**HDFS HA配置详解**

HDFS 集群中NameNode 存在单点故障（SPOF ）。对于只有一个NameNode 的集群，如果NameNode 机器出现意外downtime，那么整个集群将无法使用，直到NameNode 重新启动。HDFS 的HA 功能通过配置Active/Standby 两个NameNodes 实现在集群中对NameNode 的热备来解决上述问题。如果出现Active NN的downtime，就会切换到Standby使得NN服务不间断。HDFS HA依赖zookeeper，下面是测试的过程。

环境如下  
主机：debugo0[1-3],CentOS 6.5  
Hadoop 2.4.1  
ZooKeeper 3.4.6

|  |  |  |
| --- | --- | --- |
| HDFS | ZooKeeper |  |
| debugo01 | NN,ZKFC,JournalNode,DN | Server |
| debugo02 | NN,ZKFC,JournalNode,DN | Server |
| debugo03 | NN,JournalNode,DN | Server |

**1. 启动ZooKeeper**

编辑zookeeper配置文件

$ mkdir -p /home/hadoop/zooKeeper /home/hadoop/log/zoolog

$ cd $ZOOKEEPER\_HOME/conf

$ cp zoo\_sample.cnf zoo.cnf

$ vim zoo.cnf

tickTime=2000

initLimit=10

syncLimit=5

dataDir=/home/hadoop/zookeeper

dataLogDir=/home/hadoop/log/zoolog

clientPort=2181

将配置文件拷贝到另外两个节点，分别建立myid并启动zookeeper

$ echo "1" > /home/hadoop/zookeeper/myid

$ zkServer start

...

$ zkServer status

zkServer.sh status

JMX enabled by default

Using config: /opt/zookeeper/bin/../conf/zoo.cfg

Mode: leader

**2. 修改Hadoop配置**

core-site中需要使用ha.zookeeper.quorum设置ZooKeeper服务器节点。另外fs.defaultFS需要设置成HDFS的逻辑服务名（需与hdfs-site.xml中的dfs.nameservices一致）。

$ core-site.xml

<configuration>

<property>

<name>fs.defaultFS</name>

<value>hdfs://myhdfs</value>

</property>

<property>

<name>hadoop.tmp.dir</name>

<value>/home/hadoop/tmp</value>

</property>

<property>

<name>hadoop.logfile.size</name>

<value>104857600</value>

</property>

<property>

<name>hadoop.logfile.count</name>

<value>10</value>

</property>

<property>

<name>io.file.buffer.size</name>

<value>131072</value>

</property>

<property>

<name>ha.zookeeper.quorum</name>

<value>debugo01,debugo02,debugo03</value>

</property>

</configuration>

hdfs-site.xml中需要添加的设置较多：  
dfs.nameservices —– HDFS NN的逻辑名称，使用上面设置的myhdfs  
dfs.ha.namenodes.myhdfs —– 给定服务逻辑名称myhdfs的节点列表  
dfs.namenode.rpc-address.myhdfs.nn1 —– myhdfs中nn1节点对外服务的RPC地址  
dfs.namenode.http-address.myhdfs.nn1 —– myhdfs中nn1节点对外服务的http地址  
dfs.namenode.shared.edits.dir —– 设置一组 journalNode 的 URI 地址，active NN 将 edit log 写入这些JournalNode，而 standby NameNode 读取这些 edit log，并作用在内存中的目录树中。如果journalNode有多个节点则使用分号分割。该属性值应符合以下格式qjournal://host1:port1;host2:port2;host3:port3/journalId  
dfs.journalnode.edits.dir —– JournalNode 所在节点上的一个目录，用于存放 editlog 和其他状态信息。  
dfs.ha.automatic-failover.enabled —– 启动自动failover。自动failover依赖于zookeeper集群和ZKFailoverController（ZKFC），后者是一个zookeeper客户端，用来监控NN的状态信息。每个运行NN的节点必须要运行一个zkfc。zkfs提供了下面的功能：  
**Health monitoring** zkfc定期对本地的NN发起health-check的命令，如果NN正确返回，那么这个NN被认为是OK的。否则被认为是失效节点。  
**ZooKeeper session management** 当本地NN是健康的时候，zkfc将会在zk中持有一个session。如果本地NN又正好是active的，那么zkfc还有持有一个”ephemeral”的节点作为锁，一旦本 地NN失效了，那么这个节点将会被自动删除。  
**ZooKeeper-based election** 如果本地NN是健康的，并且zkfc发现没有其他的NN持有那个独占锁。那么他将试图去获取该锁，一旦成功，那么它就需要执行Failover，然后成为active的NN节点。Failover的过程是：第一步，对之前的NN执行fence，如果需要的话。第二步，将本地NN转换到active状态。  
启动zkfc的方法如下：hadoop-daemon.sh start zkfc。通过start-dfs.sh会自动启动该进程，一般无需手动起停。  
dfs.client.failover.proxy.provider.myhadoop —– 客户端与 active NameNode 进行交互的 Java 实现类，DFS 客户端通过该类寻找当前的active NN。  
dfs.ha.fencing.methods —– 解决HA集群脑裂问题（即出现两个 master 同时对外提供服务，导致系统处于不一致状态）。在 HDFS HA中，JournalNode 只允许一个 NameNode 写数据，不会出现两个 active NameNode 的问题，  
但是，当主备切换时，之前的 active NameNode 可能仍在处理客户端的 RPC 请求，为此，需要增加隔离机制（fencing）将之前的 active NameNode 杀死。常用的fence方法是sshfence，要指定ssh通讯使用的密钥dfs.ha.fencing.ssh.private-key-files和连接超时时间。

$ hdfs-site.xml

<configuration>

<property>

<name>dfs.nameservices</name>

<value>myhdfs</value>

</property>

<property>

<name>dfs.ha.namenodes.myhdfs</name>

<value>nn1,nn2</value>

</property>

<property>

<name>dfs.namenode.rpc-address.myhdfs.nn1</name>

<value>debugo01:8020</value>

</property>

<property>

<name>dfs.namenode.rpc-address.myhdfs.nn2</name>

<value>debugo02:8020</value>

</property>

<property>

<name>dfs.namenode.http-address.myhdfs.nn1</name>

<value>debugo01:50070</value>

</property>

<property>

<name>dfs.namenode.http-address.myhdfs.nn2</name>

<value>debugo02:50070</value>

</property>

<property>

<name>dfs.namenode.shared.edits.dir</name>

<value>qjournal://debugo01:8485;debugo02:8485;debugo03:8485/hadoop-journal</value>

</property>

<property>

<name>dfs.ha.automatic-failover.enabled</name>

<value>true</value>

</property>

<property>

<name>dfs.journalnode.edits.dir</name>

<value>/home/hadoop/journal</value>

</property>

<property>

<name>dfs.client.failover.proxy.provider.myhadoop</name>

<value>org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider</value>

</property>

<property>

<name>dfs.ha.fencing.methods</name>

<value>sshfence</value>

<description>how to communicate in the switch process</description>

</property>

<property>

<name>dfs.ha.fencing.ssh.private-key-files</name>

<value>/home/hadoop/.ssh/id\_rsa</value>

<description>the location stored ssh key</description>

</property>

<property>

<name>dfs.ha.fencing.ssh.connect-timeout</name>

<value>5000</value>

</property>

<property>

<name>dfs.datanode.data.dir</name>

<value>/home/hadoop/data</value>

</property>

<property>

<name>dfs.namenode.name.dir</name>

<value>/home/hadoop/namenode</value>

</property>

<property>

<name>dfs.namenode.handler.count</name>

<value>8</value>

</property>

<property>

<name>dfs.replication</name>

<value>2</value>

</property>

</configuration>

**3. 启动NameNode HA**

初始化zkfc

mkdir /home/hadoop/journal /home/hadoop/data /home/hadoop/namenode

hdfs zkfc -formatZK

/09/13 21:17:03 INFO zookeeper.ClientCnxn: Opening socket connection to server debugo02/192.168.46.202:2181. Will not attempt to authenticate using SASL (unknown error)

/09/13 21:17:03 INFO zookeeper.ClientCnxn: Socket connection established to debugo02/192.168.46.202:2181, initiating session

/09/13 21:17:03 INFO zookeeper.ClientCnxn: Session establishment complete on server debugo02/192.168.46.202:2181, sessionid = 0x2487208163e0000, negotiated timeout = 5000

/09/13 21:17:03 INFO ha.ActiveStandbyElector: Successfully created /hadoop-ha/myhdfs in ZK.

第一次启动格式化HDFS。 格式化HDFS的过程中，HA会journalnode通讯，所以需要先把三个节点的journalnode启动。  
hdfs journalnode  
hdfs namenode –format（其中一台）  
通过start-dfs.sh 直接启动所有服务

$ start-dfs.sh

Starting namenodes on [debugo01 debugo02]

debugo01: starting namenode, logging to /opt/hadoop/logs/hadoop-hadoop-namenode-debugo01.out

debugo02: starting namenode, logging to /opt/hadoop/logs/hadoop-hadoop-namenode-debugo02.out

debugo01: starting datanode, logging to /opt/hadoop/logs/hadoop-hadoop-datanode-debugo01.out

debugo02: starting datanode, logging to /opt/hadoop/logs/hadoop-hadoop-datanode-debugo02.out

debugo03: starting datanode, logging to /opt/hadoop/logs/hadoop-hadoop-datanode-debugo03.out

Starting journal nodes [debugo01 debugo02 debugo03]

debugo01: starting journalnode, logging to /opt/hadoop/logs/hadoop-hadoop-journalnode-debugo01.out

debugo03: starting journalnode, logging to /opt/hadoop/logs/hadoop-hadoop-journalnode-debugo03.out

debugo02: starting journalnode, logging to /opt/hadoop/logs/hadoop-hadoop-journalnode-debugo02.out

Starting ZK Failover Controllers on NN hosts [debugo01 debugo02]

debugo01: starting zkfc, logging to /opt/hadoop/logs/hadoop-hadoop-zkfc-debugo01.out

debugo02: starting zkfc, logging to /opt/hadoop/logs/hadoop-hadoop-zkfc-debugo02.out

$ jps

Jps

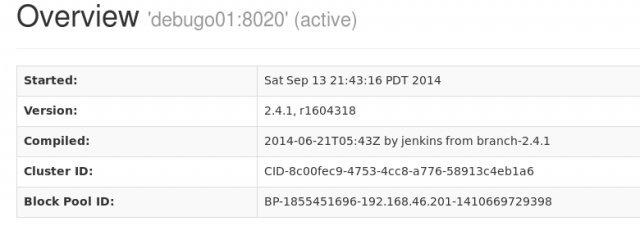
NameNode

DFSZKFailoverController

JournalNode

DataNode

QuorumPeerMain

使用浏览器访问debugo01:50070会看到该节点已经成为active  
[](http://debugo.com/wp-content/uploads/2014/09/1.png)  
先启动的namenode会成为active，在standby的日志中可以看到定期replication

-09-13 21:25:46,132 INFO org.apache.hadoop.hdfs.server.blockmanagement.CacheReplicationMonitor: Starting CacheReplication

Monitor with interval 30000 milliseconds

-09-13 21:25:46,132 INFO org.apache.hadoop.hdfs.server.blockmanagement.CacheReplicationMonitor: Rescanning because of pen

ding operations

-09-13 21:25:46,132 INFO org.apache.hadoop.hdfs.server.blockmanagement.CacheReplicationMonitor: Scanned 0 directive(s) an

d 0 block(s) in 1 millisecond(s).

....

下面需要同步一次元数据：

$ hdfs namenode –bootstrapStandby（另一台）

......

About to bootstrap Standby ID nn1 from:

Nameservice ID: myhdfs

Other Namenode ID: nn2

Other NN's HTTP address: http://debugo02:50070

Other NN's IPC address: debugo02/192.168.46.202:8020

Namespace ID: 863538584

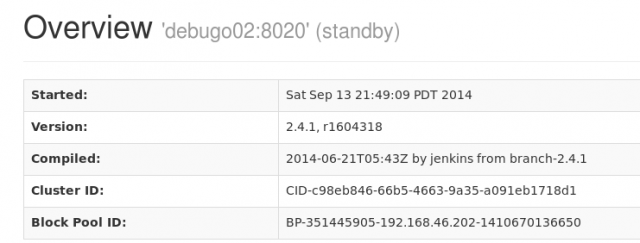
Block pool ID: BP-351445905-192.168.46.202-1410670136650

Cluster ID: CID-c98eb846-66b5-4663-9a35-a091eb1718d1

Layout version: -56

=====================================================

Re-format filesystem in Storage Directory /home/hadoop/namenode ? (Y or N) Y

这时候访问  
[](http://debugo.com/wp-content/uploads/2014/09/2.png)  
然后kill掉debugo01上的active NN进程，standby NN会成为active。

注意：手动切换时，会提示下面警告。所以一般在启动zkfc的情况下也无需进行切换。

$ hdfs haadmin -transitionToActive nn1

Automatic failover is enabled for NameNode at debugo01/192.168.46.201:8020. Refusing to manually manage HA state, since it may cause a split-brain scenario or other incorrect state.

If you are very sure you know what you are doing, please specify the forcemanual flag.

**参考**

http://www.cloudera.com/content/cloudera-content/cloudera-docs/CDH4/latest/CDH4-High-Availability-Guide/cdh4hag\_topic\_2\_3.html

http://blog.csdn.net/u010967382/article/details/30976935

http://blog.csdn.net/chenpingbupt/article/details/7922089