[【实验】Hadoop-2.8.1+zookeeper-3.4.9完全分布式环境搭建(HDFS、YARN HA)](http://blog.itpub.net/30089851/viewspace-1994585/)

**Hadoop-2.8.1+Zookeeper-3.4.9完全分布式环境搭建**

**一.版本**

|  |  |  |
| --- | --- | --- |
| 组件名 | 版本 | 说明 |
| JRE | java version "1.8.0\_45"  Java(TM) SE Runtime Environment (build 1.8.0\_45-b14)  Java HotSpot(TM) 64-Bit Server VM (build 25.45-b02, mixed mode) |  |
| Hadoop | hadoop-2.8.1.tar.gz | 主程序包 |
| Zookeeper | zookeeper-3.4.9.tar.gz | 热切,Yarn 存储数据使用的协调服务 |

**二.主机规划**

|  |  |  |  |
| --- | --- | --- | --- |
| IP | Host 及安装软件 | 部署模块 | 进程 |
| 192.168.240.11 | hadoop01:hadoop、zookeeper | DataNode  NodeManager  Zookeeper  NameNode  ResourceManager | DataNode  NodeManager  JournalNode  QuorumPeerMain  NameNode  DFSZKFailoverController  ResourceManager |
| 192.168.240.12 | hadoop02:Hadoop、zookeeper | NameNode  ResourceManager | DataNode  NodeManager  JournalNode  QuorumPeerMain  NameNode  DFSZKFailoverController  ResourceManager |
| 192.168.240.13 | hadoop03:Hadoop、zookeeper | DataNode  NodeManager  Zookeeper | DataNode  NodeManager  JournalNode  QuorumPeerMain |

**三.目录规划**

|  |  |
| --- | --- |
| 名称 | 路径 |
| $HADOOP\_HOME | /hadoop/hadoop-2.8.1 |
| Data | $HADOOP\_HOME/data |
| Log | $HADOOP\_HOME/logs |

**四.常用脚本及命令**

**1.启动集群**

start-dfs.sh

start-yarn.sh

**2.关闭集群**

stop-yarn.sh

stop-dfs.sh

**3.监控集群**

hdfs dfsadmin -report

**4.单个进程启动/关闭**

hadoop-daemon.sh start|stop namenode|datanode| journalnode

yarn-daemon.sh start |stop resourcemanager|nodemanager

**五.环境准备**

**1 .设置ip地址(5台)**

**2 .关闭防火墙(5台)**

**4 .设置主机名(3台)**

**5 .5 ip与hostname绑定(5台)**

**6 .设置5台machines,SSH互相通信**

**7 .安装JDK(5台)**

1. (1)执行命令
2. [root@sht-sgmhadoopnn-01 ~]# cd /usr/java
3. [root@sht-sgmhadoopnn-01 java]# cp /tmp/jdk-8u45-linux-x64.gz ./
4. [root@sht-sgmhadoopnn-01 java]# tar -xzvf jdk-8u45-linux-x64.gz
5. (2)vi /etc/profile 增加内容如下:
6. export JAVA\_HOME=/usr/java/jdk1.8.0\_45
7. export HADOOP\_HOME=/hadoop/hadoop-2.8.1
8. export ZOOKEEPER\_HOME=/hadoop/zookeeper
9. export PATH=.:$HADOOP\_HOME/bin:$JAVA\_HOME/bin:$ZOOKEEPER\_HOME/bin:$PATH
10. #先把HADOOP\_HOME, ZOOKEEPER\_HOME配置了
11. (3)执行 source /etc/profile
12. (4)验证:java -version

**8.创建文件夹(3台)**

mkdir /Hadoop

**六.安装Zookeeper**

**sht-sgmhadoopdn-01/02/03**

**1.下载解压**zookeeper-3.4.9.tar.gz

1. [root@hadoop01 tmp]# tar -xvf zookeeper-3.4.9.tar.gz
2. [root@hadoop02 tmp]# tar -xvf zookeeper-3.4.9.tar.gz
3. [root@hadoop03 tmp]# tar -xvf zookeeper-3.4.9.tar.gz
4. [root@hadoop01 tmp]# mv zookeeper-3.5.1-alpha /hadoop/zookeeper
5. [root@hadoop02 tmp]# mv zookeeper-3.5.1-alpha /hadoop/zookeeper
6. [root@hadoop03 tmp]# mv zookeeper-3.5.1-alpha /hadoop/zookeeper

**2.修改配置**

1. **[root@hadoop01 tmp]# cd /hadoop/zookeeper/conf**
2. **[root@hadoop01 conf]# cp zoo\_sample.cfg zoo.cfg**
3. **[root@hadoop01 conf]# vi zoo.cfg**
4. **修改dataDir**
5. **dataDir=/hadoop/zookeeper/data**
6. **添加下面三行**
7. **server.1=hadoop01:2888:3888**
8. **server.2=hadoop02:2888:3888**
9. **server.3=hadoop03:2888:3888**
10. **[root@hadoop01 conf]# cd ../**
11. **[root@hadoop01 zookeeper]# mkdir data**
12. **[root@hadoop01 zookeeper]# touch data/myid**
13. **[root@hadoop01 zookeeper]# echo 1 > data/myid**
14. **[root@hadoop01 zookeeper]# more data/myid**
15. **1**
16. **## hadoop01,02,03,也修改配置,就如下不同**
17. **[root@hadoop02 zookeeper]# echo 2 > data/myid**
18. **[root@hadoop03 zookeeper]# echo 3 > data/myid**

**七.安装Hadoop(HDFS HA+YARN HA)**

**1.下载解压hadoop-2.8.1.tar.gz**

**tar -xvf hadoop-2.8.1.tar.gz**

**mv hadoop-2.8.1 /hadoop/**

**#2.修改$HADOOP\_HOME/etc/hadoop/hadoop-env.sh 没做**

#export JAVA\_HOME="/usr/java/jdk1.8.0\_45"

系统里有：export JAVA\_HOME=${JAVA\_HOME}

**3.修改$HADOOP\_HOME/etc/hadoop/core-site.xml**

1. <?xml version="1.0" encoding="UTF-8"?>
2. <?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
3. <configuration>
4. <!--Yarn 需要使用 fs.defaultFS 指定NameNode URI -->
5. <property>
6. <name>fs.defaultFS</name>
7. <value>hdfs://mycluster</value>
8. </property>
9. <!--HDFS超级用户 -->
10. <property>
11. <name>dfs.permissions.superusergroup</name>
12. <value>root</value>
13. </property>
14. <!--==============================Trash机制======================================= -->
15. <property>
16. <!--多长时间创建CheckPoint NameNode截点上运行的CheckPointer 从Current文件夹创建CheckPoint;默认：0 由fs.trash.interval项指定 -->
17. <name>fs.trash.checkpoint.interval</name>
18. <value>0</value>
19. </property>
20. <property>
21. <!--多少分钟.Trash下的CheckPoint目录会被删除,该配置服务器设置优先级大于客户端，默认：0 不删除 -->
22. <name>fs.trash.interval</name>
23. <value>1440</value>
24. </property>
25. </configuration>

**4.修改$HADOOP\_HOME/etc/hadoop/hdfs-site.xml**

1. [root@hadoop02 sbin]# cat $HADOOP\_HOME/etc/hadoop/hdfs-site.xml
2. <?xml version="1.0" encoding="UTF-8"?>
3. <?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
4. <!--
5. Licensed under the Apache License, Version 2.0 (the "License");
6. you may not use this file except in compliance with the License.
7. You may obtain a copy of the License at
8. http://www.apache.org/licenses/LICENSE-2.0
9. Unless required by applicable law or agreed to in writing, software
10. distributed under the License is distributed on an "AS IS" BASIS,
11. WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
12. See the License for the specific language governing permissions and
13. limitations under the License. See accompanying LICENSE file.
14. -->
15. <!-- Put site-specific property overrides in this file. -->
16. <configuration>
17. <property>
18. <name>fs.defaultFS</name>
19. <value>hdfs://mycluster</value>
20. </property>
21. <property>
22. <name>dfs.permissions.superusergroup</name>
23. <value>root</value>
24. </property>
25. <!--==============================Trash introl======================================= -->
26. <property>
27. <name>fs.trash.checkpoint.interval</name>
28. <value>0</value>
29. </property>
30. <property>
31. <name>fs.trash.interval</name>
32. <value>1440</value>
33. </property>
34. <property>
35. <name>dfs.webhdfs.enabled</name>
36. <value>true</value>
37. </property>
38. <property>
39. <name>dfs.namenode.name.dir</name>
40. <value>/hadoop/hadoop-2.8.1/data/dfs/name</value>
41. <description> namenode save name table(fsimage) local dir</description>
42. </property>
43. <property>
44. <name>dfs.namenode.edits.dir</name>
45. <value>${dfs.namenode.name.dir}</value>
46. <description>namenode transaction file(edits) local dir</description>
47. </property>
48. <property>
49. <name>dfs.datanode.data.dir</name>
50. <value>/hadoop/hadoop-2.8.1/data/dfs/data</value>
51. <description>datanode save block dir</description>
52. </property>
53. <property>
54. <name>dfs.replication</name>
55. <value>3</value>
56. </property>
57. <property>
58. <name>dfs.blocksize</name>
59. <value>268435456</value>
60. </property>
61. <!--======================================================================= -->
62. <property>
63. <name>dfs.nameservices</name>
64. <value>mycluster</value>
65. </property>
66. <property>
67. <name>dfs.ha.namenodes.mycluster</name>
68. <value>nn1,nn2</value>
69. </property>
70. <property>
71. <name>dfs.namenode.rpc-address.mycluster.nn1</name>
72. <value>hadoop01:8020</value>
73. </property>
74. <property>
75. <name>dfs.namenode.rpc-address.mycluster.nn2</name>
76. <value>hadoop01:8020</value>
77. </property>
78. <!-- Hdfs HA: dfs.namenode.http-address.[nameservice ID] http -->
79. <property>
80. <name>dfs.namenode.http-address.mycluster.nn1</name>
81. <value>hadoop01:50070</value>
82. </property>
83. <property>
84. <name>dfs.namenode.http-address.mycluster.nn2</name>
85. <value>hadoop02:50070</value>
86. </property>
87. <property>
88. <name>dfs.journalnode.http-address</name>
89. <value>0.0.0.0:8480</value>
90. </property>
91. <property>
92. <name>dfs.journalnode.rpc-address</name>
93. <value>0.0.0.0:8485</value>
94. </property>
95. <property>
96. <name>dfs.namenode.shared.edits.dir</name>
97. <value>qjournal://hadoop01:8485;hadoop02:8485;hadoop03:8485/mycluster</value>
98. </property>
99. <property>
100. <name>dfs.journalnode.edits.dir</name>
101. <value>/hadoop/hadoop-2.8.1/data/dfs/jn</value>
102. </property>
103. <property>
104. <!--DataNode,Client -->
105. <name>dfs.client.failover.proxy.provider.mycluster</name>
106. <value>org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider</value>
107. </property>
108. <!--==================Namenode fencing =============================================== -->
109. <!--Failover-->
110. <property>
111. <name>dfs.ha.fencing.methods</name>
112. <value>sshfence</value>
113. </property>
114. <property>
115. <name>dfs.ha.fencing.ssh.private-key-files</name>
116. <value>/root/.ssh/id\_rsa</value>
117. </property>
118. <property>
119. <!--milliseconds fencing false -->
120. <name>dfs.ha.fencing.ssh.connect-timeout</name>
121. <value>30000</value>
122. </property>
123. <!--==================NameNode auto failover base ZKFC and Zookeeper====================== -->
124. <!--Zookeeper ZKFC monitor -->
125. <property>
126. <name>dfs.ha.automatic-failover.enabled</name>
127. <value>true</value>
128. </property>
129. <property>
130. <name>ha.zookeeper.quorum</name>
131. <value>hadoop01:2181,hadoop02:2181,hadoop03:2181</value>
132. </property>
133. <property>
134. <!--ZooKeeper time ms -->
135. <name>ha.zookeeper.session-timeout.ms</name>
136. <value>2000</value>
137. </property>
138. </configuration>

**5.修改**

#Yarn Daemon Options

#export YARN\_RESOURCEMANAGER\_OPTS

#export YARN\_NODEMANAGER\_OPTS

#export YARN\_PROXYSERVER\_OPTS

#export HADOOP\_JOB\_HISTORYSERVER\_OPTS

#Yarn Logs

export YARN\_LOG\_DIR="/hadoop/hadoop-2.8.1/logs"

**6.修改$HADOOP\_HOME/etc/hadoop/mapred-site.xml**

1. **cd $HADOOP\_HOME/etc/hadoop/**
2. **[root@**hadoop01**…]# cp mapred-site.xml.template mapred-site.xml**
3. **[root@**hadoop01**…]# vi mapred-site.xml**
4. **<configuration>**
5. **<property>**
6. **<name>mapreduce.framework.name</name>**
7. **<value>yarn</value>**
8. **</property>**
9. **<!-- JobHistory Server ============================================================== -->**
10. **<!-- MapReduce JobHistory Server address ,default: 0.0.0.0:10020 -->**
11. **<property>**
12. **<name>mapreduce.jobhistory.address</name>**
13. **<value>hadoop01:10020</value>**
14. **</property>**
15. **<!-- config MapReduce JobHistory Server web ui address ,default: 0.0.0.0:19888 -->**
16. **<property>**
17. **<name>mapreduce.jobhistory.webapp.address</name>**
18. **<value>hadoop01:19888</value>**
19. **</property></configuration>**

**7.修改$HADOOP\_HOME/etc/hadoop/yarn-site.xml**

1. **<configuration>**
2. <!-- nodemanager configure ================================================= -->
3. <property>
4. <name>yarn.nodemanager.aux-services</name>
5. <value>mapreduce\_shuffle</value>
6. </property>
7. <property>
8. <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
9. <value>org.apache.hadoop.mapred.ShuffleHandler</value>
10. </property>
11. <property>
12. <description>Address where the localizer IPC is.</description>
13. <name>yarn.nodemanager.localizer.address</name>
14. <value>0.0.0.0:23344</value>
15. </property>
16. <property>
17. <description>NM Webapp address.</description>
18. <name>yarn.nodemanager.webapp.address</name>
19. <value>0.0.0.0:23999</value>
20. </property>
21. <!-- HA configure =============================================================== -->
22. <!-- Resource Manager Configs -->
23. <property>
24. <name>yarn.resourcemanager.connect.retry-interval.ms</name>
25. <value>2000</value>
26. </property>
27. <property>
28. <name>yarn.resourcemanager.ha.enabled</name>
29. <value>true</value>
30. </property>
31. <property>
32. <name>yarn.resourcemanager.ha.automatic-failover.enabled</name>
33. <value>true</value>
34. </property>
35. <property>
36. <name>yarn.resourcemanager.ha.automatic-failover.embedded</name>
37. <value>true</value>
38. </property>
39. <!-- cluster -->
40. <property>
41. <name>yarn.resourcemanager.cluster-id</name>
42. <value>yarn-cluster</value>
43. </property>
44. <property>
45. <name>yarn.resourcemanager.ha.rm-ids</name>
46. <value>rm1,rm2</value>
47. </property>
48. <property>
49. <name>yarn.resourcemanager.ha.id</name>
50. <value>rm2</value>
51. </property>
52. <property>
53. <name>yarn.resourcemanager.scheduler.class</name>
54. <value>org.apache.hadoop.yarn.server.resourcemanager.scheduler.fair.FairScheduler</value>
55. </property>
56. <property>
57. <name>yarn.resourcemanager.recovery.enabled</name>
58. <value>true</value>
59. </property>
60. <property>
61. <name>yarn.app.mapreduce.am.scheduler.connection.wait.interval-ms</name>
62. <value>5000</value>
63. </property>
64. <!-- ZKRMStateStore configure -->
65. <property>
66. <name>yarn.resourcemanager.store.class</name>
67. <value>org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore</value>
68. </property>
69. <property>
70. <name>yarn.resourcemanager.zk-address</name>
71. <value>hadoop01:2181,hadoop02:2181,hadoop03:2181</value>
72. </property>
73. <property>
74. <name>yarn.resourcemanager.zk.state-store.address</name>
75. <value>hadoop01:2181,hadoop02:2181,hadoop03:2181</value>
76. </property>
77. <!-- ClientvisitRM'sRPCaddress (applications manager interface) -->
78. <property>
79. <name>yarn.resourcemanager.address.rm1</name>
80. <value>hadoop01:23140</value>
81. </property>
82. <property>
83. <name>yarn.resourcemanager.address.rm2</name>
84. <value>hadoop02:23140</value>
85. </property>
86. <!-- AMvisitRM'sRPCaddress(scheduler interface) -->
87. <property>
88. <name>yarn.resourcemanager.scheduler.address.rm1</name>
89. <value>hadoop01:23130</value>
90. </property>
91. <property>
92. <name>yarn.resourcemanager.scheduler.address.rm2</name>
93. <value>hadoop02:23130</value>
94. </property>
95. <!-- RM admin interface -->
96. <property>
97. <name>yarn.resourcemanager.admin.address.rm1</name>
98. <value>hadoop01:23141</value>
99. </property>
100. <property>
101. <name>yarn.resourcemanager.admin.address.rm2</name>
102. <value>hadoop02:23141</value>
103. </property>
104. <!--NMvisitRM's RPC port -->
105. <property>
106. <name>yarn.resourcemanager.resource-tracker.address.rm1</name>
107. <value>hadoop01:23125</value>
108. </property>
109. <property>
110. <name>yarn.resourcemanager.resource-tracker.address.rm2</name>
111. <value>hadoop02:23125</value>
112. </property>
113. <!-- RM web application address -->
114. <property>
115. <name>yarn.resourcemanager.webapp.address.rm1</name>
116. <value>hadoop01:8088</value>
117. </property>
118. <property>
119. <name>yarn.resourcemanager.webapp.address.rm2</name>
120. <value>hadoop02:8088</value>
121. </property>
122. <property>
123. <name>yarn.resourcemanager.webapp.https.address.rm1</name>
124. <value>hadoop01:23189</value>
125. </property>
126. <property>
127. <name>yarn.resourcemanager.webapp.https.address.rm2</name>
128. <value>hadoop02:23189</value>
129. </property>
130. **</configuration>**

**8.修改slaves(/hadoop/hadoop-2.8.1/etc/hadoop)**

[root@hadoop01 hadoop]# cd **/hadoop/hadoop-2.8.1/etc/hadoop**

[root@hadoop01 hadoop]# vi slaves

hadoop01

hadoop02

hadoop03

**9.分发文件夹**

[root@hadoop01 hadoop]# scp -r hadoop-2.8.1 root@hadoop01:/hadoop

[root@hadoop01 hadoop]# scp -r hadoop-2.8.1 root@hadoop02:/hadoop

[root@hadoop01 hadoop]# scp -r hadoop-2.8.1 root@hadoop03:/hadoop

**八.启动集群**

另外一种启动方式:<http://www.micmiu.com/bigdata/hadoop/hadoop2-cluster-ha-setup/>

**1.启动zookeeper（每台都启动）**

1. command: ./zkServer.sh start|stop|status
2. cd /hadoop/zookeeper/bin
3. [root@hadoop01 bin]# ./zkServer.sh start
4. JMX enabled by default
5. Using config: /hadoop/zookeeper/bin/../conf/zoo.cfg
6. Starting zookeeper ... STARTED
7. [root@hadoop01 bin]# jps
8. 2073 QuorumPeerMain
9. 2106 Jps
10. [root@hadoop02 bin]# ./zkServer.sh start
11. JMX enabled by default
12. Using config: /hadoop/zookeeper/bin/../conf/zoo.cfg
13. Starting zookeeper ... STARTED
14. [root@hadoop02 bin]# jps
15. 2073 QuorumPeerMain
16. 2106 Jps
17. [root@hadoop03 bin]# ./zkServer.sh start
18. JMX enabled by default
19. Using config: /hadoop/zookeeper/bin/../conf/zoo.cfg
20. Starting zookeeper ... STARTED
21. [root@hadoop03 bin]# jps
22. 2073 QuorumPeerMain
23. 2106 Jps

**2.启动hadoop(HDFS+YARN)**

***a.格式化前,先在journalnode 节点机器上先启动JournalNode进程***

1. [root@hadoop01 ~]# cd /hadoop/hadoop-2.8.1/sbin
2. [root@hadoop01 sbin]# hadoop-daemon.sh start journalnode
3. starting journalnode, logging to /hadoop/hadoop-2.8.1/logs/hadoop-root-journalnode-hadoop03.telenav.cn.out
4. [root@hadoop03 sbin]# jps
5. 16722 JournalNode
6. 16775 Jps
7. 15519 QuorumPeerMain
8. [root@hadoop02 ~]# cd /hadoop/hadoop-2.8.1/sbin
9. [root@hadoop02 sbin]# hadoop-daemon.sh start journalnode
10. starting journalnode, logging to /hadoop/hadoop-2.8.1/logs/hadoop-root-journalnode-hadoop03.telenav.cn.out
11. [root@hadoop03 sbin]# jps
12. 16722 JournalNode
13. 16775 Jps
14. 15519 QuorumPeerMain
15. [root@hadoop03 ~]# cd /hadoop/hadoop-2.8.1/sbin
16. [root@hadoop03 sbin]# hadoop-daemon.sh start journalnode
17. starting journalnode, logging to /hadoop/hadoop-2.8.1/logs/hadoop-root-journalnode-hadoop03.telenav.cn.out
18. [root@hadoop03 sbin]# jps
19. 16722 JournalNode
20. 16775 Jps
21. 15519 QuorumPeerMain

***b.NameNode格式化***

1. [root@sht-sgmhadoopnn-01 bin]# hadoop namenode -format
2. 16/02/25 14:05:04 INFO namenode.NameNode: STARTUP\_MSG:
3. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
4. STARTUP\_MSG: Starting NameNode
5. STARTUP\_MSG: host = sht-sgmhadoopnn-01.telenav.cn/172.16.101.55
6. STARTUP\_MSG: args = [-format]
7. STARTUP\_MSG: version = 2.7.2
8. STARTUP\_MSG: classpath =
9. ……………..
10. ………………
11. 16/02/25 14:05:07 INFO namenode.FSNamesystem: dfs.namenode.safemode.threshold-pct = 0.9990000128746033
12. 16/02/25 14:05:07 INFO namenode.FSNamesystem: dfs.namenode.safemode.min.datanodes = 0
13. 16/02/25 14:05:07 INFO namenode.FSNamesystem: dfs.namenode.safemode.extension = 30000
14. 16/02/25 14:05:07 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.window.num.buckets = 10
15. 16/02/25 14:05:07 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.num.users = 10
16. 16/02/25 14:05:07 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.windows.minutes = 1,5,25
17. 16/02/25 14:05:07 INFO namenode.FSNamesystem: Retry cache on namenode is enabled
18. 16/02/25 14:05:07 INFO namenode.FSNamesystem: Retry cache will use 0.03 of total heap and retry cache entry expiry time is 600000 millis
19. 16/02/25 14:05:07 INFO util.GSet: Computing capacity for map NameNodeRetryCache
20. 16/02/25 14:05:07 INFO util.GSet: VM type = 64-bit
21. 16/02/25 14:05:07 INFO util.GSet: 0.029999999329447746% max memory 889 MB = 273.1 KB
22. 16/02/25 14:05:07 INFO util.GSet: capacity = 2^15 = 32768 entries
23. 16/02/25 14:05:08 INFO namenode.FSImage: Allocated new BlockPoolId: BP-1182930464-172.16.101.55-1456380308394
24. 16/02/25 14:05:08 INFO common.Storage: Storage directory /hadoop/hadoop-2.8.1/data/dfs/name has been successfully formatted.
25. 16/02/25 14:05:08 INFO namenode.NNStorageRetentionManager: Going to retain 1 images with txid >= 0
26. 16/02/25 14:05:08 INFO util.ExitUtil: Exiting with status 0
27. 16/02/25 14:05:08 INFO namenode.NameNode: SHUTDOWN\_MSG:
28. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
29. SHUTDOWN\_MSG: Shutting down NameNode at sht-sgmhadoopnn-01.telenav.cn/172.16.101.55
30. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

***c.同步NameNode元数据***

1. 同步hadoop01 元数据到hadoop02
2. 主要是：dfs.namenode.name.dir，dfs.namenode.edits.dir还应该确保共享存储目录下(dfs.namenode.shared.edits.dir ) 包含NameNode 所有的元数据。
3. [root@hadoop01 hadoop-2.8.1]# pwd
4. /hadoop/hadoop-2.8.1
5. [root@sht-sgmhadoopnn-01 hadoop-2.8.1]# scp -r data/ root@hadoop02:/hadoop/hadoop-2.8.1
6. seen\_txid 100% 2 0.0KB/s 00:00
7. fsimage\_0000000000000000000 100% 351 0.3KB/s 00:00
8. fsimage\_0000000000000000000.md5 100% 62 0.1KB/s 00:00
9. VERSION 100% 205 0.2KB/s 00:00

***d.初始化ZFCK***

1. [root@hadoop01 bin]# hdfs zkfc -formatZK
2. ……………..
3. ……………..
4. 16/02/25 14:14:41 INFO zookeeper.ZooKeeper: Client environment:user.home=/root
5. 16/02/25 14:14:41 INFO zookeeper.ZooKeeper: Client environment:user.dir=/hadoop/hadoop-2.8.1/bin
6. 16/02/25 14:14:41 INFO zookeeper.ZooKeeper: Initiating client connection, connectString=hadoop01:2181,hadoop02:2181,hadoop03:2181 sessionTimeout=2000 watcher=org.apache.hadoop.ha.ActiveStandbyElector$WatcherWithClientRef@5f4298a5
7. 16/02/25 14:14:41 INFO zookeeper.ClientCnxn: Opening socket connection to server hadoop01.telenav.cn/172.16.101.58:2181. Will not attempt to authenticate using SASL (unknown error)
8. 16/02/25 14:14:41 INFO zookeeper.ClientCnxn: Socket connection established to hadoop01.telenav.cn/172.16.101.58:2181, initiating session
9. 16/02/25 14:14:42 INFO zookeeper.ClientCnxn: Session establishment complete on server hadoop01.telenav.cn/172.16.101.58:2181, sessionid = 0x15316c965750000, negotiated timeout = 4000
10. 16/02/25 14:14:42 INFO ha.ActiveStandbyElector: Session connected.
11. 16/02/25 14:14:42 INFO ha.ActiveStandbyElector: Successfully created /hadoop-ha/mycluster in ZK.
12. 16/02/25 14:14:42 INFO zookeeper.ClientCnxn: EventThread shut down
13. 16/02/25 14:14:42 INFO zookeeper.ZooKeeper: Session: 0x15316c965750000 closed

***e.启动HDFS 系统***

集群启动,在hadoop01执行start-dfs.sh #之前启动过zk，所以这步骤得先执行stop-dfs.sh

集群关闭,在hadoop01执行stop-dfs.sh

#####集群启动############

1. [root@hadoop01 sbin]# start-dfs.sh

####单进程启动###########

**NameNode(hadoop01, hadoop02):**

hadoop-daemon.sh start namenode

**DataNode(hadoop01, hadoop02, hadoop03):**

hadoop-daemon.sh start datanode

**JournamNode(hadoop01, hadoop02, hadoop03):**

hadoop-daemon.sh start journalnode

**ZKFC(hadoop01, hadoop02):**

hadoop-daemon.sh start zkfc

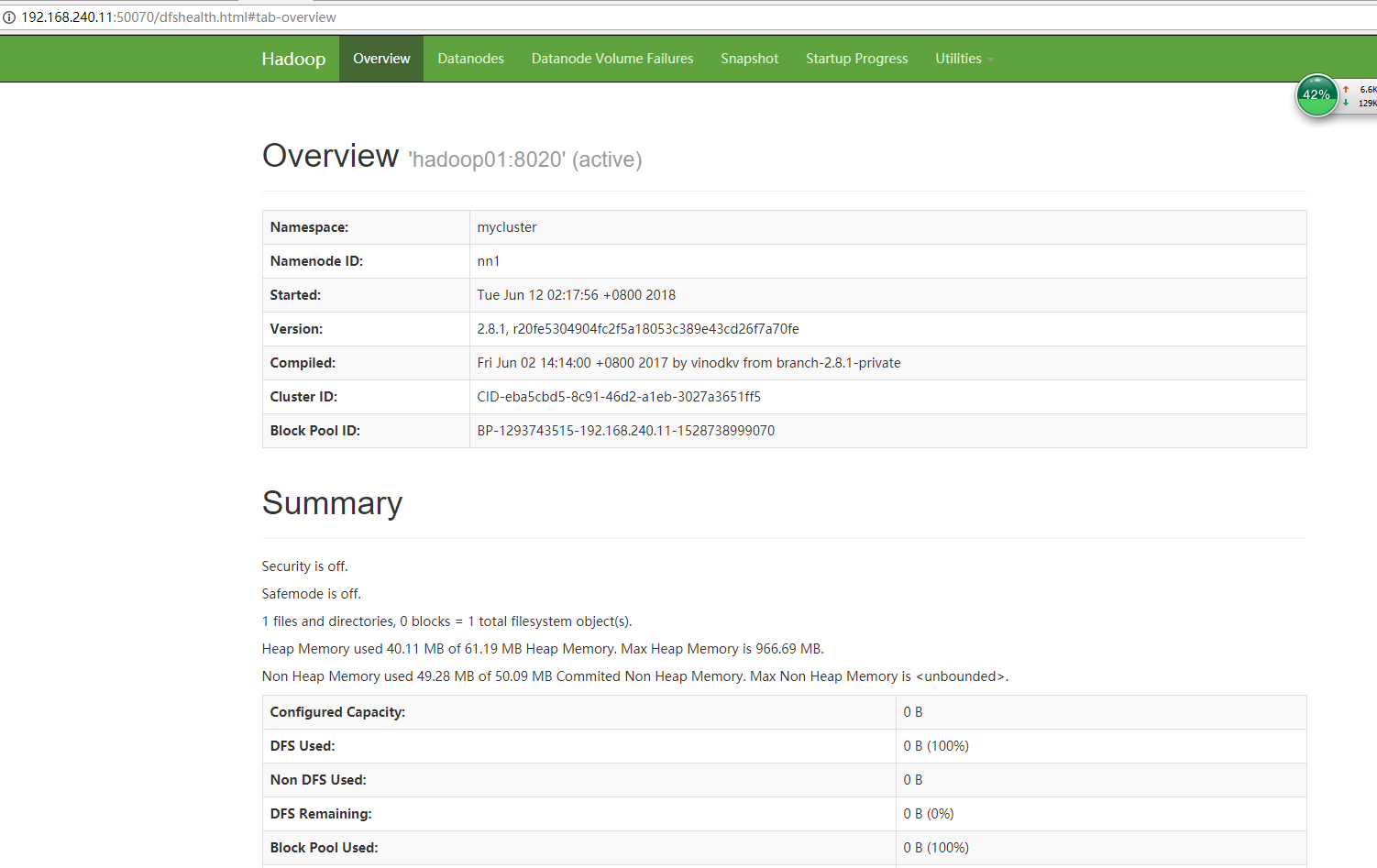
***f.验证namenode,datanode,zkfc***

**1) 进程**

1. [root@hadoop01 sbin]# jps
2. 12712 Jps
3. 12593 DFSZKFailoverController
4. 12278 NameNode
5. [root@sht-sgmhadoopnn-02 ~]# jps
6. 29714 NameNode
7. 29849 DFSZKFailoverController
8. 30229 Jps
9. [root@hadoop01 ~]# jps
10. 6348 JournalNode
11. 8775 Jps
12. 559 QuorumPeerMain
13. 8509 DataNode
14. [root@hadoop02 ~]# jps
15. 9430 Jps
16. 9160 DataNode
17. 7197 JournalNode
18. 2073 QuorumPeerMain
19. [root@hadoop03 ~]# jps
20. 16722 JournalNode
21. 17369 Jps
22. 15519 QuorumPeerMain
23. 17214 DataNode

**2) 页面**

**hadoop01:**http://192.168.240.11:50070/dfshealth.html#tab-overview**sht-sgmhadoopnn-02:**



***g.启动YARN运算框架***

#####集群启动############

**1) hadoop01启动Yarn，命令所在目录：$HADOOP\_HOME/sbin**

1. [root@hadoop01 sbin]# start-yarn.sh
2. starting yarn daemons
3. starting resourcemanager, logging to /hadoop/hadoop-2.8.1/logs/yarn-root-resourcemanager-hadoop01.telenav.cn.out
4. hadoop03: starting nodemanager, logging to /hadoop/hadoop-2.8.1/logs/yarn-root-nodemanager-hadoop03.telenav.cn.out
5. hadoop02: starting nodemanager, logging to /hadoop/hadoop-2.8.1/logs/yarn-root-nodemanager-hadoop02.telenav.cn.out
6. hadoop01: starting nodemanager, logging to /hadoop/hadoop-2.8.1/logs/yarn-root-nodemanager-hadoop01.telenav.cn.out

**2) hadoop02备机启动RM**

1. [root@hadoop02 sbin]# yarn-daemon.sh start resourcemanager
2. starting resourcemanager, logging to /hadoop/hadoop-2.8.1/logs/yarn-root-resourcemanager-hadoop02.telenav.cn.out

####单进程启动###########

**1) ResourceManager(hadoop01, hadoop02)**

yarn-daemon.sh start resourcemanager

**2) NodeManager(hadoop01, hadoop02, hadoop03)**

yarn-daemon.sh start nodemanager

######关闭#############

[root@hadoop01 sbin]# stop-yarn.sh

#包含namenode的resourcemanager进程，datanode的nodemanager进程

[root@hadoop02 sbin]# yarn-daemon.sh stop resourcemanager

***h.验证resourcemanager,nodemanager***

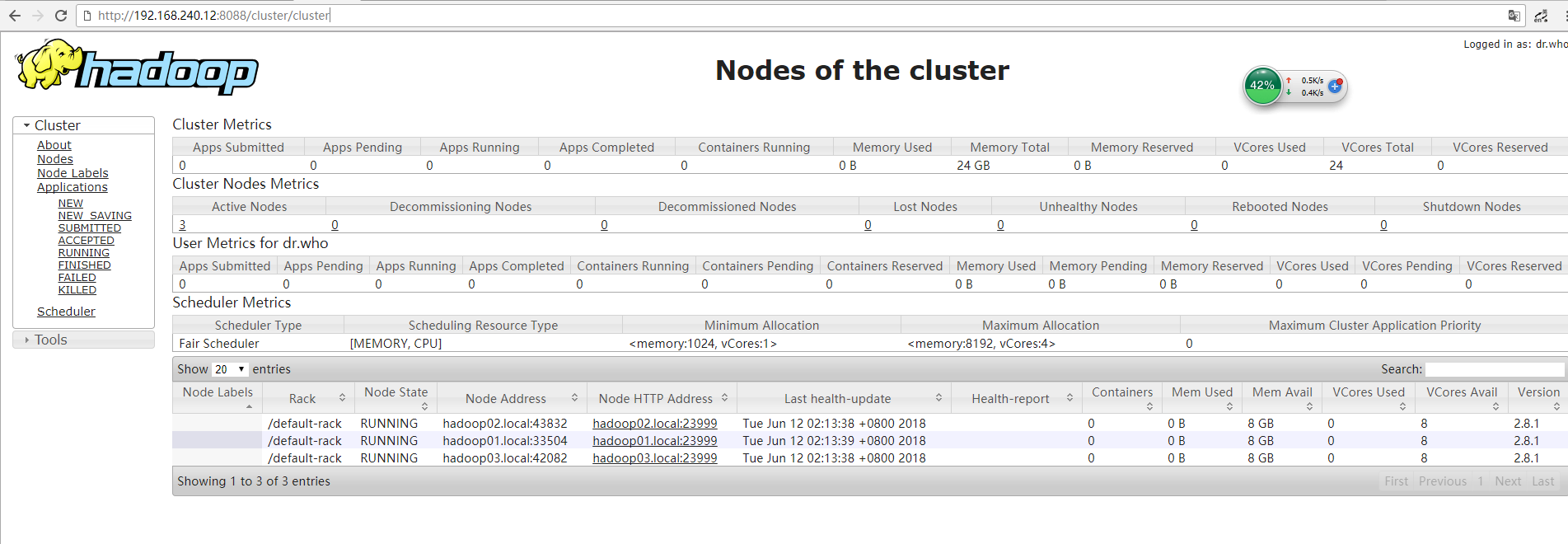
**1) 进程**

1. [root@hadoop01 sbin]# jps
2. 13611 Jps
3. 12593 DFSZKFailoverController
4. 12278 NameNode
5. 13384 ResourceManager
6. [root@sht-sgmhadoopnn-02 sbin]# jps
7. 32265 ResourceManager
8. 32304 Jps
9. 29714 NameNode
10. 29849 DFSZKFailoverController
11. [root@hadoop01 ~]# jps
12. 6348 JournalNode
13. 559 QuorumPeerMain
14. 8509 DataNode
15. 10286 NodeManager
16. 10423 Jps
17. [root@hadoop02 ~]# jps
18. 9160 DataNode
19. 10909 NodeManager
20. 11937 Jps
21. 7197 JournalNode
22. 2073 QuorumPeerMain
23. [root@hadoop03 ~]# jps
24. 18031 Jps
25. 16722 JournalNode
26. 17710 NodeManager
27. 15519 QuorumPeerMain
28. 17214 DataNode

**2) 页面**

**ResourceManger（Standby）：**<http://192.168.240.11:8088>

**ResourceManger（Active）：**<http://192.168.240.12:8088/cluster/cluster>



**九.监控集群**

[root@hadoop01 ~]# hdfs dfsadmin -report

