

HDFS 常见问题维护手册 V1.0

HDFS 常见问题维护手册 V1.0

文档版本 01
发布日期 2016-03-31

华为技术有限公司



版权所有 © 华为技术有限公司 2016。 保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

商标声明



和其他华为商标均为华为技术有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或默示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

华为技术有限公司

地址： 深圳市龙岗区坂田华为总部办公楼 邮编：518129

网址： <http://www.huawei.com>

客户服务邮箱： support@huawei.com

客户服务电话： 4008302118

目 录

HDFS.....	4
1、基本概念.....	4
【HDFS 基本架构】	4
【客户端工具】	5
hdfs dfs	5
hdfs dfsadmin.....	7
hdfs fsck.....	9
【监控项介绍】	10
HDFS WEB UI 上的 block 块信息监控.....	10
HDFS WEB UI 上的 datanode 节点信息监控.....	11
HDFS WEB UI 上的 hadoop 文件系统信息监控	12
【日志获取】	12
客户端日志.....	12
服务器端日志.....	12
2、常见问题.....	15
【启动异常】	15
[HDFS-10001]元数据丢失导致 NameNode 启动失败	15
[HDFS-10002]Editlog 损坏导致 NameNode 启动失败	16
[HDFS-10003]账号锁定导致启动组件失败.....	17
[HDFS-10004]目录没权限导致启动组件失败.....	18
【状态异常】	19
[HDFS-20001]datanode 实例状态为 concerning	19
[HDFS-20002]HDFS 服务的健康状态为 bad.....	20
[HDFS-20003]namenode 进入到 safemode	21
[HDFS-20004]NameNode 经常主备倒换	22
[HDFS-20005]磁盘空间不足，进行容量扩容，采用挂载 NAS 方式替换原有 磁盘，NFS 服务问题导致 Datanode 异常	23
[HDFS-20006]datanode 概率性出现 CPU 占用接近 1000%，导致节点丢失 （ssh 连得很慢或者不上）	24
【读写文件异常相关】	25
[HDFS-30001]文件最大打开句柄数设置太小，导致文件句柄不足.....	25
[HDFS-30002]客户端写文件 close 失败	26
[HDFS-30003]文件错误导致上传文件到 HDFS 失败	28
[HDFS-30004]网络丢包/错包导致 HDFS 写大文件失败.....	29

[HDFS-30005]HDFS 并发写操作过多造成写失败	30
[HDFS-30006]写 hdfs 文件在 close 遇到异常，再次执行 append 显示文件 lease 被老客户端占据，无法写入数据。	32
[HDFS-30007]界面配置 dfs.blocksize，将其设置为 268435456，put 数据， block 大小还是原来的大小	33
【SHELL 操作客户端相关】	33
[HDFS-40001]执行 HDFS SHELL 命令失败	33
[HDFS-40002]HDFS shell 命令使用失败.....	34
【数据分布不均衡】	35
[HDFS-50001]非 HDFS 数据残留导致数据分布不均衡	35
[HDFS-50002]客户端安装在数据节点导致数据分布不均衡.....	35
[HDFS-50003]MR 任务导致数据分布不均衡	36
【咨询类】	37
[HDFS-60001]Hdfs balance 调优参数设置.....	37
[HDFS-60002]文件副本设置方法.....	37
[HDFS-60003]HDFS 容量规划	38
[HDFS-60004]内存参数设置	39

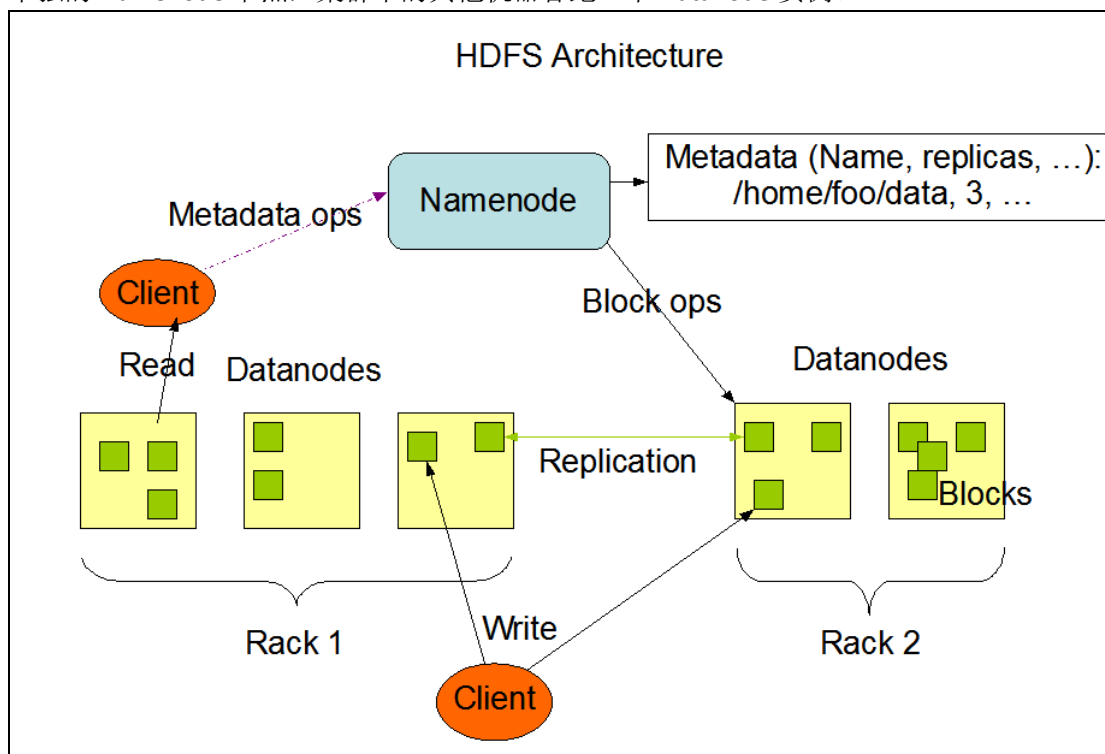
HDFS

1、基本概念

【HDFS 基本架构】

Hadoop 分布式文件系统（HDFS）是一个分布式的文件系统，运行在廉价的硬件上。它与现有的分布式文件系统有很多相似之处。然而与其他的分布式文件系统的差异也是显着的。HDFS 是高容错的，被设计成在低成本硬件上部署。HDFS 为应用数据提供高吞吐量的访问，适用于具有大规模数据集的应用程序。

HDFS 采用 master/slave 架构。一个 HDFS 集群是有一个 Namenode 和一定数目的 Datanode 组成。Namenode 是一个中心服务器，负责管理文件系统的 namespace 和客户端对文件的访问。Datanode 在集群中一般是一个节点一个，负责管理节点上它们附带的存储。在内部，一个文件其实分成一个或多个 block，这些 block 存储在 Datanode 集合里。Namenode 执行文件系统的 namespace 操作，例如打开、关闭、重命名文件和目录，同时决定 block 到具体 Datanode 节点的映射。Datanode 在 Namenode 的指挥下进行 block 的创建、删除和复制。Namenode 和 Datanode 都是设计成可以跑在普通的廉价的运行 linux 的机器上。HDFS 采用 java 语言开发，因此可以部署在很大范围的机器上。一个典型的部署场景是一台机器跑一个单独的 Namenode 节点，集群中的其他机器各跑一个 Datanode 实例。



【客户端工具】

HDFS 提供如下命令：

```
linux-suse-123:~/client # hdfs
Usage: hdfs [--config confdir] COMMAND
    where COMMAND is one of:
    dfs                run a filesystem command on the file systems supported in Hadoop.
    namenode -format    format the DFS filesystem
    secondarynamenode   run the DFS secondary namenode
    namenode            run the DFS namenode
    journalnode         run the DFS journalnode
    zkfc               run the ZK Failover Controller daemon
    datanode            run a DFS datanode
    dfsadmin           run a DFS admin client
    haadmin            run a DFS HA admin client
    colocationadmin     run a DFS Colocation Admin client
    fsck              run a DFS filesystem checking utility
    balancer          run a cluster balancing utility
    jmxget            get JMX exported values from NameNode or DataNode.
    oiv              apply the offline fsimage viewer to an fsimage
    oiv_legacy       apply the offline fsimage viewer to an legacy fsimage
    oev              apply the offline edits viewer to an edits file
    fetchdt          fetch a delegation token from the NameNode
    getconf           get config values from configuration
    groups            get the groups which users belong to
    snapshotDiff      diff two snapshots of a directory or diff the
                      current directory contents with a snapshot
    lsSnapshottableDir list all snapshottable dirs owned by the current user
                      Use -help to see options
    portmap           run a portmap service
    nfs3              run an NFS version 3 gateway
    cacheadmin        configure the HDFS cache

Most commands print help when invoked w/o parameters.
```

其中和运维关系较大的命令字有：dfs、dfsadmin、haadmin、fsck、balancer，本章将对上述命令字做相关介绍。

hdfs dfs

hdfs dfs 命令是用来操作 HDFS 中的文件的，其包含如下参数：

```

linux-suse-123:~/client # hdfs dfs
No GC_PROFILE is given. Defaults to medium.
Usage: hadoop fs [generic options]
    [-appendToFile <localsrc> ... <dst>]
    [-cat [-ignoreCrc] <src> ...]
    [-checksum <src> ...]
    [-chgrp [-R] GROUP PATH...]
    [-chmod [-R] <MODE[,MODE]... | OCTALMODE> PATH...]
    [-chown [-R] [OWNER][:[GROUP]] PATH...]
    [-copyFromLocal [-f] [-p] <localsrc> ... <dst>]
    [-copyToLocal [-p] [-ignoreCrc] [-crc] <src> ... <localdst>]
    [-count [-q] <path> ...]
    [-cp [-f] [-p] <src> ... <dst>]
    [-createSnapshot <snapshotDir> [<snapshotName>]]
    [-deleteSnapshot <snapshotDir> <snapshotName>]
    [-df [-h] [<path> ...]]
    [-du [-s] [-h] <path> ...]
    [-expunge]
    [-get [-p] [-ignoreCrc] [-crc] <src> ... <localdst>]
    [-getfacl [-R] <path>]
    [-getmerge [-nl] <src> <localdst>]
    [-help [cmd ...]]
    [-ls [-d] [-h] [-R] [<path> ...]]
    [-mkdir [-p] <path> ...]
    [-moveFromLocal <localsrc> ... <dst>]
    [-moveToLocal <src> <localdst>]
    [-mv <src> ... <dst>]
    [-put [-f] [-p] <localsrc> ... <dst>]
    [-renameSnapshot <snapshotDir> <oldName> <newName>]
    [-rm [-f] [-r|-R] [-skipTrash] <src> ...]
    [-rmdir [--ignore-fail-on-non-empty] <dir> ...]
    [-setfacl [-R] [{-b|-k} {-m|-x <acl_spec>} <path>]|[--set <acl_spec> <path>]]
    [-setrep [-R] [-w] <rep> <path> ...]
    [-stat [format] <path> ...]
    [-tail [-f] <file>]
    [-test [-defsz] <path>]
    [-text [-ignoreCrc] <src> ...]
    [-touchz <path> ...]
    [-usage [cmd ...]]

```

其中比较常用的命令有：

ls

使用方法：hdfs dfs -ls <args>

如果是文件，则按照如下格式返回文件信息：

权限 <副本数> 用户 ID 组 ID 文件大小 修改日期 修改时间 文件名

如果是目录，则返回它直接子文件的一个列表，就像在 Unix 中一样。目录返回列表的信息如下：

权限 <-> 用户 ID 组 ID 0 修改日期 修改时间 文件名

示例：hdfs dfs -ls /tmp

```

linux-suse-123:~/client # hdfs dfs -ls /
No GC_PROFILE is given. Defaults to medium.
15/06/04 20:04:11 INFO hdfs.PeerCache: SocketCache disabled.
Found 10 items
-rw-r--r--   3 hdfs   supergroup    5578859 2015-06-02 10:40 /BigData_Addon.tgz
-rw-r--r--   3 hdfs   supergroup         0 2015-04-15 11:12 /PRE_CREATE_DIR.SUCCESS
-rw-r-----  3 hdfs   hadoop      222685267 2015-01-06 09:38 /apache-flume-server.tar.gz
drwxr-x---   - flume   hadoop         0 2015-04-15 11:09 /flume
drwx-----   - hbase   hadoop         0 2015-04-15 11:09 /hbase
drwxrwxrwx   - mapred  hadoop         0 2015-04-15 11:09 /mr-history
drwxrwxrwx   - spark   supergroup     0 2015-06-03 20:16 /sparkJobHistory
drwxr-xr-x   - hdfs   supergroup     0 2015-06-04 10:03 /system
drwxrwxrwx   - hdfs   hadoop         0 2015-06-04 11:36 /tmp
drwxrwxrwx   - hdfs   hadoop         0 2015-04-15 11:14 /user

```

返回值：成功返回 0，失败返回-1。

du

使用方法：hdfs dfs -du URI [URI ...]

显示目录中所有文件的大小，或者当只指定一个文件时，显示此文件的大小。

示例：hdfs dfs -du -h /

```
linux-suse-123:~/client # hdfs dfs -du -h /  
No GC_PROFILE is given. Defaults to medium.  
15/06/04 20:23:47 INFO hdfs.PeerCache: SocketCache disabled.  
5.3 M      /BigData_Addon.tgz  
0          /PRE_CREATE_DIR.SUCCESS  
212.4 M    /apache-flume-server.tar.gz  
0          /flume  
0          /hbase  
0          /mr-history  
179        /sparkJobHistory  
0          /system  
13.7 G     /tmp  
0          /user
```

返回值：成功返回 0，失败返回-1。

expunge

使用方法：hdfs dfs -expunge

清空回收站。如果 HDFS 中可用空间意外地缩小了，并且不知道被什么文件占用了，那么很可能是被回收站占用了；回收站的数据默认是 24 小时后删除，在确认回收站的数据无效之后可以通过使用该命令将回收站清空，回收 HDFS 空间。

rm

使用方法：hdfs dfs -rm [-f] [-r|-R] [-skipTrash] <src> ...

删除指定的文件或者文件。如果删除目录，需要指定[-r]参数。如果递归删除，需要指定[-R]参数。如果直接删除文件而不将文件放入回收站，需要指定[-skipTrash]参数。

示例：hdfs dfs -rm /user/hadoop/emptydir/file1

返回值：成功返回 0，失败返回-1。

hdfs dfsadmin

hdfs dfsadmin 命令是 HDFS 中的管理命令，通过该命令可以对 HDFS 进行管理操作：


```
linux-suse-123:~/client # hdfs dfsadmin
No GC_PROFILE is given. Defaults to medium.
Usage: java DFSAdmin
Note: Administrative commands can only be run as the HDFS superuser.
    [-report]
    [-safemode enter | leave | get | wait]
    [-allowSnapshot <snapshotDir>]
    [-disallowSnapshot <snapshotDir>]
    [-saveNamespace]
    [-rollEdits]
    [-restoreFailedStorage true|false|check]
    [-refreshNodes]
    [-finalizeUpgrade]
    [-rollingUpgrade [<query|prepare|finalize>]]
    [-metasave filename]
    [-refreshServiceAcl]
    [-refreshUserToGroupsMappings]
    [-refreshSuperUserGroupsConfiguration]
    [-refreshCallQueue]
    [-printTopology]
    [-refreshNamenodes datanodehost:port]
    [-deleteBlockPool datanode-host:port blockpoolId [force]]
    [-setQuota <quota> <dirname>...<dirname>]
    [-clrQuota <dirname>...<dirname>]
    [-setSpaceQuota <quota> <dirname>...<dirname>]
    [-clrSpaceQuota <dirname>...<dirname>]
    [-setBalancerBandwidth <bandwidth in bytes per second>]
    [-fetchImage <local directory>]
    [-shutdownDatanode <datanode_host:ipc_port> [upgrade]]
    [-getDatanodeInfo <datanode_host:ipc_port>]
    [-help [cmd]]
```

以下将对比较常见的命令做重点介绍：

Report

使用方法：hdfs dfsadmin -report

该命令将获取 HDFS 整体健康概况和每个 DataNode 节点的使用概况。举例说明：

```

linux-suse-123:~/client # hdfs dfsadmin -report
No GC_PROFILE is given. Defaults to medium.
15/06/04 21:02:36 INFO hdfs.PeerCache: SocketCache disabled.
Configured Capacity: 95015632896 (88.49 GB)
Present Capacity: 94474309632 (87.99 GB)
DFS Remaining: 49200480256 (45.82 GB)
DFS Used: 45273829376 (42.16 GB)
DFS Used%: 47.92%
Under replicated blocks: 0
Blocks with corrupt replicas: 0
Missing blocks: 0

-----
Datanodes available: 3 (3 total, 0 dead)

Live datanodes:
Name: 192.188.45.123:25009 (linux-suse-123)
Hostname: linux-suse-123
Rack: /default/rack0
Decommission Status : Normal
Configured Capacity: 31671877632 (29.50 GB)
DFS Used: 15090585600 (14.05 GB)
Non DFS Used: 180441088 (172.08 MB)
DFS Remaining: 16400850944 (15.27 GB)
DFS Used%: 47.65%
DFS Remaining%: 51.78%
Configured Cache Capacity: 0 (0 B)
Cache Used: 0 (0 B)
Cache Remaining: 0 (0 B)

```

该命令首先列出 HDFS 整体的健康状况，例如，DFS Used 表示当前 HDFS 存储空间已经使用了多少，Block with corrupt replices 表示当前不正常的副本数(包括丢失的和错误的副本)，Missing block 表示当前丢失的块有多少，该值不为 0 表示可能存在文件损坏。通过这个值可以大致地了解当前 HDFS 的监控状况。

随后是列出 HDFS 中的每个 DataNode 的状况，在上述显示中当前正常工作的 DataNode 节点有 3 个，Dead DataNode 节点有 0 个。表示每一个 DataNode 正在正常工作。针对每一个 DataNode，列出其当前的状态，其中主要是空间使用量。

通过上述命令可以简单地了解系统的运行状况。

hdfs fsck

使用方法: hdfs fsck <path> [-list-corruptfileblocks | [-move | -delete | -openforwrite] [-files [-blocks [-locations | -racks]]]]

<path> : fsck 文件检查路径

-move : 将损坏的文件移动到/lost+found

-delete : 删除损坏的文件

-openforwrite : 显示正在打开写操作的文件

-files : 显示文件信息

-list-corruptfileblocks : 显示丢失块信息列表及所属文件信息
-blocks : 显示 block 块信息
-locations: 显示每个 block 块的位置
-racks : 显示 datanode 的机架信息

该命令可以检查整个文件系统的健康状况。例如，检查是否有 block 丢失，文件是否有损坏，并且可以删除损坏的文件。如果检查的 HDFS 数据量很大，该命令会对 namenode 造成较大压力，建议谨慎使用，尽量选择在业务压力较小的时候执行。

示例：`hdfs fsck /user/hadoop/emptydir/file1 -files -blocks -locations`

【监控项介绍】

HDFS WEB UI 上的 block 块信息监控

在 HDFS WEB UI 上的 Overview 页面：

查看 HDFS 启动时间、版本信息、编译时间、集群 ID 等信息：

Hadoop	Overview	Datanodes	Snapshot	Startup Progress	Utilities -
Overview '51-196-26-23:25000' (active)					
Started:	Wed Jun 03 16:43:25 CST 2015				
Version:	V100R001C00, rb5bbc284ee145a51ea0e13ab132a9e6891a3ed24				
Compiled:	13 Dec 2014 22:16:41 by datasight from Hadoop2.4_Dev				
Cluster ID:	myhacluster				
Block Pool ID:	BP-888144158-51.196.26.21-1433216445454				

查看是否进入到 safemode 模式、文件和目录数、block 块数、内存、HDFS 容量、节点状态等信息：

Summary

Security is on.

Safemode is off.

71 files and directories, 17 blocks = 88 total filesystem object(s).

Heap Memory used 207.81 MB of 1017.63 MB Heap Memory. Max Heap Memory is 1.99 GB.

Non Heap Memory used 45.31 MB of 132.94 MB Committed Non Heap Memory. Max Non Heap Memory is 176 MB.

Configured Capacity:	2.88 TB
DFS Used:	41.34 MB
Non DFS Used:	155.5 GB
DFS Remaining:	2.73 TB
DFS Used%:	0%
DFS Remaining%:	94.73%
Block Pool Used:	41.34 MB
Block Pool Used%:	0%
DataNodes usages% (Min/Median/Max/stdDev):	0.00% / 0.00% / 0.00% / 0.00%
Live Nodes	3 (Decommissioned: 0)
Dead Nodes	0 (Decommissioned: 0)
Decommissioning Nodes	0
Number of Under-Replicated Blocks	0
Number of Blocks Pending Deletion	0

HDFS WEB UI 上的 datanode 节点信息监控

在 HDFS WEB UI 上的 Datanodes 页面:

查看每个 datanode 节点状态信息、容量信息、磁盘使用情况、block 块数、坏磁盘数等。

Hadoop	Overview	Datanodes	Snapshot	Startup Progress	Utilities
--------	----------	-----------	----------	------------------	-----------

Datanode Information

In operation

Node	Last contact	Admin State	Capacity	Used	Non DFS Used	Remaining	Blocks	Block pool used	Failed Volumes	Version
51-196-26-23 (51.196.26.23:25009)	1	In Service	984.31 GB	13.78 MB	50.21 GB	934.09 GB	13	13.78 MB (0%)	0	V100R001C00
51-196-26-22 (51.196.26.22:25009)	1	In Service	984.31 GB	13.78 MB	52.48 GB	931.81 GB	13	13.78 MB (0%)	0	V100R001C00
51-196-26-21 (51.196.26.21:25009)	1	In Service	984.31 GB	13.78 MB	52.81 GB	931.48 GB	13	13.78 MB (0%)	0	V100R001C00

Decommissioning

Node	Last contact	Under replicated blocks	Blocks with no live replicas	Under Replicated Blocks In files under construction
------	--------------	-------------------------	------------------------------	--

HDFS WEB UI 上的 hadoop 文件系统信息监控

在 HDFS WEB UI 上点击 Utilities 选择 Browse the file system，查看 hadoop 文件系统信息：

Browse Directory

/

Go!

Permission	Owner	Group	Size	Replication	Block Size	Name
-rw-r--r--	hdfs	supergroup	0 B	3	128 MB	PRE_CREATE_DIR.SUCCESS
drwxr-x---	hdfs	hadoop	0 B	0	0 B	encryption
drwxr-x---	flume	hadoop	0 B	0	0 B	flume
drwx-----	hbase	hadoop	0 B	0	0 B	hbase
drwxrwxrwx	mapred	hadoop	0 B	0	0 B	mr-history
drwxrwxrwx	hdfs	hadoop	0 B	0	0 B	tmp
drwxrwxrwx	hdfs	hadoop	0 B	0	0 B	user

【日志获取】

HDFS 定位日志主要包括客户端日志和服务端日志：

客户端日志

开启客户端 debug 日志：
在 客 户 端 安 装 路 径 `/HDFS/hadoop/etc/hadoop/hadoop-env.sh` 添 加 `export HADOOP_ROOT_LOGGER=DEBUG,RFA`
默认存放路径：
`客户端安装路径/HDFS/hadoop/logs/hadoop.log`

服务器端日志

服务器端日志主要包括：dn、jn、nm、zkfc。
从 Manager WEB UI 获取服务端日志方式如下：


```

[root@51-196-26-21 Bigdata]# ll
total 88
drwx-----. 8 omm wheel 4096 Jun  3 14:14 audit
drwx-----. 5 omm wheel 4096 Jun  4 20:36 controller
drwx-----. 7 omm wheel 4096 Jun  2 11:37 dbservice
drwx-----. 3 omm wheel 4096 Jun  2 11:51 hbase
drwx-----. 6 omm wheel 4096 Jun  2 11:35 hdfs
drwx-----. 4 omm wheel 4096 Jun  3 14:14 hive
drwx-----. 2 omm wheel 4096 Jun  3 17:21 httpd
drwx-----. 2 omm wheel 4096 Jun  4 21:00 kerberos
drwxr-----. 2 omm wheel 4096 Jun  2 11:34 ldapclient
drwxr-----. 2 omm wheel 4096 Jun  3 16:59 ldapserver
drwxr-x---. 2 omm wheel 4096 Jun  2 11:00 logman
drwx-----. 6 omm wheel 4096 Jun  2 11:31 nodeagent
drwx-----. 2 omm wheel 4096 Jun  4 21:03 okerberos
drwxr-----. 2 omm wheel 4096 Jun  4 09:19 oldapserver
drwxr-x---. 5 omm wheel 4096 May 29 19:36 omm
drwxr-xr-x. 2 omm wheel 4096 Jun  3 16:49 patch
drwx-----. 2 omm wheel 4096 Jun  3 09:55 timestamp
drwx-----. 2 omm wheel 4096 Jun  4 09:19 tomcat
drwxrwx---. 2 omm wheel 4096 Jun  4 09:07 update
drwx-----. 2 omm wheel 4096 Jun  3 17:19 watchdog
drwx-----. 3 omm wheel 4096 Jun  2 11:35 yarn
drwx-----. 3 omm wheel 4096 Jun  2 11:39 zookeeper
[root@51-196-26-21 Bigdata]# cd hdfs/
[root@51-196-26-21 hdfs]# ll
total 16
drwx-----. 2 omm wheel 4096 Jun  3 16:43 dn
drwx-----. 2 omm wheel 4096 Jun  3 16:42 jn
drwx-----. 2 omm wheel 4096 Jun  4 02:34 nn
drwx-----. 2 omm wheel 4096 Jun  3 16:43 zkfc
[root@51-196-26-21 hdfs]# █

[root@51-196-26-21 hdfs]# cd dn/
[root@51-196-26-21 dn]# ll
total 6548
-rw-----. 1 omm wheel      836 Jun  2 11:35 cleanupDetail.log
-rw-----. 1 omm wheel  13446 Jun  4 20:52 datanode-omm-gc.log
-rw-----. 1 omm wheel      0 Jun  2 11:35 hadoop.log
-rw-----. 1 omm wheel 6651205 Jun  4 21:07 hadoop-omm-datanode-51-196-26-21.log
-rw-----. 1 omm wheel      65 Jun  3 16:43 hadoop-omm-datanode-51-196-26-21.out
-rw-----. 1 omm wheel      65 Jun  3 16:25 hadoop-omm-datanode-51-196-26-21.out.1
-rw-----. 1 omm wheel      65 Jun  3 14:48 hadoop-omm-datanode-51-196-26-21.out.2
-rw-----. 1 omm wheel      65 Jun  3 14:41 hadoop-omm-datanode-51-196-26-21.out.3
-rw-----. 1 omm wheel      65 Jun  2 11:41 hadoop-omm-datanode-51-196-26-21.out.4
-rw-----. 1 omm wheel     855 Jun  3 16:43 jsvc.err
-rw-----. 1 omm wheel     545 Jun  3 16:43 jsvc.out
-rw-----. 1 omm wheel     620 Jun  3 16:43 prestartDetail.log
[root@51-196-26-21 dn]# █

```

2、常见问题

【启动异常】

[HDFS-10001]元数据丢失导致 NameNode 启动失败

【问题背景与现象】

元数据丢失导致 NameNode 启动失败。

【原因分析】

1. 分析 namenode 日志（/var/log/Bigdata/hdfs/nn/hadoop-omm-namendoe-XXX.log）:

```
2018-08-22    13:23:38,550    ERROR    [main]    Exception    in    namenode    join
org.apache.hadoop.hdfs.server.namenode.NameNode.main(NameNode.java:1259)
org.apache.hadoop.hdfs.server.common.InconsistentFSStateException: Directory /srv/BigData/hadoop/data2/namenode/data is
in an inconsistent state: storage directory does not exist or is not accessible.
at org.apache.hadoop.hdfs.server.namenode.FSImage.recoverStorageDirs(FSImage.java:310)
at org.apache.hadoop.hdfs.server.namenode.FSImage.recoverTransitionRead(FSImage.java:221)
at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.loadFSImage(FSNamesystem.java:568)
at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.loadFromDisk(FSNamesystem.java:447)
at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.loadFromDisk(FSNamesystem.java:411)
at org.apache.hadoop.hdfs.server.namenode.NameNode.loadNamesystem(NameNode.java:393)
at org.apache.hadoop.hdfs.server.namenode.NameNode.initializeNamesystem(NameNode.java:441)
at org.apache.hadoop.hdfs.server.namenode.NameNode.initialize(NameNode.java:423)
at org.apache.hadoop.hdfs.server.namenode.NameNode.<init>(NameNode.java:615)
at org.apache.hadoop.hdfs.server.namenode.NameNode.<init>(NameNode.java:596)
at org.apache.hadoop.hdfs.server.namenode.NameNode.createNameNode(NameNode.java:1196)
at org.apache.hadoop.hdfs.server.namenode.NameNode.main(NameNode.java:1255)
```

2. 客户的两个 namenode，
dfs.name.dir 配置项值为/srv/BigData/hadoop/data2/namenode/data

其中正常节点该目录下存在数据文件：

```
IHADOOP-9:/srv/BigData/hadoop/data2/namenode/data # ll
drwxr-xr-x 2 omm wheel 36864 Nov 17 21:34 current
-rwxr-xr-x 1 omm wheel    0 Nov 18 11:26 in_use.lock
```

非正常节点该目录下为空：

```
IHADOOP-21:/srv/BigData/hadoop/data2/namenode/data #ll
```

3. 判断为非正常节点数据丢失，导致启动失败。

【解决办法】

1. 进入 namenode 正常节点(主机 9)的/srv/BigData/hadoop/data2/namenode/data/下，

-
- 将 current 目录和 in_use.lock 拷贝到非正常节点(主机 21)相同目录下。
 - 拷贝完后需要递归修改 21 节点上新拷贝文件的权限为 omm:wheel
 - 从页面重启 hdfs 服务

[HDFS-10002]Editlog 损坏导致 NameNode 启动失败

【问题背景与现象】

Editlog 损坏导致 NameNode 启动失败

【原因分析】

- 分析 namenode 日志 (/var/log/Bigdata/hdfs/nn/hadoop-omm-namendoe-XXX.log):

```
2015-07-07 19:23:16,974 | FATAL | main | Exception in namenode join |
org.apache.hadoop.hdfs.server.namenode.NameNode.main(NameNode.java:1453)
org.apache.hadoop.hdfs.server.namenode.EditLogInputException: Error replaying edit log at offset 0. Expected transaction
ID was 344074
at org.apache.hadoop.hdfs.server.namenode.FSEditLogLoader.loadEditRecords(FSEditLogLoader.java:193)
at org.apache.hadoop.hdfs.server.namenode.FSEditLogLoader.loadFSEdits(FSEditLogLoader.java:133)
at org.apache.hadoop.hdfs.server.namenode.FSImage.loadEdits(FSImage.java:805)
at org.apache.hadoop.hdfs.server.namenode.FSImage.loadFSImage(FSImage.java:665)
at org.apache.hadoop.hdfs.server.namenode.FSImage.recoverTransitionRead(FSImage.java:272)
at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.loadFSImage(FSNamesystem.java:893)
at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.loadFromDisk(FSNamesystem.java:640)
at org.apache.hadoop.hdfs.server.namenode.NameNode.loadNamesystem(NameNode.java:508)
at org.apache.hadoop.hdfs.server.namenode.NameNode.initialize(NameNode.java:564)
at org.apache.hadoop.hdfs.server.namenode.NameNode.<init>(NameNode.java:729)
at org.apache.hadoop.hdfs.server.namenode.NameNode.<init>(NameNode.java:713)
at org.apache.hadoop.hdfs.server.namenode.NameNode.createNameNode(NameNode.java:1375)
at org.apache.hadoop.hdfs.server.namenode.NameNode.main(NameNode.java:1447)
Caused by: org.apache.hadoop.hdfs.server.namenode.RedundantEditLogInputStream$PrematureEOFException: got
premature end-of-file at txid 343894; expected file to go up to 344093
at
org.apache.hadoop.hdfs.server.namenode.RedundantEditLogInputStream.nextOp(RedundantEditLogInputStream.java:194)
at org.apache.hadoop.hdfs.server.namenode.EditLogInputStream.readOp(EditLogInputStream.java:85)
at org.apache.hadoop.hdfs.server.namenode.EditLogInputStream.skipUntil(EditLogInputStream.java:151)
at
org.apache.hadoop.hdfs.server.namenode.RedundantEditLogInputStream.nextOp(RedundantEditLogInputStream.java:178)
at org.apache.hadoop.hdfs.server.namenode.EditLogInputStream.readOp(EditLogInputStream.java:85)
at org.apache.hadoop.hdfs.server.namenode.FSEditLogLoader.loadEditRecords(FSEditLogLoader.java:180)
... 12 more
```

- 根据日志分析发现是由于 editlog 损坏导致 namenode 启动失败。

【解决办法】

在 namenode 节点执行下面操作，重新生成 editlog 文件。

1. su - omm
2. cd /opt/huawei/Bigdata/etc/*_*_Namenode
3. cat ENV_VARS
4. 文件内容复制粘贴到文本编辑器中，将每一行增加 export，并且注意 HADOOP_OPTS 参数的值需增加引号，且需要去掉 "\"，例如：

```
HADOOP_OPTS="-DIgnoreReplayReqDetect
-Djava.security.auth.login.config=/opt/huawei/Bigdata/etc/4_28_NameNode/jaas.conf
-Dzookeeper.server.principal=zookeeper/hadoop.hadoop.com
-Djava.security.krb5.conf=/opt/huawei/Bigdata/etc/4_26_KerberosClient/kdc.conf"
```

5. export 上述文件中的各环境变量
6. 执行修复操作

```
cd /opt/huawei/Bigdata/DataSight_FM_BasePlatform_V100R001C00_Hadoop/hadoop/bin
./hdfs namenode -initializeSharedEdits
```

出现选项的时候，选择 Y

[HDFS-10003]账号锁定导致启动组件失败

【问题背景与现象】

新安装集群，启动备 NameNode 失败，启动 DataNode 失败。

启动备 NameNode 的时候，显示认证失败，导致启动失败。

```
/home/omm/kerberos/bin/kinit -k -t /opt/huawei/Bigdata/etc/2_15_NameNode/hdfs.keytab hdfs/
hadoop.hadoop.com -c /opt/huawei/Bigdata/etc/2_15_NameNode/11846 failed
export key tab file for hdfs/hadoop.hadoop.com failed.
export and check keytab file failed
, errMsg=]]] for NameNode#192.168.1.92@192-168-1-92.
[2015-07-11 02:34:33] RoleInstance started failure for ROLE[name: NameNode].
[2015-07-11 02:34:34] Failed to complete the instances start operation. Current operation entities:
[NameNode#192.168.1.92@192-168-1-92], Failure entites : [NameNode#192.168.1.92@192-168-1-92].
Operation Failed.
Failed to complete the instances start operation. Current operation entities: [NameNode#192.168.1.92@192-168-1-92], Failure
entites: [NameNode#192.168.1.92@192-168-1-92].
```

【原因分析】

1. 查看 kerberos 日志，/var/log/Bigdata/kerberos/krb5kdc.log，发现有集群外的 IP 使用 hdfs 用户连接，导致多次认证失败，导致 hdfs 账号被锁定。

```
Jul 11 02:49:16 192-168-1-91 krb5kdc[1863](info): AS_REQ (2 etypes {18 17}) 192.168.1.93: NEEDED_PREAUTH:
hdfs/hadoop.hadoop.com@HADOOP.COM for krbtgt/HADOOP.COM@HADOOP.COM, Additional pre-authentication
```

```
required
Jul 11 02:49:16 192-168-1-91 krb5kdc[1863](info): preauth (encrypted_timestamp) verify failure: Decrypt integrity check
failed
Jul 11 02:49:16 192-168-1-91 krb5kdc[1863](info): AS_REQ (2 etypes {18 17}) 192.168.1.93: PREAUTH_FAILED:
hdfs/hadoop.hadoop.com@HADOOP.COM for krbtgt/HADOOP.COM@HADOOP.COM, Decrypt integrity check failed
```

【解决办法】

该进入集群外的节点（如 192.168.1.93），断开对 HDFS 的认证。
等待 5 分钟，此账号就会被解锁。

[HDFS-10004]目录没权限导致启动组件失败

【问题背景与现象】

新安装集群，启动 DataNode 失败。
启动 DataNode 的时候，显示启动失败。具体信息 hadoop-omm-datanode-主机名.out

【原因分析】

1. 查看 datanode 的 out 日志：/var/log/Bigdata/hdfs/dn/hadoop-omm-datanode-主机名.out

```
ulimit -a for secure datanode user omm
standard in must be a tty
```

2. 查看 datanode 运行日志：/var/log/Bigdata/hdfs/dn/hadoop-omm-datanode-主机名.log

```
2015-07-14 23:17:10,276 | WARN | main | Invalid dfs.datanode.data.dir /srv/BigData/hadoop/data1/dn : |
DataNode.java:1948
java.io.FileNotFoundException: File file:/srv/BigData/hadoop/data1/dn does not exist
```

3. 检查发现是 datanode 的目录没有权限：

```
ll /srv/BigData/hadoop
d----- . 5 omm wheel 4096 Jul 15 02:02 data1
```

【解决办法】

修改目录权限：
chown -R omm:wheel /srv/BigData/hadoop/data1/
chmod 700 /srv/BigData/hadoop/data1

【状态异常】

[HDFS-20001]datanode 实例状态为 concerning

【问题背景与现象】

datanode 实例状态为 concerning，hdfs 健康状态为 bad。

【原因分析】

1. 分析 DataNode 日志（/var/log/Bigdata/hdfs/dn/hadoop-omm-datanode-XXX.log）

```
org.apache.hadoop.hdfs.server.common.InconsistentFSStateException: Directory /srv/BigData/hadoop/data2/dn is in an
inconsistent state: Root /srv/BigData/hadoop/data2/dn: DatanodeUuid=79c427a7-867c-4ba1-9593-593d4c97ee7b, does not
match 0b63f82f-1582-43e6-82a1-4b8c34df78fc from other StorageDirectory.

    at org.apache.hadoop.hdfs.server.datanode.DataStorage.setFieldsFromProperties(DataStorage.java:399)
    at org.apache.hadoop.hdfs.server.datanode.DataStorage.setFieldsFromProperties(DataStorage.java:354)
    at org.apache.hadoop.hdfs.server.common.StorageInfo.readProperties(StorageInfo.java:228)
    at org.apache.hadoop.hdfs.server.datanode.DataStorage.doTransition(DataStorage.java:457)
    at org.apache.hadoop.hdfs.server.datanode.DataStorage.recoverTransitionRead(DataStorage.java:226)
    at org.apache.hadoop.hdfs.server.datanode.DataStorage.recoverTransitionRead(DataStorage.java:254)
    at org.apache.hadoop.hdfs.server.datanode.DataNode.initStorage(DataNode.java:990)
    at org.apache.hadoop.hdfs.server.datanode.DataNode.initBlockPool(DataNode.java:956)
    at
    org.apache.hadoop.hdfs.server.datanode.BPOfferService.verifyAndSetNamespaceInfo(BPOfferService.java:278)
    at
    org.apache.hadoop.hdfs.server.datanode.BPServiceActor.connectToNNAndHandshake(BPServiceActor.java:220)
    at org.apache.hadoop.hdfs.server.datanode.BPServiceActor.run(BPServiceActor.java:816)
    at java.lang.Thread.run(Thread.java:745)
```

2. 通过 datanode 日志发现是由于 data2 这个目录的 datanodeuuid 与其他目录的 datanodeuuid 不一致导致 datanode 注册失败，发现是有人手动拷贝修改过 /srv/BigData/hadoop/data2/dn/current 这个目录下面的 VERSION 文件。

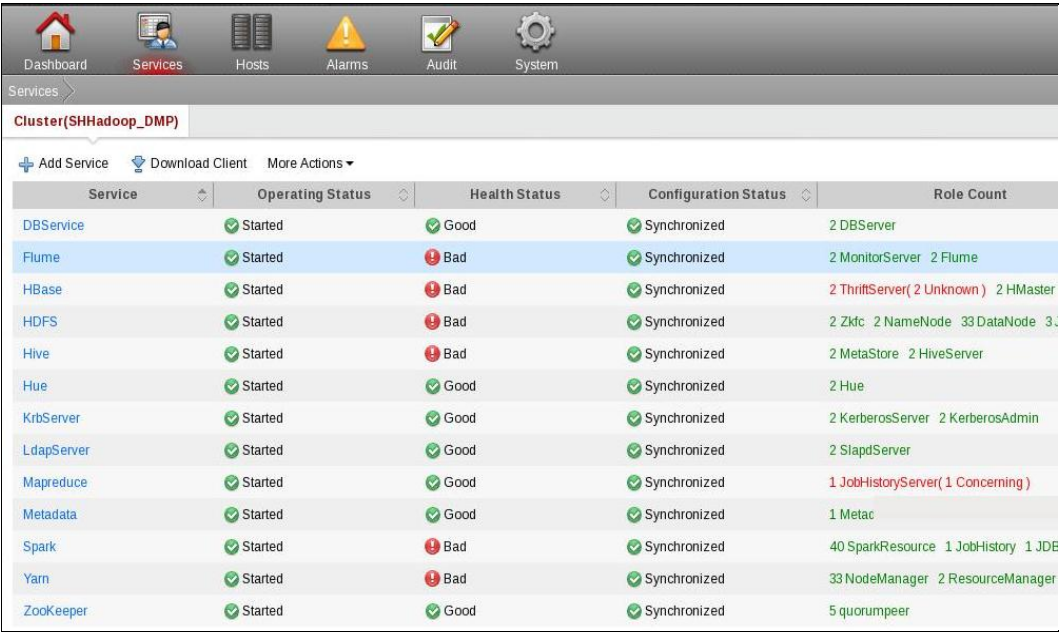
【解决办法】

1. 将 /srv/BigData/hadoop/data2/dn/current 目录下面的 VERSION 文件里面的 datanodeuuid 改成跟/srv/BigData/hadoop/data1/dn/current 目录下的 VERSION 文件一致
2. 从 web 页面重启有问题的 datanode 实例

[HDFS-20002]HDFS 服务的健康状态为 bad

【问题背景与现象】

HDFS 服务的健康状态为 bad。



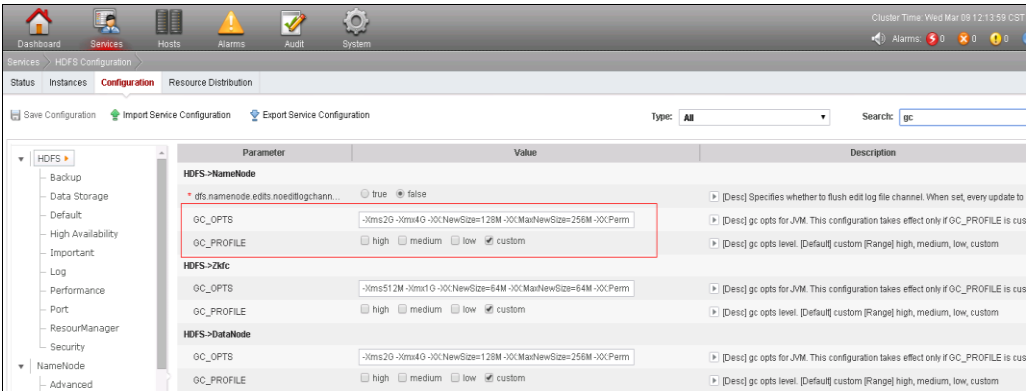
Service	Operating Status	Health Status	Configuration Status	Role Count
DBService	Started	Good	Synchronized	2 DBServer
Flume	Started	Bad	Synchronized	2 MonitorServer 2 Flume
HBase	Started	Bad	Synchronized	2 ThriftServer(2 Unknown) 2 HMaster
HDFS	Started	Bad	Synchronized	2 Zkfc 2 NameNode 33 DataNode 33
Hive	Started	Bad	Synchronized	2 MetaStore 2 HiveServer
Hue	Started	Good	Synchronized	2 Hue
KrbServer	Started	Good	Synchronized	2 KerberosServer 2 KerberosAdmin
LdapServer	Started	Good	Synchronized	2 SlapdServer
Mapreduce	Started	Good	Synchronized	1 JobHistoryServer(1 Concerning)
Metadata	Started	Good	Synchronized	1 Metac
Spark	Started	Bad	Synchronized	40 SparkResource 1 JobHistory 1 JDB
Yarn	Started	Bad	Synchronized	33 NodeManager 2 ResourceManager
ZooKeeper	Started	Good	Synchronized	5 quorumpeer

【原因分析】

- 通过 Namenode 日志发现存在大量 java.lang.OutOfMemoryError: Java heap space，这说明是 namendoe 内存不足导致 HDFS 服务不可用

【解决办法】

- 登录 web 页面，参照 HDFS 容量规划调整 namendoe GC 内存，并重启 namendoe 实例

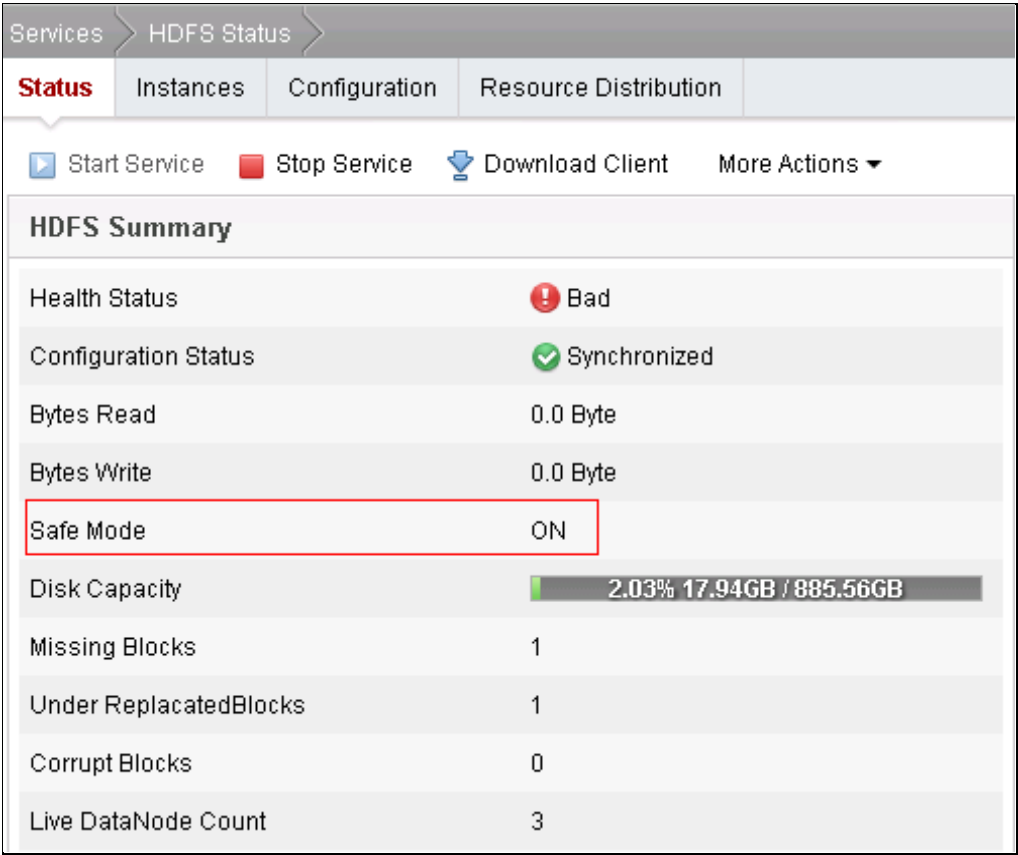


Parameter	Value	Description
HDFS->NameNode		
* dfs.namenode.edits.flushlogchann...	<input type="radio"/> true <input checked="" type="radio"/> false	[Desc] Specifies whether to flush edit log file channel. When set, every update to
GC_OPTS	-Xms2G -Xmx4G -XX:NewSize=128M -XX:MaxNewSize=256M -XX:Perm	[Desc] gc opts for JVM. This configuration takes effect only if GC_PROFILE is cus
GC_PROFILE	<input type="checkbox"/> high <input type="checkbox"/> medium <input type="checkbox"/> low <input checked="" type="checkbox"/> custom	[Desc] gc opts level. [Default] custom [Range] high, medium, low, custom
HDFS->Zkfc		
GC_OPTS	-Xms512M -Xmx1G -XX:NewSize=64M -XX:MaxNewSize=64M -XX:Perm	[Desc] gc opts for JVM. This configuration takes effect only if GC_PROFILE is cus
GC_PROFILE	<input type="checkbox"/> high <input type="checkbox"/> medium <input type="checkbox"/> low <input checked="" type="checkbox"/> custom	[Desc] gc opts level. [Default] custom [Range] high, medium, low, custom
HDFS->DataNode		
GC_OPTS	-Xms20 -Xmx4G -XX:NewSize=128M -XX:MaxNewSize=256M -XX:Perm	[Desc] gc opts for JVM. This configuration takes effect only if GC_PROFILE is cus
GC_PROFILE	<input type="checkbox"/> high <input type="checkbox"/> medium <input type="checkbox"/> low <input checked="" type="checkbox"/> custom	[Desc] gc opts level. [Default] custom [Range] high, medium, low, custom

[HDFS-20003]namenode 进入到 safemode

【问题背景与现象】

namenode 进入到 safemode。



【原因分析】

1. 分析 namenode 日志（/var/log/Bigdata/hdfs/nn/hadoop-omm-namendoe-XXX.log）:

```
2015-06-12 11:41:53,944 | INFO | Socket Reader #1 for port 25000 | Auth successful for dmp_operator2 (auth:PROXY)
v2015-06-12 11:41:54,163 | INFO | IPC Server handler 57 on 25000 | IPC Server handler 57 on 25000, call
org.apache.hadoop.ha.HAServiceProtocol.monitorHealth from 10.63.53.128:44396 Call#289361 Retry#0 |
org.apache.hadoop.ipc.Server$Handler.run(Server.java:2034)
org.apache.hadoop.ha.HealthCheckFailedException: The NameNode has no resources available
at org.apache.hadoop.hdfs.server.namenode.NameNode.monitorHealth(NameNode.java:1466)
at org.apache.hadoop.hdfs.server.namenode.NameNodeRpcServer.monitorHealth(NameNodeRpcServer.java:1158)
at
org.apache.hadoop.ha.protocolPB.HAServiceProtocolServerSideTranslatorPB.monitorHealth(HAServiceProtocolServerSi
deTranslatorPB.java:75)
at
org.apache.hadoop.ha.proto.HAServiceProtocolProtos$HAServiceProtocolService$2.callBlockingMethod(HAServiceProt
ocolProtos.java:4458)
```

```
at org.apache.hadoop.ipc.ProtobufRpcEngine$Server$ProtoBufRpcInvoker.call(ProtobufRpcEngine.java:585)
at org.apache.hadoop.ipc.RPC$Server.call(RPC.java:928)
at org.apache.hadoop.ipc.Server$Handler$1.run(Server.java:2013)
at org.apache.hadoop.ipc.Server$Handler$1.run(Server.java:2009)
at java.security.AccessController.doPrivileged(Native Method)
at javax.security.auth.Subject.doAs(Subject.java:415)
at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1612)
at org.apache.hadoop.ipc.Server$Handler.run(Server.java:2007)
```

2. 日志显示 namenode 资源不足，检查客户环境中 dfs.namenode.edits.dir 配置路径为：/srv/BigData/namenode，该目录挂载在/目录磁盘下，在进入 safemode 这段时间发现磁盘超过阈值告警，提示/目录磁盘空间不足。
3. 通过上述分析，可以确认是由于磁盘空间不足，导致 namenode 进入到 safemode。

【解决办法】

清理 namenode 磁盘空间。

[HDFS-20004]NameNode 经常主备倒换

【问题背景与现象】

NameNode 经常主备倒换。

【原因分析】

1. 查看 HDFS 的日志和 GC 日志，发现 GC 耗费了将近 2 分钟，导致 ZKFC 任务主 NameNode 故障，导致主备切换。

HDFS 日志：

```
2015-06-10 07:19:39,557 | INFO | Socket Reader #1 for port 25000 | Authorization successful for bdim@HADOOP.COM
(auth:TOKEN) for protocol=interface org.apache.hadoop.hdfs.protocol.ClientProtocol |
org.apache.hadoop.security.authorize.ServiceAuthorizationManager.authorize(ServiceAuthorizationManager.java:114)
2015-06-10 07:21:20,228 | WARN |
org.apache.hadoop.hdfs.server.blockmanagement.PendingReplicationBlocks$PendingReplicationMonitor@71da7b2a |
PendingReplicationMonitor timed out blk_1244913381_171247396 |
org.apache.hadoop.hdfs.server.blockmanagement.PendingReplicationBlocks$PendingReplicationMonitor.pendingReplicat
ionCheck(PendingReplicationBlocks.java:249)
2015-06-10 07:21:20,229 | INFO | CacheReplicationMonitor(934749367) | Rescanning after 128980 milliseconds |
org.apache.hadoop.hdfs.server.blockmanagement.CacheReplicationMonitor.run(CacheReplicationMonitor.java:177)
2015-06-10 07:21:20,229 | WARN | org.apache.hadoop.util.JvmPauseMonitor$Monitor@5456abe4 | Detected pause in
JVM or host machine (eg GC): pause of approximately 100504ms
GC pool 'ParNew' had collection(s): count=1 time=769ms
GC pool 'ConcurrentMarkSweep' had collection(s): count=1 time=99875ms |
org.apache.hadoop.util.JvmPauseMonitor$Monitor.run(JvmPauseMonitor.java:169)
```

GC 日志：

```

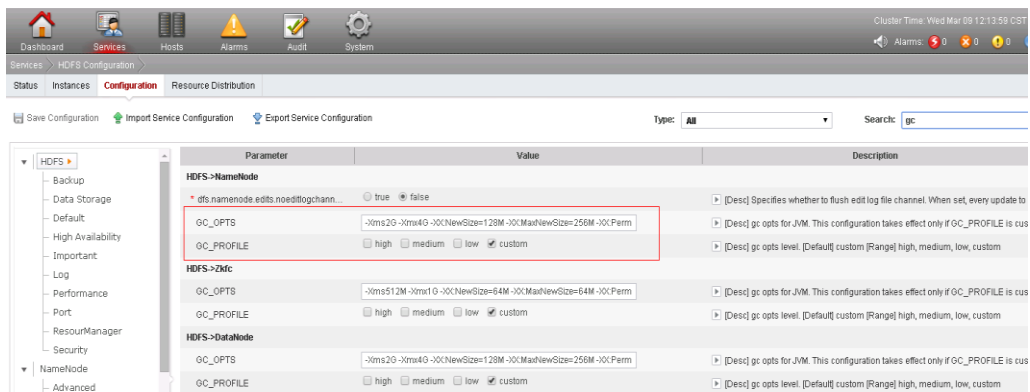
2015-06-10T07:19:39.584+0800: 4156979.605: [GC2015-06-10T07:19:39.584+0800: 4156979.605: [ParNew (promotion
failed): 3010696K->2957240K(3225600K), 0.7691140 secs]2015-06-10T07:19:40.353+0800: 4156980.374: [CMS:
70943439K->20657733K(97079296K), 99.8744960 secs] 73952939K->20657733K(100304896K), [CMS Perm :
51526K->51359K(131072K)], 100.6438720 secs] [Times: user=104.64 sys=0.13, real=100.66 secs]

Heap
par new generation total 3225600K, used 96103K [0x00007f83f2e00000, 0x00007f84cda00000, 0x00007f84cda00000)
eden space 2867200K, 3% used [0x00007f83f2e00000, 0x00007f83f8bda768, 0x00007f84a1e00000)
from space 358400K, 0% used [0x00007f84a1e00000, 0x00007f84a1e00000, 0x00007f84b7c00000)
to space 358400K, 0% used [0x00007f84b7c00000, 0x00007f84b7c00000, 0x00007f84cda00000)
concurrent mark-sweep generation total 97079296K, used 20657733K [0x00007f84cda00000, 0x00007f9bf2e00000,
0x00007f9bf2e00000)
concurrent-mark-sweep perm gen total 131072K, used 51402K [0x00007f9bf2e00000, 0x00007f9bfae00000,
0x00007f9bfae00000)

```

【解决办法】

1. 登录 web 页面，参照 HDFS 容量规划调整 namendoe GC 内存，并重启 namendoe 实例



[HDFS-20005]磁盘空间不足，进行容量扩容，采用挂载 **NAS** 方式替换原有磁盘，**NFS** 服务问题导致 **Datanode** 异常

【问题背景与现象】

磁盘空间不足，进行容量扩容，采用挂载 **NAS** 方式替换原有磁盘。

强制删除 2 个 datanode 后，发生丢块现象，由于另外一个 datanode 处于 concerning 状态，显示丢块。

【原因分析】

1. 一个 datanode 挂载的 NAS 盘的 NFS 服务有问题，datanode 初始化的时候无法获取文件锁资源异常，启动不了。
2. 一个 datanode 挂载 NAS 正常，但是更换 NAS 磁盘，出现启动不了。

【解决办法】

1. 对前一个 datanode 问题，通过重启 rpcbind 服务解决：
`service rpcbind restart`
2. 对后一个 datanode 问题，原因是锁文件资源被占用，暂时通过删除 `/srv/BigData/hadoop/dataX/current/in_use.lock` 文件规避解决出现问题和挂载 NAS 盘的 NFS 服务相关。

[HDFS-20006]datanode 概率性出现 CPU 占用接近 1000%，导致节点丢失（ssh 连得很慢或者不上）

【问题背景与现象】

datanode 概率性出现 CPU 占用接近 1000%，导致节点丢失。

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	MEM	TIME+	COMMAND
60636	omm	20	0	9445m	1.7g	16m	S	299	1.3	1952.06	java.exe -Dproc_datanode -outfile /var/log/Bigdata/hdfs/dn/jsvc.out -errfile /var/log/Bigdata/hdfs/dn/jsvc.err -pidfil
32428	oosadm	20	0	18116	3784	1828	R	155	0.0	1:17.63	/opt/tsp/manager/rtp/python/bin/python /opt/tsp/manager/agent-1.3.10.200/tools/pyscript/sysappctrl.py -cmd status -te
32410	oosadm	20	0	55016	8048	2836	R	155	0.0	1:59.80	/opt/tsp/manager/rtp/python/bin/python /opt/tsp/manager/agent-1.3.10.200/tools/pyscript/watchdog.py -cmd status
32412	oosadm	20	0	36752	3512	2340	R	155	0.0	1:50.32	/opt/tsp/manager/rtp/python/bin/python /opt/tsp/manager/agent-1.3.10.200/tools/pyscript/sysappctrl.py -cmd procinfo -
32484	omm	20	0	12800	1476	1124	R	155	0.0	0:10.73	/bin/bash -c /opt/huawei/Bigdata/jdk1.7.0_80/bin/java -server -Xmx1024m -Djava.io.tmpdir=/export/data/jam/nm/localdi
32341	oosadm	20	0	57760	8688	3000	R	139	0.0	3:29.41	/opt/tsp/manager/rtp/python/bin/python /opt/tsp/manager/agent/tools/pyscript/syscollector.py svs /opt/tsp/manager/var
32531	omm	20	0	11176	640	468	R	106	0.0	0:04.19	-bash -c echo \$QMS_RUN_PATH
32443	root	20	0	0	0	0	S	81	0.0	0:11.87	lsnsnsnsnsnsnsns

【原因分析】

1. datanode 有许多写失败的日志。

```
2015-08-31 11:29:34,184 [ERROR] DataXceiver for client DFSCClient_NONMAPREDUCE_1675952887_23 at /192.168.8.40:44514 [Receiving block BP-125271511-192.168.8.29-1440656260530:blk_1074766997_1034914] | TSP21:25009:DataXceiver error processing WRITE_BLOCK operation src:/192.168.8.40:44514 dst:/192.168.8.64:25009 | DataXceiver.java:258
java.io.IOException: Premature EOF from inputStream
    at org.apache.hadoop.io.IOUtils.readFully(IOUtils.java:194)
    at org.apache.hadoop.hdfs.protocol.datatransfer.PacketReceiver.doReadFully(PacketReceiver.java:213)
    at org.apache.hadoop.hdfs.protocol.datatransfer.PacketReceiver.doRead(PacketReceiver.java:134)
    at org.apache.hadoop.hdfs.protocol.datatransfer.PacketReceiver.receiveNextPacket(PacketReceiver.java:109)
    at org.apache.hadoop.hdfs.server.datanode.BlockReceiver.receivePacket(BlockReceiver.java:446)
    at org.apache.hadoop.hdfs.server.datanode.BlockReceiver.receiveBlock(BlockReceiver.java:707)
    at org.apache.hadoop.hdfs.server.datanode.DataXceiver.writeBlock(DataXceiver.java:748)
    at org.apache.hadoop.hdfs.protocol.datatransfer.Receiver.opWriteBlock(Receiver.java:124)
    at org.apache.hadoop.hdfs.protocol.datatransfer.Receiver.processOp(Receiver.java:71)
    at org.apache.hadoop.hdfs.server.datanode.DataXceiver.run(DataXceiver.java:240)
    at java.lang.Thread.run(Thread.java:745)
2015-08-31 11:29:35,147 [INFO] DataXceiver for client DFSCClient_NONMAPREDUCE_402997805_1 at /192.168.8.30:59449 [Sending block BP-125271511-192.168.8.29-1440656260530:blk_1074181856_446655] | src:/192.168.8.64:25009, dest:/192.168.8.30:59449, bytes: 16826, op:HDFS_READ, c1ID: DFSCClient_NONMAPREDUCE_402997805_1, offset: 0, srvid: 9d1d30a5-046d-438b-83c9-2c6c54c6bd12, blockid: BP-125271511-192.168.8.29-1440656260530:blk_1074181856_446655, duration: 78832 | BlockSender.java:738
2015-08-31 11:29:35,269 [INFO] org.apache.hadoop.util.JvmPauseMonitor$Monitor@551bd2a0 | Detected pause in JVM or host machine (eg GC): pause of approximately 7480ms
No GCs detected | JvmPauseMonitor.java:172
2015-08-31 11:29:36,985 [INFO] org.apache.hadoop.util.JvmPauseMonitor$Monitor@551bd2a0 | Detected pause in JVM or host machine (eg GC): pause of approximately 1215ms
No GCs detected | JvmPauseMonitor.java:172
2015-08-31 11:29:43,067 [INFO] DataXceiver for client DFSCClient_NONMAPREDUCE_1675952887_23 at /192.168.8.33:35530 [Receiving block BP-125271511-192.168.8.29-1440656260530:blk_1074767006_1034923] | Exception for BP-125271511-192.168.8.29-1440656260530:blk_1074767006_1034923 | BlockReceiver.java:742
java.io.IOException: Premature EOF from inputStream
```

2. 短时间内写入大量文件导致这种情况，因此 datanode 内存不足。

```
Line 153101: 2015-08-31 11:24:29,313 [INFO] org.apache.hadoop.util.JvmPauseMonitor$Monitor@551bd2a0 | Detected pause in JVM or host machine (eg GC): pause of approximately 119ms
Line 153132: 2015-08-31 11:24:42,689 [WARN] org.apache.hadoop.util.JvmPauseMonitor$Monitor@551bd2a0 | Detected pause in JVM or host machine (eg GC): pause of approximately 1127ms
Line 153135: 2015-08-31 11:24:45,810 [INFO] org.apache.hadoop.util.JvmPauseMonitor$Monitor@551bd2a0 | Detected pause in JVM or host machine (eg GC): pause of approximately 100ms
Line 153138: 2015-08-31 11:24:49,801 [INFO] org.apache.hadoop.util.JvmPauseMonitor$Monitor@551bd2a0 | Detected pause in JVM or host machine (eg GC): pause of approximately 1067ms
Line 153155: 2015-08-31 11:25:10,167 [WARN] org.apache.hadoop.util.JvmPauseMonitor$Monitor@551bd2a0 | Detected pause in JVM or host machine (eg GC): pause of approximately 1232ms
```

【解决办法】

1. 检查 datanode 内存配置，以及机器内存。适当增加 datanode 内存。

【读写文件异常相关】

[HDFS-30001]文件最大打开句柄数设置太小，导致文件句柄不足

【问题背景与现象】

写文件到 HDFS 很慢，或者写文件失败。

【原因分析】

1. DataNode 日志，存在异常提示 java.io.IOException: Too many open files

/var/log/Bigdata/hdfs/dn/hadoop-omm-datanode-XXX.log

```
-05-19 17:18:59,126 | WARN | org.apache.hadoop.hdfs.server.datanode.DataXceiverServer@142ff9fa |  
YSDN12:25009:DataXceiverServer: |  
org.apache.hadoop.hdfs.server.datanode.DataXceiverServer.run(DataXceiverServer.java:160)  
java.io.IOException: Too many open files  
at sun.nio.ch.ServerSocketChannelImpl.accept0(Native Method)  
at sun.nio.ch.ServerSocketChannelImpl.accept(ServerSocketChannelImpl.java:241)  
at sun.nio.ch.ServerSocketAdaptor.accept(ServerSocketAdaptor.java:100)  
at org.apache.hadoop.hdfs.net.TcpPeerServer.accept(TcpPeerServer.java:134)  
at org.apache.hadoop.hdfs.server.datanode.DataXceiverServer.run(DataXceiverServer.java:137)  
at java.lang.Thread.run(Thread.java:745)
```


2. 如果某个 datanode 日志中打印 Too many open files，说明该节点文件句柄不足，导致打开文件句柄失败，然后就会重试往其他 datanode 节点写数据，最终表现为写文件很慢或者写文件失败。

【解决办法】

1. 用 `ulimit -a` 命令查看有问题节点文件句柄数最多设置是多少，如果很小，建议修改成 640000。

```
[omm@189-39-150-167 ~]$ ulimit -a  
core file size          (blocks, -c) 0  
data seg size           (kbytes, -d) unlimited  
scheduling priority     (-e) 0  
file size               (blocks, -f) unlimited  
pending signals         (-i) 256551  
max locked memory       (kbytes, -l) 64  
max memory size         (kbytes, -m) unlimited  
open files              (-n) 640000  
pipe size               (512 bytes, -p) 8  
POSIX message queues    (bytes, -q) 819200  
real-time priority      (-r) 0  
stack size              (kbytes, -s) 10240  
cpu time                (seconds, -t) unlimited  
max user processes      (-u) 60000  
virtual memory          (kbytes, -v) unlimited  
file locks              (-x) unlimited
```

2. `vi /etc/security/limits.d/90-nofile.conf` 编辑这个文件，修改文件句柄数设置。如果没有这个文件，可以新建一个文件，并按照下图内容修改。

	<code>hard</code>	<code>nofile</code>	<code>640000</code>
<code>*</code>	<code>soft</code>	<code>nofile</code>	<code>640000</code>

3. 重新打开一个终端，用 `ulimit -a` 命令查看是否修改成功，如果没有，请重新按照上述步骤重新修改。
4. 从 web 页面重启 datanode 实例。

[HDFS-30002]客户端写文件 close 失败

【问题背景与现象】

客户端写文件 close 失败，客户端提示数据块没有足够非副本数。

客户端日志：

```
2015-05-27      19:00:52.811      [pool-2-thread-3]      ERROR:
/tsp/nedata/collect/UGW/ugwufdr/20150527/10/6_20150527105000_20150527105500_SR5S14_1432723806338_128_11.pkg
.tmp1432723806338 close hdfs sequence file fail (SequenceFileInfoChannel.java:444)
java.io.IOException: Unable to close file because the last block does not have enough number of replicas.
    at org.apache.hadoop.hdfs.DFSOutputStream.completeFile(DFSOutputStream.java:2160)
    at org.apache.hadoop.hdfs.DFSOutputStream.close(DFSOutputStream.java:2128)
    at org.apache.hadoop.fs.FSDataOutputStream$PositionCache.close(FSDataOutputStream.java:70)
    at org.apache.hadoop.fs.FSDataOutputStream.close(FSDataOutputStream.java:103)
    at com.huawei.pai.collect2.stream.SequenceFileInfoChannel.close(SequenceFileInfoChannel.java:433)
    at
com.huawei.pai.collect2.stream.SequenceFileWriterToolChannel$FileCloseTask.call(SequenceFileWriterToolChannel.java:804
)
    at
com.huawei.pai.collect2.stream.SequenceFileWriterToolChannel$FileCloseTask.call(SequenceFileWriterToolChannel.java:792
)
    at java.util.concurrent.FutureTask.run(FutureTask.java:262)
    at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1145)
    at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:615)
    at java.lang.Thread.run(Thread.java:745)
```

【原因分析】

1. HDFS 客户端开始写 Block

HDFS 客户端是在 2015-05-27 18:50:24,232 开始写 /20150527/10/6_20150527105000_20150527105500_SR5S14_1432723806338_128_11.pkg.tmp1432723806338 的。其中分配的块是 blk_1099105501_25370893：

```
2015-05-27 18:50:24,232 | INFO | IPC Server handler 30 on 25000 | BLOCK* allocateBlock:
/20150527/10/6_20150527105000_20150527105500_SR5S14_1432723806338_128_11.pkg.tmp1432723806338. BP-
```

```
1803470917-172.29.57.33-1428597734132 blk_1099105501_25370893{blockUCState=UNDER_CONSTRUCTION,
primaryNodeIndex=-1, replicas=[ReplicaUnderConstruction[[DISK]DS-b2d7b7d0-f410-4958-8eba-
6deecbca2f87:NORMAL|RBW], ReplicaUnderConstruction[[DISK]DS-76bd80e7-ad58-49c6-bf2c-
03f91caf750f:NORMAL|RBW]]}
org.apache.hadoop.hdfs.server.namenode.FSNamesystem.saveAllocatedBlock(FSNamesystem.java:3166)
```

2. 写完之后 HDFS 客户端调用了 fsync:

```
2015-05-27 19:00:22,717 | INFO | IPC Server handler 22 on 25000 | BLOCK* fsync:
20150527/10/6_20150527105000_20150527105500_SR5S14_1432723806338_128_11.pkg.tmp1432723806338 for
DFSClient_NONMAPREDUCE_-120525246_15
org.apache.hadoop.hdfs.server.namenode.FSNamesystem.fsync(FSNamesystem.java:3805)
```

3. HDFS 客户端调用 close 关闭文件，NameNode 收到客户端的 close 请求之后就会检查最后一个块的完成状态，只有当有足够的 DataNode 上报了块完成才可用关闭文件，检查块完成的状态是通过 checkFileProgress 函数检查的，打印如下：

```
2015-05-27 19:00:27,603 | INFO | IPC Server handler 44 on 25000 | BLOCK* checkFileProgress:
blk_1099105501_25370893{blockUCState=COMMITTED, primaryNodeIndex=-1,
replicas=[ReplicaUnderConstruction[[DISK]DS-ef5fd3c9-5088-4813-ae9a-34a0714ec3a3:NORMAL|RBW],
ReplicaUnderConstruction[[DISK]DS-f863e30f-ce5b-48cc-9cca-72f64c558adc:NORMAL|RBW]]} has not reached
minimal replication 1
org.apache.hadoop.hdfs.server.namenode.FSNamesystem.checkFileProgress(FSNamesystem.java:3197)
2015-05-27 19:00:28,005 | INFO | IPC Server handler 45 on 25000 | BLOCK* checkFileProgress:
blk_1099105501_25370893{blockUCState=COMMITTED, primaryNodeIndex=-1,
replicas=[ReplicaUnderConstruction[[DISK]DS-ef5fd3c9-5088-4813-ae9a-34a0714ec3a3:NORMAL|RBW],
ReplicaUnderConstruction[[DISK]DS-f863e30f-ce5b-48cc-9cca-72f64c558adc:NORMAL|RBW]]} has not reached
minimal replication 1
org.apache.hadoop.hdfs.server.namenode.FSNamesystem.checkFileProgress(FSNamesystem.java:3197)
2015-05-27 19:00:28,806 | INFO | IPC Server handler 63 on 25000 | BLOCK* checkFileProgress:
blk_1099105501_25370893{blockUCState=COMMITTED, primaryNodeIndex=-1,
replicas=[ReplicaUnderConstruction[[DISK]DS-ef5fd3c9-5088-4813-ae9a-34a0714ec3a3:NORMAL|RBW],
ReplicaUnderConstruction[[DISK]DS-f863e30f-ce5b-48cc-9cca-72f64c558adc:NORMAL|RBW]]} has not reached
minimal replication 1
org.apache.hadoop.hdfs.server.namenode.FSNamesystem.checkFileProgress(FSNamesystem.java:3197)
2015-05-27 19:00:30,408 | INFO | IPC Server handler 43 on 25000 | BLOCK* checkFileProgress:
blk_1099105501_25370893{blockUCState=COMMITTED, primaryNodeIndex=-1,
replicas=[ReplicaUnderConstruction[[DISK]DS-ef5fd3c9-5088-4813-ae9a-34a0714ec3a3:NORMAL|RBW],
ReplicaUnderConstruction[[DISK]DS-f863e30f-ce5b-48cc-9cca-72f64c558adc:NORMAL|RBW]]} has not reached
minimal replication 1
org.apache.hadoop.hdfs.server.namenode.FSNamesystem.checkFileProgress(FSNamesystem.java:3197)
2015-05-27 19:00:33,610 | INFO | IPC Server handler 37 on 25000 | BLOCK* checkFileProgress:
blk_1099105501_25370893{blockUCState=COMMITTED, primaryNodeIndex=-1,
replicas=[ReplicaUnderConstruction[[DISK]DS-ef5fd3c9-5088-4813-ae9a-34a0714ec3a3:NORMAL|RBW],
ReplicaUnderConstruction[[DISK]DS-f863e30f-ce5b-48cc-9cca-72f64c558adc:NORMAL|RBW]]} has not reached
minimal replication 1
org.apache.hadoop.hdfs.server.namenode.FSNamesystem.checkFileProgress(FSNamesystem.java:3197)
```

```
2015-05-27 19:00:40,011 | INFO | IPC Server handler 37 on 25000 | BLOCK* checkFileProgress:
blk_1099105501_25370893{blockUCState=COMMITTED, primaryNodeIndex=-1,
replicas=[ReplicaUnderConstruction[[DISK]DS-ef5fd3c9-5088-4813-ae9a-34a0714ec3a3:NORMAL|RBW],
ReplicaUnderConstruction[[DISK]DS-f863e30f-ce5b-48cc-9cca-72f64c558adc:NORMAL|RBW]]} has not reached
minimal replication 1
org.apache.hadoop.hdfs.server.namenode.FSNamesystem.checkFileProgress(FSNamesystem.java:3197)
```

4. NameNode 打印了多次 checkFileProgress 是由于 HDFS 客户端多次尝试 close 文件，但是由于当前状态不满足要求，导致 close 失败，HDFS 客户端 retry 的次数是由参数 dfs.client.block.write.locateFollowingBlock.retries 决定的，该参数默认是 5，所以在 NameNode 的日志中看到了 6 次 checkFileProgress 打印。
5. 但是再过 0.5s 之后，DataNode 就上报块已经成功写入了：

```
2015-05-27 19:00:40,608 | INFO | IPC Server handler 60 on 25000 | BLOCK* addStoredBlock: blockMap updated:
192.168.10.21:25009 is added to blk_1099105501_25370893{blockUCState=COMMITTED, primaryNodeIndex=-1,
replicas=[ReplicaUnderConstruction[[DISK]DS-ef5fd3c9-5088-4813-ae9a-34a0714ec3a3:NORMAL|RBW],
ReplicaUnderConstruction[[DISK]DS-f863e30f-ce5b-48cc-9cca-72f64c558adc:NORMAL|RBW]]} size 11837530 |
org.apache.hadoop.hdfs.server.blockmanagement.BlockManager.logAddStoredBlock(BlockManager.java:2393)
2015-05-27 19:00:48,297 | INFO | IPC Server handler 37 on 25000 | BLOCK* addStoredBlock: blockMap updated:
192.168.10.10:25009 is added to blk_1099105501_25370893 size 11837530 |
org.apache.hadoop.hdfs.server.blockmanagement.BlockManager.logAddStoredBlock(BlockManager.java:2393)
```

6. DataNode 上报块写成功通知延迟的原因可能有：网络瓶颈导致、CPU 瓶颈导致。
7. 如果此时再次调用 close 或者 close 的 retry 的次数增多，那么 close 都将返回成功。建议适当增大参数 dfs.client.block.write.locateFollowingBlock.retries 的，默认值为 5 次，尝试的时间间隔为 400ms、800ms、1600ms、3200ms、6400ms、12800ms，那么 close 函数最多需要 25.2 秒才能返回。

【解决办法】

规避办法：

1. 可以通过调整客户端参数 dfs.client.block.write.locateFollowingBlock.retries 的值来增加 retry 的次数，可以将值设置为 6，那么中间睡眠等待的时间为 400ms、800ms、1600ms、3200ms、6400ms、12800ms，也就是说 close 函数最多要 50.8 秒才能返回。

【备注说明】

一般出现上述现象，说明集群负载很大，通过调整参数只是临时规避这个问题，建议还是降低集群负载。例如：避免把所有 CPU 都分配 MR 跑任务。

[HDFS-30003]文件错误导致上传文件到 HDFS 失败

【问题背景与现象】

用 hadoop dfs -put 把本地文件拷贝到 HDFS 上，有报错。

上传部分文件后，报错失败，从 namenode 原生页面看，临时文件大小不再变化。

【原因分析】

1. 查看 namenode 日志，发现该文件一直在被尝试写，直到最终失败。

/var/log/Bigdata/hdfs/nn/hadoop-omm-namenode-主机名.log

```
2015-07-13 10:05:07,847 | WARN | org.apache.hadoop.hdfs.server.namenode.LeaseManager$Monitor@36fea922 | DIR*
NameSystem.internalReleaseLease: Failed to release lease for file /hive/order/OS_ORDER_8.txt_COPYING_. Committed
blocks are waiting to be minimally replicated. Try again later. | FSNamesystem.java:3936
2015-07-13 10:05:07,847 | ERROR | org.apache.hadoop.hdfs.server.namenode.LeaseManager$Monitor@36fea922 | Cannot
release the path /hive/order/OS_ORDER_8.txt_COPYING_ in the lease [Lease. Holder:
DFSClient_NONMAPREDUCE_-1872896146_1, pendingcreates: 1] | LeaseManager.java:459
org.apache.hadoop.hdfs.protocol.AlreadyBeingCreatedException: DIR* NameSystem.internalReleaseLease: Failed to release
lease for file /hive/order/OS_ORDER_8.txt_COPYING_. Committed blocks are waiting to be minimally replicated. Try
again later.
at FSNamesystem.internalReleaseLease(FSNamesystem.java:3937)
```

2. 根因分析：被上传的文件损坏，因此会上传失败。
3. 验证办法：cp 或者 scp 被拷贝的文件，也会失败，确认文件本身已损坏。

【解决办法】

1. 文件本身损坏造成的此问题，采用正常文件进行上传。

[HDFS-30004]网络丢包/错包导致 HDFS 写大文件失败

【问题背景与现象】

双平面，新安装集群，3 个 datanode 节点，HDFS 显示正常，但是客户端往集群 put 失败。

往 HDFS 写文件失败，客户端日志显示没有足够的副本数。

```
2015-07-22 17:11:10,095 | ERROR | HiveServer2-Background-Pool: Thread-585 | Job Submission failed with exception
'java.io.IOException(Unable to close file because the last block does not have enough number of replicas.)'
java.io.IOException: Unable to close file because the last block does not have enough number of replicas.
```

【原因分析】

1. 查看 NameNode 的 audit 日志，运行正常。查看 namenode 日志，发现 datanode 节点之间块传输出错。

```
2015-07-22 17:11:00,278 | INFO | DataXceiver for client DFSClient_NONMAPREDUCE_1825323278_585 at
/192.168.106.12:57651 [Receiving block BP-672666964-192.168.106.10-1437377674081:blk_1073746882_6066]
| Exception for
BP-672666964-192.168.106.10-1437377674081:blk_1073746882_6066 | BlockReceiver.java:742
java.io.IOException: Premature EOF from inputStream
```

2. 在这两个节点服务 IP (business ip)，之间拷贝 (scp) 数据，拷贝速度是 50kb/s。使用 ifconfig 命令，查看到大量的丢包错包 (ping 命令也可以看到)。


```
51-196-106-12:/opt # ifconfig
eth0      Link encap:Ethernet  HWaddr 9C:37:F4:06:42:BF
          inet addr:192.168.106.12  Bcast:192.168.255.255  Mask:255.255.0.0
          inet6 addr: fe80::9e37:f4ff:fe06:42bf/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:7651125 errors:506281 dropped:76171 overruns:0 frame:506281
          TX packets:8995644 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3890564273 (3710.3 Mb)  TX bytes:3850128648 (3671.7 Mb)
```

3. 经过检查硬件，是光模块问题，会导致网络丢包、错包情况。

```
[root@host1 quorumpeer]# ping 192.168.121.5
PING 192.168.121.5 (192.168.121.5) 56(84) bytes of data.
64 bytes from 192.168.121.5: icmp_seq=12 ttl=64 time=0.119 ms
64 bytes from 192.168.121.5: icmp_seq=13 ttl=64 time=0.115 ms
^C
--- 192.168.121.5 ping statistics ---
20 packets transmitted, 2 received, 90% packet loss, time 19610ms
rtt min/avg/max/mdev = 0.115/0.117/0.119/0.002 ms
```

【解决办法】

1. 规避办法：不使用该机器。
2. 解决办法：硬件问题（光模块问题），更换硬件

[HDFS-30005]HDFS 并发写操作过多造成写失败

【问题背景与现象】

在某集群中，HDFS 运行正常，仅有一个 DataNode，最小副本数为 1。某组件在向 HDFS 写入数据失败。

在其他组件使用 HDFS 的 API 接口向 HDFS 写入数据时，发现选择 DataNode 失败，如下图。

```
07-08 00:19:52,330|ERROR|pai.mst.as.MstRunner|com.huawei.pai.common.util.iohelper.IOStreamHelper
113|IOException in closing org.codehaus.jackson.impl.WriterBasedGenerator@70afd866
org.apache.hadoop.ipc.RemoteException(java.io.IOException): File /tsp/configdata/traut/persist/umts could
only be replicated to 0 nodes instead of minReplication (=1). There are 1 datanode(s) running and no
node(s) are excluded in this operation.
    at org.apache.hadoop.hdfs.server.blockmanagement.BlockManager.chooseTarget(BlockManager.java:1470)
    at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.getAdditionalBlock(FSNamesystem.java:2707)
    at org.apache.hadoop.hdfs.server.namenode.NameNodeRpcServer.addBlock(NameNodeRpcServer.java:587)
    at org.apache.hadoop.hdfs.protocolPB.ClientNameNodeProtocolServerSideTranslatorPB.addBlock(ClientNameNodeProt
ocolServerSideTranslatorPB.java:442)
    at org.apache.hadoop.hdfs.protocol.proto.ClientNameNodeProtocolProtos$ClientNameNodeProtocol$2.callBlockingMe
thod(ClientNameNodeProtocolProtos.java)
    at org.apache.hadoop.ipc.ProtobufRpcEngine$Server$ProtoBufRpcInvoker.call(ProtobufRpcEngine.java:585)
    at org.apache.hadoop.ipc.RPC$Server.call(RPC.java:928)
    at org.apache.hadoop.ipc.Server$Handler$1.run(Server.java:2013)
    at org.apache.hadoop.ipc.Server$Handler$1.run(Server.java:2009)
    at java.security.AccessController.doPrivileged(Native Method)
    at javax.security.auth.Subject.doAs(Subject.java:415)
    at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1641)
    at org.apache.hadoop.ipc.Server$Handler.run(Server.java:2007)

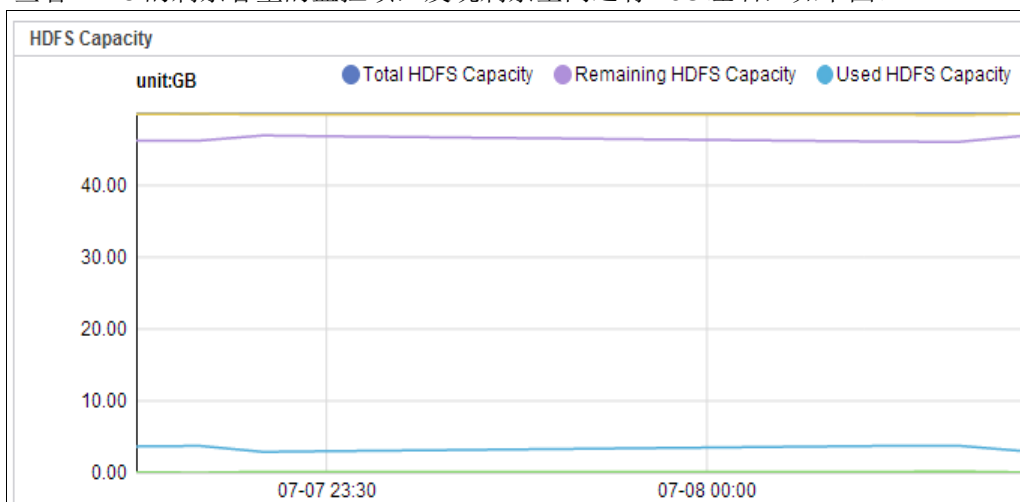
    at org.apache.hadoop.ipc.Client.call(Client.java:1410)
    at org.apache.hadoop.ipc.Client.call(Client.java:1363)
    at org.apache.hadoop.ipc.ProtobufRpcEngine$Invoker.invoke(ProtobufRpcEngine.java:206)
    at com.sun.proxy.$Proxy21.addBlock(Unknown Source)
```

【原因分析】

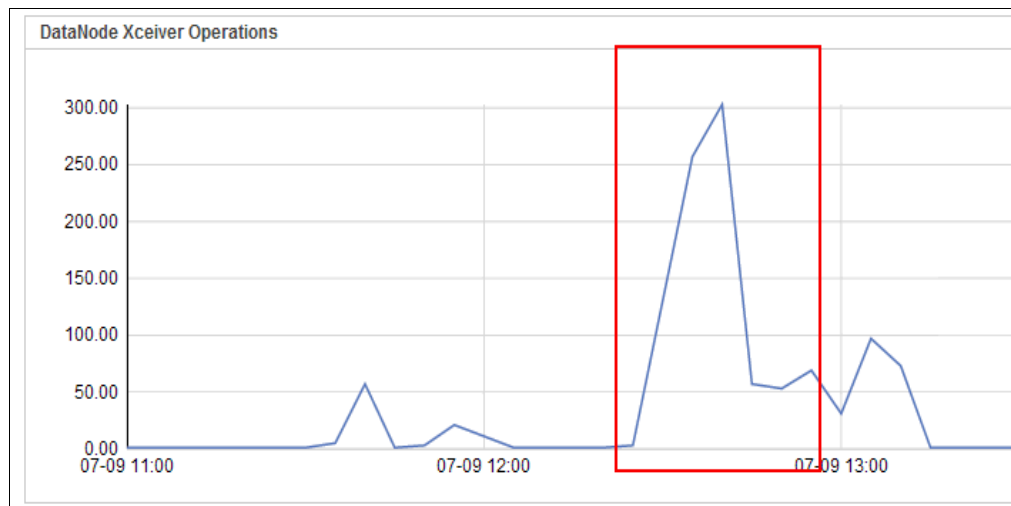
1. 查看 NameNode 日志，发现仅有的 DataNode 的磁盘剩余空间不足。

```
2015-07-08 00:19:59,837 | WARN | IPC Server handler 35 on 25000 | Failed to place enough replicas, still
in need of 1 to reach 1. [10.183.23.204:25009: Storage
[DISK]DS-de03be13-bfa9-40d1-adcb-e55692f1d7ab:NORMALat node /default/rack0/10.183.23.204:25009
because th is not chosene node does not have enough space 10.183.23.204:25009: Storage
[DISK]DS-06a3db7f-5966-4e8e-8b22-739293d9f9bf:NORMALat node /default/rack0/10.183.23.204:25009 is not
chosen because the node does not have enough space ] | BlockPlacementPolicyDefault.java:345
2015-07-08 00:19:59,837 | DEBUG | IPC Server handler 35 on 25000 | PrivilegedActionException
as:ossuser/hadoop@HADOOP.COM (auth:KERBEROS) cause:java.io.IOException: File
/tsp/configdata/filemanager/QuotaMonitorInfo.json could only be replicated to 0 nodes instead of
minReplication (=1). There are 1 datanode(s) running and no node(s) are excluded in this operation. |
UserGroupInformation.java:1645
2015-07-08 00:19:59,837 | INFO | IPC Server handler 35 on 25000 | IPC Server handler 35 on 25000, call
org.apache.hadoop.hdfs.protocol.ClientProtocol.addBlock from 10.183.23.33:44784 Call#2353 Retry#0 |
Server.java:2034
java.io.IOException: File /tsp/configdata/filemanager/QuotaMonitorInfo.json could only be replicated to 0
nodes instead of minReplication (=1). There are 1 datanode(s) running and no node(s) are excluded in this
operation.
    at org.apache.hadoop.hdfs.server.blockmanagement.BlockManager.chooseTarget(BlockManager.java:1470)
    at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.getAdditionalBlock(FSNamesystem.java:2707)
    at org.apache.hadoop.hdfs.server.namenode.NameNodeRpcServer.addBlock(NameNodeRpcServer.java:587)
    at
```

2. 查看 HDFS 的剩余容量的监控项，发现剩余空间还有 40G 左右，如下图。



3. 根据 HDFS 的剩余空间来看，不应该出现写数据时的 DataNode 空间不足的错误。但确实出现了这样的错误提示，那说明这 40G 的剩余空间已经被预占用了。查看 HDFS 的读写操作数（DataNode Xceiver Operations）指标，可以看到出现写操作失败时的读写操作数最高达 300 个，如下图。这是非常高的读写并发量。



根据 HDFS 的分配一个块的原则，每分配 1 个块，就需要预占 5 个块的空间。所以，以每个写操作申请 1 个块的最小量计算，300 个并发量的写操作需要一次申请的 DataNode 大小为： $300 \times 5 \times 128\text{M} = 192\text{G}$
所以，这么高的预占用容量，远远高于 HDFS 的剩余容量，最后导致了写入失败。

【解决办法】

增加 HDFS 容量，或者减小该组件写 DataNode 的并发数。

[HDFS-30006]写 hdfs 文件在 close 遇到异常，再次执行 append 显示文件 lease 被老客户端占据，无法写入数据。

【问题背景与现象】

关于 FusionInsight_V100R002C30。3 个节点的集群，客户端调用 create 或者 append 接口写 hdfs 文件时重启两个 DN。

3 个节点的集群，客户端调用 create 或者 append 接口写 hdfs 文件时重启两个 DN，DN 启动之后仍然无法写入数据。

【原因分析】

1. 客户端调用 create 或者 append 接口写 hdfs 文件时，如果 Pipeline 的 DN 出现问题，HDFS 将会更新 pipeline，若无法恢复的情况下将抛出异常
2. 客户端在捕获异常之后对输出流进行 close，若 close 成功，NN 将会释放该文件的 lease。若 close 失败，NN 将会占据该文件的 lease，直到 lease 超过软连接 60s 才可以被其他客户端抢占。

客户端出现异常：

```
DataStreamException | org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.run(DFSOutputStream.java:627)
java.io.IOException: Failed to replace a bad datanode on the existing pipeline due to no more good datanodes being available to try.
```

```
(Nodes: current=[192.168.26.49:25009], original=[192.168.26.49:25009]). The current failed datanode replacement policy is DEFAULT, and a client may configure this via 'dfs.client.block.write.replace-datanode-on-failure.policy' in its configuration.
```

3. 当在 close 失败的情况,如果客户端直接使用 **append** 接口去操作文件将会出现异常。
namenode 异常:

```
“2015-07-21 20:27:22,152 | WARN | IPC Server  
handler 13 on 25000 | DIR* NameSystem.append: failed to create file /flume/zyh/TMP_over_12.log.tmp for  
DFSClient_NONMAPREDUCE_-1730950729_27 for client 192.168.26.48 because current leaseholder is trying to recreate  
file. | FSNamesystem.java:2395“
```

最终文件一致处于无法写入的状态。

【解决办法】

在遇到 close 异常的时候,如果需要再次进行 **append** 操作,则可以先检查该文件的 **lease** 是否被其他的客户端占据,如果没有则可以进行 **append**,如果该文件还被其他客户端打开,则等待 NN 释放 **lease**。异常关闭的文件在 NN 中将会维持 60s 的 **lease** 被旧的客户端占据,60s 之后其他客户端可以对该文件进行操作。客户端代码需要处理 close 异常情况。

[HDFS-30007]界面配置 **dfs.blocksize**, 将其设置为 268435456, put 数据, block 大小还是原来的大小

【问题背景与现象】

界面配置 **dfs.blocksize**, 将其设置为 268435456, put 数据, block 大小还是原来的大小。

【原因分析】

1. 客户端的 **hdfs-site.xml** 文件中的 **dfs.blocksize** 大小没有更改, 以客户端配置为准。

【解决办法】

1. 确保 **dfs.blocksize** 为 512 的倍数。
2. 重新下载安装客户端或者更改客户端配置。
3. **dfs.blocksize** 是客户端配置, 以客户端为准。若客户端不配置, 以服务端为准。

【SHELL 操作客户端相关】

[HDFS-40001]执行 HDFS SHELL 命令失败

【问题背景与现象】

客户端执行 HDFS 命令失败。

客户端日志:

```
[omm@SCCHDPHIV02103 2_24_HiveServer]$ hadoop fs -du -h /warehouse/plutolog.db/hive_appaccessinfo_fat
shell-init: error retrieving current directory: getcwd: cannot access parent directories: No such file or directory
chdir: error retrieving current directory: getcwd: cannot access parent directories: No such file or directory
chdir: error retrieving current directory: getcwd: cannot access parent directories: No such file or directory
chdir: error retrieving current directory: getcwd: cannot access parent directories: No such file or directory

No GC_PROFILE is given. Defaults to medium.

Error occurred during initialization of VM

java.lang.Error: Properties init: Could not determine current working directory.
    at java.lang.System.initProperties(Native Method)
    at java.lang.System.initializeSystemClass(System.java:1119)
```

【原因分析】

1. 根据客户端日志显示，发现是不能访问父目录地址，提示没有这个目录。
2. 检查当前目录路径，发现父目录已经被其他终端删除，所以判断是由于父目录被删除导致命令执行失败。

【解决办法】

1. 切换目录执行，或者重新打开一个 shell 终端执行。

[HDFS-40002]HDFS shell 命令使用失败

【问题背景与现象】

在集群外的机器，执行 `hadoop fs -text /tt` 失败。报错为空指针，且在集群内机器执行 OK，在集群外机器执行失败。

错误日志：

```
2015-05-28 08:39:32,384 | WARN | IPC Server handler 53 on 25000 | Exception running
/opt/huawei/Bigdata/etc/1_6_NameNode/topo.sh 10.64.35.139 |
ExitCodeException exitCode=1: 'import site' failed; use -v for traceback
Traceback (most recent call last):
  File "/opt/huawei/Bigdata/etc/1_6_NameNode/topology.py", line 2, in <module>
    from string import join
ImportError: No module named string
```

【原因分析】

1. omm 用户执行 `sh /opt/huawei/Bigdata/etc/1_6_NameNode/topo.py` 确认是 python 命令原因
2. 在 root 下和 omm 下分别执行 `which python`，不是都指向 `/usr/bin/python`，安装了多个版本 python
3. 运行 `/usr/bin/python`，报错。和客户确认，在该 datanode 节点他安装了新 python。

【解决办法】

1. 删除新的 python。不建议客户在运行 hadoop 集群上随意安装软件，尤其 python 这种语言级的。

【数据分布不均衡】

[HDFS-50001]非 HDFS 数据残留导致数据分布不均衡

【问题背景与现象】

数据出现不均衡，磁盘 1 过满而其他磁盘未写满。

HDFS DataNode 数据存储目录配置为 /export/data1/dfs--/export/data12/dfs

目前看到的现象是大量数据都是存储到了/export/data1/dfs，其他盘的数据比较均衡

【原因分析】

1. 客户磁盘为卸载重装，有一个目录在上次卸载时未卸载干净。

【解决办法】

1. 手动清理未卸载干净的数据。

[HDFS-50002]客户端安装在数据节点导致数据分布不均衡

【问题背景与现象】

HDFS 的 datanode 数据分布不均匀，在 101 节点上磁盘使用达到 100%，其他节点空闲很多。

【原因分析】

1. 客户端安装在 101 节点，根据 HDFS 数据副本机制，第一个副本会存放在本地机器，最终导致 101 节点磁盘被占满，其他节点空闲很多。

【解决办法】

1. 将客户端安装在管理节点或者集群外某台集群，然后做 balance 均衡数据。

【备注说明】

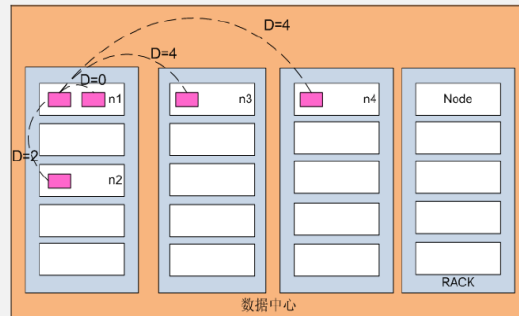
数据副本机制

副本距离计算公式：

- $\text{distance}(d1/r1/n1, d1/r1/n1)=0$ 同一台服务器的距离为0
- $\text{distance}(d1/r1/n1, d1/r1/n2)=2$ 同一机架不同的服务器距离为2
- $\text{distance}(d1/r1/n1, d1/r2/n3)=4$ 不同机架的服务器距离为4

副本放置策略：

- 第一个副本在本地机器
- 第二个副本在远端机架
- 第三个副本看之前的两个副本是否在同一机架，如果是则选择其他机架，否则选择和第一个副本相同机架的不同节点。
- 第四个及以上，随机选择副本存放位置



副本放置策略

[HDFS-50003]MR 任务导致数据分布不均衡

【问题背景与现象】

HDFS 的 datanode 数据分布不均匀，某个节点空间使用率增长很快，其他节点增长比较缓慢平均。

【原因分析】

1. 查看是否有正在执行的 MR 任务，并且查看是否有 map 或者 reduce 正在该节点执行。
2. 如果刚好有 task 在该节点长时间写数据，根据 HDFS 数据副本机制，会导致本节点空间使用率增长很快，最终造成数据倾斜。

SUCCESSFUL MAP attempts in job_1457003801642_0015									
Attempt	State	Status	Node	Logs	Start Time	Finish Time	Elapsed Time		
attempt_1457003801642_0015_m_000000_0	SUCCEEDED	Copying hdfs://hacluster/tmp/FusionInsight_HD_V100R002C30LCN001_RHEL.tar.gz to hdfs://189.39.151.223:25000/tmp/FusionInsight_HD_V100R002C30LCN001_RHEL.tar.gz	hdfs://189.39.151.223:25000/tmp/FusionInsight_HD_V100R002C30LCN001_RHEL.tar.gz	logs	Tue Mar 8 16:55:46 +0800 2016	Tue Mar 8 16:55:52 +0800 2016	5sec		
Attempt	State	Status	Node	Logs	Start Time	Finish Time	Elapsed Time		

【解决办法】

1. 如果该任务可以 kill，可以先 kill 掉这个任务。
2. 如果磁盘空间使用率没有自己降下来，就需要做 balance 均衡数据。

【咨询类】

[HDFS-60001]Hdfs balance 调优参数设置

【问题背景与现象】

当 hdfs 集群各个 datanode 存储的数据不均衡时，需要使用 hdfs balance 功能，调整相关参数可以提升 balance 性能

【原因分析】

修改如下参数：

1. dfs.datanode.balance.bandwidthPerSec =1GB (if possible keep 5 Gb also).
说明：这个参数要看组网情况，如果是用 10G 网络，可以调整为 5Gb。
2. dfs.datanode.max.transfer.threads = 8192
3. dfs.namenode.replication.max-streams=20
4. dfs.namenode.handler.count=25
5. dfs.datanode.handler.count= 20
6. dfs.datanode.balance.max.concurrent.moves=30

【解决办法】

见上述问题分析。

[HDFS-60002]文件副本设置方法

【问题背景与现象】

dfs.replication 在 client 和 server 不同配置下显示不同的效果。

客户端配置 dfs.replication=3，

更改服务端配置 dfs.replication=2 并重启服务，put 数据备份数仍然是 3。

客户端不配置，服务端配置 dfs.replication=2，put 数据备份数为 3 却不是 2。

【原因分析】

针对上述现象，作了以下几组实验了说明该参数配置问题：

关于dfs.replication参数客户端/服务端配置情况的测试情况					
		client	server	actual	
				Default replication factor	Average block replication
test1	dfs.replication	-	3	3	3
test2	dfs.replication	-	2	2	3
test3	dfs.replication	2	3	3	2
test4	dfs.replication	2	2	2	2

结论:

1. 当客户端不配置该参数，则默认将只用备份数为 3。
2. 服务端的 `dfs.replication` 配置影响 Default replication factor，该因子为 hdfs 检查备份数是否符合要求的因子，若不符合会上报复制备份操作。
3. 实际文件的备份数由客户端提供，若配置则按配置生效，若不配置，则默认使用备份数为 3，不是 server 端 `hdfs-default` 配置参数，由代码配置。

【解决办法】

在客户端的 `hdfs-site.xml` 配置 `dfs.replication` 参数来更改上传 hdfs 的文件备份数。

1. 若是 hdfs，则在 `$Client/HDFS/hadoop/etc/hadoop/hdfs-site.xml` 配置；
2. 若是 spark 应用，则在 `$Client/Spark/spark/conf/hdfs-site.xml` 配置。

[HDFS-60003]HDFS 容量规划

HDFS DataNode 以 Block 的形式，保存用户的文件和目录，同时在 NameNode 中生成一个文件对象，对应 DataNode 中每个文件、目录和 Block。

NameNode 中文件对象需要占用一定的内存，消耗内存大小随文件对象的生成而线性递增。DataNode 实际保存的文件和目录越多，NameNode 文件对象总量增加，需要消耗更多的内存，使集群现有硬件可能会难以满足业务需求，且导致集群难以扩展。

规划存储大量文件的 HDFS 系统容量，就是规划 NameNode 的容量规格和 DataNode 的容量规格，并根据容量设置参数。

容量规格

● NameNode 容量规格

在 NameNode 中，每个文件对象对应 DataNode 中的一个文件、目录或 Block。

一个文件至少占用一个 Block，默认每个 Block 大小为“134217728”即 128MB，对应参数为“`dfs.blocksize`”。默认情况下一个文件小于 128MB 时，只占用一个 Block；文件大于 128MB 时，占用 Block 数为：文件大小 ÷ 128MB。目录不占用 Block。

根据“`dfs.blocksize`”，NameNode 的文件对象数计算方法如下：

NameNode 文件对象数计算：

单个文件大小	文件对象数
小于 128MB	1（对应文件）+1（对应 Block）=2
大于 128MB（例如 128G）	1（对应文件）+1,024（对应 $128GB \div 128MB = 1024$ Block）=1,025

主备 NameNode 支持最大文件对象的数量为 300,000,000（最多对应 150,000,000 个小文件）。“`dfs.namenode.max.objects`”规定当前系统可生成的文件对象数，默认值为“0”表示不限制。

● DataNode 容量规格

在 HDFS 中，Block 以副本的形式存储在 DataNode 中，默认副本数为“3”，对应参数为“`dfs.replication`”。

集群中所有 DataNode 角色实例保存的 Block 总数为：HDFS Block * 3。集群中每个 DataNode 实例平均保存的 Blocks = HDFS Block * 3 ÷ DataNode 节点数。

DataNode 支持规格：

项目	规格
单个 DataNode 实例支持最大 Block 副本数	5,000,000
单个 DataNode 实例上单个磁盘支持最大 Block 副本数	500,000
单个 DataNode 实例支持最大 Block 副本数需要的最小磁盘数	10

DataNode 节点数规划:

HDFS Block 数	最少 DataNode 角色实例数
10,000,000	$10,000,000 \div 5,000,000 = 6$
50,000,000	$50,000,000 \div 5,000,000 = 30$
100,000,000	$100,000,000 \div 5,000,000 = 60$

查看 HDFS 容量状态

● NameNode 信息

登录 FusionInsight Manager, 选择“服务管理 > HDFS > NameNode(主)”, 单击“Overview”, 查看“Summary”显示的当前 HDFS 中文件对象、文件数量、目录数量和 Block 数量信息。

● DataNode 信息

登录 FusionInsight Manager, 选择“服务管理 > HDFS > NameNode(主)”, 单击“DataNodes”, 查看所有告警 DataNode 节点的 Block 数量信息。

[HDFS-60004]内存参数设置

● NameNode JVM 参数配置规则

NameNode JVM 参数“GC_OPTS”默认值为:

```
-Xms2G -Xmx4G -XX:NewSize=128M -XX:MaxNewSize=256M -XX:MetaspaceSize=128M -XX:MaxMetaspaceSize=128M
-XX:CMSFullGCsBeforeCompaction=1 -XX:MaxDirectMemorySize=1G -XX:+UseConcMarkSweepGC -
XX:+CMSParallelRemarkEnabled -XX:+UseCMSCompactAtFullCollection -XX:CMSInitiatingOccupancyFraction=65 -
Xloggc:{BigdataLogHome}/hdfs/nn/namenode-omm-gc.log -XX:+PrintGCDetails -
Dsun.rmi.dgc.client.gcInterval=0x7FFFFFFF -Dsun.rmi.dgc.server.gcInterval=0x7FFFFFFF -XX:-
OmitStackTraceInFastThrow -XX:+PrintGCDateStamps -XX:+UseGCLogFileRotation -XX:NumberOfGCLogFiles=10 -
XX:GCLogFileSize=1M
```

NameNode 文件数量和 NameNode 使用的内存大小成比例关系, 文件对象变化时请修改默认值中的“-Xms2G -Xmx4G -XX:NewSize=128M -XX:MaxNewSize=256M”。参考值如下表所示。

NameNode JVM 配置:

文件对象数量	参考值
10,000,000	“-Xms6G -Xmx6G -XX:NewSize=512M -XX:MaxNewSize=512M”
20,000,000	“-Xms12G -Xmx12G -XX:NewSize=1G -XX:MaxNewSize=1G”
50,000,000	“-Xms32G -Xmx32G -XX:NewSize=2G -XX:MaxNewSize=3G”
100,000,000	“-Xms64G -Xmx64G -XX:NewSize=4G -XX:MaxNewSize=6G”
200,000,000	“-Xms96G -Xmx96G -XX:NewSize=8G -XX:MaxNewSize=9G”
300,000,000	“-Xms164G -Xmx164G -XX:NewSize=12G -XX:MaxNewSize=12G”

● DataNode JVM 参数配置规则

DataNode JVM 参数 “GC_OPTS” 默认值为:

-Xms2G -Xmx4G -XX:NewSize=128M -XX:MaxNewSize=256M -XX:MetaspaceSize=128M -XX:MaxMetaspaceSize=128M
-XX:CMSFullGCsBeforeCompaction=1 -XX:MaxDirectMemorySize=1G -XX:+UseConcMarkSweepGC -
XX:+CMSParallelRemarkEnabled -XX:+UseCMSCompactAtFullCollection -XX:CMSInitiatingOccupancyFraction=65 -
Xloggc:{BigdataLogHome}/hdfs/dn/datanode-omm-gc.log -XX:+PrintGCDetails -
Dsun.rmi.dgc.client.gcInterval=0x7FFFFFFFFFFFFFFFE -Dsun.rmi.dgc.server.gcInterval=0x7FFFFFFFFFFFFFFFE -XX:-
OmitStackTraceInFastThrow -XX:+PrintGCDateStamps -XX:+UseGCLogFileRotation -XX:NumberOfGCLogFiles=10 -
XX:GCLogFileSize=1M

集群中每个 DataNode 实例平均保存的 Blocks= HDFS Block * 3 ÷ DataNode 节点数，单个 DataNode 实例平均 Block 数量变化时请修改默认值中的 “-Xms2G -Xmx4G -XX:NewSize=128M -XX:MaxNewSize=256M”。参考值如下表所示。

DataNode JVM 配置:

单个 DataNode 实例平均 Block 数量	参考值
2,000,000	“-Xms4G -Xmx4G -XX:NewSize=256M -XX:MaxNewSize=256M”
5,000,000	“-Xms8G -Xmx8G -XX:NewSize=512M -XX:MaxNewSize=512M”