Item 19
git_comments:

1. run a get and see?
2. Expected.

git_commits:

1. **summary:** HBASE-4016 HRegion.incrementColumnValue() doesn't have a consistent behavior when the field that we are incrementing is less than 8 bytes long
   message: HBASE-4016 HRegion.incrementColumnValue() doesn't have a consistent behavior when the field that we are incrementing is less than 8 bytes long git-svn-id: https://svn.apache.org/repos/asf/hbase/trunk@1140907 13f79535-47bb-0310-9956-ffa450edef68

github_issues:

github_issues_comments:

github_pulls:

github_pulls_comments:

github_pulls_reviews:

jira_issues:

1. **summary:** HRegion.incrementColumnValue() doesn't have a consistent behavior when the field that we are incrementing is less than 8 bytes long
   description: We wanted to use an int (32-bit) atomic counter and we initialize it with a certain value when the row is created. Later, we increment the counter using HTable.incrementColumnValue(). This call results in one of two outcomes. 1. The call succeeds, but the column value now is a long (64-bit) and is corrupt (by additional data that was in the buffer read). 2. Throws IOException/IllegalArgumentException. Java.io.IOException: java.io.IOException: java.lang.IllegalArgumentException: offset (65547) + length (8) exceed the capacity of the array: 65551 at org.apache.hadoop.hbase.util.Bytes.explainWrongLengthOrOffset(Bytes.java:502) at org.apache.hadoop.hbase.util.Bytes.toLong(Bytes.java:480) at org.apache.hadoop.hbase.regionserver.HRegion.incrementColumnValue(HRegion.java:3139) at org.apache.hadoop.hbase.regionserver.HRegionServer.incrementColumnValue(HRegionServer.java:2468) at sun.reflect.GeneratedMethodAccessor24.invoke(Unknown Source) at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25) at java.lang.reflect.Method.invoke(Method.java:597) at org.apache.hadoop.hbase.ipc.HBaseRPC$Server.call(HBaseRPC.java:570) at org.apache.hadoop.hbase.ipc.HBaseServer$Handler.run(HBaseServer.java:1039) Based on our incorrect usage of counters (initializing it with a 32 bit value and later using it as a counter), I would expect that we fail consistently with mode 2 rather than silently corrupting data with mode 1. However, the exception is thrown only rarely and I am not sure what determines the case to be executed. I am wondering if this has something to do with flush. Here is a HRegion unit test that can reproduce this problem. http://paste.lisp.org/display/122822 We modified our code to initialize the counter with a 64 bit value. But, I was also wondering if something has to change in HRegion.incrementColumnValue() to handle inconsistent counter sizes gracefully without corrupting existing data. Please let me know if you need additional information.

jira_issues_comments:

1. **body:** Your code is storing strings in the cells, but incrementColumnValue expects a big-endian encoded long, not a string.
   label: code-design
2. Todd, I reviewed this with Praveen before he created this jira and his point is that the error reporting seems buggy when trying to increment a cell that doesn't contain a long, not that a string or int cannot be incremented.

3. **body:** The cell that is being incremented, (row1,fam1,qual1) store an int (4 bytes) not a string. I think you are confusing the row key with the cell value. There was a minor issue with the old patch and I have revised it to fix it. The new one is available at http://paste.lisp.org/+2MRQ/1 I am reopening the issue. **label:** code-design
4. Fixed this. Btw - Praveen, your test is broken. even if you comment out l long result; try { result = region.incrementColumnValue(row1, fam1, qual1, 1, true); fail("Expected to fail here"); } catch (Exception exception) { // Expected. }, the assertICV's still fail.
5. checks length of stored value before incrementing.
6. also fixed Praveen's tests.
7. Applied to TRUNK. Thanks for the patch Li Pi.
8. Integrated in HBase-TRUNK #1996 (See [https://builds.apache.org/job/HBase-TRUNK/1996/]) HBASE-4016 HRegion.incrementColumnValue() doesn't have a consistent behavior when the field that we are incrementing is less than 8 bytes long stack : Files : * /hbase/trunk/src/main/java/org/apache/hadoop/hbase/regionserver/HRegion.java * /hbase/trunk/CHANGES.txt * /hbase/trunk/src/test/java/org/apache/hadoop/hbase/regionserver/TestHRegion.java * /hbase/trunk/src/main/java/org/apache/hadoop/hbase/regionserver/HRegionServer.java
9. This issue was closed as part of a bulk closing operation on 2015-11-20. All issues that have been resolved and where all fixVersions have been released have been closed (following discussions on the mailing list).