**hadoop学习（八）Hadoop2.2.0+HA+zookeeper3.4.5详细配置过程+错误处理(1)**

(http://www.it165.net/admin/html/201404/2728.html)

心血之作，在熟悉hadoop2架构的过程耽误了太长时间，在搭建环境过程遇到一些问题，这些问题一直卡在那儿，不得以解决，耽误了时间。最后，千寻万寻，把问题解决，多谢在过程提供帮助的大侠。这篇文章中，我也会把自己遇到的问题给列出来，帮助后来者进一步的学习。

这篇文章结合自己实际测试过程，花费太多心血整理。转载注明出处：

http://blog.csdn.net/yczws1/article/details/23566383

前言

本文主要通过对hadoop2.2.0集群配置的过程加以梳理，所有的步骤都是通过自己实际测试。文档的结构也是根据自己的实际情况而定，同时也会加入自己在实际过程遇到的问题。搭建环境过程不重要，重要点在于搭建过程中遇到的问题，解决问题的过程。

可能自己遇到的问题在一些由经验的老者手上都不是问题，但是这些问题着实让自己耽误了很长时间，最后问题解决也是费了太大心血。也通过这篇文档，表现出来，算是总结，为后者提供意见。

Hadoop2.2.0体系结构

要想理解本节内容，首先需要了解hadoop1的体系结构。这里不过多的介绍基于hadoop1的体系架构，早在之前，曾搭建hadoop1.2.1伪分布式集群，详细请看hadoop学习（一）hadoop-1.2.1伪分布式配置及遇到的问题。这里主要介绍hadoop2的体系架构。

hadoop1的核心组成是两部分，即HDFS和MapReduce。在hadoop2中变为HDFS和Yarn。

新的HDFS中的NameNode不再是只有一个了，可以有多个（目前只支持2个）。每一个都有相同的职能。

这两个NameNode的地位如何：一个是active状态的，一个是standby状态的。当 集群运行时，只有active状态的NameNode是正常工作的，standby状态的NameNode是处于待命状态的，时刻同步active状态 NameNode的数据。一旦active状态的NameNode不能工作，通过手工或者自动切换，standby状态的NameNode就可以转变为 active状态的，就可以继续工作了。这就是高可靠。

当NameNode发生故障时，他们的数据如何保持一致：在这里，2个NameNode的数据其实是实时共享的。新HDFS采用了一种共享机制，JournalNode集群或者NFS进行共享。NFS是操作系统层面的，JournalNode是hadoop层面的，我们这里使用JournalNode集群进行数据共享。

如何实现NameNode的自动切换：这就需要使用ZooKeeper集群进行选择了。HDFS集群中的两个NameNode都在ZooKeeper中注册，当active状态的NameNode出故障时，ZooKeeper能检测到这种情况，它就会自动把standby状态的NameNode切换为active状态。

HDFS Federation（HDFS联盟）：联盟的出现是有原因的。我们知道 NameNode是核心节点，维护着整个HDFS中的元数据信息，那么其容量是有限的，受制于服务器的内存空间。当NameNode服务器的内存装不下数据后，那么HDFS集群就装不下数据了，寿命也就到头了。因此其扩展性是受限的。HDFS联盟指的是有多个HDFS集群同时工作，那么其容量理论上就不受限了，夸张点说就是无限扩展。你可以理解成，一个总集群中，可以虚拟出两个或两个以上的单独的小集群，各个小集群之间数据是实时共享的。因为hadoop集群中已经不在单独存在namenode和datanode的概念。当一个其中一个小集群出故障，可以启动另一个小集群中的namenode节点，继续工作。因为数据是实时共享，即使namenode或datanode一起死掉，也不会影响整个集群的正常工作。

集群节点任务安排：

这点很重要，我们事先一定要先理解，节点之间任务是如何安排的。如果事先不理解为什么是这样，后面还会遇到更多的问题。这就需要，理解journalnode、zookeeper、datanode、namenode之间关系。自己也是在这上面耽误了很长时间，希望读者这点多注意下。

6台主机。



Journalnode和zookeeper保持奇数点，这点大家要有个概念，最少不少于3个节点。这里暂不讲解。

两个namenode上面已经说明，其实在hadoop2中几点之间namenode和datanode之间的划分已经不是那么明确了。这只是采用后4台机器作为namenode。这里也存在一个问题：如果把datanode和namenode放在一起，对数据的读取IO的效率肯定会有一定的影响，不同机器之间还是要通过网线和http请求完成数据之间的共享。实际中，两者是可以在一起。但是我不知道在一起和不在一起之间的主要区别在哪儿，上面的解释只是个人意见，如果读者有更好的意见可以留言，大家一起讨论。

在集群搭建之间，各主机设置静态IP、更改主机名称、主机之间ssh互联等相关设置这里不在多讲。如有需要，请参考：hadoop学习（五）Hadoop2.2.0完全分布式安装详解（1）配置文档。

下面就进入正式的集群的安装过程：

下面所有的过程都是在hadoop1机器上完成的，之后把文件复制到其他节点中。

Zookeeper安装过程：

1、下载解压zookeeper

下载地址：http://mirror.bit.edu.cn/apache/zookeeper/zookeeper-3.4.5/

解压到指定目录：这里目录：/home/tom/yarn/hadoop-2.2.0/app/

在hadoop目录中创建app目录。把文件解压到hadoop的app目录中，是为了以后整个项目可以整体移植。包括后面，我们会安装HBase、Hive等软件，都是解压到app的目录中。

2、修改配置文件

2.1进入zookeeper中conf目录：

拷贝命名zoo\_sample.cfg 为zoo.cfg。我们一般不修改配置文件默认的示例文件，修改赋值其子文件。

编辑zoo.cfg

view sourceprint?

01.tickTime=2000

02.initLimit=10

03.syncLimit=5

04.dataDir=/home/tom/yarn/hadoop-2.2.0/app/zookeeper-3.4.5/zkdata

05.dataLogDir=/home/tom/yarn/hadoop-2.2.0/app/zookeeper-3.4.5/zkdatalog

06.clientPort=2181

07.server.1=hadoop1:2888:3888

08.server.2=hadoop2:2888:3888

09.server.3=hadoop3:2888:3888

10.server.4=hadoop4:2888:3888

11.server.5=hadoop5:2888:3888

2.2创建zkdata和zkdatalog两个文件夹

在zookeeper的目录中，创建上述两个文件夹。进入zkdata文件夹，创建文件myid，填入1。这里写入的1，是在zoo.cfg文本中的server.1中的1。当我们把所有文件都配置完毕，我们把hadoop1中yarn目录复制到其它机器中，我们在修改每台机器中对应的myid文本，hadoop2中的myid写入2。其余节点，安照上面配置，依此写入相应的数字。Zkdatalog文件夹，是为了指定zookeeper产生日志指定相应的路径。

3、添加环境变量

本机环境变量添是在/etc/profile目录中添加的。

view sourceprint?

1.export ZOOKEEPER\_HOME=/home/tom/yarn/hadoop-2.2.0/app/zookeeper-3.4.5

2.PATH=$ZOOKEEPER\_HOME/bin:$PATH

添加ZOOKEEPER\_HOME/bin目录可以在原有的PATH后面加入

：$ZOOKEEPER\_HOME/bin

关于环境变量修改/etc目录下的profile文件，也可以在根目录下的.bashrc目录下添加环境变量。这两者有什么区别：.bashrc是对当前目录用户的环境变量，profile文件是对所有用户都开放的目录。当系统加载文件中，先从profile找相应的路劲，如果没有会在.bashrc文件中找对应的环境变量路径。这两者大家稍至了解。

然后 source /etc/profile

上面3个步骤就安装zookeeper完毕。然后就是测试zookeeper，这个放到后面等hadoop1上整体配置完毕，scp到其它主机上后，再一起测试。

Hadoop配置

1、下载解压hadoop2.2.0

路径：http://apache.dataguru.cn/hadoop/common/hadoop-2.2.0/

解压到：/home/tom/yarn/下。其实这一步应该在解压zookeeper之前。不再多讲。

2、修改配置文件

这里要修改配置文件一共包括6个，分别是在hadoop-env.sh、core-site.xml、hdfs-site.xml、mapred-site.xml、 yarn-site.xml和slaves。

修改文件的目录地址：/home/tom/yarn/hadoop-2.2.0/etc/hadoop/

2.1文件hadoop-env.sh

添加jdk环境变量：

export JAVA\_HOME=/usr/lib/jvm/jdk1.7.0\_45

2.2文件coer-site.xml

view sourceprint?

01.<configuration>

02.<property>

03.<name>fs.defaultFS</name>

04.<value>hdfs://cluster1</value>

05.</property>

06.【这里的值指的是默认的HDFS路径。这里只有一个HDFS集群，在这里指定！该值来自于hdfs-site.xml中的配置】

07.<property>

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

09.<value>/home/tom/yarn/yarn\_data/tmp</value>

10.</property>

11.【这里的路径默认是NameNode、DataNode、JournalNode等存放数据的公共目录。用户也可以自己单独指定这三类节点的目录。这里的yarn\_data/tmp目录与文件都是自己创建的】

12.<property>

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

14.<value>hadoop1:2181,hadoop2:2181,hadoop3:2181,hadoop4:2181,hadoop5:2181</value>

15.</property>

16.【这里是ZooKeeper集群的地址和端口。注意，数量一定是奇数，且不少于三个节点】

17.</configuration>

2.3文件hdfs-site.xml

view sourceprint?

01.重点核心文件：

02.<configuration>

03.<property>

04.<name>dfs.replication</name>

05.<value>2</value>

06.</property>

07.【指定DataNode存储block的副本数量。默认值是3个，我们现在有4个DataNode，该值不大于4即可。】

08.<property>

09.<name>dfs.permissions</name>

10.<value>false</value>

11.</property>

12.<property>

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

14.<value>false</value>

15.</property>

16.<property>

17.<name>dfs.nameservices</name>

18.<value>cluster1</value>

19.</property>

20.【给hdfs集群起名字】

21.<property>

22.<name>dfs.ha.namenodes.cluster1</name>

23.<value>hadoop1,hadoop2</value>

24.</property>

25.【指定NameService是cluster1时的namenode有哪些，这里的值也是逻辑名称，名字随便起，相互不重复即可】

26.<property>

27.<name>dfs.namenode.rpc-address.cluster1.hadoop1</name>

28.<value>hadoop1:9000</value>

29.</property>

30.【指定hadoop101的RPC地址】

31.<property>

32.<name>dfs.namenode.http-address.cluster1.hadoop1</name>

33.<value>hadoop1:50070</value>

34.</property>

35.【指定hadoop101的http地址】

36.<property>

37.<name>dfs.namenode.rpc-address.cluster1.hadoop2</name>

38.<value>hadoop2:9000</value>

39.</property>

40.<property>

41.<name>dfs.namenode.http-address.cluster1.hadoop2</name>

42.<value>hadoop2:50070</value>

43.</property>

44.<property>

45.<name>dfs.namenode.servicerpc-address.cluster1.hadoop1</name>

46.<value>hadoop1:53310</value>

47.</property>

48.<property>

49.<name>dfs.namenode.servicerpc-address.cluster1.hadoop2</name>

50.<value>hadoop2:53310</value>

51.</property>

52.<property>

53.<name>dfs.ha.automatic-failover.enabled.cluster1</name>

54.<value>true</value>

55.</property>

56.【指定cluster1是否启动自动故障恢复，即当NameNode出故障时，是否自动切换到另一台NameNode】

57.<!--指定JournalNode -->

58.<property>

59.<name>dfs.namenode.shared.edits.dir</name> <value>qjournal://hadoop1:8485;hadoop2:8485;hadoop3:8485;hadoop4:8485;hadoop5:8485/cluster1</value>

60.</property>

61.【指定cluster1的两个NameNode共享edits文件目录时，使用的JournalNode集群信息】

62.<property>

63.<name>dfs.client.failover.proxy.provider.cluster1</name> <value>org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider</value>

64.</property>

65.【指定cluster1出故障时，哪个实现类负责执行故障切换】

66.<property>

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

68.<value>/home/tom/yarn/yarn\_data/tmp/journal</value>

69.</property>

70.【指定JournalNode集群在对NameNode的目录进行共享时，自己存储数据的磁盘路径。tmp路径是自己创建，journal是启动journalnode自动生成】

71.<property>

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

73.<value>sshfence</value>

74.</property>

75.【一旦需要NameNode切换，使用ssh方式进行操作】

76.<property>

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

78.<value>/home/tom/.ssh/id\_rsa</value>

79.</property>

80.【如果使用ssh进行故障切换，使用ssh通信时用的密钥存储的位置】

81.<property>

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

83.<value>10000</value>

84.</property>

85.<property>

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

87.<value>100</value>

88.</property>

89.</configuration>

2.4文件mapred-site.xml

view sourceprint?

1.<configuration>

2.<property>

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

4.<value>yarn</value>

5.</property>

6.</configuration>

7.【指定运行mapreduce的环境是yarn，与hadoop1不同的地方】

2.5文件yarn-site.xml

view sourceprint?

01.<configuration>

02.<property>

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

04.<value>hadoop1</value>

05.</property>

06.【自定义ResourceManager的地址，还是单点】

07.<property>

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

09.<value>mapreduce.shuffle</value>

10.</property>

11.</configuration>

2.6文件slaves

添加：这里指定哪台机器是datanode，这里指定4台机器。你甚至可以把集群所有机器都当做datanode

view sourceprint?

1.hadoop3

2.hadoop4

3.hadoop5

4.hadoop6

3、添加环境变量

环境变量的添加方法大都相同。这里给出我所有环境变量配置，大家可以根据自己的需要参考一下。

这里我们只要添加HADOOP\_HOME环境变量。

view sourceprint?

01.JAVA\_HOME=/usr/lib/jvm/jdk1.7.0\_51

02.export PATH=$PATH:$JAVA\_HOME/bin

03.export HBASE\_HOME=/home/tom/hadoop-2.2.0/app/hbase-0.94.6-cdh4.4.0

04.export HIVE\_HOME=/home/tom/hadoop-2.2.0/app/hive-0.12.0/

05.export HADOOP\_HOME=/home/tom/hadoop-2.2.0

06.export PATH=$PATH:$HBASE\_HOME/bin:$HIVE\_HOME/bin:$HADOOP\_HOME/bin:$HADOOP\_HOME/sbin

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

08.export ZOOKEEPER\_HOME=/home/tom/yarn/hadoop-2.2.0/app/zookeeper-3.4.5

09.export PATH=$PATH:$ZOOKEEPER\_HOME/bin

4、复制到其它节点

在hadoop的的根目录下（即：/home/tom目录下）：因为我们所有的环境都装载在hadoop1的tom目录下。

执行：

view sourceprint?

1.scp -r yarn hadoop2:/home/tom

2.scp -r yarn hadoop3:/home/tom

3.scp -r yarn hadoop4:/home/tom

4.scp -r yarn hadoop5:/home/tom

5.scp -r yarn hadoop6:/home/tom

注意点：

1、因为我们是把整个yarn目录复制到其他节点中，zookeeper也包含在内。事先我们定义zookeeper是在1-5台机器上部署。这里我们虽然把zookeeper拷贝到6机器中，但是我们再zookeeper配置文件中没有配置6机器的节点，在启动zookeeper的时候，6机器也不需要启动。

2、现在要做的是进入zookeeper目录下的zkdata目录，修改myid文件：各个myid内容对应zoo.cfg文件中server对应的编号。

按照上面的3个大步骤，以及在注释中自己要创建的文件夹，指定相应的路径之后，整体的hadoop环境算是搭建完毕。下面就是等测试。

看似简单的不能再简单的搭建过程，这是你弄明白之后的事情。在从hadoop1到2之间的过度，主要的变化是namenode和mapreduce到yarn架构之间的变化。就在这简单的配置过程中，加上可参考网上众多配置教程，也耽误了太长时间。不是文件难配置，而是在出现问题，不知道怎么解决，就一直卡在那儿。咨询过一些大牛，但是他们也是搪塞，没有给出真正问题的原因。其中有一个问题，在QQ群中，咨询过一个人，从他那边才得到启发，把其中的一个问题给解决掉。这也是我们遇到的问题，没有一个平台，导致在一些别人看似不是问题的问题上耽误太长时间。

上面的任务完成后。下面才是非常关键的步骤，任务的重中之重，同时也是问题出现的地方，也是卡时间最多的地方：测试整个集群的性能。由下一篇博客介绍。

**Hadoop 2.4.0+zookeeper3.4.6+hbase0.98.3分布式集群搭建**

（http://weir2009.iteye.com/blog/2082719）

|  |  |  |  |
| --- | --- | --- | --- |
| Ip | 主机名 | 程序 | 进程 |
| 192.168.137.11 | h1 | Jdk  Hadoop  hbase | Namenode  DFSZKFailoverController  Hamster |
| 192.168.137.12 | h2 | Jdk  Hadoop  hbase | Namenode  DFSZKFailoverController  Hamster |
| 192.168.137.13 | h3 | Jdk  Hadoop | resourceManager |
| 192.168.137.14 | h4 | Jdk  Hadoop  Zookeeper  hbase | Datanode  nodeManager  JournalNode  QuorumPeerMain  HRegionServer |
| 192.168.137.15 | h5 | Jdk  Hadoop  Zookeeper  Hbase | Datanode  nodeManager  JournalNode  QuorumPeerMain  HRegionServer |
| 192.168.137.16 | h6 | Jdk  Hadoop  Zookeeper  hbase | Datanode  nodeManager  JournalNode  QuorumPeerMain  HRegionServer |

准备工作

1.       修改Linux主机名

Vim /etc/sysconfig/network

添加 HOSTNAME=h1

2.       修改IP

***vim /etc/sysconfig/network-scripts/ifcfg-eth0***

***修改IPADDR=192.168.137.11***

3.       修改主机名和IP的映射关系

Vim /etc/hosts

添加192.168.137.11  h1

4.       关闭防火墙

 service iptables stop

5.       ssh免登陆

ssh-keygen –t rsa //产生公钥和私钥

拷贝公钥到其他电脑(h2为主机名)

ssh-copy-id -i h2

6.       安装JDK，配置环境变量等

这里可以在一台电脑上配置，然后拷贝到其他电脑

scp –r /home/jdk/  h2:/home/

都做完可以重启一下电脑

安装zookeeper

解压 tar –zxvf zookeeper-3.4.6.tar.gz

1.修改配置文件conf/ zoo\_sample.cfg  为zoo.cfg

mv zoo\_sample.cfg zoo.cfg

打开修改内容：

dataDir=/home/gj/zookeeper-3.4.6/data  //数据目录,可随意定义

最后面添加：

server.1=h4:2888:3888

server.2=h5:2888:3888

server.3=h6:2888:3888

// server.X=A:B:C

其中X是一个数字, 表示这是第几号server.

A是该server所在的IP地址.

B配置该server和集群中的leader交换消息所使用的端口.

C配置选举leader时所使用的端口.

注意这里需要创建data文件夹

进入data文件夹创建文件myid  内容为1

1表示这是第几号server, 与server.X=A:B:C中的X对应

2.将配置到的zookeeper拷贝到其他电脑（h2,h3）上

使用 scp -r 命令

分别修改 myid文件内容为2,3

1.       启动三个节点的 bin目录下的./zkServer.sh start

也可以将zookeeper 配置到环境变量里面

安装hadoop

修改文件：

1.hadoop-env.sh

export JAVA\_HOME=/usr/hadoop/jdk  //添加java环境

2.core-site.xml

<configuration>

 <!--指定hdfs的nameservice为ns1-->

 <property>

   <name>fs.defaultFS</name>

   <value>hdfs://ns1</value>

 </property>

<!--指定hadoop数据存放目录-->

 <property>

   <name>hadoop.tmp.dir</name>

   <value>/root/hadoop/hadoop-2.4.0/tmp</value>

 </property>

<!--指定zookeeper地址-->

 <property>

   <name>ha.zookeeper.quorum</name>

   <value>h4:2181,h5:2181,h6:2181</value>

 </property>

</configuration>

2.       hdfs-site.xml

<configuration>

<!--指定hdfs的nameservice为ns1，需要和core-site.xml中的保持一致 -->

<property>

   <name>dfs.nameservices</name>

   <value>ns1</value>

</property>

<!-- ns1下面有两个NameNode，分别是nn1，nn2 -->

<property>

   <name>dfs.ha.namenodes.ns1</name>

   <value>nn1,nn2</value>

</property>

<!-- nn1的RPC通信地址 -->

<property>

   <name>dfs.namenode.rpc-address.ns1.nn1</name>

   <value>h1:9000</value>

</property>

<!-- nn1的http通信地址 -->

<property>

    <name>dfs.namenode.http-address.ns1.nn1</name>

    <value>h1:50070</value>

</property>

<!-- nn2的RPC通信地址 -->

<property>

        <name>dfs.namenode.rpc-address.ns1.nn2</name>

        <value>h2:9000</value>

</property>

<!-- nn2的http通信地址 -->

<property>

        <name>dfs.namenode.http-address.ns1.nn2</name>

        <value>h2:50070</value>

</property>

<!-- 指定NameNode的元数据在JournalNode上的存放位置 -->

<property>

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

        <value>qjournal://h4:8485;h5:8485;h6:8485/ns1</value>

</property>

<!-- 指定JournalNode在本地磁盘存放数据的位置 -->

<property>

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

        <value>/root/hadoop/hadoop-2.4.0/journal</value>

</property>

<!-- 开启NameNode失败自动切换 -->

<property>

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

        <value>true</value>

</property>

<!-- 配置失败自动切换实现方式 -->

<property>

        <name>dfs.client.failover.proxy.provider.ns1</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>

<!-- 使用隔离机制时需要ssh免登陆 -->

<property>

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

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

</property>

</configuration>

4. mapred-site.xml.template 重命名为mapred-site.xml

<configuration>

<!-- 指定mr框架为yarn方式 -->

<property>

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

        <value>yarn</value>

</property>

</configuration>

5. yarn-site.xml

<configuration>

<!-- 指定resourcemanager地址 -->

        <property>

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

                <value>h3</value>

        </property>

<!-- 指定nodemanager启动时加载server的方式为shuffle server -->

        <property>

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

                <value>mapreduce\_shuffle</value>

        </property>

</configuration>

6.slaves

  h4

  h5

  h6

将在一台电脑上配置好的hadoop拷贝到其他电脑

启动hadoop  和zookeeper（已配置到环境变量里面）

1.先启动zookeeper在h4,h5,h6上

  zkServer.sh start

 查看状态zkServer.sh status（会发现有一个leader，两个follower）

2. 启动journalnode(在h1上启动)

   hadoop-daemons.sh start journalnode

3．格式化HDFS(在h1上启动)

hadoop namenode –format

此时会在hadoop目录里面产生tmp文件夹，将这个文件夹拷贝到h2上

3.       格式化ZK(在h1上启动)

hdfs zkfc –formatZK

4.       启动hadoop(在h1上启动)

start-all.sh

此时可能在h3上的resourceManager没有启动，可以进入h3启动start-yarn.sh

这时就可以通过web查看hadoop集群的各个状态，也可以用jps 命令查看进程

hbase 集群配置

1.       conf/hbase-env.sh

java\_home=java路径

export HBASE\_MANAGES\_ZK=false

使用独立的ZooKeeper时需要修改HBASE\_MANAGES\_ZK值为false，为不使用默认ZooKeeper实例。

2. conf/hbase-site.xml

<property>  
 <name>hbase.rootdir</name>  
 <value>hdfs://h1:9000/hbase</value>  
</property>  
<property>  
 <name>hbase.cluster.distributed</name>  
 <value>true</value>  
</property>  
<property>  
<name>hbase.master</name>  
<value>h1:60000</value>  
</property>  
 <property>  
 <name>hbase.master.port</name>  
 <value>60000</value>  
 <description>The port master should bind to.</description>  
 </property>  
   
 <property>  
   <name>hbase.zookeeper.quorum</name>  
   <value>h4,h5,h6</value>  
 </property>

3.conf/ regionservers

h4

h5

h6

启动hbase

在h1 上

start-hbase.sh

在h2上

start-hbase.sh

这是可以通过web查看hbase的状态  ，会发现像namenode一样有一个Active 状态的hmaster和Standby 状态的hmaster

至此完成集群。

**Hadoop2的ResourceManager高可用配置**

**（http://slaytanic.blog.51cto.com/2057708/1423227）**

Hadoop 2.2没怎么关注过，太新，bug太多。2.4出来以后关注了一些东西，比如2.4里面直接带了ResourceManager的高可用，这点比较吸引人。之前2.2没注意有没有，貌似是没有，然后CDH自己出了一个解决方案，这次2.4的更新直接自己带了，还不错，这样就全了，Namenode有HA和Federation，RM也有了HA，而且也可以通过ZKFC自动做故障切换。大概从2.4开始，Hadoop就可以往生产环境逐渐切换了。

直接记录配置 RM HA 的最小需求和配置项。跟NN的HA一样，RM的HA也需要两台机器硬件配置相同，这个没什么可解释的了，当初1代的时候，NN和SNN就必须是一模一样的硬件配置。就像配置NN的HA一样，RM的HA也需要给出servicename。以下配置是配置RM自动失效恢复的配置项，大概没多少人会用到手动恢复吧，用手动恢复就把zookeeper部分给干掉就行了。

<property>

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

<value>true</value>

</property>

<property>

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

<value>rm1,rm2</value>

</property>

<property>

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

<value>192.168.1.2</value>

</property>

<property>

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

<value>192.168.1.3</value>

</property>

<property>

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

<value>true</value>

</property>

<property>

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

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

</property>

<property>

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

<value>192.168.1.2:2181,192.168.1.3:2181</value>

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

</property>

<property>

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

<value>yarn-ha</value>

</property>

将配置写入yarn-site.xml里面，然后分别在两台服务器正常启动RM就可以了，就像启动NN HA一样的方式 sudo -u yarn yarn-daemon.sh start resourcemanager

另外，在Hadoop的各种HA中，有个隐藏属性是很多人不知道的，就是强制切换，一般来说，我们通过命令行切换HA，需要去运行

sudo -u hdfs hdfs haadmin -transitionToActive/transitionToStandby

或者

sudo -u yarn yarn rmadmin -transitionToActive/transitionToStandby

但是，这种方式在启用了ZKFC做自动失效恢复的状态下是不允许修改的，提示信息里只说了可以强制执行，但是没有提供命令，其实强制切换主备命令很简单。加个forcemanual就好了。

sudo -u hdfs hdfs haadmin -transitionToActive --forcemanual nn1

但是这样做的后果是，ZKFC将停止工作，你将不会再有自动故障切换的保障，但是有些时候，这是必须的，特别是有时候，Hadoop的NN在ZKFC正常工作的情况下，也会出现两个standby，两个standby的问题就在于诸如Hive和Pig这种东西，会直接报一个什么 Operation category READ is not supported in state standby 什么什么的，甚至你看着明明一个是active，一个是standby，也会报这个错误，这时候就必须手动强制切换了，强制切换完以后，别忘了重新启动ZKFC就好了。这个强制切换的要求就是用户必须没有任何对元数据的操作，这样才能有效的防止脑裂的发生。应该来说，进入安全模式再切换会比较稳妥一些。

补充: Hadoop ResourceManager的实现不像namenode只能有两个做HA，ResourceManager的HA可以多台。

[**Hadoop常见错误解决**](http://www.cnblogs.com/haoxinyue/archive/2013/06/06/hadoop.html)

(<http://www.cnblogs.com/haoxinyue/archive/2013/06/06/hadoop.html>)

1. 通过命令和查看日志文件查看hadoop启动和运行情况

在NameNode端，可以通过

tail -100 /var/log/hadoop/hadoop/hadoop-hadoop-namenode-hadoop-namenode.log

查看NameNode的运行日志

在DataNode端也可以通过

cat /var/log/hadoop/hadoop/hadoop-hadoop-datanode-hadoop-datanode1.log

查看DataNode的运行日志。

通过jps命令分别在datanode和namenode端运行，查看已启动的服务。

2. NameNode不能启动：

Cannot lock storage ……tmp/dfs/name. The directory is already locked.

也许是因为执行hadoop的账号对这个文件夹tmp/dfs/name没有权限。可以使用如下命令解决

chown -R hadoop:hadoop /usr/hadoop

3. DataNode不能启动：

在客户端日志显示 namenode namespaceID = 1713611278; datanode namespaceID = 596511341

这个问题基本上是因为在namenode端多次运行hadoop namenode –format 导致的。在hadoop的core-site.xml文件中（不同的hadoop版本名字会有不同）找到<name>hadoop.tmp.dir</name>，清空对应的文件夹。举例：

[hadoop@hadoop-datanode1 hadoop]$ cat core-site.xml

<?xml version="1.0"?>

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

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

<configuration>

<!--global properties -->

<property>

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

<value>/usr/hadoop/tmp</value>

</property>

[hadoop@hadoop-datanode1 tmp]$ rm -rf /usr/hadoop/tmp/\*

然后重新启动hadoop，在datanode端用jps看是否datanode已经启动了。

4. 运行wordcount程序时fs找不到文件夹：

Input path does not exist: hdfs://localhost:9000/user/input

在集群环境中，处理的文件都是在hdfs中，所以必须把要处理的文件复制到hadoop 某个文件夹中。下面举例，在fs中新建文件夹，把提前做好的wordcount文件复制带hdfs中，最后运行程序。

[hadoop@hadoop-namenode ~]$ hadoop fs -mkdir /tmp/wordcount/input

[hadoop@hadoop-namenode ~]$ hadoop fs -put /home/hadoop/wordcount/input /tmp/wordcount/input

[hadoop@hadoop-namenode ~]$ hadoop fs -ls /tmp/wordcount/input

hadoop jar /home/hadoop/hadoop-examples-1.1.2.jar wordcount /tmp/wordcount/input/input /tmp/wordcount/output

查看结果

hadoop fs -cat /tmp/wordcount/output/part-r-00000