**HDFS 设备规划**

1. 首先，搭建五台小集群，虚拟机的话，创建五个

下面为这五台机器分别分配IP地址及相应的角色：集群有个特点，五机子用户名最好一致，要不你就创建一个组，把这些用户放到组里面去，五台的用户名都是hadoop,主机名分别为：master，second，data1，data2，data3

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 192.168.20.21  master | 192.168.20.22  second | 192.168.20.23  data1 | 192.168.20.24  data2 | 192.168.20.25  data3 |
| namenode | 是（主） | 是（从） | 否 | 否 | 否 |
| datanode | 是 | 是 | 是 | 是 | 是 |
| journalnode | 是 | 是 | 是 | 是 | 是 |
| zookeeper | 是 | 是 | 是 | 是 | 是 |
| zkfc | 是 | 是 | 否 | 否 | 否 |

**文件路径规划：**

**/usr/local/install： 下保存各种下载的tar 包**

**/usr/local/hadoop： Hadoop HDFS 的安装路径**

**/usr/local/zookeeper： ZooKeeper 的安装路径**

**/usr/local/java ： JDK 的安装路径**

**安装配置jdk,SSH**

 hadoop集群可以用root用户，或者新建一个hadoop用户；

如果用户名不一致，你就要创建一个用户组，把用户放到用户组下面：

 sudoaddgroup hadoop     创建hadoop用户组

 sudoadduser -ingrouphadoop one  创建一个one用户，归到hadoop组下

 二.由于用户是普通用户，没有root一些权限，所以修改hadoop用户权限

 用root权限，修改sudoers文件

 nano    /etc/sudoers   打开文件，修改hadoop用户权限，如果你创建的是one用户，就one ALL=(ALL:ALL) ALL

1. 在这五台机子上分别设置/etc/hosts及/etc/hostname

hosts这个文件用于定于主机名与IP地址之间的对用关系

sudo -i  获取最高权限

nano /etc/hosts:

ctrl+o:保存，然后回车，ctrl+x:退出

hostname 这个文件用于定义主机名的，

注意：主机是主机名，从机就是从机名，例如：主namenode在这里就是master，从namenode在这里就是second，datanode在这里就是data1，

然后你可以输入：ping second,看能不能ping通

四.要在这五台机子上均安装jdk,ssh,并配置好环境变量，五台机子都是这个操作：：

做好准备工作，下载jdk-8u66-linux-x64.tar 这个软件包

1.sudo apt-get install openssh-server    下载SSH

   ssh     查看，代表安装成功

2. tar zxvfjdk-8u66-linux-x64.tar 解压jdk

3.sudo nano /etc/profile,在最下面加入这几句话，保存,这是配置java环境变量

4.source  /etc/profile  使其配置生效

 验证jdk是否安装成功，敲命令

5.java -version     可以看到JDK版本信息，代表安装成功

6：配置SSH  免密码登陆,记住，要在root用户下执行的

ssh-keygen -t dsa -P ''

之后一路回 车（产生秘钥，会自动产生一个.ssh文件

7.cd   /root/.ssh  进入ssh文件夹

 cat/root/.ssh/id\_dsa.pub >>/root/.ssh/authorized\_keys

 把id\_dsa.pub 追加到授权的 key 里面去

8. ssh localhost     此时已经可以进行ssh localhost的无密码登陆

9.拷贝id\_dsa.pub文件到其他机器

192.168.9.21  操作：

scp  .ssh/id\_dsa.pub   root@192.168.9.22:/root/.ssh/21.pub

scp  .ssh/id\_dsa.pub   root@192.168.9.23:/root/.ssh/21.pub

scp  .ssh/id\_dsa.pub   root@192.168.9.24:/root .ssh/21.pub

scp  .ssh/id\_dsa.pub   root@192.168.9.25:/root/.ssh/21.pub

在192.168.9.22，192.168.9.23，192.168.9.24，192.168.9.25 下重复上述操作.

所有设备都完成后，执行下面操作

10.公钥都追加到 那个授权文件里

在192.168.9.21机子上操作：

cat  .ssh/21.pub >> .ssh/authorized\_keys

cat  .ssh/22.pub >> .ssh/authorized\_keys

cat  .ssh/23.pub >> .ssh/authorized\_keys

cat  .ssh/24.pub >> .ssh/authorized\_keys

cat  .ssh/25.pub >> .ssh/authorized\_keys

11.验证ssh 192.168.9.21 hostname

master

**搭建Zookeeper集群**

1. 下载zookeeper-3.4.6版本：zookeeper-3.4.6.tar.gz,放在/usr/local/insalll下面
2. 创建zookeeper 目录 mkdir /usr/local/zookeeper
3. 拷贝安装文件

cp /usr/local/insalll/ zookeeper-3.4.6.tar.gz /usr/local/zookeeper

cd/usr/local/zookeeper

tar  zxvf  zookeeper-3.4.5.tar.gz       直接进行解压

1. 配置etc/profile

sudonano  etc/profile      在末尾加入下面配置

export ZOOKEEPER\_HOME=/usr/local/zookeeperzookeeper-3.4.6

export PATH=$ZOOKEEPER\_HOME/bin:$ZOOKEEPER\_HOME/conf:$PATH

source /etc/profile    使其配置生效

1. 创建目录：

mkdir/usr/local/zookeeper/idt/data

mkdir/usr/local/zookeeper/idt/datalog

6. 配置zookeeper-3.4.6/conf/zoo.cfg文件，这个文件本身是没有的，有个zoo\_sample.cfg模板

    cd    zookeeper-3.4.6/conf      进入conf目录

    cp  zoo\_sample.cfg   zoo.cfg     拷贝模板

    sudonano  zoo.cfg       修改zoo.cfg文件，红色是修改部分

zoo.cfg :

# The number of milliseconds of each tick

tickTime=2000

# The number of ticks that the initial

# synchronization phase can take

initLimit=10

# The number of ticks that can pass between

# sending a request and getting an acknowledgement

syncLimit=5

# the directory where the snapshot is stored.

# do not use /tmp for storage, /tmp here is just

# example sakes.

dataDir=/usr/local/zookeeper/idt/data

dataLogDir=/usr/local/zookeeper/idt/datalog

# the port at which the clients will connect

clientPort=2181

# the maximum number of client connections.

# increase this if you need to handle more clients

#maxClientCnxns=60

#

# Be sure to read the maintenance section of the

# administrator guide before turning on autopurge.

#

# http://zookeeper.apache.org/doc/current/zookeeperAdmin.html#sc\_maintenance

#

# The number of snapshots to retain in dataDir

#autopurge.snapRetainCount=3

# Purge task interval in hours

# Set to "0" to disable auto purge feature

#autopurge.purgeInterval=1

server.1=master:2888:3888

server.2=second:2888:3888

server.3=data1:2888:3888

server.4=data2:2888:3888

server.5=data3:2888:3888

---------------------------------------------------------------------

注意：创建dataDir参数指定的目录，创建data文件夹，在这个文件夹下，还要创建一个文本myid

cd  /usr/local/zookeeper

mkdir  data     创建data

cd  /usr/local/zookeeper/idt/data     进入data文件夹下

touch  myid   创建文本myid，在这个文本内写入1，因为server.1=master:2888:3888server指定的是1,

在其余机子配置，second下面的myid是2，data1下面myid是3，

Data2下面myid是4，data3下面myid是5，这些都是根据server来的

7.主机配置完以后，其他设备上安装zookeeper

记住:::::修改从机的myid.从机也要配置etc/profile

5.启动zookeeper，先hadoop集群启动

zkServer.sh  start     这个启动是主机从机都要输入启动命令

bin/zkServer.sh  status 在不同的机器上使用该命令，其中二台显示follower，一台显示leader

zkCli.sh  -server 192.168.9.21:2181  启动客户端脚本

quit   退出

help  可是查看帮助命令

这样zookeeper集群就配置完了

     配置hadoop集群2.6.3版本HDFS的HA配置

1. 把hadoop-2.6.3.tar.gz文件放在 /usr/local/install路径下

mkdir /usr/local/hadoop

cp/usr/local/install /usr/local/hadoop

1. 进行解压

tar  zxvf  hadoop-2.6.3.tar.gz

3.hadoop配置过程，

 配置之前，需要在hadoop本地文件系统创建以下文件夹：

 mkdir /usr/local/hadoop/idt/dfs/name

mkdir/usr/local/hadoop/idt/dfs/data

mkdir /usr/local/hadoop/idt/tmp/journal

touch idt/excludes

 给这些文件要赋予权限

 sudochmod 777 /usr/local/hadoop/idt/dfs/

sudochmod 777 /usr/local/hadoop/idt/tmp

  这里要涉及到的配置文件有7个：

~/hadoop-2.6.3/etc/hadoop/hadoop-env.sh

~/hadoop-2.6.3/etc/hadoop/slaves

~/hadoop-2.6.3/etc/hadoop/core-site.xml

~/hadoop-2.6.3/etc/hadoop/hdfs-site.xml

以上个别文件默认不存在的，可以复制相应的template文件获得。

cd /usr/local/hadoop/hadoop-2.6.3/etc/hadoop    进入到hadoop配置文件的目录中

4.配置hadoop-env.sh

 sudonano  /home/hadoop/hadoop-2.6.3/etc/hadoop/hadoop-env.sh

export JAVA\_HOME=/home/hadoop/jdk1.8.0\_66   配置jdk

5.配置slaves,写入一下内容

master

second

data1

data2

data3

6.配置core-site.xml 对于nc-idt的集群，我们统一为：idt

<?xml version="1.0" encoding="UTF-8"?>

<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<!--

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software

distributed under the License is distributed on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and

limitations under the License. See accompanying LICENSE file.

-->

<!-- Put site-specific property overrides in this file. -->

<configuration>

<property>

<name>fs.defaultFS</name>

<value>hdfs://idt</value>

</property>

<property>

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

<value>/usr/local/hadoop/idt/tmp</value>

</property>

<property>

<name>dfs.hosts.exclude</name>

<value>/usr/local/hadoop/idt/excludes</value>

<final>true</final>

</property>

<property>

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

<value>master:2181,second:2181,data1:2181,data2:2181,data3:2181</value>

</property>

<property>

<name>ha.zookeeper.session-timeout.ms</name>

<value>1000</value>

</property>

</configuration>

7.配置hdfs-site.xml

<?xml version="1.0" encoding="UTF-8"?>

<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<!--

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software

distributed under the License is distributed on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and

limitations under the License. See accompanying LICENSE file.

-->

<!-- Put site-specific property overrides in this file. -->

<configuration>

<property>

<name>dfs.replication</name>

<value>3</value>

</property>

<property>

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

<value>/usr/local/hadoop/idt/dfs/name</value>

</property>

<property>

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

<value>/usr/local/hadoop/idt/dfs/data</value>

</property>

<property>

<name>dfs.webhdfs.enabled</name>

<value>true</value>

</property>

<property>

<name>dfs.permissions</name>

<value>false</value>

</property>

<property>

<name>dfs.permissions.enabled</name>

<value>false</value>

</property>

<property>

<name>dfs.nameservices</name>

<value>idt</value>

</property>

<property>

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

<value>master,second</value>

</property>

<property>

<name>dfs.namenode.rpc-address.idt.master</name>

<value>master:9000</value>

</property>

<property>

<name>dfs.namenode.rpc-address.idt.second</name>

<value>second:9000</value>

</property>

<property>

<name>dfs.namenode.servicerpc-address.idt.master</name>

<value>master:53310</value>

</property>

<property>

<name>dfs.namenode.servicerpc-address.idt.second</name>

<value>second:53310</value>

</property>

<property>

<name>dfs.namenode.http-address.idt.master</name>。

<value>master:50070</value>

</property>

<property>

<name>dfs.namenode.http-address.idt.second</name>

<value>second:50070</value>

</property>

<property>

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

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

</property>

<property>

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

<value>sshfence</value>

</property>

<property>

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

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

</property>

<property>

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

<value>30000</value>

</property>

<property>

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

<value>/usr/local/hadoop/idt/tmp/journal</value>

</property>

<property>

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

<value>true</value>

</property>

<property>

<name>ha.failover-controller.cli-check.rpc-timeout.ms</name>

<value>60000</value>

</property>

<property>

<name>ipc.client.connect.timeout</name>

<value>60000</value>

</property>

<property>

<name>dfs.image.transfer.bandwidthPerSec</name>

<value>4194304</value>

</property>

</configuration>

 master配置完以后，可是直接把hadoop配置文件复制到其他从机，这样可以节省时间

 命令是在root用户下进行：这个只需要在master运行就可以了

  scp  -r /usr/local/hadoop/ root[@second:/usr/local](mailto:one@ubuntu2:/home/one/)

  scp  -r /usr/local/hadoop/ root[@data1:/usr/local](mailto:one@ubuntu2:/home/one/)

 scp  -r /usr/local/hadoop/ root[@data2:/usr/local](mailto:one@ubuntu2:/home/one/)

 scp  -r /usr/local/hadoop/ root[@data3:/usr/local](mailto:one@ubuntu2:/home/one/)

**启动过程(顺序很重要)**

0、首先把各个zookeeper起来，如果zookeeper集群还没有启动的话。

./bin/zkServer.sh start    记住每台机子都要启动

1、然后在master 节点执行如下命令，创建命名空间

./bin/hdfs zkfc -formatZK

2、在每个节点用如下命令启日志程序

./sbin/hadoop-daemon.sh start journalnode

3、在主master节点 格式化namenode和journalnode目录

./bin/hdfs namenode -format -clusterId idt

4、在master节点启动

./sbin/hadoop-daemon.sh start namenode

5、在seond节点执行第一行命令，这个是把备namenode节点的目录格式化并把元数据从主namenode节点copy过来，并且这个命令不会把journalnode目录再格式化了！然后用第二个命令启动备namenode进程！

./bin/hdfs namenode –bootstrapStandby (成功可能性小，直接执行下面)

(如果不能成功，直接将master 的/usr/local/Hadoop/idt/dfs/name 拷贝过来，同样的路径下覆盖就行)，然后启动下面：

./sbin/hadoop-daemon.sh start namenode

6、在两个namenode节点都执行以下命令

./sbin/hadoop-daemon.sh start zkfc

起来后对应jps进程为：DFSZKFailoverController

7、在所有datanode节点都执行以下命令启动datanode

./sbin/hadoop-daemon.sh start datanode

**startupphase**

下次启动的时候，就直接执行以下命令就可以全部启动所有进程和服务了：

但是还是要先启动zookeeper,启动日志程序，然后在全部启动

1、在每个节点启动zookeeper

./usr/local/zookeeper/zookeeper-3.4.6/bin/zkServer.sh start

2、在每个节点用如下命令启日志程序

./sbin/hadoop-daemon.sh start journalnode

3、在master上启动hadoop

./sbin/start-dfs.sh

4、在两个namenode节点都执行以下命令 (如果没有起来)

./sbin/hadoop-daemon.sh start zkfc

然后访问以下两个地址查看启动的两个namenode的状态：

http://192.168.9.21:50070/dfshealth.jsp

http://192.168.9.22:50070/dfshealth.jsp

**stop phase**

停止所有HDFS相关的进程服务，执行以下命令：

./sbin/stop-dfs.sh

**试HDFS的HA功能**

在任意一台namenode机器上通过jps命令查找到namenode的进程号，然后通过kill -9的方式杀掉进程，观察另一个namenode节点是否会从状态standby变成active状态。

hd@hd0:/opt/hadoop/apps/hadoop$ jps

1686 JournalNode

1239 QuorumPeerMain

1380 NameNode

2365 Jps

1863 DFSZKFailoverController

kill -9 1380

然后观察原来是standby状态的namenode机器的zkfc日志，若最后一行出现如下日志，则表示切换成功：

2013-12-31 16:14:41,114 INFOorg.apache.hadoop.ha.ZKFailoverController: Successfully transitioned NameNodeat hd0/192.168.9.22:53310 to active state

这时再通过命令启动被kill掉的namenode进程

./sbin/hadoop-daemon.sh start namenode

对应进程的zkfc最后一行日志如下：

2013-12-31 16:14:55,683 INFOorg.apache.hadoop.ha.ZKFailoverController: Successfully transitioned NameNodeat hd2/192.168.0.21:53310 to standby state

可以在两台namenode机器之间来回kill掉namenode进程以检查HDFS的HA配置！