this makes coordinator process equivalent of a (coordinator+overlord) process. originally, I had `beOverlord ` set to false by default but it was decided to make it true by default to encourage customers to use it. In any case, let us hold on from merging this till next dev sync and we can rethink the decision on that.
9. Since it's a major change I think it's best to make it false by default in the code (for better compat with existing installs). But we can certainly change the quickstart configs to include beOverlord=true. I find that generally people start from the quickstart configs and copy them to start a prod config, so that'll have the effect of encouraging people to use it.
10. @jon-wei @gianm @fjy @cheddar it was discussed in the dev sync today and conclusion was to keep `beOverlord:true` by default
11. @gianm by integration testing you mean... use this config while running the existing integration tests, that will automatically happen on our builds given that it is enabled by default . on our builds we'll have both (coordinator+overlord) and overlord running when this PR is merged. or did you mean a separate integration test.. but what would that do and will it test anything extra?
12. @himanshug by integration testing, I mean having two test configs: - one with separate coordinator and overlord - one that _only_ has a coordinator to make sure it's actually able to perform these functions With the current test config (coordinator+overlord) it's possible the coordinator is unable to properly be an overlord, but we don't know because the standalone overlord is taking over.
13. **body:** @himanshug now that I started think about doc updates, I think we'll need some to avoid confusing users if we are going to actively encourage this setting to be on, either by making it default in code, or by setting it true in the quickstart sample configs. That's because if the config is on by default, or true in the sample configs, then new users will not really come into the project with a concept of an "overlord node" (they'll think the coordinator does everything) and so many existing docs are going to be confusing. Most of this is not necessary if we're doing this "quietly" (not on by default or in sample configs) since the risk of user confusion with a quiet addition of the feature is low. Some doc places that should get an update if we have beOverlord on by default, or true in sample configs: - tutorial/cluster.md: maybe we should change the recommendation from "coordination server to host the Coordinator and Overlord processes" to just having a coordinator acting as overlord. If we make this change then the open ports list needs adjustment as well. - configuration/indexing-service.md (maybe add an infobox that if beOverlord is true, all the refs to "overlord" here should apply to coordinator instead) - design/indexing-service.md should mention that overlord functionality can be handled by the Coordinator node as well - batch-ingestion.md, tasks.md reference "posted to a running instance of a Druid overlord" and "Once an overlord node accepts

a task", those should mention coordinators acting as overlords are fine too - tutorial/quickstart.md: references to running overlord and to "overlord console" - tutorial/tutorial-batch.md: references to "POST to the Druid Overlord" and "overlord console" - Various docs: Assumption that overlord functionality is on port 8090
label: documentation
14. **body:** @himanshug can you also please write a blurb for the release notes about how to do a rolling update? We generally include those new features that need "special" migration procedures.
label: documentation
15. @gianm I have changed it slightly to allow for full rolling upgrade. Now if coordinator is configured to acts as overlord then it insists on another configuration overlordService to be provided which it uses to publish as overlord. that will take care of rolling upgrades. see `coordinator.md` updates. i have also made coordinator to not be overlord by default in this PR, so nothing changes for the users by this PR being merged. That makes this PR merge-able without risks and don't want this to block 0.10.0 release. We will use that configuration internally on some of the clusters to gain more confidence on it. I will do a separate PR to change quickstart guide and for integration testing configs which wouldn't necessarily be in 0.10.0 milestone.
16. @pjain1 @fjy @jon-wei @nishantmonu51 anyone interested to take another look after the most recent changes?

github_pulls_reviews:

1. default true?
2. can we call this coordinateIngestion
3. handleIngestion?
4. standalone?
5. **body:** have u tested this? are u sure both consoles work?
label: test
6. yes, i tested this. with this flag, coordinator console - http://host:port/ overlord console - http://host:port/console.html
7. this seems related to discussion in https://groups.google.com/d/topic/druid-development/6Lz7CGgkgBI/discussion, so we should resolve that thread first.
8. made true by default
9. changed
10. can you also verify that http://coordinator-host:port/#/indexing-service works ?
11. **body:** Would prefer a tighter check here. As-is, setting to "no" or "FALSE" or "false " with some whitespace would make the coordinator be an overlord. How about throwing an exception if the property is set, but is not "true" or "false"? Or, at the very least, log a warning.
label: code-design
12. The need to change `druid.selectors.indexing.serviceName` means migration can't be done totally smoothly: - If you set beOverlord on coordinators first, one of them might grab leadership and now other servers can't find the overlord service (since they would look for the old serviceName and the new leader won't announce on that one). - If you update druid.selectors.indexing.serviceName on other servers first, then they won't be able to find the current leader if it's still an overlord. This could be fixed by letting the coordinator announce the overlord service name for a while too, just so stuff can find it. Then the order for migration would be: first update coordinators to beOverlord and announce both services, then update other Druid servers to set druid.selectors.indexing.serviceName equal to the coordinator name, then (optionally) update coordinators to stop announcing the old overlord service name.
13. that sounds good to enable rolling deploy, will explore how to enable that. we probably will have to create another ServiceAnnouncer at coordinator using druid.selectors.indexing.serviceName property . however, that would mean users would need to correctly specify `druid.selectors.indexing.serviceName` at coordinators also which wasn't necessary till now.
14. considering, user needs to take explicit step to make things work, I am thinking of actually keeping `beOverlord` false by default and rather test it on our clusters first. @gianm @fjy what do you think?
15. I'm happy with beOverlord being false by default. At the same time we can set it to true in the quickstart configs and remove overlord from the quickstart. That gets us a lot of the simplicity benefit for new users without making life difficult for already existing deployments.
16. **body:** Btw, that makes me think: could you please update indexing-service.md docs too? They say a lot of "this property must be set on overlord" etc etc. If beOverlord is true and they're combined, the

properties actually need to be set on the coordinator. Some language clarifying that would be helpful since without it, there is potential for users to be even more confused than they are now!
label: documentation

17. yes it works

jira_issues:

jira_issues_comments: