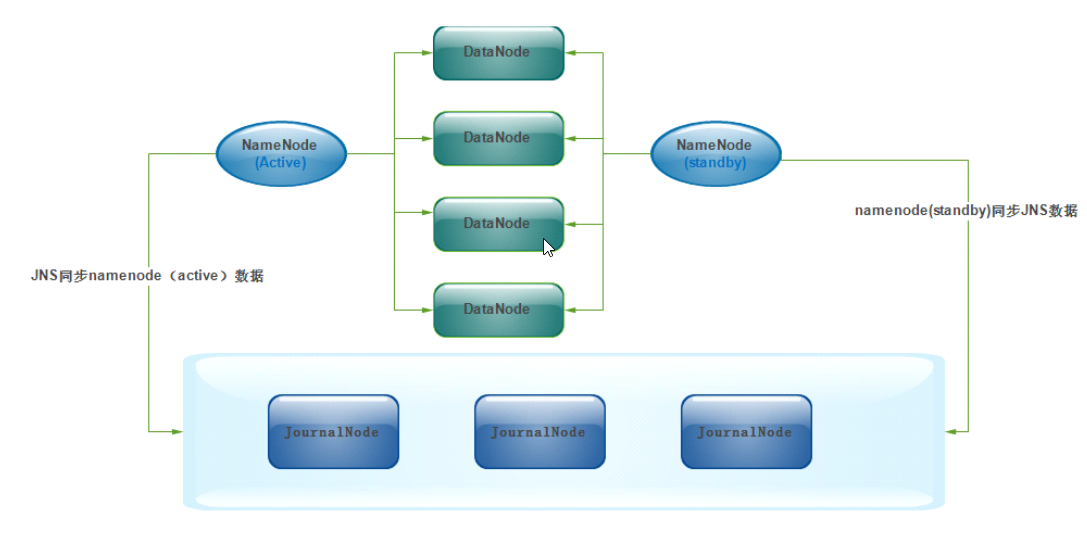
# Hadoop2.0 HA搭建步骤

## 准备工作

6台虚拟机，内存512M，hadoop1~6

修改静态IP：192.168.65.121 ~ 126

## 架构图



## 集群节点分配

* **hadoop1**

Zookeeper

NameNode(active)

Resourcemanager (active)

* **hadoop2**

Zookeeper

NameNode (standby)

* **hadoop3**

Zookeeper

ResourceManager(standby)

* **hadoop4**

DataNode

NodeManager

JournalNode

* **hadoop5**

DataNode

NodeManager

JournalNode

* **hadoop6**

DataNode

NodeManager

JournalNode

## 安装步骤

### 固化IP

修改配置文件

cd /etc/sysconfig/network-scripts #进入网络配置目录

dir ifcfg\* #找到网卡配置文件

ifcfg-ens16777736 ifcfg-lo

vi ifcfg-ens16777736

配置文件内容

TYPE=Ethernet

BOOTPROTO=static #改成static，针对NAT

NAME=eno16777736

UUID=4cc9c89b-cf9e-4847-b9ea-ac713baf4cc8

DEVICE=eno16777736

ONBOOT=yes #开机启动此网卡

IPADDR=192.168.163.129 #固定IP地址

NETMASK=255.255.255.0 #子网掩码

GATEWAY=192.168.163.2 #网关和NAT自动配置的相同，不同则无法登录

DNS1=192.168.163.2 #和网关相同

重启网络

service network restart

ping www.baidu.com

### 永久关闭每台机器的防火墙

systemctl stop firewalld.service #关闭防火墙服务

systemctl disable firewalld.service #禁止防火墙开启启动

firewall-cmd --state #检查防火墙状态

### 为每台机器配置主机名

hadoop1,hadoop2 ……

以及hosts文件

配置主机名

执行：vim /etc/hostname 修改为hadoop1~6

然后执行 hostname 主机名

达到不重启生效目的

配置hosts文件

执行：vim /etc/hosts

示例：

127.0.0.1 localhost

::1 localhost

192.168.65.121 hadoop1

192.168.65.122 hadoop2

192.168.65.123 hadoop3

192.168.65.124 hadoop4

192.168.65.125 hadoop5

192.168.65.126 hadoop6

### 通过远程命令将配置好的hosts文件scp到其他5台节点上

执行：scp /etc/hosts hadoop2:/etc

### 为每台机器配置ssh免秘钥登录

执行：ssh-keygen

ssh-copy-id root@hadoop1 （分别发送到6台节点上）

vim /root/.ssh/known\_hosts 检查是否配置成功

### 为每台机器安装jdk和配置JAVA\_HOME

vim /etc/profile

在尾行添加

JAVA\_HOME=/home/app/jdk1.8.0\_65

JAVA\_BIN=/home/app/jdk1.8.0\_65/bin

HADOOP\_HOME=/home/app/hadoop-2.7.1

PATH=$JAVA\_HOME/bin:$HADOOP\_HOME/bin:$HADOOP\_HOME/sbin:$PATH

CLASSPATH=.:$JAVA\_HOME/lib/dt.jar:$JAVA\_HOME/lib/tools.jar

export JAVA\_HOME JAVA\_BIN HADOOP\_HOME PATH CLASSPATH

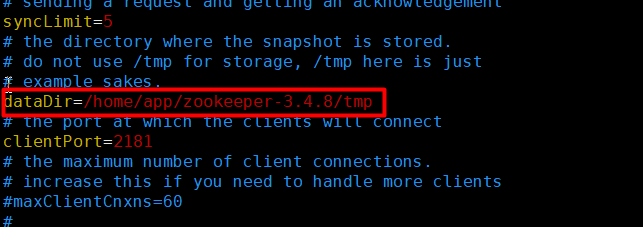
### 前三台机器安装和配置zookeeper

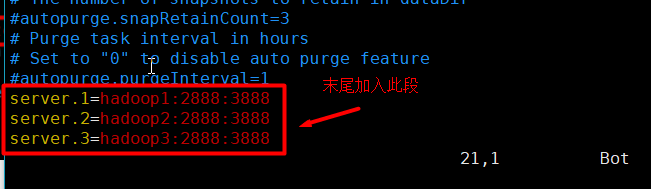
解压安装包 tar -xvf zookeeper-3.4.8.tar.gz

进入conf目录 cd zookeeper-3.4.8/conf/

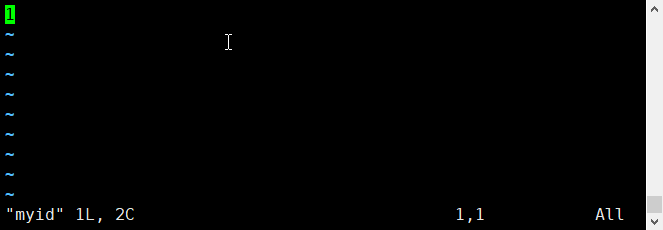
复制zoo\_sample.cfg 为zoo.cfg: cp zoo\_sample.cfg zoo.cfg

编辑zoo.cfg：vim zoo.cfg





Zookeeper根目录中创建tmp文件夹，tmp文件夹中创建myid文件，编辑文本：1（2,3）



拷贝整个zookeeper目录到hadoop2，hadoop3并修改myid文件分别为1,2

scp -r zookeeper-3.4.8 hadoop2:/home/app

### 安装和配置01节点的hadoop

创建/home/app目录

解压安装包

### 配置hadoop-env.sh

cd etc/hadoop

配置 hadoop-env.sh

vim hadoop-env.sh

#JDK安装目录，虽然系统配置了JAVA\_HOME，但有时无法正确识别，最后进行配置

export JAVA\_HOME=/home/app/jdk1.8.0\_65

#指定hadoop的配置文件目录，不运行hadoop可以不指定

export HADOOP\_CONF\_DIR=/home/app/hadoop-2.7.1/etc/hadoop

配置jdk安装所在目录

配置hadoop配置文件所在目录

### 配置core-site.xml

<configuration>

<!--用来指定hdfs的老大，ns为固定属性名，表示两个namenode-->

<property>

<name>fs.defaultFS</name>

<value>hdfs://ns</value>

</property>

<!--用来指定hadoop运行时产生文件的存放目录-->

<property>

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

<value>/home/app/hadoop-2.7.1/tmp</value>

</property>

<!--执行zookeeper地址-->

<property>

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

<value>hadoop1:2181,hadoop2:2181,hadoop3:2181</value>

</property>

</configuration>

### 配置01节点的hdfs-site.xml

配置

<configuration>

<!--执行hdfs的nameservice为ns,和core-site.xml保持一致-->

<property>

<name>dfs.nameservices</name>

<value>ns</value>

</property>

<!--ns下有两个namenode,分别是nn1,nn2-->

<property>

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

<value>nn1,nn2</value>

</property>

<!--nn1 的 RPC 通信地址-->

<property>

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

<value>hadoop1:9000</value>

</property>

<!--nn1的http通信地址-->

<property>

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

<value>hadoop1:50070</value>

</property>

<!--nn2的RPC通信地址-->

<property>

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

<value>hadoop2:9000</value>

</property>

<!--nn2 的 http 通信地址-->

<property>

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

<value>hadoop2:50070</value>

</property>

<!--指定namenode的元数据在JournalNode上的存放位置,这样，namenode2可以

从 jn 集群里获取

最新的namenode的信息，达到热备的效果-->

<property>

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

<value>qjournal://hadoop4:8485;hadoop5:8485;hadoop6:8485/ns</value>

</property>

<!--指定 JournalNode 存放数据的位置-->

<property>

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

<value>/home/app/hadoop-2.7.1/journal</value>

</property>

<!--开启namenode故障时自动切换-->

<property>

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

<value>true</value>

</property>

<!--配置切换的实现方式-->

<property>

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

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

</property>

<!--配置隔离机制-->

<property>

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

<value>sshfence</value>

</property>

<!--配置隔离机制的ssh登录秘钥所在的位置-->

<property>

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

<value>/root/.ssh/id\_rsa</value>

</property>

<!--配置namenode数据存放的位置,可以不配置，如果不配置，默认用的是

core-site.xml里配置的hadoop.tmp.dir的路径-->

<property>

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

<value>file:///home/app/hadoop-2.7.1/tmp/namenode</value>

</property>

<!--配置datanode数据存放的位置,可以不配置，如果不配置，默认用的是

core-site.xml 里配置的 hadoop.tmp.dir 的路径-->

<property>

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

<value>file:///home/app/hadoop-2.7.1/tmp/datanode</value>

</property>

<!--配置 block 副本数量-->

<property>

<name>dfs.replication</name>

<value>3</value>

</property>

<!--设置hdfs的操作权限，false表示任何用户都可以在hdfs上操作文件-->

<property>

<name>dfs.permissions</name>

<value>false</value>

</property>

</configuration>

### 配置 mapred-site.xml

配置代码：

<configuration>

<property>

<!--指定mapreduce运行在yarn上-->

<name>mapreduce.framework.name</name>

<value>yarn</value>

</property>

</configuration>

### 配置yarn-site.xml

配置代码：

<configuration>

<!--开启YARN HA -->

<property>

<name>yarn.resourcemanager.ha.enabled</name>

<value>true</value>

</property>

<!--指定两个resourcemanager的名称-->

<property>

<name>yarn.resourcemanager.ha.rm-ids</name>

<value>rm1,rm2</value>

</property>

<!--配置rm1，rm2的主机-->

<property>

<name>yarn.resourcemanager.hostname.rm1</name>

<value>hadoop1</value>

</property>

<property>

<name>yarn.resourcemanager.hostname.rm2</name>

<value>hadoop3</value>

</property>

<!--开启yarn恢复机制-->

<property>

<name>yarn.resourcemanager.recovery.enabled</name>

<value>true</value>

</property>

<!--执行rm恢复机制实现类-->

<property>

<name>yarn.resourcemanager.store.class</name>

<value>org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore</value>

</property>

<!--配置zookeeper的地址-->

<property>

<name>yarn.resourcemanager.zk-address</name>

<value>hadoop1:2181,hadoop2:2181,hadoop3:2181</value>

<description>For multiple zk services, separate them with comma</description>

</property>

<!--指定YARN HA的名称-->

<property>

<name>yarn.resourcemanager.cluster-id</name>

<value>yarn-ha</value>

</property>

<!--指定 yarn 的老大 resoucemanager 的地址-->

<property>

<name>yarn.resourcemanager.hostname</name>

<value>hadoop1</value>

</property>

<!--NodeManager获取数据的方式-->

<property>

<name>yarn.nodemanager.aux-services</name>

<value>mapreduce\_shuffle</value>

</property>

</configuration>

### 配置slaves文件

配置代码：

hadoop4

hadoop5

hadoop6

### 根据配置文件，创建相关的文件夹，用来存放对应数据

在hadoop-2.7.1目录下创建:

①journal目录

②创建tmp目录

③在tmp目录下，分别创建namenode目录和datanode目录

### 配置 hadoop 的环境变量（可不配）

JAVA\_HOME=/home/app/jdk1.8.0\_65

JAVA\_BIN=/home/app/jdk1.8.0\_65/bin

HADOOP\_HOME=/home/app/hadoop-2.7.1

PATH=$JAVA\_HOME/bin:$HADOOP\_HOME/bin:$HADOOP\_HOME/sbin:$PATH

CLASSPATH=.:$JAVA\_HOME/lib/dt.jar:$JAVA\_HOME/lib/tools.jar

export JAVA\_HOME JAVA\_BIN HADOOP\_HOME PATH CLASSPATH

### 通过scp 命令，将hadoop安装目录远程copy到其他5台机器上

比如向hadoop02节点传输：

scp -r hadoop-2.7.1 root@hadoop2:/home/app

### 启动zookeeper集群

在Zookeeper安装目录的bin目录下执行：sh zkServer.sh start

### 格式化zookeeper

在zk的leader节点上执行：

hdfs zkfc -formatZK，这个指令的作用是在zookeeper集群上生成ha节点 （ns节点）

注：18--24步可以用一步来替代：进入hadoop安装目录的sbin目录，执行： sh start-dfs.sh 。 但建议还是按部就班来执行，比较可靠。

### 启动journalnode集群

在4、5、6节点上执行：

切换到hadoop安装目录的sbin目录下，执行：

sh hadoop-daemons.sh start journalnode

然后执行jps命令查看进程。

### 格式化1节点的namenode

在1节点上执行：

hadoop namenode -format

### 启动1节点的namenode

在 1 节点上执行：

sh hadoop-daemon.sh start namenode

### 把2节点的namenode节点变为standby namenode节点

在2节点上执行：

hdfs namenode -bootstrapStandby

### 启动 2 节点的 namenode 节点

在2节点上执行：

sh hadoop-daemon.sh start namenode

### 在4,5,6节点上启动datanode节点

在 4,5,6 节点上执行： sh hadoop-daemon.sh start datanode

### 启动zkfc（启动FalioverControllerActive)

在1,2节点上执行：

sh hadoop-daemon.sh start zkfc

### 在1节点上启动主Resourcemanager

在1节点上执行：start-yarn.sh

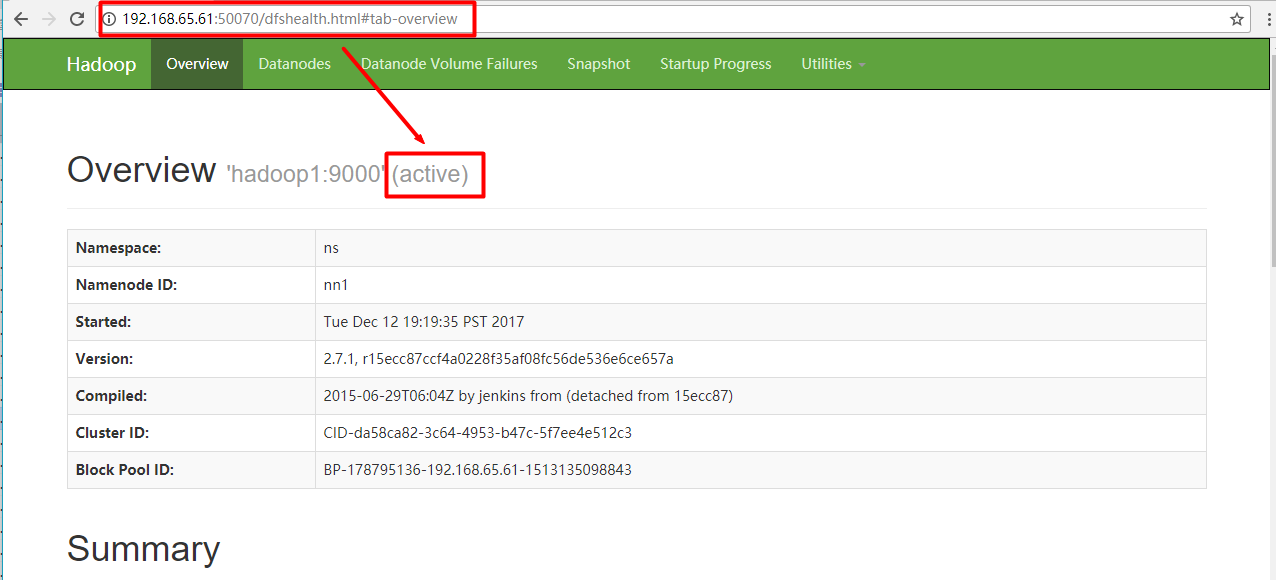
启动成功后，4,5,6节点上应该有nodemanager 的进程

### 在 3 节点上启动副 Resoucemanager

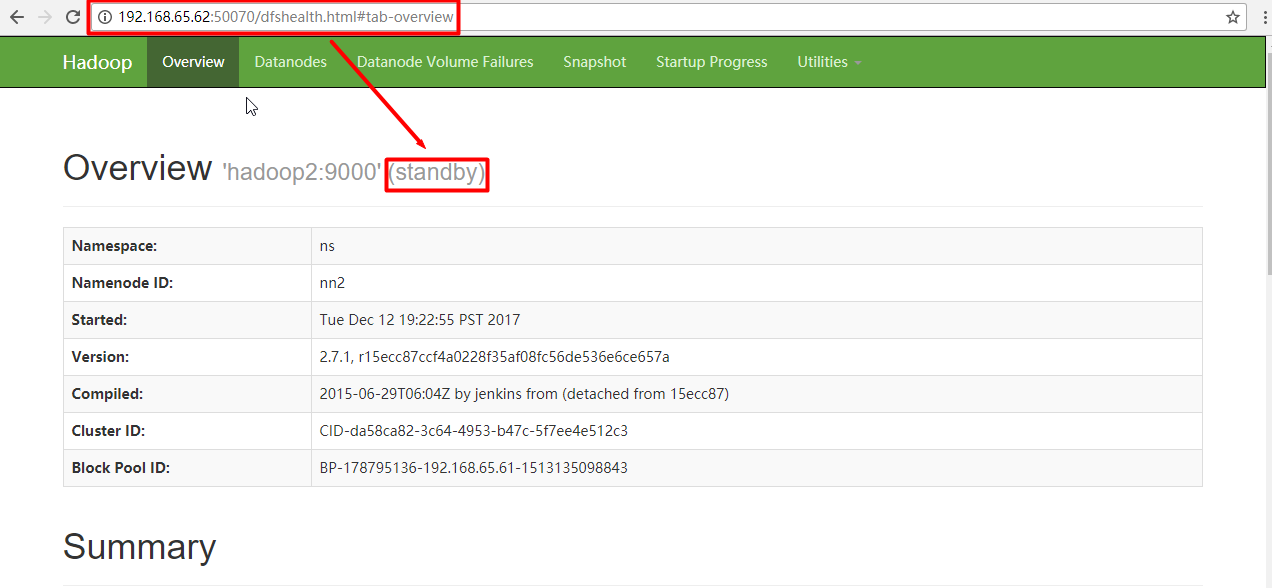
在3节点上执行：sh yarn-daemon.sh start resourcemanager

## 测试

输入地址： http://192.168.65.111:50070 ，查看 namenode 的信息，是active状态



输入地址：[http://192.168.65.22:50070查看namenode的信息，是standby状态](http://192.168.234.22:50070，查看namenode的信息，是standby状态)



然后停掉01节点的namenode,此时发现standby的namenode变为active。

如果出现standby节点不能正确替代active的情况检查fuser是否正常

yum provides “\*/fuser”

在启动namenode的两台节点上安装fuser,分别执行

yum -y install psmisc