Item 142
git_comments:

1. Copy blob operation in Azure storage is very costly. It will be highly likely throttled during Azure storage gc. Short term fix will be using a more intensive exponential retry policy when the cluster is getting throttled.
2. Re-throw exception as an Azure storage exception.

git_commits:

1. **summary:** HADOOP-11693. Azure Storage FileSystem rename operations are throttled too aggressively to complete HBase WAL archiving. Contributed by Duo Xu.
   message: HADOOP-11693. Azure Storage FileSystem rename operations are throttled too aggressively to complete HBase WAL archiving. Contributed by Duo Xu. (cherry picked from commit 7a346bcb4fa5b56191ed00a39e72e51c9bdf1b56)

github_issues:

github_issues_comments:

github_pulls:

github_pulls_comments:

github_pulls_reviews:

jira_issues:

1. **summary:** Azure Storage FileSystem rename operations are throttled too aggressively to complete HBase WAL archiving.
   description: One of our customers' production HBase clusters was periodically throttled by Azure storage, when HBase was archiving old WALs. HMaster aborted the region server and tried to restart it. However, since the cluster was still being throttled by Azure storage, the upcoming distributed log splitting also failed. Sometimes hbase:meta table was on this region server and finally showed offline, which cause the whole cluster in bad state. {code} 2015-03-01 18:36:45,623 ERROR org.apache.hadoop.hbase.master.HMaster: Region server workernode4.hbaseproddb4001.f5.internal.cloudapp.net,60020,1424845421044 reported a fatal error: ABORTING region server workernode4.hbaseproddb4001.f5.internal.cloudapp.net,60020,1424845421044: IOE in log roller Cause: org.apache.hadoop.fs.azure.AzureException: com.microsoft.windowsazure.storage.StorageException: The server is busy. at org.apache.hadoop.fs.azurenative.AzureNativeFileSystemStore.rename(AzureNativeFileSystemStore.java:2446) at org.apache.hadoop.fs.azurenative.AzureNativeFileSystemStore.rename(AzureNativeFileSystemStore.java:2367) at org.apache.hadoop.fs.azurenative.NativeAzureFileSystem.rename(NativeAzureFileSystem.java:1960) at org.apache.hadoop.hbase.util.FSUtils.renameAndSetModifyTime(FSUtils.java:1719) at org.apache.hadoop.hbase.regionserver.wal.FSHLog.archiveLogFile(FSHLog.java:798) at org.apache.hadoop.hbase.regionserver.wal.FSHLog.cleanOldLogs(FSHLog.java:656) at org.apache.hadoop.hbase.regionserver.wal.FSHLog.rollWriter(FSHLog.java:593) at org.apache.hadoop.hbase.regionserver.LogRoller.run(LogRoller.java:97) at java.lang.Thread.run(Thread.java:745) Caused by: com.microsoft.windowsazure.storage.StorageException: The server is busy. at com.microsoft.windowsazure.storage.StorageException.translateException(StorageException.java:163) at com.microsoft.windowsazure.storage.core.StorageRequest.materializeException(StorageRequest.java:306) at com.microsoft.windowsazure.storage.core.ExecutionEngine.executeWithRetry(ExecutionEngine.java:229) at com.microsoft.windowsazure.storage.blob.CloudBlob.startCopyFromBlob(CloudBlob.java:762) at org.apache.hadoop.fs.azurenative.StorageInterfaceImpl$CloudBlobWrapperImpl.startCopyFromBlob(StorageInterfaceImpl.java:350) at org.apache.hadoop.fs.azurenative.AzureNativeFileSystemStore.rename(AzureNativeFileSystemStore.java:2439) ... 8 more 2015-03-01 18:43:29,072 ERROR org.apache.hadoop.hbase.executor.EventHandler: Caught throwable while processing event M_META_SERVER_SHUTDOWN java.io.IOException: failed log splitting for workernode13.hbaseproddb4001.f5.internal.cloudapp.net,60020,1424845307901, will retry at org.apache.hadoop.hbase.master.handler.MetaServerShutdownHandler.process(MetaServerShutdownHandler.java:71) at org.apache.hadoop.hbase.executor.EventHandler.run(EventHandler.java:128) at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1145) at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:615) at java.lang.Thread.run(Thread.java:745) Caused by: org.apache.hadoop.fs.azure.AzureException: com.microsoft.windowsazure.storage.StorageException: The server is busy. at org.apache.hadoop.fs.azurenative.AzureNativeFileSystemStore.rename(AzureNativeFileSystemStore.java:2446) at org.apache.hadoop.fs.azurenative.NativeAzureFileSystem$FolderRenamePending.execute(NativeAzureFileSystem.java:393) at org.apache.hadoop.fs.azurenative.NativeAzureFileSystem.rename(NativeAzureFileSystem.java:1973) at org.apache.hadoop.hbase.master.MasterFileSystem.getLogDirs(MasterFileSystem.java:319) at org.apache.hadoop.hbase.master.MasterFileSystem.splitLog(MasterFileSystem.java:406) at org.apache.hadoop.hbase.master.MasterFileSystem.splitMetaLog(MasterFileSystem.java:302) at org.apache.hadoop.hbase.master.MasterFileSystem.splitMetaLog(MasterFileSystem.java:293) at org.apache.hadoop.hbase.master.handler.MetaServerShutdownHandler.process(MetaServerShutdownHandler.java:64) ... 4 more Caused by: com.microsoft.windowsazure.storage.StorageException: The server is busy. at com.microsoft.windowsazure.storage.StorageException.translateException(StorageException.java:163) at com.microsoft.windowsazure.storage.core.StorageRequest.materializeException(StorageRequest.java:306) at com.microsoft.windowsazure.storage.core.ExecutionEngine.executeWithRetry(ExecutionEngine.java:229) at com.microsoft.windowsazure.storage.blob.CloudBlob.startCopyFromBlob(CloudBlob.java:762) at org.apache.hadoop.fs.azurenative.StorageInterfaceImpl$CloudBlobWrapperImpl.startCopyFromBlob(StorageInterfaceImpl.java:350) at org.apache.hadoop.fs.azurenative.AzureNativeFileSystemStore.rename(AzureNativeFileSystemStore.java:2439) ... 11 more Sun

Mar 01 18:59:51 GMT 2015, org.apache.hadoop.hbase.client.RpcRetryingCaller@aa93ac7, org.apache.hadoop.hbase.NotServingRegionException: org.apache.hadoop.hbase.NotServingRegionException: Region hbase:meta,,1 is not online on workernode13.hbaseproddb4001.f5.internal.cloudapp.net,60020,1425235081338 at org.apache.hadoop.hbase.regionserver.HRegionServer.getRegionByEncodedName(HRegionServer.java:2676) at org.apache.hadoop.hbase.regionserver.HRegionServer.getRegion(HRegionServer.java:4095) at org.apache.hadoop.hbase.regionserver.HRegionServer.scan(HRegionServer.java:3076) at org.apache.hadoop.hbase.protobuf.generated.ClientProtos$ClientService$2.callBlockingMethod(ClientProtos.java:28861) at org.apache.hadoop.hbase.ipc.RpcServer.call(RpcServer.java:2008) at org.apache.hadoop.hbase.ipc.CallRunner.run(CallRunner.java:92) at org.apache.hadoop.hbase.ipc.SimpleRpcScheduler.consumerLoop(SimpleRpcScheduler.java:160) at org.apache.hadoop.hbase.ipc.SimpleRpcScheduler.access$000(SimpleRpcScheduler.java:38) at org.apache.hadoop.hbase.ipc.SimpleRpcScheduler$1.run(SimpleRpcScheduler.java:110) at java.lang.Thread.run(Thread.java:745) {code} When archiving old WALs, WASB will do rename operation by copying src blob to destination blob and deleting the src blob. Copy blob is very costly in Azure storage and during Azure storage gc, it will be highly likely throttled. The throttling by Azure storage usually ends within 15mins. Current WASB retry policy is exponential retry, but only last at most for 2min. Short term fix will be adding a more intensive exponential retry when copy blob is throttled.

jira_issues_comments:

1. Good find Duo, this explains some throttling I've seen already! | Copy blob is very costly in Azure storage and during Azure storage gc, it will be highly likely throttled. Why can't we just "move" (CopyFromBlob) it to a different container? If I'm not mistaken it is a pretty cheap linkage operation, O(1) so to say.
2. **body:** [~thomas.jungblut] We discussed with Azure storage team. The problem here is during copying blob, temp tables will be created. Azure storage has cleaner threads to clean these temp tables. When the number of temp tables reaches a certain number, Azure storage will throtte the copy blob operation so that no more temp tables will be created. However, during Azure storage gc, cleaner is blocked, which restricts the number of copy blob operations meanwhile. It seems not simply a linkage operation. I do not fully know the details though.
   label: code-design
3. [~cnauroth] Please take a look.
4. [~onpduo] you are right- I was mistaking it with deletes of containers vs. blobs. Can you please shed some light on why we archive old WALs? I would assume we can just queue them up for deletion and delete them at a rate that doesn't cause throttling while it was splitting logs. I know this might be a very "localized" solution to HBase, do you see any better fix than just changing the retry backoffs?
5. [~thomas.jungblut] Here's some words from [~enis], {code} "There is currently two services which may keep the files in the archive dir. First is a TTL process, which ensures that the WAL files are kept at least for 10 min. This is mainly for debugging. The other one is replication. If you have replication setup, the replication processes will hang on to the WAL files until they are replicated. There was some related discussion about directly deleting those files, but that was not implemented AFAIK. HBase assumes that rename() is a cheap operation, and uses rename for not only WAL files but for data files as well." {code} However, in cloud, especially in Azure storage, rename() is not a cheap operation. Currently Azure storage gc only happens on page blobs and not as frequently as you imagined. So changing retry backoffs on copyblob operation seems the only short term fix here.
6. [~apurtell], although the issue title mentions HBase, the root cause for this problem actually resides in Hadoop Common, specifically the Azure Storage {{FileSystem}} implementation. I have updated the issue title to try to make this clearer. It looks like I don't have access to move HBASE issues, so would you mind moving this back to HADOOP? Thanks!
7. Sure, moving it back! Thanks [~cnauroth]
8. Patch looks good to me. Is copyBlob the only operation that may get ServerBusy ?
9. [~enis] Currently yes. Since our WASB driver is slow due to the synchronized hsync() method when writing WALs to Azure storage, we have not seen other operations being throttled.
10. {color:red}-1 overall{color}. Here are the results of testing the latest attachment http://issues.apache.org/jira/secure/attachment/12702958/HADOOP-11681.01.patch against trunk revision d6e05c5. {color:green}+1 @author{color}. The patch does not contain any @author tags. {color:green}+1 tests included{color}. The patch appears to include 1 new or modified test files. {color:green}+1 javac{color}. The applied patch does not increase the total number of javac compiler warnings. {color:green}+1 javadoc{color}. There were no new javadoc warning messages. {color:green}+1 eclipse:eclipse{color}. The patch built with eclipse:eclipse. {color:red}-1 findbugs{color}. The patch appears to introduce 1 new Findbugs (version 2.0.3) warnings. {color:green}+1 release audit{color}. The applied patch does not increase the total number of release audit warnings. {color:green}+1 core tests{color}. The patch passed unit tests in hadoop-tools/hadoop-azure. Test results: https://builds.apache.org/job/PreCommit-HADOOP-Build/5896//testReport/ Findbugs warnings: https://builds.apache.org/job/PreCommit-HADOOP-Build/5896//artifact/patchprocess/newPatchFindbugsWarningshadoop-azure.html Console output: https://builds.apache.org/job/PreCommit-HADOOP-Build/5896//console This message is automatically generated.
11. [~cnauroth] Could you take a look?
12. **body:** Hi, [~onpduo]. I have just one question. Right now, the patch waits to see if it encounters a {{SERVER_BUSY}} error, and then restarts the operation with the redefined retry policy. Is there any reason not to just use this retry policy right from the beginning on the initial call to {{startCopyFromBlob}}? The patch will need to be reformatted to fit Hadoop coding conventions. We indent by 2 spaces, and we wrap lines that exceed 80 characters. Thanks!
    label: code-design
13. **body:** [~cnauroth] There is a default retry policy for all the Azure storage calls, and it is initialized with the Azure storage client. {code} private static final int DEFAULT_MIN_BACKOFF_INTERVAL = 1 * 1000; // 1s private static final int DEFAULT_MAX_BACKOFF_INTERVAL = 30 * 1000; // 30s private static final int DEFAULT_BACKOFF_INTERVAL = 1 * 1000; // 1s private static final int DEFAULT_MAX_RETRY_ATTEMPTS = 15; {code} The backoff interval is 1s. Azure storage throttling issue caused by Azure storage GC happens very rarely. Until now only one customer met this issue and only last 10-15 mins every day. Below is the retry policy for copyblob, {code} private static final int DEFAULT_COPYBLOB_MIN_BACKOFF_INTERVAL = 3 * 1000; // 3s private static final int

DEFAULT_COPYBLOB_MAX_BACKOFF_INTERVAL = 90 * 1000; // 90s private static final int DEFAULT_COPYBLOB_BACKOFF_INTERVAL = 30 * 1000; // 30s private static final int DEFAULT_COPYBLOB_MAX_RETRY_ATTEMPTS = 15; {code} The backoff is longer, 15s. We set these values in order to let it retry as much as 15min. We only apply this to the rare Azure storage GC case so we do not lose performance in the normal cases. I have changed the format.
label: code-design

14. {color:red}-1 overall{color}. Here are the results of testing the latest attachment http://issues.apache.org/jira/secure/attachment/12703803/HADOOP-11681.02.patch against trunk revision a5cf985. {color:green}+1 @author{color}. The patch does not contain any @author tags. {color:green}+1 tests included{color}. The patch appears to include 1 new or modified test files. {color:green}+1 javac{color}. The applied patch does not increase the total number of javac compiler warnings. {color:green}+1 javadoc{color}. There were no new javadoc warning messages. {color:green}+1 eclipse:eclipse{color}. The patch built with eclipse:eclipse. {color:red}-1 findbugs{color}. The patch appears to introduce 1 new Findbugs (version 2.0.3) warnings. {color:green}+1 release audit{color}. The applied patch does not increase the total number of release audit warnings. {color:green}+1 core tests{color}. The patch passed unit tests in hadoop-tools/hadoop-azure. Test results: https://builds.apache.org/job/PreCommit-HADOOP-Build/5913//testReport/ Findbugs warnings: https://builds.apache.org/job/PreCommit-HADOOP-Build/5913//artifact/patchprocess/newPatchFindbugsWarningshadoop-azure.html Console output: https://builds.apache.org/job/PreCommit-HADOOP-Build/5913//console This message is automatically generated.
15. **body:** Thanks for the explanation about normal case vs. the new backoff policy. That makes sense. The eclipse:eclipse failure looks unrelated. I couldn't reproduce it locally. Sorry to nitpick, but there are still some lines in {{AzureNativeFileSystemStore}} that exceed the 80 character limit. I know there are some existing lines in this file that already break the rule. Don't worry about cleaning up all of the existing code, but please make sure all lines touched in the patch adhere to the 80 character limit. The findbugs warning is legitimate. I'm not sure why it's triggering now with this patch, as it appears the problem existed before the patch. We can fix this by changing the {{catch (Exception e)}} so that there are 2 separate catch clauses for {{catch (StorageException e)}} and {{catch (URISyntaxException e)}}. Each one can be rethrown wrapped as an {{AzureException}}. We're almost there. Thanks, Duo!
    label: code-design
16. Please disregard the mention of an eclipse:eclipse failure in my last comment. That comment was meant for a different patch. Sorry about that.
17. **body:** [~cnauroth] I have wrapped those lines execeeding 80 characters, and indent by 2 characters.
    label: code-design
18. {color:green}+1 overall{color}. Here are the results of testing the latest attachment http://issues.apache.org/jira/secure/attachment/12704005/HADOOP-11681.03.patch against trunk revision 344d7cb. {color:green}+1 @author{color}. The patch does not contain any @author tags. {color:green}+1 tests included{color}. The patch appears to include 1 new or modified test files. {color:green}+1 javac{color}. The applied patch does not increase the total number of javac compiler warnings. {color:green}+1 javadoc{color}. There were no new javadoc warning messages. {color:green}+1 eclipse:eclipse{color}. The patch built with eclipse:eclipse. {color:green}+1 findbugs{color}. The patch does not introduce any new Findbugs (version 2.0.3) warnings. {color:green}+1 release audit{color}. The applied patch does not increase the total number of release audit warnings. {color:green}+1 core tests{color}. The patch passed unit tests in hadoop-tools/hadoop-azure. Test results: https://builds.apache.org/job/PreCommit-HADOOP-Build/5919//testReport/ Console output: https://builds.apache.org/job/PreCommit-HADOOP-Build/5919//console This message is automatically generated.
19. I have committed this to trunk, branch-2 and branch-2.7. Duo, thank you for contributing the patch and incorporating the feedback. Thomas and Enis, thank you for helping with code review.
20. FAILURE: Integrated in Hadoop-trunk-Commit #7306 (See [https://builds.apache.org/job/Hadoop-trunk-Commit/7306/]) HADOOP-11693. Azure Storage FileSystem rename operations are throttled too aggressively to complete HBase WAL archiving. Contributed by Duo Xu. (cnauroth: rev 7a346bcb4fa5b56191ed00a39e72e51c9bdf1b56) * hadoop-common-project/hadoop-common/CHANGES.txt * hadoop-tools/hadoop-azure/src/main/java/org/apache/hadoop/fs/azure/AzureNativeFileSystemStore.java * hadoop-tools/hadoop-azure/src/test/java/org/apache/hadoop/fs/azure/MockStorageInterface.java * hadoop-tools/hadoop-azure/src/main/java/org/apache/hadoop/fs/azure/StorageInterfaceImpl.java * hadoop-tools/hadoop-azure/src/main/java/org/apache/hadoop/fs/azure/StorageInterface.java
21. FAILURE: Integrated in Hadoop-Yarn-trunk #864 (See [https://builds.apache.org/job/Hadoop-Yarn-trunk/864/]) HADOOP-11693. Azure Storage FileSystem rename operations are throttled too aggressively to complete HBase WAL archiving. Contributed by Duo Xu. (cnauroth: rev 7a346bcb4fa5b56191ed00a39e72e51c9bdf1b56) * hadoop-tools/hadoop-azure/src/main/java/org/apache/hadoop/fs/azure/AzureNativeFileSystemStore.java * hadoop-tools/hadoop-azure/src/test/java/org/apache/hadoop/fs/azure/MockStorageInterface.java * hadoop-tools/hadoop-azure/src/main/java/org/apache/hadoop/fs/azure/StorageInterfaceImpl.java * hadoop-common-project/hadoop-common/CHANGES.txt * hadoop-tools/hadoop-azure/src/main/java/org/apache/hadoop/fs/azure/StorageInterface.java
22. FAILURE: Integrated in Hadoop-Yarn-trunk-Java8 #130 (See [https://builds.apache.org/job/Hadoop-Yarn-trunk-Java8/130/]) HADOOP-11693. Azure Storage FileSystem rename operations are throttled too aggressively to complete HBase WAL archiving. Contributed by Duo Xu. (cnauroth: rev 7a346bcb4fa5b56191ed00a39e72e51c9bdf1b56) * hadoop-common-project/hadoop-common/CHANGES.txt * hadoop-tools/hadoop-azure/src/main/java/org/apache/hadoop/fs/azure/StorageInterfaceImpl.java * hadoop-tools/hadoop-azure/src/test/java/org/apache/hadoop/fs/azure/MockStorageInterface.java * hadoop-tools/hadoop-azure/src/main/java/org/apache/hadoop/fs/azure/StorageInterface.java * hadoop-tools/hadoop-azure/src/main/java/org/apache/hadoop/fs/azure/AzureNativeFileSystemStore.java
23. FAILURE: Integrated in Hadoop-Hdfs-trunk #2062 (See [https://builds.apache.org/job/Hadoop-Hdfs-trunk/2062/]) HADOOP-11693. Azure Storage FileSystem rename operations are throttled too aggressively to complete HBase WAL archiving. Contributed by Duo Xu. (cnauroth: rev 7a346bcb4fa5b56191ed00a39e72e51c9bdf1b56) * hadoop-tools/hadoop-azure/src/test/java/org/apache/hadoop/fs/azure/MockStorageInterface.java * hadoop-common-project/hadoop-common/CHANGES.txt * hadoop-tools/hadoop-azure/src/main/java/org/apache/hadoop/fs/azure/AzureNativeFileSystemStore.java * hadoop-tools/hadoop-azure/src/main/java/org/apache/hadoop/fs/azure/StorageInterfaceImpl.java * hadoop-tools/hadoop-azure/src/main/java/org/apache/hadoop/fs/azure/StorageInterface.java

24. FAILURE: Integrated in Hadoop-Hdfs-trunk-Java8 #121 (See [https://builds.apache.org/job/Hadoop-Hdfs-trunk-Java8/121/]) HADOOP-11693. Azure Storage FileSystem rename operations are throttled too aggressively to complete HBase WAL archiving. Contributed by Duo Xu. (cnauroth: rev 7a346bcb4fa5b56191ed00a39e72e51c9bdf1b56) * hadoop-tools/hadoop-azure/src/main/java/org/apache/hadoop/fs/azure/StorageInterface.java * hadoop-tools/hadoop-azure/src/test/java/org/apache/hadoop/fs/azure/MockStorageInterface.java * hadoop-tools/hadoop-azure/src/main/java/org/apache/hadoop/fs/azure/AzureNativeFileSystemStore.java * hadoop-tools/hadoop-azure/src/main/java/org/apache/hadoop/fs/azure/StorageInterfaceImpl.java * hadoop-common-project/hadoop-common/CHANGES.txt
25. FAILURE: Integrated in Hadoop-Mapreduce-trunk-Java8 #130 (See [https://builds.apache.org/job/Hadoop-Mapreduce-trunk-Java8/130/]) HADOOP-11693. Azure Storage FileSystem rename operations are throttled too aggressively to complete HBase WAL archiving. Contributed by Duo Xu. (cnauroth: rev 7a346bcb4fa5b56191ed00a39e72e51c9bdf1b56) * hadoop-tools/hadoop-azure/src/main/java/org/apache/hadoop/fs/azure/AzureNativeFileSystemStore.java * hadoop-tools/hadoop-azure/src/main/java/org/apache/hadoop/fs/azure/StorageInterface.java * hadoop-tools/hadoop-azure/src/test/java/org/apache/hadoop/fs/azure/MockStorageInterface.java * hadoop-tools/hadoop-azure/src/main/java/org/apache/hadoop/fs/azure/StorageInterfaceImpl.java * hadoop-common-project/hadoop-common/CHANGES.txt
26. FAILURE: Integrated in Hadoop-Mapreduce-trunk #2080 (See [https://builds.apache.org/job/Hadoop-Mapreduce-trunk/2080/]) HADOOP-11693. Azure Storage FileSystem rename operations are throttled too aggressively to complete HBase WAL archiving. Contributed by Duo Xu. (cnauroth: rev 7a346bcb4fa5b56191ed00a39e72e51c9bdf1b56) * hadoop-tools/hadoop-azure/src/test/java/org/apache/hadoop/fs/azure/MockStorageInterface.java * hadoop-tools/hadoop-azure/src/main/java/org/apache/hadoop/fs/azure/StorageInterfaceImpl.java * hadoop-tools/hadoop-azure/src/main/java/org/apache/hadoop/fs/azure/AzureNativeFileSystemStore.java * hadoop-tools/hadoop-azure/src/main/java/org/apache/hadoop/fs/azure/StorageInterface.java * hadoop-common-project/hadoop-common/CHANGES.txt
27. Hi [~cnauroth] I have submitted a new patch, which does the retries in WASB rather than rely on Azure Storage SDK. As I looked into the source code this week, Azure Storage SDK regards storage exception as non-retryable, so when throttling happens, the current code might still not work. Could you reopen this JIRA and review it ASAP? Thanks.
28. Hello [~onpduo]. The HADOOP-11693 patch already shipped in Apache Hadoop 2.7.0. At this point, please create a new jira to track the new change instead of attaching new patches here. BTW, I noticed that this new patch appears to be in a multi-byte character encoding (UTF-16 with BOM?). When you create the new jira, please attach an ASCII patch file.