Item 94
git_comments:

git_commits:

1. **summary:** [SPARK-27812][K8S][2.4] Bump K8S client version to 4.6.1
   message: [SPARK-27812][K8S][2.4] Bump K8S client version to 4.6.1 # What changes were proposed in this pull request? Backport of #26093 to `branch-2.4` ### Why are the changes needed? https://issues.apache.org/jira/browse/SPARK-27812 https://issues.apache.org/jira/browse/SPARK-27927 We need this fix https://github.com/fabric8io/kubernetes-client/pull/1768 that was released on version 4.6 of the client. The root cause of the problem is better explained in https://github.com/apache/spark/pull/25785 ### Does this PR introduce any user-facing change? No ### How was this patch tested? This patch was tested manually using a simple pyspark job ```python from pyspark.sql import SparkSession if __name__ == '__main__': spark = SparkSession.builder.getOrCreate() ``` The expected behaviour of this "job" is that both python's and jvm's process exit automatically after the main runs. This is the case for spark versions <= 2.4. On version 2.4.3, the jvm process hangs because there's a non daemon thread running ``` "OkHttp WebSocket https://10.96.0.1/..." #121 prio=5 os_prio=0 tid=0x00007fb27c005800 nid=0x24b waiting on condition [0x00007fb300847000] "OkHttp WebSocket https://10.96.0.1/..." #117 prio=5 os_prio=0 tid=0x00007fb28c004000 nid=0x247 waiting on condition [0x00007fb300e4b000] ``` This is caused by a bug on `kubernetes-client` library, which is fixed on the version that we are upgrading to. When the mentioned job is run with this patch applied, the behaviour from spark <= 2.4.0 is restored and both processes terminate successfully Closes #26152 from igorcalabria/k8s-client-update-2.4. Authored-by: igor.calabria <igor.calabria@ubee.in> Signed-off-by: Dongjoon Hyun <dhyun@apple.com>

github_issues:

github_issues_comments:

github_pulls:

1. **title:** [SPARK-27812][K8S] Bump K8S client version to 4.6.1
   body: ### What changes were proposed in this pull request? Updated kubernetes client. ### Why are the changes needed? https://issues.apache.org/jira/browse/SPARK-27812 https://issues.apache.org/jira/browse/SPARK-27927 We need this fix https://github.com/fabric8io/kubernetes-client/pull/1768 that was released on version 4.6 of the client. The root cause of the problem is better explained in https://github.com/apache/spark/pull/25785 ### Does this PR introduce any user-facing change? Nope, it should be transparent to users ### How was this patch tested? This patch was tested manually using a simple pyspark job ```python from pyspark.sql import SparkSession if __name__ == '__main__': spark = SparkSession.builder.getOrCreate() ``` The expected behaviour of this "job" is that both python's and jvm's process exit automatically after the main runs. This is the case for spark versions <= 2.4. On version 2.4.3, the jvm process hangs because there's a non daemon thread running ``` "OkHttp WebSocket https://10.96.0.1/..." #121 prio=5 os_prio=0 tid=0x00007fb27c005800 nid=0x24b waiting on condition [0x00007fb300847000] "OkHttp WebSocket https://10.96.0.1/..." #117 prio=5 os_prio=0 tid=0x00007fb28c004000 nid=0x247 waiting on condition [0x00007fb300e4b000] ``` This is caused by a bug on `kubernetes-client` library, which is fixed on the version that we are upgrading to. When the mentioned job is run with this patch applied, the behaviour from spark <= 2.4.3 is restored and both processes terminate successfully

github_pulls_comments:

1. @igorcalabria thank you for writing this patch I have ran this internally on my cluster and can confirm that non-daemon threads no longer block the JVM from exiting. Against branch 2.4 and master: ====================== I see the following non-daemon threads exist when running spark-submit against the following pyspark job: ``` from pyspark.sql import SparkSession if __name__ == '__main__': spark = SparkSession.builder.getOrCreate() ``` After the python process exited, the JVM process remained and I see the following using `jstack`: ``` "OkHttp WebSocket https://10.96.0.1/..." #121 prio=5 os_prio=0 tid=0x00007fb27c005800 nid=0x24b waiting on condition [0x00007fb300847000] "OkHttp WebSocket https://10.96.0.1/..." #117 prio=5 os_prio=0 tid=0x00007fb28c004000 nid=0x247 waiting on

condition [0x00007fb300e4b000] ``` Against this patch: ============= I no longer see the following non-daemon threads and the JVM is able to exit after the completion of the python process. @mccheah @erikerlandson @felixcheung @holdenk we should backport this fix to branch 2.4 as well and cut a new release with this patch.

2. ok to test
3. @igorcalabria . The following claims is not matched with your PR description. > Standard kubernetes tests should validate this patch. Please describe how to test this for the following for the reviewer. (Technically, this PR need a test case for that. Or, at least, manual testing process should be described in the PR description and should be the commit log.) > https://issues.apache.org/jira/browse/SPARK-27812 > https://issues.apache.org/jira/browse/SPARK-27927
4. **[Test build #111979 has finished] (https://amplab.cs.berkeley.edu/jenkins/job/SparkPullRequestBuilder/111979/testReport)** for PR 26093 at commit [`7f15bac`] (https://github.com/apache/spark/commit/7f15bacbb9b8ac4a448e3f0fca8213632c07c52d). * This patch passes all tests. * This patch merges cleanly. * This patch adds no public classes.
5. > @igorcalabria . The following claims is not matched with your PR description. > > > Standard kubernetes tests should validate this patch. > > Please describe how to test this for the following for the reviewer. (Technically, this PR need a test case for that. Or, at least, manual testing process should be described in the PR description and should be the commit log.) > > > https://issues.apache.org/jira/browse/SPARK-27812 > > https://issues.apache.org/jira/browse/SPARK-27927 @dongjoon-hyun A manual test is described above by @ifilonenko. If you want, I could add an integration test for this specific case. I skipped it because other PRs updating kubernetes-client version didn't add any tests and the README in `kubernetes/integration-tests` states that the framework is under review and subject to major changes
6. **[Test build #112032 has finished] (https://amplab.cs.berkeley.edu/jenkins/job/SparkPullRequestBuilder/112032/testReport)** for PR 26093 at commit [`5d6bcd7`] (https://github.com/apache/spark/commit/5d6bcd790b2d37fcce45852dbd03b4e5959bd298). * This patch **fails build dependency tests**. * This patch merges cleanly. * This patch adds no public classes.
7. If it's easy to write a test for this, OK. If it's complex or would take a long time to run, I can see just verifying the small behavior change with a manual test.
8. Kubernetes integration test starting URL: https://amplab.cs.berkeley.edu/jenkins/job/SparkPullRequestBuilder-K8s/17039/
9. Kubernetes integration test status success URL: https://amplab.cs.berkeley.edu/jenkins/job/SparkPullRequestBuilder-K8s/17039/
10. It's pretty difficult to cover all the code paths for changes in a downstream library. @dongjoon-hyun @srowen is there a standard protocol requiring an additional test for downstream dependencies? In this case, we're picking up a fix to an internal implementation detail in the downstream library, and I'd imagine we'd trust the downstream library to have already done its due diligence to test their fix. Regression tests exist to catch new problems introduced by the dependency bumps as we go along. I'd agree that a manual test demonstrating the desired end behavior is sufficient here.
11. Kubernetes integration test starting URL: https://amplab.cs.berkeley.edu/jenkins/job/SparkPullRequestBuilder-K8s/17042/
12. Kubernetes integration test status success URL: https://amplab.cs.berkeley.edu/jenkins/job/SparkPullRequestBuilder-K8s/17042/
13. Hi, @igorcalabria and @mccheah . I'm okay as long as the manual description becomes the PR description and the commit log. > Please describe how to test this for the following for the reviewer. (Technically, this PR need a test case for that. Or, at least, manual testing process should be described in the PR description and should be the commit log.)
14. **[Test build #112035 has finished] (https://amplab.cs.berkeley.edu/jenkins/job/SparkPullRequestBuilder/112035/testReport)** for PR 26093 at commit [`b31b375`] (https://github.com/apache/spark/commit/b31b375266b0a4d9d717f95aa1af8d4a02b05015). * This patch passes all tests. * This patch merges cleanly. * This patch adds no public classes.
15. +1 on my end, I have tested using manual tests. An integration test seems like a bit of overkill, although a simple jstack check in the test would be sufficient.
16. > @mccheah @erikerlandson @felixcheung @holdenk we should backport this fix to branch 2.4 as well and cut a new release with this patch. Will there be a 2.4.5 ?

17. @erikerlandson I think there should be. This bug will cause pyspark resources to hang unless some external process does resource reaping. @dongjoon-hyun I agree, the PR description should change
18. There will be a 2.4.5 eventually, I'm sure. 2.4.x is a sort of "LTS" branch.
19. Of course, @erikerlandson . As @srowen said, it's the `LTS` branch. Since `2.4.0` was released on `November, 2018`, `branch-2.4` should be maintained at least `May, 2020` (18 months). After that, as a `LTS` branch, I believe `branch-2.4` will be maintained for a while. Since `2.4.4` was released on September, 2019. `2.4.5` will arrive early 2020 and so on. I hope more regular releases. But, the `2.4.x` release schedule heavily depends on the patches on `branche-2.4` and the volunteer for release manager.
20. @dongjoon-hyun I have updated the description to include a manual test. I believe that both jira issues fix claims are now supported by the manual test case provided in the description. Please let me know if you need anything else. Cheers,
21. Thank you for updating, @igorcalabria ! I'll follow the verification steps and try to land this.
22. Also, cc @jiangxb1987 since he is a release manager for `3.0.0-preview`.
23. I'm +1 on backporting even though it's a version change I think we need this in 2.4.X
24. Actually if were going to update to 4.6.1 (and it looks like might as well) @ifilonenko would you have the cycles to manually verify with that version as well?
25. @dongjoon-hyun bumped version to 4.6.1. Following other reviewers suggestions, I also removed explicit okhttp dep.
26. Kubernetes integration test starting URL: https://amplab.cs.berkeley.edu/jenkins/job/SparkPullRequestBuilder-K8s/17209/
27. Kubernetes integration test status success URL: https://amplab.cs.berkeley.edu/jenkins/job/SparkPullRequestBuilder-K8s/17209/
28. **[Test build #112221 has finished] (https://amplab.cs.berkeley.edu/jenkins/job/SparkPullRequestBuilder/112221/testReport)** for PR 26093 at commit [`a5eb702`] (https://github.com/apache/spark/commit/a5eb702a6f41f6e76934d1ec5eee3f7a0374690b). * This patch passes all tests. * This patch merges cleanly. * This patch adds no public classes.
29. Is this tested with `Python 3` on the master branch?
30. Thank you so much everyone. Especially, welcome, @igorcalabria . I added you to the Apache Spark contributor group. Could you make a backporting PR against branch-2.4, @igorcalabria ?

github_pulls_reviews:

1. You need to change the `kubernetes.client.version` in `resource-managers/kubernetes/integration-tests/pom.xml` as well. and bump: ``` <dependency> <groupId>com.squareup.okhttp3</groupId> <artifactId>okhttp</artifactId> - <version>3.8.1</version> + <version>3.12.0</version> </dependency> ``` as I believe there is a dependency here https://mvnrepository.com/artifact/io.fabric8/kubernetes-client/4.6.0
2. Ok. Is there a reason why okhttp is added as an explicit dependency(kubernetes-client already brings it)? If we really need a specific version, shouldn't we at least exclude it from `kubernetes-client`? I'm saying this because it's very easy to end up with a unexpected version on the classpath. `kubernetes-client` uses okhttp's version 3.12 since version 4.1.2 but we're only updating it now.
3. It may have been to make sure that another dependency on okhttp didn't win out and override to a lower version. It can just be updated here, or you can try removing it and examining the transitive dependencies afterwards. If it's already at 3.12.0, then it can likely be omitted.
4. That is a good point, @mccheah do we have an opinion on this? I am okay with removing the explicit dependency since `kubernetes-client` already brings it in. If there was a reason, I can't seem to remember, to explicitly state, let's exclude it?
5. This one looks okay to me.
6. agreed we should remove explicit dep to avoid a mismatch failure in the future

jira_issues:

jira_issues_comments: