Hadoop元数据合并异常及解决方法

这几天观察了一下Standby NN上面的日志，发现每次Fsimage合并完之后，Standby NN通知Active NN来下载合并好的Fsimage的过程中会出现以下的异常信息：

```
2014-04-23 14:42:54,964 ERROR org.apache.hadoop.hdfs.server.namenode.ha.
   StandbyCheckpointer: Exception in doCheckpoint
java.net.SocketTimeoutException: Read timed out
        at java.net.SocketInputStream.socketRead0(Native Method)
        at java.net.SocketInputStream.read(SocketInputStream.java:152)
        at java.net.SocketInputStream.read(SocketInputStream.java:122)
        at java.io.BufferedInputStream.fill(BufferedInputStream.java:235)
        at java.io.BufferedInputStream.read1(BufferedInputStream.java:275)
        at java.io.BufferedInputStream.read(BufferedInputStream.java:334)
        at sun.net.www.http.HttpClient.parseHTTPHeader(HttpClient.java:687)
        at sun.net.www.http.HttpClient.parseHTTP(HttpClient.java:633)
        at sun.net.www.protocol.http.HttpURLConnection.getInputStream
(HttpURLConnection.java:1323)
        at java.net.HttpURLConnection.getResponseCode(HttpURLConnection.java:468)
        at org.apache.hadoop.hdfs.server.namenode.TransferFsImage.doGetUrl
(TransferFsImage.java:268)
        at org.apache.hadoop.hdfs.server.namenode.TransferFsImage.getFileClient
(TransferFsImage.java:247)
        at org.apache.hadoop.hdfs.server.namenode.TransferFsImage.
uploadImageFromStorage(TransferFsImage.java:162)
        at org.apache.hadoop.hdfs.server.namenode.ha.StandbyCheckpointer.
doCheckpoint(StandbyCheckpointer.java:174)
        at org.apache.hadoop.hdfs.server.namenode.ha.StandbyCheckpointer.
access$1100(StandbyCheckpointer.java:53)
        at org.apache.hadoop.hdfs.server.namenode.ha.StandbyCheckpointer
$CheckpointerThread.doWork(StandbyCheckpointer.java:297)
        at org.apache.hadoop.hdfs.server.namenode.ha.StandbyCheckpointer
$CheckpointerThread.access$300(StandbyCheckpointer.java:210)
        at org.apache.hadoop.hdfs.server.namenode.ha.StandbyCheckpointer
$CheckpointerThread$1.run(StandbyCheckpointer.java:230)
        at org.apache.hadoop.security.SecurityUtil.doAsLoginUserOrFatal
(SecurityUtil.java:456)
        at org.apache.hadoop.hdfs.server.namenode.ha.StandbyCheckpointer
$CheckpointerThread.run(StandbyCheckpointer.java:226)
```

上面的代码贴出来有点乱啊，可以看下下面的图片截图：

StandbyCheckpointer

于是习惯性的去Google了一下，找了好久也没找到类似的信息。只能自己解决。我们通过分析日志发现更奇怪的问题，上次Checkpoint的时间一直都不变（一直都是Standby NN启动的时候第一次Checkpoint的时间），如下：

```
2014-04-23 14:50:54,429 INFO
org.apache.hadoop.hdfs.server.namenode.ha.StandbyCheckpointer: Triggering
checkpoint because it has been 70164 seconds since the last checkpoint,
which exceeds the configured interval 600
```

难道这是Hadoop的bug？于是我就根据上面的错误信息去查看源码，经过仔细的分析，发现上述的问题都是由StandbyCheckpointer类输出的：

```
private void doWork() {
    // Reset checkpoint time so that we don't always checkpoint
    // on startup.
    lastCheckpointTime = now();
    while (shouldRun) {
      try {
        Thread.sleep(1000 * checkpointConf.getCheckPeriod());
      } catch (InterruptedException ie) {
      }
      if (!shouldRun) {
        break;
      }
      try {
        // We may have lost our ticket since last checkpoint, log in again,
        // just in case
        if (UserGroupInformation.isSecurityEnabled()) {
          UserGroupInformation.getCurrentUser().checkTGTAndReloginFromKeytab();
        }

        long now = now();
        long uncheckpointed = countUncheckpointedTxns();
        long secsSinceLast = (now - lastCheckpointTime)/1000;
```

```
        boolean needCheckpoint = false;
        if (uncheckpointed >= checkpointConf.getTxnCount()) {
          LOG.info("Triggering checkpoint because there have been " +
              uncheckpointed + " txns since the last checkpoint, which " +
              "exceeds the configured threshold " +
              checkpointConf.getTxnCount());
          needCheckpoint = true;
        } else if (secsSinceLast >= checkpointConf.getPeriod()) {
          LOG.info("Triggering checkpoint because it has been " +
              secsSinceLast + " seconds since the last checkpoint, which " +
              "exceeds the configured interval " + checkpointConf.getPeriod());
          needCheckpoint = true;
        }

        synchronized (cancelLock) {
          if (now < preventCheckpointsUntil) {
            LOG.info("But skipping this checkpoint since we are about"+
            " to failover!");
            canceledCount++;
            continue;
          }
          assert canceler == null;
          canceler = new Canceler();
        }

        if (needCheckpoint) {
          doCheckpoint();
          lastCheckpointTime = now;
        }
      } catch (SaveNamespaceCancelledException ce) {
        LOG.info("Checkpoint was cancelled: " + ce.getMessage());
        canceledCount++;
      } catch (InterruptedException ie) {
        // Probably requested shutdown.
        continue;
      } catch (Throwable t) {
        LOG.error("Exception in doCheckpoint", t);
      } finally {
        synchronized (cancelLock) {
          canceler = null;
        }
      }
    }
  }
}
```

上面的异常信息是由doCheckpoint()函数执行的过程中出现问题而抛出来的，这样导致lastCheckpointTime = now;语句永远执行不到。那么为什么doCheckpoint()执行过程会出现异常？？根据上述堆栈信息的跟踪，发现是由TransferFsImage类的doGetUrl函数中的下面语句导致的：

```
if (connection.getResponseCode() != HttpURLConnection.HTTP_OK) {
```

由于connection无法得到对方的响应码而超时。于是我就想到是否是我的集群socket超时设置的有问题？？后来经过各种分析发现不是。于是我只能再看看代码，我发现了上述代码的前面有如下设置：

```
 if (timeout <= 0) {
     Configuration conf = new HdfsConfiguration();
     timeout = conf.getInt(DFSConfigKeys.DFS_IMAGE_TRANSFER_TIMEOUT_KEY,
         DFSConfigKeys.DFS_IMAGE_TRANSFER_TIMEOUT_DEFAULT);
}

if (timeout > 0) {
     connection.setConnectTimeout(timeout);
     connection.setReadTimeout(timeout);
}

if (connection.getResponseCode() != HttpURLConnection.HTTP_OK) {
     throw new HttpGetFailedException(
         "Image transfer servlet at " + url +
         " failed with status code " + connection.getResponseCode() +
         "₩nResponse message:₩n" + connection.getResponseMessage(),
         connection);
}
```

DFS_IMAGE_TRANSFER_TIMEOUT_KEY这个时间是由dfs.image.transfer.timeout参数所设置的，默认值为10 * 60 * 1000，单位为毫秒。然后我看了一下这个属性的解释：

Timeout for image transfer in milliseconds. This timeout and the related dfs.image.transfer.bandwidthPerSec parameter should be configured such that normal image transfer can complete within the timeout. This timeout prevents client hangs when the sender

fails during image transfer, which is particularly important during checkpointing. Note that this timeout applies to the entirety of image transfer, and is not a socket timeout.

这才发现问题，这个参数的设置和dfs.image.transfer.bandwidthPerSec息息相关，要保证Active NN在dfs.image.transfer.timeout时间内把合并好的Fsimage从Standby NN上下载完，要不然会出现异常。然后我看了一下我的配置

```
<property>
        <name>dfs.image.transfer.timeout</name>
        <value>60000</value>
</property>

<property>
        <name>dfs.image.transfer.bandwidthPerSec</name>
        <value>1048576</value>
</property>
```

60秒超时，一秒钟拷贝1MB，而我的集群上的元数据有800多MB，显然是不能在60秒钟拷贝完，后来我把dfs.image.transfer.timeout设置大了，观察了一下，集群再也没出现过上述异常信息，而且以前的一些异常信息也由于这个而解决了。。