Item 29
git_comments:

git_commits:

1. **summary:** HBASE-2231 Compaction events should be written to HLog; LOGGING ADDENDUM
   message: HBASE-2231 Compaction events should be written to HLog; LOGGING ADDENDUM git-svn-id: https://svn.apache.org/repos/asf/hbase/trunk@1485873 13f79535-47bb-0310-9956-ffa450edef68

github_issues:

github_issues_comments:

github_pulls:

github_pulls_comments:

github_pulls_reviews:

jira_issues:

1. **summary:** Compaction events should be written to HLog
   description: The sequence for a compaction should look like this: # Compact region to "new" files # Write a "Compacted Region" entry to the HLog # Delete "old" files This deals with a case where the RS has paused between step 1 and 2 and the regions have since been reassigned.

jira_issues_comments:

1. Essentially, we're relying on the following sequence of events in the case of a pause: # The RS starts a compaction # GC or whatever kicks in, causing the RS to be kicked out of the ZK quorum # Master notices this and opens the RS's latest HLog for append. This steals the write lease on the HLog file, bumps generation stamps, etc. # RS comes back to life, finishes the compaction, writes "Compacted Region" to HLog, and calls hflush() # hflush() fails since the original writer lease is no longer valid. Thus, the region server aborts and does not futz with the region's on-HDFS data (thus not interfering with the new server of this region) This same sequence occurs for the case when the region tries to hflush edits in-flight RPCs from before the pause. There was some question about a problem where the regionserver also rolls its HLog right after the pause. In this case, the master might not catch the rolled HLog file, and the RS wouldn't be properly interrupted. Perhaps one solution for this is that, when rolling an HLog, you must open the new HLog first, then append an "HLog rolled" message to the old HLog, then start writing into the new one. There can then be a protocol on the HMaster to determine that it has really stolen the lease for the latest HLog. Regardless of the above question, the HMaster will also need to be sure to take the lease on the last HLog immediately, rather than starting to roll forward from the first HLog without looking at the latest ones.
2. Just wanted to attach evidence to the issue pointed out by Todd - "the regionserver also rolls its HLog right after the pause. In this case, the master might not catch the rolled HLog file, and the RS wouldn't be properly interrupted" From an email exchange between a few of us: > Look at these snippets from the log: > 2010-02-15 16:23:56,493 INFO org.apache.hadoop.hbase.regionserver.HRegionServer: Got ZooKeeper event, state: Disconnected, type: None, path: null > 2010-02-15 16:23:57,848 WARN org.apache.hadoop.hbase.regionserver.HLog: IPC Server handler 30 on 60020 took 1200ms appending an edit to hlog; editcount=121 > 2010-02-15 16:24:04,697 INFO org.apache.hadoop.hbase.regionserver.HRegionServer: Got ZooKeeper event, state: Expired, type: None, path: null > 2010-02-15 16:24:04,697 WARN org.apache.zookeeper.ClientCnxn: Exception closing session 0x326c5173bc80000 to sun.nio.ch.SelectionKeyImpl@5534afcb > > This means master did not take possession of the latest hlog. So the RS actually appended into the log AFTER the zk timeout happened. The suggestion from yesterday's discussion (first open the last hlog) should take care of this edit.
3. A offline discussion among Karthik and Arvind resulted in the idea that the master should rename the directory where hlogs are written. It should then open the last file for "append" to ensure that the old region server cannot write anymore to it. After that, the master is free to safely process all the logs in the

hlog directory. This would need that the region server use org.apache.hadoop.fs.create() while crearing a hlog file (which allows failing a create if any intermediate directory in the path is absent).

4. bq. This would need that the region server use org.apache.hadoop.fs.create() while crearing a hlog file (which allows failing a create if any intermediate directory in the path is absent). Which API does this? I was under the impression that all of the available create() APIs recursively mkdired the containing directory (though maybe I'm out of date or just plain wrong)
5. org.apache.hadoop.fs.FileContext.create() has a "option" flag that allows an app to specify whether intermediate directories should be created or not.
6. yea, but that API isn't til 0.22. Backporting FileContext is a no-go imo.
7. I'm going to suck it up and take this one on, I think - it's fairly easy to reproduce by doing heavy writes with WAL disabled for those writes, and then kill -STOP a region server for 65 seconds or so.
8. linking HBASE-2519, which causes this to currently present as silent data loss in some situations
9. Here's a test case which shows the silent dataloss after causing IOEs on the next compaction of the affected region.
10. My plan is this: - HLog already has a special METAFAMILY/METAROW concept that can be used to put out-of-band info in the editlog. - Currently we write a "COMPLETE_CACHE_FLUSH" entry in there for cache flushes - Do a similar thing, but with COMPACTION_FINISHED. In this case we'll also have a qualifier COMPACTION_DATA which will contain (a) the compacted file path in the compaction dir, and (b) the files that fed into the compaction - During HLog replay, if we see one of these entries, we try to "finish" the compaction by moving the compacted file into place and removing the source files (these are idempotent operations, so it's fine if the RS already did it)
11. Bulk move of 0.20.5 issues into 0.21.0 after vote that we merge branch into TRUNK up on list.
12. **body:** Here's a patch for trunk with tests, etc. This still needs cleanup, etc, please don't commit yet. I'll do some cleanup and then post a reviewboard asap.
    label: code-design
13. I've been thinking about this a bit, and I want to think about a different design. The flaw with the design as I outlined above is that we have to read through the entire reconstruction HLog until we find the special entries telling us to "finish off" the compaction for the dead regionserver. This is fine in today's world, but I think in some instances it would be better to finish the compaction and get the directory in a reasonable state *before* replaying the logs. This would allow, for example, the recovering regionserver to start serving the region in "stale readonly" mode before log replay is done. I'm thinking about a different design in which we add an /hbase/ops-in-progress/regionserver/ directory in zookeeper. When we are about to start moving files around after a compaction, we write our intention in there, and when we're done, we remove it. Thus the HMaster can take of finishing or aborting the compaction immediately when it starts recovery, and the recovering regionserver doesn't have to worry about the transition in progress. Thoughts?
14. **body:** Do you want to revive this issue Todd? If so, my thought is that we should just rollback whatever that was in progress in a way that the sleeping node couldn't mess up when it wakes up. I'm not super keen on using ZK for that, seems overly complicated.
    label: code-design
15. **body:** The original plan seems good, or have things changed too much? If we can piggypack on HLog, that would be better since as JD pointed out ZK can be tricky. Given that this issue is so small, should we hold up 0.90 for this or not?
    label: code-design
16. If someone wants to take this over, that's cool - I'm unlikely to have time in the next week and a half, though. It is a nasty bug that I've seen in real life, though.
17. Todd, you got a fix for this or will we punt to 0.92?
18. Let's punt, though I'm working on it and maybe it will happen tomorrow :)
19. **body:** Setting to 0.92 but hope to get this in a dot release or later RC of 0.90 if 0.92 takes too long
    label: code-design
20. hi folks, can we get this into some release of 0.90?
21. yet another approach would be the following: the hdfs files remain inside a path that has the currently assigned regionname encoded in it. For example, a path of the form /hbase/tablename/regionname/currently-assigned-regionserver/.... when a master assigns a region to a regionserver, it makes the change in zookeeper and then issues one rename call to hdfs to rename the top level region directory name to the appropriate regionserver name. In some ways, this is similar to IO-fencing in the sense that any old region servers cannot operate on any of the existing files of this region. Recovery from a master restart and/or region-server restart should be simple, and I can list those steps

later. This has some drawbacks, especially since this makes the region data have variable path names at different points in time, could cause some issues in bulk-loading via map-reduce.

22. Marking patch available
23. Moving out of 0.92.0. Pull back in if someone wants to work on it.
24. Moving out of 0.94. Probably not a blocker if it can go unfixed for a long time.
25. Todd's patch is not small. Now that HLog interface has been introduced, rebase is needed.
26. Testcase tailored to 0.94 branch. I found that TestIOFencing hung: {code} "main" prio=5 tid=7fdd52801800 nid=0x10b415000 waiting on condition [10b413000] java.lang.Thread.State: TIMED_WAITING (sleeping) at java.lang.Thread.sleep(Native Method) at org.apache.hadoop.hbase.TestIOFencing.testFencingAroundCompaction(TestIOFencing.java:208) {code} at the sleep() call below: {code} compactingRegion.allowCompactions(); while (compactingRegion.compactCount == 0) { Thread.sleep(1000); {code}
27. Here is the test for trunk. Hanging at mo. Let me see why.
28. Fixed hang. Test passes now but it should not. We have become lax if a file is missing when we go to compact -- perhaps because region double-assigned -- since we made the CompactionRequest object changes. I think the reasoning is that by the time our compaction runs, another may have run before us so it is ok if the geography is different. Need to fix this first. Let me dig more.
29. So, test passes because we are able to read from deleted file because dfsclient has block locations cached. I suppose this makes sense it is just a little unexpected (It took me a while to figure it). Not sure how test failed in the past. My guess is that in the past we were strict when you went to compact that all referenced files had to be present. That is no longer the case in that we seem to be loosey-goosey about it to handle the case where a compaction may have run before ours. Let me look into this more.
30. This test case fails when files have been removed by a compaction. Not to forward port Todd's fancy WAL writing.
31. Not done yet. pb'd the compaction description added to the WAL.
32. Here is finished patch but it seems that IOFencing does not work. After recovering lease and assigning new regions, the dying server can still append the WAL 'successfully'.
33. is there any documentation on how IO fencing is supposed to work in HBase? It wouldn't surprise me if you can still append to a moved file given how HDFS works.
34. ping? Perhaps we should introduce file creation operation of some sort to the end of compaction instead.
35. [~sershe] Sorry. Owe you response. Need to restudy patch and reconstitute context. HBASE-2645 is related. Writer over there gets stuck on a sync and eventually fails complaining about lease. Would think we'd see similar over here when we try to write the compaction edit to the WAL on the dying regionserver.
36. Could it be related to the issue fixed in HBASE-7878, where recoverLease does not really recover lease when splitting start when the lease is not recovered yet, so due to timing data loss is possible? Or was it a different behavior here on dead writer's side
37. *and the splitting starts when
38. **body:** I have rebased Stack's v4 patch, and made some changes. From my testing, it seems that IO Fencing for the WAL works (tested with HBASE-7878, not HBASE-8389). [~stack] do you remember how did you test and conclude that it does not work? This is pretty important, so I can dedicate some more time on testing it. Some of the changes from patch v4 to v5 are: - Renamed Compaction -> CompactionDescriptor - Compaction output can be a list of files, not a single file. - Added compaction WAL edit replay. Now we recognize and replay the compaction edit on region open. This is needed because if we fail after we snyc() the wal, we might still try to delete the files. - Added 2 more tests for above condition. - HLog does not know about compaction wal edit. - Compaction wal edit log goes through the normal append code path. I've tried to capture the Fencing / Idempotency semantics in the following javadoc excerpt: Compaction event should be idempotent, since there is no IO Fencing for the region directory in hdfs. A region server might still try to complete the compaction after it lost the region. That is why the following events are carefully ordered for a compaction: 1. Compaction writes new files under region/.tmp directory (compaction output) 2. Compaction atomically moves the temporary file under region directory 3. Compaction appends a WAL edit containing the compaction input and output files. Forces sync on WAL. 4. Compaction deletes the input files from the region directory. Failure conditions are handled like this: - If RS fails before 2, compaction wont complete. Even if RS lives on and finishes the compaction later, it will only write the new data file to the region directory. Since we already have this data, this will be idempotent but we will have a redundant copy of the data. - If RS fails between 2 and 3, the region will have a redundant copy of the data. The RS that failed won't be able to finish snyc() for WAL because of lease recovery in WAL. - If RS fails after 3, the region region server who opens the region will pick up the the compaction marker from the WAL and replay it by removing

the compaction input files. Failed RS can also attempt to delete those files, but the operation will be idempotent
label: code-design

39. {color:red}-1 overall{color}. Here are the results of testing the latest attachment http://issues.apache.org/jira/secure/attachment/12580216/hbase-2231_v5.patch against trunk revision . {color:green}+1 @author{color}. The patch does not contain any @author tags. {color:green}+1 tests included{color}. The patch appears to include 17 new or modified tests. {color:red}-1 patch{color}. The patch command could not apply the patch. Console output: https://builds.apache.org/job/PreCommit-HBASE-Build/5426//console This message is automatically generated.
40. **body:** bq. .... it seems that IO Fencing for the WAL works Excellent (You are a better man than me!) bq. stack do you remember how did you test and conclude that it does not work? This is pretty important, so I can dedicate some more time on testing it. Straight hbase and its bundled hadoop from circa November 2012. Looking at the differences between v5 and v4, I do not see anything that could have changed the mechanics unless its the hdfs difference, or very likely, I just wasn't testing properly. Patch looks great. {code} + repeated string compactionInput = 4; + repeated string compactionOutput = 5; + required string storeHomeDir = 6; {code} Are these full paths in above or partials? Hopefully the latter (in case hbase.rootdir gets moved). This is good change: {code} - if (HLogUtil.isMetaFamily(kv.getFamily())) continue; + if (WALEdit.isMetaEditFamily(kv.getFamily())) continue; {code} Do we need this class, CompactionDescriptor? Why not just use the pb class of same name? It has good accessors, etc., already? Or its constructor should take the pb CompactionDescriptor? Should toByteArray add pb magic preamble? (Probably not -- too much). Is there enough to ensure that this is a compaction edit? {code} + if (kv.matchingFamily(WALEdit.METAFAMILY) || !Bytes.equals(key.getEncodedRegionName(), this.getRegionInfo().getEncodedNameAsBytes())) { + //this is a special edit, we should handle it + CompactionDescriptor compaction = WALEdit.getCompaction(kv); {code} In other words, any other kinds of edits w/ a METAFAMILY cf? Is this used? createFlush Patch looks good. IOFencing works? I was seeing that I could write the edit anyways even after the master had recovered the lease. Maybe it s hbase-7878 returning when it shouldn't have been (we weren't checking the return flag?) Thanks Enis. +1 on commit.
label: code-design
41. **body:** bq. Are these full paths in above or partials? Hopefully the latter (in case hbase.rootdir gets moved). Good point, I think they are absolute paths for now. I'll change that. bq. Do we need this class, CompactionDescriptor? I don't know, you tell me :) It was in the v4 patch, so I kept it. Let me get rid of it, we can just use the PB object directly as you said. bq. Is there enough to ensure that this is a compaction edit? If METAFAMILY we call getCompaction() which does check the row, cf, and column. If not just returns null. I was thinking of adding a new Type in KeyValue, smt like Marker, so that we do not hack around METAFAMILY. wdyt, I can do a follow up issue. bq. createFlush Is this used? It is not used after HBASE-7329. I should remove it. One thing we discussed offline with Sergey was that, if WAL disabled or deferred, a RS that lost the region, might still try to do a flush on entries that did not go into WAL. In these case, we might be writing a file to the region after the we lost the control of the region, but it should be fine since either: (1) the entries in the flushed file went into WAL already, so we would just be adding a redundant file to the region. (2) the entries did not go into WAL, but will be flushed into the region. They will or will not be immediately seen by the region, because the region might be already open. They will be seen by subsequent region opens. Both cases should be fine as far as i can tell though, because for (1), we rely on idempotent edits, for (2) since wal is deferred/skipped these cases would be semantically acceptable.
label: code-design
42. Updated patch. Let's see what hadoopqa thinks.
43. **body:** {color:red}-1 overall{color}. Here are the results of testing the latest attachment http://issues.apache.org/jira/secure/attachment/12580486/hbase-2231_v6.patch against trunk revision . {color:green}+1 @author{color}. The patch does not contain any @author tags. {color:green}+1 tests included{color}. The patch appears to include 17 new or modified tests. {color:green}+1 hadoop1.0{color}. The patch compiles against the hadoop 1.0 profile. {color:green}+1 hadoop2.0{color}. The patch compiles against the hadoop 2.0 profile. {color:red}-1 javadoc{color}. The javadoc tool appears to have generated 2 warning messages. {color:green}+1 javac{color}. The applied patch does not increase the total number of javac compiler warnings. {color:green}+1 findbugs{color}. The patch does not introduce any new Findbugs (version 1.3.9) warnings. {color:red}-1 release audit{color}. The applied patch generated 1 release audit warnings (more than the trunk's current 0 warnings). {color:red}-1 lineLengths{color}. The patch introduces lines longer than 100 {color:green}+1 site{color}. The mvn site goal succeeds with this patch. {color:red}-1 core tests{color}. The patch failed these unit tests:

org.apache.hadoop.hbase.mapreduce.TestHFileOutputFormat Test results: https://builds.apache.org/job/PreCommit-HBASE-Build/5448//testReport/ Release audit warnings: https://builds.apache.org/job/PreCommit-HBASE-Build/5448//artifact/trunk/patchprocess/patchReleaseAuditProblems.txt Findbugs warnings: https://builds.apache.org/job/PreCommit-HBASE-Build/5448//artifact/trunk/patchprocess/newPatchFindbugsWarningshbase-prefix-tree.html Findbugs warnings: https://builds.apache.org/job/PreCommit-HBASE-Build/5448//artifact/trunk/patchprocess/newPatchFindbugsWarningshbase-client.html Findbugs warnings: https://builds.apache.org/job/PreCommit-HBASE-Build/5448//artifact/trunk/patchprocess/newPatchFindbugsWarningshbase-common.html Findbugs warnings: https://builds.apache.org/job/PreCommit-HBASE-Build/5448//artifact/trunk/patchprocess/newPatchFindbugsWarningshbase-protocol.html Findbugs warnings: https://builds.apache.org/job/PreCommit-HBASE-Build/5448//artifact/trunk/patchprocess/newPatchFindbugsWarningshbase-server.html Findbugs warnings: https://builds.apache.org/job/PreCommit-HBASE-Build/5448//artifact/trunk/patchprocess/newPatchFindbugsWarningshbase-hadoop1-compat.html Findbugs warnings: https://builds.apache.org/job/PreCommit-HBASE-Build/5448//artifact/trunk/patchprocess/newPatchFindbugsWarningshbase-examples.html Findbugs warnings: https://builds.apache.org/job/PreCommit-HBASE-Build/5448//artifact/trunk/patchprocess/newPatchFindbugsWarningshbase-hadoop-compat.html Console output: https://builds.apache.org/job/PreCommit-HBASE-Build/5448//console This message is automatically generated.
label: code-design

44. **body:** Rebased patch, and fixed javadoc warnings. Test and release audit warnings seem unrelated.
label: code-design
45. bq. I don't know, you tell me It was in the v4 patch, so I kept it. Let me get rid of it, we can just use the PB object directly as you said. I don't remember what I was thinking back then (unsmile). bq. I was thinking of adding a new Type in KeyValue, smt like Marker, so that we do not hack around METAFAMILY. wdyt, I can do a follow up issue. It is fine as is for now. If we want more special markers, lets do the special KV type (good idea btw). bq. It is not used after HBASE-7329. I should remove it. Out of scope for this. New issue? bq. if WAL disabled or deferred... What is this? If either of the above, you are playing w/ fire anyways so you could lose data? Let me look at new version of patch.
46. +1 on patch.
47. **body:** {color:red}-1 overall{color}. Here are the results of testing the latest attachment http://issues.apache.org/jira/secure/attachment/12580586/hbase-2231_v7.patch against trunk revision . {color:green}+1 @author{color}. The patch does not contain any @author tags. {color:green}+1 tests included{color}. The patch appears to include 17 new or modified tests. {color:green}+1 hadoop1.0{color}. The patch compiles against the hadoop 1.0 profile. {color:green}+1 hadoop2.0{color}. The patch compiles against the hadoop 2.0 profile. {color:green}+1 javadoc{color}. The javadoc tool did not generate any warning messages. {color:green}+1 javac{color}. The applied patch does not increase the total number of javac compiler warnings. {color:green}+1 findbugs{color}. The patch does not introduce any new Findbugs (version 1.3.9) warnings. {color:red}-1 release audit{color}. The applied patch generated 1 release audit warnings (more than the trunk's current 0 warnings). {color:red}-1 lineLengths{color}. The patch introduces lines longer than 100 {color:green}+1 site{color}. The mvn site goal succeeds with this patch. {color:red}-1 core tests{color}. The patch failed these unit tests: org.apache.hadoop.hbase.security.access.TestAccessController org.apache.hadoop.hbase.backup.TestHFileArchiving org.apache.hadoop.hbase.TestIOFencing org.apache.hadoop.hbase.client.TestAdmin Test results: https://builds.apache.org/job/PreCommit-HBASE-Build/5454//testReport/ Release audit warnings: https://builds.apache.org/job/PreCommit-HBASE-Build/5454//artifact/trunk/patchprocess/patchReleaseAuditProblems.txt Findbugs warnings: https://builds.apache.org/job/PreCommit-HBASE-Build/5454//artifact/trunk/patchprocess/newPatchFindbugsWarningshbase-protocol.html Findbugs warnings: https://builds.apache.org/job/PreCommit-HBASE-Build/5454//artifact/trunk/patchprocess/newPatchFindbugsWarningshbase-client.html Findbugs warnings: https://builds.apache.org/job/PreCommit-HBASE-Build/5454//artifact/trunk/patchprocess/newPatchFindbugsWarningshbase-examples.html Findbugs warnings: https://builds.apache.org/job/PreCommit-HBASE-Build/5454//artifact/trunk/patchprocess/newPatchFindbugsWarningshbase-hadoop1-compat.html Findbugs warnings: https://builds.apache.org/job/PreCommit-HBASE-

Build/5454//artifact/trunk/patchprocess/newPatchFindbugsWarningshbase-prefix-tree.html Findbugs warnings: https://builds.apache.org/job/PreCommit-HBASE-Build/5454//artifact/trunk/patchprocess/newPatchFindbugsWarningshbase-common.html Findbugs warnings: https://builds.apache.org/job/PreCommit-HBASE-Build/5454//artifact/trunk/patchprocess/newPatchFindbugsWarningshbase-server.html Findbugs warnings: https://builds.apache.org/job/PreCommit-HBASE-Build/5454//artifact/trunk/patchprocess/newPatchFindbugsWarningshbase-hadoop-compat.html Console output: https://builds.apache.org/job/PreCommit-HBASE-Build/5454//console This message is automatically generated.
label: code-design
48. Tests might be failing because of HBASE-5930. I've put an addendum there.
49. Reattach for qa.
50. **body:** {color:red}-1 overall{color}. Here are the results of testing the latest attachment http://issues.apache.org/jira/secure/attachment/12580625/hbase-2231_v7.patch against trunk revision . {color:green}+1 @author{color}. The patch does not contain any @author tags. {color:green}+1 tests included{color}. The patch appears to include 17 new or modified tests. {color:green}+1 hadoop1.0{color}. The patch compiles against the hadoop 1.0 profile. {color:green}+1 hadoop2.0{color}. The patch compiles against the hadoop 2.0 profile. {color:green}+1 javadoc{color}. The javadoc tool did not generate any warning messages. {color:green}+1 javac{color}. The applied patch does not increase the total number of javac compiler warnings. {color:green}+1 findbugs{color}. The patch does not introduce any new Findbugs (version 1.3.9) warnings. {color:red}-1 release audit{color}. The applied patch generated 1 release audit warnings (more than the trunk's current 0 warnings). {color:red}-1 lineLengths{color}. The patch introduces lines longer than 100 {color:green}+1 site{color}. The mvn site goal succeeds with this patch. {color:red}-1 core tests{color}. The patch failed these unit tests: org.apache.hadoop.hbase.backup.TestHFileArchiving Test results: https://builds.apache.org/job/PreCommit-HBASE-Build/5461//testReport/ Release audit warnings: https://builds.apache.org/job/PreCommit-HBASE-Build/5461//artifact/trunk/patchprocess/patchReleaseAuditProblems.txt Findbugs warnings: https://builds.apache.org/job/PreCommit-HBASE-Build/5461//artifact/trunk/patchprocess/newPatchFindbugsWarningshbase-protocol.html Findbugs warnings: https://builds.apache.org/job/PreCommit-HBASE-Build/5461//artifact/trunk/patchprocess/newPatchFindbugsWarningshbase-client.html Findbugs warnings: https://builds.apache.org/job/PreCommit-HBASE-Build/5461//artifact/trunk/patchprocess/newPatchFindbugsWarningshbase-examples.html Findbugs warnings: https://builds.apache.org/job/PreCommit-HBASE-Build/5461//artifact/trunk/patchprocess/newPatchFindbugsWarningshbase-hadoop1-compat.html Findbugs warnings: https://builds.apache.org/job/PreCommit-HBASE-Build/5461//artifact/trunk/patchprocess/newPatchFindbugsWarningshbase-prefix-tree.html Findbugs warnings: https://builds.apache.org/job/PreCommit-HBASE-Build/5461//artifact/trunk/patchprocess/newPatchFindbugsWarningshbase-common.html Findbugs warnings: https://builds.apache.org/job/PreCommit-HBASE-Build/5461//artifact/trunk/patchprocess/newPatchFindbugsWarningshbase-server.html Findbugs warnings: https://builds.apache.org/job/PreCommit-HBASE-Build/5461//artifact/trunk/patchprocess/newPatchFindbugsWarningshbase-hadoop-compat.html Console output: https://builds.apache.org/job/PreCommit-HBASE-Build/5461//console This message is automatically generated.
label: code-design
51. I've committed this to trunk and 0.95. Thanks Stack.
52. Attaching 0.95 version of the patch, had to resolve some minor conflicts.
53. Integrated in HBase-TRUNK #4082 (See [https://builds.apache.org/job/HBase-TRUNK/4082/]) HBASE-2231 Compaction events should be written to HLog (Stack & Enis) (Revision 1476414) Result = FAILURE enis : Files : * /hbase/trunk/hbase-client/src/main/java/org/apache/hadoop/hbase/protobuf/ProtobufUtil.java * /hbase/trunk/hbase-protocol/src/main/java/org/apache/hadoop/hbase/protobuf/generated/WAL.java * /hbase/trunk/hbase-protocol/src/main/protobuf/WAL.proto * /hbase/trunk/hbase-server/src/main/java/org/apache/hadoop/hbase/mapreduce/WALPlayer.java * /hbase/trunk/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/HRegion.java * /hbase/trunk/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/HStore.java * /hbase/trunk/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/Store.java * /hbase/trunk/hbase-

server/src/main/java/org/apache/hadoop/hbase/regionserver/wal/FSHLog.java * /hbase/trunk/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/wal/HLog.java * /hbase/trunk/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/wal/HLogUtil.java * /hbase/trunk/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/wal/WALEdit.java * /hbase/trunk/hbase-server/src/test/java/org/apache/hadoop/hbase/HBaseTestingUtility.java * /hbase/trunk/hbase-server/src/test/java/org/apache/hadoop/hbase/MiniHBaseCluster.java * /hbase/trunk/hbase-server/src/test/java/org/apache/hadoop/hbase/TestFullLogReconstruction.java * /hbase/trunk/hbase-server/src/test/java/org/apache/hadoop/hbase/TestIOFencing.java * /hbase/trunk/hbase-server/src/test/java/org/apache/hadoop/hbase/coprocessor/TestRowProcessorEndpoint.java * /hbase/trunk/hbase-server/src/test/java/org/apache/hadoop/hbase/regionserver/TestHRegion.java

54. Integrated in hbase-0.95 #164 (See [https://builds.apache.org/job/hbase-0.95/164/]) HBASE-2231 Compaction events should be written to HLog (Stack & Enis) (Revision 1476419) Result = FAILURE enis : Files : * /hbase/branches/0.95/hbase-client/src/main/java/org/apache/hadoop/hbase/protobuf/ProtobufUtil.java * /hbase/branches/0.95/hbase-protocol/src/main/java/org/apache/hadoop/hbase/protobuf/generated/WAL.java * /hbase/branches/0.95/hbase-protocol/src/main/protobuf/WAL.proto * /hbase/branches/0.95/hbase-server/src/main/java/org/apache/hadoop/hbase/mapreduce/WALPlayer.java * /hbase/branches/0.95/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/HRegion.java * /hbase/branches/0.95/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/HStore.java * /hbase/branches/0.95/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/Store.java * /hbase/branches/0.95/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/wal/FSHLog.java * /hbase/branches/0.95/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/wal/HLog.java * /hbase/branches/0.95/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/wal/HLogUtil.java * /hbase/branches/0.95/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/wal/WALEdit.java * /hbase/branches/0.95/hbase-server/src/test/java/org/apache/hadoop/hbase/HBaseTestingUtility.java * /hbase/branches/0.95/hbase-server/src/test/java/org/apache/hadoop/hbase/MiniHBaseCluster.java * /hbase/branches/0.95/hbase-server/src/test/java/org/apache/hadoop/hbase/TestFullLogReconstruction.java * /hbase/branches/0.95/hbase-server/src/test/java/org/apache/hadoop/hbase/TestIOFencing.java * /hbase/branches/0.95/hbase-server/src/test/java/org/apache/hadoop/hbase/coprocessor/TestRowProcessorEndpoint.java * /hbase/branches/0.95/hbase-server/src/test/java/org/apache/hadoop/hbase/regionserver/TestHRegion.java
55. Integrated in hbase-0.95-on-hadoop2 #82 (See [https://builds.apache.org/job/hbase-0.95-on-hadoop2/82/]) HBASE-2231 Compaction events should be written to HLog (Stack & Enis) (Revision 1476419) Result = FAILURE enis : Files : * /hbase/branches/0.95/hbase-client/src/main/java/org/apache/hadoop/hbase/protobuf/ProtobufUtil.java * /hbase/branches/0.95/hbase-protocol/src/main/java/org/apache/hadoop/hbase/protobuf/generated/WAL.java * /hbase/branches/0.95/hbase-protocol/src/main/protobuf/WAL.proto * /hbase/branches/0.95/hbase-server/src/main/java/org/apache/hadoop/hbase/mapreduce/WALPlayer.java * /hbase/branches/0.95/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/HRegion.java * /hbase/branches/0.95/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/HStore.java * /hbase/branches/0.95/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/Store.java * /hbase/branches/0.95/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/wal/FSHLog.java * /hbase/branches/0.95/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/wal/HLog.java * /hbase/branches/0.95/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/wal/HLogUtil.java * /hbase/branches/0.95/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/wal/WALEdit.java * /hbase/branches/0.95/hbase-server/src/test/java/org/apache/hadoop/hbase/HBaseTestingUtility.java * /hbase/branches/0.95/hbase-server/src/test/java/org/apache/hadoop/hbase/MiniHBaseCluster.java * /hbase/branches/0.95/hbase-server/src/test/java/org/apache/hadoop/hbase/TestFullLogReconstruction.java * /hbase/branches/0.95/hbase-server/src/test/java/org/apache/hadoop/hbase/TestIOFencing.java * /hbase/branches/0.95/hbase-server/src/test/java/org/apache/hadoop/hbase/coprocessor/TestRowProcessorEndpoint.java * /hbase/branches/0.95/hbase-server/src/test/java/org/apache/hadoop/hbase/regionserver/TestHRegion.java
56. Integrated in HBase-TRUNK-on-Hadoop-2.0.0 #512 (See [https://builds.apache.org/job/HBase-TRUNK-on-Hadoop-2.0.0/512/]) HBASE-2231 Compaction events should be written to HLog (Stack & Enis) (Revision 1476414) Result = FAILURE enis : Files : * /hbase/trunk/hbase-

client/src/main/java/org/apache/hadoop/hbase/protobuf/ProtobufUtil.java * /hbase/trunk/hbase-protocol/src/main/java/org/apache/hadoop/hbase/protobuf/generated/WAL.java * /hbase/trunk/hbase-protocol/src/main/protobuf/WAL.proto * /hbase/trunk/hbase-server/src/main/java/org/apache/hadoop/hbase/mapreduce/WALPlayer.java * /hbase/trunk/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/HRegion.java * /hbase/trunk/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/HStore.java * /hbase/trunk/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/Store.java * /hbase/trunk/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/wal/FSHLog.java * /hbase/trunk/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/wal/HLog.java * /hbase/trunk/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/wal/HLogUtil.java * /hbase/trunk/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/wal/WALEdit.java * /hbase/trunk/hbase-server/src/test/java/org/apache/hadoop/hbase/HBaseTestingUtility.java * /hbase/trunk/hbase-server/src/test/java/org/apache/hadoop/hbase/MiniHBaseCluster.java * /hbase/trunk/hbase-server/src/test/java/org/apache/hadoop/hbase/TestFullLogReconstruction.java * /hbase/trunk/hbase-server/src/test/java/org/apache/hadoop/hbase/TestIOFencing.java * /hbase/trunk/hbase-server/src/test/java/org/apache/hadoop/hbase/coprocessor/TestRowProcessorEndpoint.java * /hbase/trunk/hbase-server/src/test/java/org/apache/hadoop/hbase/regionserver/TestHRegion.java

57. {quote} If RS fails after 3, the region region server who opens the region will pick up the the compaction marker from the WAL and replay it by removing the compaction input files. Failed RS can also attempt to delete those files, but the operation will be idempotent {quote} One issue on the above is that WAL replay skips all flushed WAL edits. Let's say after the compaction marker WAL edit we have a region flush. Therefore, the wal compaction event won't be replayed.
58. **body:** It seems that although very unlikely this can still happen: HStore Compaction: 1. write compaction file 2. Append Compaction to WAL (new sequenceId) 3. store.writeLock.lock() 4. modify store files 5. store.writeLock.unlock() 6. delete files HRegion flush: a. startSeqId = wal.startCacheFlush() -> this has to come after 2 b. for each store c. store.writeLock.lock() d. add store file e. store.writeLock.unlock() f. update completeSequenceId If we get an execution where steps 1 and 2 completed, then flush starts with steps a..f and completes before we make sure that the compaction files are deleted, then the compact entry might be skipped on replay. It seems we cannot extend the HStore write lock to guard against this easily. I was not able to find a solution that is not ugly for this.
    label: code-design
59. Integrated in hbase-0.95 #178 (See [https://builds.apache.org/job/hbase-0.95/178/]) HBASE-8478 HBASE-2231 breaks TestHRegion#testRecoveredEditsReplayCompaction under hadoop2 profile (Revision 1479717) Result = FAILURE enis : Files : * /hbase/branches/0.95/hbase-server/src/main/java/org/apache/hadoop/hbase/util/FSUtils.java * /hbase/branches/0.95/hbase-server/src/test/java/org/apache/hadoop/hbase/regionserver/TestHRegion.java
60. Integrated in HBase-TRUNK #4099 (See [https://builds.apache.org/job/HBase-TRUNK/4099/]) HBASE-8478 HBASE-2231 breaks TestHRegion#testRecoveredEditsReplayCompaction under hadoop2 profile (Revision 1479716) Result = SUCCESS enis : Files : * /hbase/trunk/hbase-server/src/main/java/org/apache/hadoop/hbase/util/FSUtils.java * /hbase/trunk/hbase-server/src/test/java/org/apache/hadoop/hbase/regionserver/TestHRegion.java
61. Integrated in hbase-0.95-on-hadoop2 #89 (See [https://builds.apache.org/job/hbase-0.95-on-hadoop2/89/]) HBASE-8478 HBASE-2231 breaks TestHRegion#testRecoveredEditsReplayCompaction under hadoop2 profile (Revision 1479717) Result = FAILURE enis : Files : * /hbase/branches/0.95/hbase-server/src/main/java/org/apache/hadoop/hbase/util/FSUtils.java * /hbase/branches/0.95/hbase-server/src/test/java/org/apache/hadoop/hbase/regionserver/TestHRegion.java
62. Integrated in HBase-TRUNK-on-Hadoop-2.0.0 #518 (See [https://builds.apache.org/job/HBase-TRUNK-on-Hadoop-2.0.0/518/]) HBASE-8478 HBASE-2231 breaks TestHRegion#testRecoveredEditsReplayCompaction under hadoop2 profile (Revision 1479716) Result = FAILURE enis : Files : * /hbase/trunk/hbase-server/src/main/java/org/apache/hadoop/hbase/util/FSUtils.java * /hbase/trunk/hbase-server/src/test/java/org/apache/hadoop/hbase/regionserver/TestHRegion.java
63. **body:** The new INFO logging seems kinda chatty (and it shows up on multiple lines): {noformat} 2013-05-23 11:38:12,955 INFO [RegionServer:0;172.21.3.117,57009,1369334164639-smallCompactions-1369334211720] org.apache.hadoop.hbase.regionserver.wal.HLogUtil: Appended compaction marker tableName: "TestTable" encodedRegionName: "338447e35d9c2b5b0a38e982c08cbf09" familyName: "info" compactionInput: "515e42ec2ef746fa86a5ca7acbc58aed" compactionInput: "8abc6ae8c4754484b2628a4f4909342c" compactionInput: "303b7f1767b24408a55813142a5618c8"

compactionInput: "0f93806c7ecf43a487f0c5dcc2de911f" compactionOutput: "1bff72a0b8664a0b8072049a1899e908" storeHomeDir: "info" {noformat} I'm not sure what the value is. **label:** code-design

64. Hi, Prakash Khemani is no longer at Facebook so this email address is no longer being monitored. If you need assistance, please contact another person who is currently at the company.
65. bq. The new INFO logging seems kinda chatty Yeah, I also noticed that today. Let me do an addendum to this to change to debug.
66. How about this addendum?
67. Trace level it on commit? +1
68. Here is what I applied (hope you don't mind Enis). It makes the logging of compaction append trace level.
69. Np at all.
70. Integrated in hbase-0.95 #212 (See [https://builds.apache.org/job/hbase-0.95/212/]) HBASE-2231 Compaction events should be written to HLog; LOGGING ADDENDUM (Revision 1485874) Result = FAILURE stack : Files : * /hbase/branches/0.95/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/wal/HLogUtil.java
71. Integrated in hbase-0.95-on-hadoop2 #110 (See [https://builds.apache.org/job/hbase-0.95-on-hadoop2/110/]) HBASE-2231 Compaction events should be written to HLog; LOGGING ADDENDUM (Revision 1485874) Result = FAILURE stack : Files : * /hbase/branches/0.95/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/wal/HLogUtil.java
72. Integrated in HBase-TRUNK-on-Hadoop-2.0.0 #541 (See [https://builds.apache.org/job/HBase-TRUNK-on-Hadoop-2.0.0/541/]) HBASE-2231 Compaction events should be written to HLog; LOGGING ADDENDUM (Revision 1485873) Result = FAILURE stack : Files : * /hbase/trunk/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/wal/HLogUtil.java
73. Integrated in HBase-TRUNK #4141 (See [https://builds.apache.org/job/HBase-TRUNK/4141/]) HBASE-2231 Compaction events should be written to HLog; LOGGING ADDENDUM (Revision 1485873) Result = SUCCESS stack : Files : * /hbase/trunk/hbase-server/src/main/java/org/apache/hadoop/hbase/regionserver/wal/HLogUtil.java