HADOOP-15171

sershe

2018-01-13

Contents

Chapter 1

Root issue HADOOP-15171

1.1 Summary

native ZLIB decompressor produces 0 bytes on the 2nd call; also incorrrectly handles some zlib errors

1.2 Description

While reading some ORC file via direct buffers, Hive gets a 0-sized buffer for a particular compressed segment of the file. We narrowed it down to Hadoop native ZLIB codec; when the data is copied to heap-based buffer and the JDK Inflater is used, it produces correct output. Input is only 127 bytes so I can paste it here.
All the other (many) blocks of the file are decompressed without problems by the same code.

```
2018-01-13T02:47:40,815 TRACE [IO-Elevator-Thread-0 (1515637158315_0079_1_00_000000_0)]
encoded.EncodedReaderImpl: Decompressing 127 bytes to dest buffer pos 524288,
limit 786432
2018-01-13T02:47:40,816  WARN [IO-Elevator-Thread-0 (1515637158315_0079_1_00_000000_0)]
encoded.EncodedReaderImpl: The codec has produced 0 bytes for 127 bytes
at pos 0, data hash 1719565039: [e3 92 e1 62 66 60 60 10 12 e5 98 e0 27
c4 c7 f1 e8 12 8f 40 c3 7b 5e 89 09 7f 6e 74 73 04 30 70 c9 72 b1 30 14
4d 60 82 49 37 bd e7 15 58 d0 cd 2f 31 a1 a1 e3 35 4c fa 15 a3 02 4c 7a
51 37 bf c0 81 e5 02 12 13 5a b6 9f e2 04 ea 96 e3 62 65 b8 c3 b4 01 ae
fd d0 72 01 81 07 87 05 25 26 74 3c 5b c9 05 35 fd 0a b3 03 50 7b 83 11
c8 f2 c3 82 02 0f 96 0b 49 34 7c fa ff 9f 2d 80 01 00
2018-01-13T02:47:40,816  WARN [IO-Elevator-Thread-0 (1515637158315_0079_1_00_000000_0)]
encoded.EncodedReaderImpl: Fell back to JDK decompressor with memcopy; got
155 bytes
```

Hadoop version is based on 3.1 snapshot.
The size of libhadoop.so is 824403 bytes, and libgplcompression is 78273 FWIW. Not sure how to extract versions from those.

1.3 Attachments

No attachments

1.4 Comments

1. **sershe:** [~jnp] [~hagleitn] fyi
2. **stevel@apache.org:** Irrespective of the fix, sounds like the hadoop decompressor code should do a followup check that the #of bytes returned is non-zero
3. **sershe:** [~stevel@apache.org] [~jnp] is it possible to get some traction on this actually? We now also have to work around this in ORC project, and this is becoming a pain
4. **gopalv:** bq. this is becoming a pain

   This is a huge perf hit right now, the workaround is much slower than the original codepath.
5. **stevel@apache.org:** There was another JIRA on this wasn't there? Sergei, can you find it?
6. **sershe:** Tentative cause (still confirming) - calling end() on ZlibDirectDecompressor breaks other unrelated ZlibDirectDecompressor-s. So it may not be related to buffers as such.
7. **sershe:** Hmm, nm, it might be a red herring
8. **sershe:** Ok, here's repro code. I get the bar exception if dd2 is added. Note that dd2 is not used for anything and is not related in any way to dd1.

   If I instead end dd1 and then reuse it, I get a NPE in Java code. But if I end dd2, internals of Java object are not affected in dd1; looks like the native side has some issue.

```
    dest.position(startPos);

    dest.limit(startLim);

    src.position(startSrcPos);

    src.limit(startSrcLim);



    ZlibDecompressor.ZlibDirectDecompressor dd1 = new ZlibDecompressor.ZlibDirectDe
0);

    dd1.decompress(src, dest);

    dest.limit(dest.position()); // Set the new limit to where the decompressor
stopped.

    dest.position(startPos);
    if (dest.remaining() == 0) {

      throw new RuntimeException("foo");

    }



    ZlibDecompressor.ZlibDirectDecompressor dd2 = new ZlibDecompressor.ZlibDirectDe
0);

    dest.position(startPos);

    dest.limit(startLim);

    src.position(startSrcPos);

    src.limit(startSrcLim);

    dd2.end();

    dd1.decompress(src, dest);

    dest.limit(dest.position()); // Set the new limit to where the decompressor
stopped.

    dest.position(startPos);

    if (dest.remaining() == 0) {

      throw new RuntimeException("bar");

    }
```

As a side note, Z_BUF_ERROR error in the native code is not processed correctly. See the detailed example for this error here http://zlib.net/zlib_how.html ; given that neither the Java nor native code handles partial reads; and nothing propagates the state to the caller, this should throw an error just like Z_DATA_ERROR.

The buffer address null checks should probably also throw and not exit silently.

Z_NEED_DICT handling is also suspicious. Does anything actually handle this?

9. **sershe:** Update: turns out, end() was a red herring after all - any reuse of the same object without calling reset causes the issue.

Given that the object does not support the zlib library model of repeatedly calling inflate with more data, it basically never makes sense to call decompress without calling reset.

Perhaps the call should be built in? I cannot find whether zlib itself actually requires one to reset (at least, for the continuous decompression case, it doesn't look like it's the case), so perhaps cleanup could be improved too.

At any rate, error handling should be fixed to not return 0.

10. **Michael South:** Issue should be closed, unfounded. The Hive Orc driver creates a decompression object and repeatedly calling it to deflate Orc blocks. Its treating each block as an entirely separate chunk (stream), completely decompressing each with one call to ...{{_inflateBytesDirect()}}. However, it wasn't calling {{inflateReset()}} or {{inflateEnd()}} / {{inflateInit()}} between the streams, which naturally left things in a confused state. It appears to be fixed in trunk Hive.

    Also, returning 0 for {{Z_BUF_ERROR}} or {{Z_NEED_DICT}} is correct, and should not throw an error. The Java decompression object is agnostic as to whether the application is working in stream or all-at-once mode. The only determination of which mode is active is whether the application (Hive Orc driver in this case) is passing the entire input in one chunk and is allocating sufficient space for all of the output. Therefore, the application must check for a zero return. If no-progress (zero return) is an impossible situation then it can throw an exception; otherwise it needs to look at one or more of ...{{_finished()}}, ...{{_getRemaining()}}, and/or ...{{_needDict()}} to figure out what's needed to make further progress. (It would be nice if JNI exposed the {{avail_out}} field, but if it's not an input or dictionary issue it must be a full output buffer.)

    There *is* a very minor bug in ...{{inflateBytesDirect()}}. It's calling {{inflate()}} with {{Z_PARTIAL_FLUSH}}, which only applies to {{deflate()}}. It should be {{Z_NO_FLUSH}}. However, in the current zlib code (1.2.11) the {{flush}} parameter only affects the return code, and it only checks whether or not it is {{Z_FINISH}}.

    Edit: The Zlib docs (overall, very excellent) kind of assume you realize that the internals, allocated in ...{{init()}}, are only valid for one stream run. The docs *do* say that ...{{end()}} deallocates these internal buffers; therefore to reuse the base compressor / decompressor you need to call ...{{init()}} again to re-allocate them (otherwise NPE). The docs also state that ...{{reset()}} is equivalent to calling ...{{end()}} followed by ...{{init()}}, only by resetting the internals and not deallocating and reallocating them.

11. **Michael South:** Apology: A raw "closed, unfounded" is too brusque. [~sershe] did a really excellent job analyzing the bug, creating a minimal testcase, and identifying where the issue is. I can't imagine how many hours he spent plowing through the Orc and Zlib codebases and rerunning tests.

Chapter 2

Connected issue HIVE-18452

2.1 Summary

work around HADOOP-15171

2.2 Description

2.3 Attachments

1. HIVE-18452.patch

2.4 Comments

1. **sershe:** This also adds a safety flag to turn off codec pool. I suspected the pool was at fault, and since it could easily have bugs it's good to have a safety option.
2. **sershe:** [~prasanth_j] can you take a look?
3. **hiveqa:** | (x) *{color:red}-1 overall{color}* |

   \\

   \\

   || Vote || Subsystem || Runtime || Comment ||

   || || || || {color:brown} Prechecks {color} ||

   | {color:blue}0{color} | {color:blue} findbugs {color} | {color:blue} 0m 1s{color} | {color:blue} Findbugs executables are not available. {color} |

   | {color:green}+1{color} | {color:green} @author {color} | {color:green} 0m 0s{color} | {color:green} The patch does not contain any @author tags. {color} |

   || || || || {color:brown} master Compile Tests {color} ||

   | {color:blue}0{color} | {color:blue} mvndep {color} | {color:blue} 1m 25s{color} | {color:blue} Maven dependency ordering for branch {color} |

| {color:green}+1{color} | {color:green} mvninstall {color} | {color:green} 5m 21s{color} | {color:green} master passed {color} |

| {color:green}+1{color} | {color:green} compile {color} | {color:green} 1m 32s{color} | {color:green} master passed {color} |

| {color:green}+1{color} | {color:green} checkstyle {color} | {color:green} 1m 2s{color} | {color:green} master passed {color} |

| {color:green}+1{color} | {color:green} javadoc {color} | {color:green} 1m 16s{color} | {color:green} master passed {color} |

|| || || || {color:brown} Patch Compile Tests {color} ||

| {color:blue}0{color} | {color:blue} mvndep {color} | {color:blue} 0m 21s{color} | {color:blue} Maven dependency ordering for patch {color} |

| {color:green}+1{color} | {color:green} mvninstall {color} | {color:green} 1m 54s{color} | {color:green} the patch passed {color} |

| {color:green}+1{color} | {color:green} compile {color} | {color:green} 1m 30s{color} | {color:green} the patch passed {color} |

| {color:green}+1{color} | {color:green} javac {color} | {color:green} 1m 30s{color} | {color:green} the patch passed {color} |

| {color:red}-1{color} | {color:red} checkstyle {color} | {color:red} 0m 31s{color} | {color:red} ql: The patch generated 8 new + 128 unchanged - 1 fixed = 136 total (was 129) {color} |

| {color:red}-1{color} | {color:red} checkstyle {color} | {color:red} 0m 12s{color} | {color:red} llap-server: The patch generated 1 new + 38 unchanged - 1 fixed = 39 total (was 39) {color} |

| {color:green}+1{color} | {color:green} whitespace {color} | {color:green} 0m 0s{color} | {color:green} The patch has no whitespace issues. {color} |

| {color:green}+1{color} | {color:green} javadoc {color} | {color:green} 1m 15s{color} | {color:green} the patch passed {color} |

|| || || || {color:brown} Other Tests {color} ||

| {color:green}+1{color} | {color:green} asflicense {color} | {color:green} 0m 11s{color} | {color:green} The patch does not generate ASF License warnings. {color} |

| {color:black}{color} | {color:black} {color} | {color:black} 17m 17s{color} | {color:black} {color} |

\\

\\

|| Subsystem || Report/Notes ||

| Optional Tests | asflicense javac javadoc findbugs checkstyle compile |

| uname | Linux hiveptest-server-upstream 3.16.0-4-amd64 #1 SMP Debian 3.16.36-1+deb8u1 (2016-09-03) x86_64 GNU/Linux |

| Build tool | maven |

| Personality | /data/hiveptest/working/yetus/dev-support/hive-personality.sh |

| git revision | master / b1cdbc6 |

| Default Java | 1.8.0_111 |

| checkstyle | http://104.198.109.242/logs//PreCommit-HIVE-Build-8610/yetus/diff-checkstyle-ql.txt |

| checkstyle | http://104.198.109.242/logs//PreCommit-HIVE-Build-8610/yetus/diff-checkstyle-llap-server.txt |

| modules | C: common ql llap-server U: . |

| Console output | http://104.198.109.242/logs//PreCommit-HIVE-Build-8610/yetus.txt |

| Powered by | Apache Yetus http://yetus.apache.org |

This message was automatically generated.

4. **hiveqa:**

   Here are the results of testing the latest attachment:
   https://issues.apache.org/jira/secure/attachment/12905979/HIVE-18452.patch

   {color:red}ERROR:{color} -1 due to no test(s) being added or modified.

   {color:red}ERROR:{color} -1 due to 25 failed/errored test(s), 11558 tests executed
   Failed tests:

```
org.apache.hadoop.hive.cli.TestCliDriver.testCliDriver[materialized_view_create_rew
(batchId=16)
org.apache.hadoop.hive.cli.TestCliDriver.testCliDriver[ppd_join5] (batchId=35)
org.apache.hadoop.hive.cli.TestCliDriver.testCliDriver[ppd_windowing2]
(batchId=11)
org.apache.hadoop.hive.cli.TestMiniLlapCliDriver.testCliDriver[llap_smb]
(batchId=151)
org.apache.hadoop.hive.cli.TestMiniLlapLocalCliDriver.testCliDriver[bucket_map_join
(batchId=170)
org.apache.hadoop.hive.cli.TestMiniLlapLocalCliDriver.testCliDriver[bucketsortoptim
```

```
(batchId=152)
org.apache.hadoop.hive.cli.TestMiniLlapLocalCliDriver.testCliDriver[hybridgrace_has
(batchId=157)
org.apache.hadoop.hive.cli.TestMiniLlapLocalCliDriver.testCliDriver[insert_values_o
(batchId=165)
org.apache.hadoop.hive.cli.TestMiniLlapLocalCliDriver.testCliDriver[llap_acid]
(batchId=169)
org.apache.hadoop.hive.cli.TestMiniLlapLocalCliDriver.testCliDriver[llap_acid_fast]
(batchId=160)
org.apache.hadoop.hive.cli.TestMiniLlapLocalCliDriver.testCliDriver[mergejoin]
(batchId=165)
org.apache.hadoop.hive.cli.TestMiniLlapLocalCliDriver.testCliDriver[sysdb]
(batchId=160)
org.apache.hadoop.hive.cli.TestMiniSparkOnYarnCliDriver.testCliDriver[bucketizedhiv
(batchId=178)
org.apache.hadoop.hive.cli.TestNegativeCliDriver.testCliDriver[authorization_part]
(batchId=94)
org.apache.hadoop.hive.cli.TestNegativeCliDriver.testCliDriver[materialized_view_no
(batchId=94)
org.apache.hadoop.hive.cli.TestNegativeCliDriver.testCliDriver[materialized_view_no
(batchId=94)
org.apache.hadoop.hive.cli.TestNegativeCliDriver.testCliDriver[stats_aggregator_err
(batchId=94)
org.apache.hadoop.hive.cli.TestSparkCliDriver.testCliDriver[ppd_join5]
(batchId=121)
org.apache.hadoop.hive.metastore.TestEmbeddedHiveMetaStore.testTransactionalValidat
(batchId=214)
org.apache.hadoop.hive.ql.io.TestDruidRecordWriter.testWrite (batchId=254)
org.apache.hadoop.hive.ql.lockmgr.TestDbTxnManager2.testMergeUnpartitioned01
(batchId=291)
org.apache.hadoop.hive.ql.parse.TestReplicationScenarios.testConstraints
(batchId=226)
org.apache.hive.jdbc.TestSSL.testConnectionMismatch (batchId=232)
org.apache.hive.jdbc.TestSSL.testConnectionWrongCertCN (batchId=232)
org.apache.hive.jdbc.TestSSL.testMetastoreConnectionWrongCertCN (batchId=232)
```

Test results: https://builds.apache.org/job/PreCommit-HIVE-Build/8610/testReport
Console output: https://builds.apache.org/job/PreCommit-HIVE-Build/8610/console
Test logs: http://104.198.109.242/logs/PreCommit-HIVE-Build-8610/

Messages:

```
Executing org.apache.hive.ptest.execution.TestCheckPhase
Executing org.apache.hive.ptest.execution.PrepPhase
Executing org.apache.hive.ptest.execution.YetusPhase
Executing org.apache.hive.ptest.execution.ExecutionPhase
Executing org.apache.hive.ptest.execution.ReportingPhase
Tests exited with: TestsFailedException: 25 tests failed
```

This message is automatically generated.

ATTACHMENT ID: 12905979 - PreCommit-HIVE-Build

5. **ashutoshc:** +1
6. **sershe:** Committed to master. Thanks for the review!
7. **ashutoshc:** This jira is resolved and released with Hive 3.0 If you find an issue with it, please create a new jira.