

Public signup for this instance is **disabled**. Our [Jira Guidelines page](#) explains how to get an account.



Hadoop Common / HADOOP-3707

## Frequent DiskOutOfSpaceException on almost-full datanodes

Details

Type:	Bug	Status:	<span>CLOSED</span>
Priority:	Blocker	Resolution:	Fixed
Affects Version/s:	0.17.0	Fix Version/s:	0.17.2
Component/s:	None		
Labels:	None		
Release Note:	NameNode keeps a count of number of blocks scheduled to be written to a datanode and uses it to avoid allocating more blocks than a datanode can hold.		

Description

On a datanode which is completely full (leaving reserve space), we frequently see target node reporting,

```
2008-07-07 16:54:44,707 INFO org.apache.hadoop.dfs.DataNode: Receiving block blk_3328886742742952100 src: /11.1.11.111:22222 de
2008-07-07 16:54:44,708 INFO org.apache.hadoop.dfs.DataNode: writeBlock blk_3328886742742952100 received exception org.apache.hi
2008-07-07 16:54:44,708 ERROR org.apache.hadoop.dfs.DataNode: 33.3.33.33:22222:DataXceiver: org.apache.hadoop.util.DiskChecker$
    at org.apache.hadoop.dfs.FSDataset$FSVolumeSet.getNextVolume(FSDataset.java:444)
    at org.apache.hadoop.dfs.FSDataset.writeToBlock(FSDataset.java:716)
    at org.apache.hadoop.dfs.DataNode$BlockReceiver.<init>(DataNode.java:2187)
    at org.apache.hadoop.dfs.DataNode$DataXceiver.writeBlock(DataNode.java:1113)
    at org.apache.hadoop.dfs.DataNode$DataXceiver.run(DataNode.java:976)
    at java.lang.Thread.run(Thread.java:619)
```






Sender reporting

```
2008-07-07 16:54:44,712 INFO org.apache.hadoop.dfs.DataNode: 11.1.11.111:22222:Exception writing block blk_3328886742742952100
java.io.IOException: Broken pipe
    at sun.nio.ch.FileDispatcher.write0(Native Method)
    at sun.nio.ch.SocketDispatcher.write(SocketDispatcher.java:29)
    at sun.nio.ch.IOUtil.writeFromNativeBuffer(IOUtil.java:104)
    at sun.nio.ch.IOUtil.write(IOUtil.java:75)
    at sun.nio.ch.SocketChannelImpl.write(SocketChannelImpl.java:334)
    at org.apache.hadoop.net.SocketOutputStream$Writer.performIO(SocketOutputStream.java:53)
    at org.apache.hadoop.net.SocketIOWithTimeout.doIO(SocketIOWithTimeout.java:140)
    at org.apache.hadoop.net.SocketOutputStream.write(SocketOutputStream.java:144)
    at org.apache.hadoop.net.SocketOutputStream.write(SocketOutputStream.java:105)
    at java.io.BufferedOutputStream.flushBuffer(BufferedOutputStream.java:65)
    at java.io.BufferedOutputStream.write(BufferedOutputStream.java:109)
    at java.io.DataOutputStream.write(DataOutputStream.java:90)
    at org.apache.hadoop.dfs.DataNode$BlockReceiver.receiveChunk(DataNode.java:2292)
    at org.apache.hadoop.dfs.DataNode$BlockReceiver.receivePacket(DataNode.java:2411)
    at org.apache.hadoop.dfs.DataNode$BlockReceiver.receiveBlock(DataNode.java:2476)
    at org.apache.hadoop.dfs.DataNode$DataXceiver.writeBlock(DataNode.java:1204)
    at org.apache.hadoop.dfs.DataNode$DataXceiver.run(DataNode.java:976)
    at java.lang.Thread.run(Thread.java:619)
```

Since it's not constantly happening, my guess is whenever datanode gets some small space available, namenode over-assigns blocks which can fail the block pipeline.  
(Note, before 0.17, namenode was much slower in assigning blocks)

Attachments


<a href="#">HADOOP-3707-branch-017.patch</a>	8 kB	10/Jul/08 23:57
<a href="#">HADOOP-3707-branch-017.patch</a>	8 kB	09/Jul/08 19:13
<a href="#">HADOOP-3707-branch-017.patch</a>	5 kB	08/Jul/08 23:55
<a href="#">HADOOP-3707-branch-018.patch</a>	8 kB	15/Jul/08 00:44

 <a href="#">HADOOP-3707-trunk.patch</a>	8 kB	15/Jul/08 00:48
 <a href="#">HADOOP-3707-trunk.patch</a>	8 kB	14/Jul/08 22:01
 <a href="#">HADOOP-3707-trunk.patch</a>	8 kB	10/Jul/08 19:08
 <a href="#">HADOOP-3707-trunk.patch</a>	5 kB	10/Jul/08 03:51
 <a href="#">HADOOP-3707-trunk.patch</a>	8 kB	09/Jul/08 22:44


▼ Activity



 [Koji Noguchi](#) created issue - 07/Jul/08 17:21

▼  [Hairong Kuang](#) added a comment - 07/Jul/08 17:40

I believe that this is triggered by much faster block replication scheduling introduced by [HADOOP-2696](#). Currently when the namenode choosing a replication target, it only looks at the remaining disk space provided by a datanode through the previous heartbeat but do not look at the space that will be taken by the blocks that have been scheduled to replicate to this datanode. If the previous heartbeat says it has space for 5 blocks but the namenode can schedule as many as 50 blocks to this datanode, the overassigned 45 blocks will fail with the DiskOutOfSpaceException. It looks that the namenode should look at the scheduled blocks as well when choosing a replication target.


▼  [Raghu Angadi](#) added a comment - 07/Jul/08 17:57

Yes. Considering load scheduled also avoids related problems like [HADOOP-3633](#) and [HADOOP-3685](#).


▼  [Raghu Angadi](#) added a comment - 08/Jul/08 05:59 - edited

Proposal that Hairong and I discussed : (this seems safe enough for 0.17 and 0.18) :


- Each datanode maintains a counter `approxBlocksSheduled`.
  - incremented each time a block is scheduled to a datanode
  - decremented when datanode receives 'block received' message from a datanode.
  - No list of block ids is maintained.
  - Not every block scheduled will eventually receive a 'block received' message. it will be corrected over time, as described later below.
- disk space left on datanode will be `(freespace_reported_in_heartbeat - (approxBlocksScheduled+prevApproxBlockScheduled)*defaultBlockSize)`
- 'approx' in the name of the variable is deliberate since it is not expected to be very accurate. It will be handled like this :
  - another variable 'prevApproxBlocksScheduled' is maintained.
  - Every 5 minutes or so, value of 'prev' will be ignored. 'prev' will be set to current value and current will be set to zero.
  - So if there are some blocks that are not reported back by the datanode, they will eventually get adjusted (usually 10 min; bit longer if datanode is continuously receiving blocks).
  - Its not an error if NameNode receives 'block received' message and this counter is zero.
- This count will also be useful for throttling number of blocks scheduled for replication.. (may be the limit could be something large like 50 or 100).

 [Raghu Angadi](#) made changes - 08/Jul/08 20:59

Field	Original Value	New Value
Assignee		<a href="#">Raghu Angadi</a> [ <a href="#">rangadi</a> ]


 [Raghu Angadi](#) made changes - 08/Jul/08 23:53

Fix Version/s	0.18.0 [ <a href="#">12312972</a> ]
Fix Version/s	0.17.2 [ <a href="#">12313296</a> ]
Fix Version/s	0.19.0 [ <a href="#">12313211</a> ]
Priority	Major [ <a href="#">3</a> ]
	Blocker [ <a href="#">1</a> ]

▼  [Raghu Angadi](#) added a comment - 08/Jul/08 23:55

patch for branch-0.17 is attached. It seems to work well.


---

 Raghu Angadi made changes - 08/Jul/08 23:55

Attachment


[HADOOP-3707-branch-017.patch](#) [ 12385552 ]

---

▼  Raghu Angadi added a comment - 09/Jul/08 19:13

Updated patch adds a test case.


---

 Raghu Angadi made changes - 09/Jul/08 19:13

Attachment


[HADOOP-3707-branch-017.patch](#) [ 12385663 ]

---

▼  Hairong Kuang added a comment - 09/Jul/08 21:45


Raghu, could you please upload a patch for the trunk?

---

▼  Raghu Angadi added a comment - 09/Jul/08 22:44

Patch for trunk is attached.


---

 Raghu Angadi made changes - 09/Jul/08 22:44

Attachment

[HADOOP-3707-trunk.patch](#) [ 12385682 ]


---

 Raghu Angadi made changes - 10/Jul/08 03:51

Attachment

[HADOOP-3707-trunk.patch](#) [ 12385698 ]

---

 Raghu Angadi made changes - 10/Jul/08 19:08

Attachment

[HADOOP-3707-trunk.patch](#) [ 12385795 ]

---

▼  Konstantin Shvachko added a comment - 10/Jul/08 21:57


This is a nice idea, but I am not happy that

- it is rather complex,
- it is very approximate,
- it is not localized in one place of the code, and therefore hard to maintain.

One disadvantage of the approach is that the name-node forgets about scheduled replication after a while (10 min?).

Which means that if you do a lot of slow writes or replications then you can schedule a lot for one node then forget about it and then schedule the same amount again.

1. I think you don't need the approximation, and hence can remove `prevApproxBlockScheduled`, because we know exactly when to increment and when to decrement the scheduling counters.
    - increment when block is scheduled for replication and when a new block is allocated, this can be in one common place at `chooseTargets()`.
    - decrement when `blockReceived()` and when the block moves from `pendingReplications` back to `neededReplications`.  
Need to find one common place for the call.
  2. May be we can solve this using simpler things. Like
    - slow down the replication scheduler.
    - use bigger values for `dfs.datanode.du.pct` or `dfs.datanode.du.reserved`.
- 

▼  Raghu Angadi added a comment - 10/Jul/08 22:10

> I think you don't need the approximation,

I don't see how it can possibly be accurate without having a lot more code and more memory to maintain more state in `NameNode`. For e.g. say `NameNode` returns 3 datanodes to a client to write to, and then client immediately dies. If there are no such errors, then it is accurate.

increment when block is scheduled for replication and when a new block is allocated, this can be in one common place at `chooseTargets()`.

chooseTarget() is not always called to schedule writes. And blocks allocated may not be used by the caller of chooseTargets(). It calls in two places instead of one.. as an alternative, we could have a common method that is invoked whenever NN asks a DN or a client to write to a DN.


> May be we can solve this using simpler things. Like

May be. Though I don't see why NameNode can't do this itself.. like this patch.

> slow down the replication scheduler.

I think slowing down activities is pretty defensive. Also replication is not the only cause: should we slow down client writes too?

I don't mean to say this is the best solution. Is there a better solution that is comparably simple and safe for 0.17 and 0.18?

▼  Raghu Angadi added a comment - 10/Jul/08 22:37

Advantages of the current patch :

- fixes a real problem observed by users and increases robustness of DFS.
- is certainly an improvement over what we have.
- has no regressions.
- does not slow down any NameNode activity.
- in my opinion does not increase or decrease the complexity or change the nature of the big beast "FSNamesystem".
- once it works well, the counter can be used for other scheduling activities.
- I don't think "approx" in the name should distract much.. it is as accurate as it can be.. and we deal with small departures from accuracy in case of errors. It is only guilty of living with uncertainty 😊.

Of course we can change the patch, for e.g. we can increase the "roll interval" from 5 minutes.

▼  Raghu Angadi added a comment - 10/Jul/08 23:57

Attaching patch for 0.17.2. It has two cosmetic changes compared to earlier 0.17 patch. Diff between the patches :


```
- private final int BLOCKS_SCHEDULED_Roll_INTERVAL_MSEC = 300 * 1000; // 5 min
+ private static final int BLOCKS_SCHEDULED_ROLL_INTERVAL = 300 * 1000; // 5 min
...
-     BLOCKS_SCHEDULED_Roll_INTERVAL_MSEC) {
+     BLOCKS_SCHEDULED_ROLL_INTERVAL) {
...

```

Konstantin,


for 0.18 and trunk, I am looking into if the counter could be decremented when a block moves from neededReplications to pendingReplications and when the lease expires.. which makes the counter more accurate and we could increase the "roll interval" to something like 1 hour.

If you think patch for branch 0.17 is good enough (i.e. it does not make anything worse), could you +1 it for 0.17? 0.17.2 release is waiting only on this patch.


 Raghu Angadi made changes - 10/Jul/08 23:57

Attachment

[HADOOP-3707-branch-017.patch](#) [ 12385814 ]

▼  Raghu Angadi added a comment - 11/Jul/08 00:19

'ant test-patch' and 'ant test-core' succeeded for 0.17.


 Raghu Angadi made changes - 14/Jul/08 18:08

Fix Version/s


0.17.2 [ 12313296 ]

Release Note

committed this patch to 0.17. I will temporarily remove 0.17 from the 'fix versions' so that it does not appear as an unresolved blocker for 0.17.


▼  Raghu Angadi added a comment - 14/Jul/08 18:08

committed this patch to 0.17. I will temporarily remove 0.17 from the 'fix versions' so that it does not appear as an unresolved blocker for 0.17.

 Raghu Angadi made changes - 14/Jul/08 18:08


Release Note

committed this patch to 0.17. I will temporarily remove 0.17 from the 'fix versions' so that it does not appear as an unresolved blocker for 0.17.

 [Raghu Angadi](#) made changes - 14/Jul/08 22:01



Attachment

[HADOOP-3707-trunk.patch](#) [ 12386017 ]

 [Raghu Angadi](#) made changes - 15/Jul/08 00:44

Attachment

[HADOOP-3707-branch-018.patch](#) [ 12386026 ]


  [Raghu Angadi](#) added a comment - 15/Jul/08 00:48 - edited

Attached patches for 0.18 and trunk.

Increased the 'roll time' to 10 minutes from 5 minutes.


As Konstantin suggested, we could improve the count by updating the count when a block moves from 'pendingReplications' to 'neededReplications' and when a file is abandoned. I am not sure yet sure of a clean patch for that. For now we could file another jira to improve this. I think these improvements will make the counter more dependable and might be used in more scheduling decisions.

I hope the attached patches fix this jira and improve current block allocation.

 [Raghu Angadi](#) made changes - 15/Jul/08 00:48

Attachment



[HADOOP-3707-trunk.patch](#) [ 12386028 ]

 [Raghu Angadi](#) made changes - 15/Jul/08 00:48

Status

Open [ 1 ]

Patch Available [ 10002 ]

  [Hadoop QA](#) added a comment - 16/Jul/08 06:07

+1 overall. Here are the results of testing the latest attachment

<http://issues.apache.org/jira/secure/attachment/12386028/HADOOP-3707-trunk.patch>  
against trunk revision 677127.

+1 @author. The patch does not contain any @author tags.

+1 tests included. The patch appears to include 4 new or modified tests.

+1 javadoc. The javadoc tool did not generate any warning messages.

+1 javac. The applied patch does not increase the total number of javac compiler warnings.

+1 findbugs. The patch does not introduce any new Findbugs warnings.

+1 release audit. The applied patch does not increase the total number of release audit warnings.

+1 core tests. The patch passed core unit tests.

+1 contrib tests. The patch passed contrib unit tests.



Test results: <http://hudson.zones.apache.org/hudson/job/Hadoop-Patch/2867/testReport/>

Findbugs warnings: <http://hudson.zones.apache.org/hudson/job/Hadoop-Patch/2867/artifact/trunk/build/test/findbugs/newPatchFindbugsWarnings.html>


Checkstyle results: <http://hudson.zones.apache.org/hudson/job/Hadoop-Patch/2867/artifact/trunk/build/test/checkstyle-errors.html>

Console output: <http://hudson.zones.apache.org/hudson/job/Hadoop-Patch/2867/console>

This message is automatically generated.

  [Raghu Angadi](#) added a comment - 16/Jul/08 19:15

I just committed this.

 [Raghu Angadi](#) made changes - 16/Jul/08 19:15

Fix Version/s

0.17.2 [ 12313296 ]

Release Note

NameNode keeps a count of number of blocks scheduled to be written to a datanode and uses it to

avoid allocating more blocks than a datanode can hold.


Resolution

Status

Patch Available [ 10002 ]


Fixed [ 1 ]

Resolved [ 5 ]

 Owen O'Malley made changes - 18/Aug/08 23:19


Fix Version/s0.18.0 [ 12312972 ]

Fix Version/s0.19.0 [ 12313211 ]


 Owen O'Malley made changes - 18/Aug/08 23:19

StatusResolved [ 5 ]




Closed [ 6 ]

 Hudson added a comment - 22/Aug/08 12:34

Integrated in Hadoop-trunk #581 (See <http://hudson.zones.apache.org/hudson/job/Hadoop-trunk/581/>)


 Owen O'Malley made changes - 08/Jul/09 16:43

Component/sdfs [ 12310710 ]


Transition	Time In Source Status	Execution Times
 Raghu Angadi made transition - 15/Jul/08 00:48 <div>OPEN → PATCH AVAILABLE</div>	7d 7h 27m	1
 Raghu Angadi made transition - 16/Jul/08 19:15 <div>PATCH AVAILABLE → RESOLVED</div>	1d 18h 26m	1
 Owen O'Malley made transition - 18/Aug/08 23:19 <div>RESOLVED → CLOSED</div>	33d 4h 4m	1

People


Assignee:

 Raghu Angadi


Reporter:

 Koji Noguchi

Votes:

 0 Vote for this issue

Watchers:

 0 Start watching this issue

Dates

Created:

07/Jul/08 17:21

Updated:

08/Jul/09 16:43

Resolved:

16/Jul/08 19:15