HADOOP-17258

dongjoon

2020-09-11

Contents

Chapter 1

Root issue HADOOP-17258

1.1 Summary

MagicS3GuardCommitter fails with ‘pendingset‘ already exists

1.2 Description

In ‘trunk/branch-3.3/branch-3.2‘, ‘MagicS3GuardCommitter.innerCommitTask‘ has ‘false‘ at ‘pendingSet.save‘.

```
    try {
      pendingSet.save(getDestFS(), taskOutcomePath, false);
    } catch (IOException e) {
      LOG.warn("Failed to save task commit data to {} ",
          taskOutcomePath, e);
      abortPendingUploads(context, pendingSet.getCommits(), true);
      throw e;
    }
```

And, it can cause a job failure like the following.

```
WARN TaskSetManager: Lost task 1562.1 in stage 1.0 (TID 1788, 100.92.11.63, executor 26): org.apache.spark.SparkException: Task failed while writing rows.
    at org.apache.spark.sql.execution.datasources.FileFormatWriter$.org$apache$spark$sql$execution$datasources$FileFormatWriter$$executeTask(FileFormatWriter.scala:257)
    at org.apache.spark.sql.execution.datasources.FileFor
matWriter$$anonfun$write$1.apply(FileFormatWriter.scala:170)
    at org.apache.spark.sql.execution.datasources.FileFormatWriter$$anonfun$write$1.apply(FileFormatWriter.scala:169)
    at org.apache.spark.scheduler.ResultTask.runTask(ResultTask.scala:90)
    at org.apache.spark.scheduler.Task.run(Task.scala:123)
    at org.apache.spark.executor.Executor$TaskRunner$$anonfun$10.apply(Executor.scala:40
8)
    at org.apache.spark.util.Utils$.tryWithSafeFinally(Utils.scala:1360)
    at org.apache.spark.executor.Executor$TaskRunner.run(Executor.scala:414)
    at java.base/java.util.concurrent.ThreadPoolExecutor.runWorker(Unknown Source)
    at java.base/java.util.concurrent.ThreadPoolExecutor$Worker.run(Unknown Source)
    at java.base/java.lang.Thread.run(Unknown Source)
Caused by: org.apache.hado
op.fs.FileAlreadyExistsException: s3a://xxx/__magic/app-attempt-0000/task_20200911063607_0001_m_001562.pendingset already exists
    at org.apache.hadoop.fs.s3a.S3AFileSystem.create(S3AFileSystem.java:761)
    at org.apache.hadoop.fs.FileSystem.create(FileSystem.java:1118)
    at org.apache.hadoop.fs.FileSystem.create(FileSystem.java:1098)
    at org.apache.hadoop.fs.FileSystem.create(FileSystem.j
ava:987)
    at org.apache.hadoop.util.JsonSerialization.save(JsonSerialization.java:269)
    at org.apache.hadoop.fs.s3a.commit.files.PendingSet.save(PendingSet.java:170)
    at org.apache.hadoop.fs.s3a.commit.magic.MagicS3GuardCommitter.innerCommitTask(
        MagicS3GuardCommitter.java:220)
    at org.apache.hadoop.fs.s3a.commit.magic.MagicS3GuardCommitter.commitTask(
        MagicS3GuardCommitter.java:165)

 at org.apache.spark.mapred.SparkHadoopMapRedUtil$.performCommit$1(SparkHadoopMapRedUtil
     .scala:50)
    at org.apache.spark.mapred.SparkHadoopMapRedUtil$.commitTask(SparkHadoopMapRedUtil.
        scala:77)
    at org.apache.spark.internal.io.HadoopMapReduceCommitProtocol.commitTask(
        HadoopMapReduceCommitProtocol.scala:244)
    at org.apache.spark.sql.execution.datasources.FileFormatDataWriter.commit(FileForm
atDataWriter.scala:78)
    at org.apache.spark.sql.execution.datasources.FileFormatWriter$$anonfun$org$apache$
        spark$sql$execution$datasources$FileFormatWriter$$executeTask$3.apply(
        FileFormatWriter.scala:247)
    at org.apache.spark.sql.execution.datasources.FileFormatWriter$$anonfun$org$apache$
        spark$sql$execution$datasources$FileFormatWriter$$executeTask$3.apply(
        FileFormatWriter.scala:242)
\end{ls
tlisting} \ \newline%
\newline%
\begin{lstlisting}
20/09/11 07:44:38 ERROR TaskSetManager: Task 957.1 in stage 1.0 (TID 1412) can not write
     to output file: org.apache.hadoop.fs.FileAlreadyExistsException: s3a://xxx/t/
    __magic/app-attempt-0000/task_20200911073922_0001_m_000957.pendingset already exists
    ; not retrying
```

The above happens in EKS with S3 environment and the job failure happens when some executor containers are killed by K8s

1.3 Attachments

No attachments

1.4 Comments

1. **dongjoon:** This happens frequently in a situation where some Spark executors die.

2. **stevel@apache.org:**

   That failure implies that attempt 1 succeeded & created the file, so TA 2 couldn't. Maybe the thing to do here is on that second write, say "the file is there, so let's gracefully abort our commit, relying on attempt #1 to be correct. Then it should abort its uploads and return success

   Makes sense?

What happens to the whole job? Does it fail?

(FWIW, failures during task commit is something the s3a committers MUST be resilient to. Spark expects commits to be atomic and exactly one attempt to succeed, but it doesn't care which (more specifically: your code mustn't care)

3. **dongjoon:** Thank you for commenting, [~stevel@apache.org]. Yes, it makes sense. Currently, this causes the job failure at the almost end of execution. I used DynamoDB and S3Guard and S3Magic committer and have been monitoring the target directory.
4. **dongjoon:** Also, cc [~chaosun]
5. **stevel@apache.org:** OK. So we're failing in createFile right now. What do you think should be done?

bq. This causes the job failure at the almost end of execution.

well, technically it's always the end of the execution :)

Looking at the stack trace, here's the code:

```

    try {

      pendingSet.save(getDestFS(), taskOutcomePath, false);

    } catch (IOException e) {

      LOG.warn("Failed to save task commit data to {} ",

          taskOutcomePath, e);

      abortPendingUploads(context, pendingSet.getCommits(), true);

      throw e;

    }
```

change:

```



} catch (FileAlreadyExistsException e) {

  log.warn("Another attempt has already succeeded")

      abortPendingUploads(context, pendingSet.getCommits(), true);

      return true;

} catch (IOException e) { /* as before */ }
```

I'd wondered if we could be clever, load the previous pendingset and abort those, but that would be a disaster if the second attempt crashed during that rollback.

But by overwriting the file, the pending commits from it will be lost; we will rely on the final list and abort of all uploads to clean up and so avoid bills for incomplete uploads. People should always have their buckets set up to delete pending uploads after a few days anyway.

6. **stevel@apache.org:** Fancy submitting a patch? Easier to get reviews if you code/test and I review/commit

7. **dongjoon:** The return value is ‘PendingSet‘ instead of ‘boolean‘, isn't it?

   - https://github.com/apache/hadoop/blob/trunk/hadoop-tools/hadoop-aws/src/main/java/org/apache/hadoop/fs

```

private PendingSet innerCommitTask(...)
```

8. **stevel@apache.org:** merged to branch 3.3 & trunk -thanks!