# Hadoop

## 环境准备

1. 至少3台服务器
2. Jdk1.7
3. Hadoop-2.x

注：这里是安装原生的hadoop集群，jdk和hadoop的安装包可以自由选择，如果是安装CDH可以忽略这些步骤，这里不赘述CDH的安装，后面会专门阐述CDH的安装步骤。

## 安装

### 2.1创建hadoop用户

useradd hadoop

passwd hadoop

### 2.2安装jdk

1. 解压：tar -zxvf jdk-1.7.gz
2. 移动：mv jdk1.7 /usr/jdk1.7
3. 编辑配置：vi /etc/profile

JAVA\_HOME=/usr/jdk1.7

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

PATH=$JAVA\_HOME/bin:$PATH

export JAVA\_HOME CLASSPATH PATH

1. 立即生效配置信息：. /etc/profile（或source /etc/profile）
2. 验证：java -version

### 2.3安装ssh

ssh-keygen -t dsa -P '' -f ~/.ssh/id\_dsa

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

注意：在hadoop用户下，需要给authorized\_keys赋予600的权限，不然免密码登陆无效

补充：a.查看防火墙： chkconfig --list |grep iptables

Iptables 0:off 1:off 2:on 3:on 4:on 5:on 6:off

b.关闭chkconfig iptables off

验证：chkconfig --list |grep iptables

### 2.4安装hadoop

1. 解压：tar -zxvf hadoop-2.x.gz
2. 移动：mv hadoop-2.x /home/hadoop/source/hadoop2.5.1（路径自选）
3. 配置环境变量：vi /etc/profile

HADOOP\_DEV\_HOME=/home/hadoop/source/hadoop-2.5.1

HADOOP\_MAPARED\_HOME=${HADOOP\_DEV\_HOME}

HADOOP\_COMMON\_HOME=${HADOOP\_DEV\_HOME}

HADOOP\_HDFS\_HOME=${HADOOP\_DEV\_HOME}

YARN\_HOME=${HADOOP\_DEV\_HOME}

HADOOP\_CONF\_DIR=${HADOOP\_DEV\_HOME}/etc/hadoop

HDFS\_CONF\_DIR=${HADOOP\_DEV\_HOME}/etc/hadoop

YARN\_CONF\_DIR=${HADOOP\_DEV\_HOME}/etc/hadoop

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

PATH=$JAVA\_HOME/bin:$PATH:$HADOOP\_DEV\_HOME/bin:$HADOOP\_DEV\_HOME/sbin:$HIVE\_HOME/bin:$HBASE\_HOME/bin

export JAVA\_HOME CLASSPATH PATH HADOOP\_DEV\_HOME HADOOP\_MAPARED\_HOME HADOOP\_COMMON\_HOME HADOOP\_HDFS\_HOME YARN\_HOME HADOOP\_CONF\_DIR HDFS\_CONF\_DIR YARN\_CONF\_DIR

1. 立即生效配置信息：. /etc/profile（或source /etc/profile）
2. 验证环境变量：echo $HADOOP\_DEV\_HOME
3. 配置hadoop的conf：

注：hadoop2.x以后的版本conf文件夹改为etc文件夹了

A.修改hadoop-env.sh的JAVA\_HOME

export JAVA\_HOME=/usr/jdk1.7

B.配置hdfs-site.xml

<configuration>

<property>

<name>dfs.replication</name>

<value>3</value>

</property>

<property>

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

<value>file:///home/hadoop/hdfs/name</value>

<final>true</final>

</property>

<property>

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

<value>file:///home/hadoop/hdfs/data</value>

<final>true</final>

</property>

<property>

<name>dfs.namenode.secondary.http-address</name>

<value>10.211.55.12:9001</value>

</property>

<property>

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

<value>true</value>

</property>

<property>

<name>dfs.permissions</name>

<value>false</value>

</property>

</configuration>

C.配置core-site.xml

<configuration>

<property>

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

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

</property>

<property>

<name>fs.default.name</name>

<value>hdfs://10.211.55.12:9000</value>

</property>

</configuration>

D.配置mapred-site.xml

<configuration>

<property>

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

<value>yarn</value>

</property>

<property>

<name>mapreduce.jobhistory.address</name>

<value>10.211.55.12:10020</value>

</property>

<property>

<name>mapreduce.jobhistory.webapp.address</name>

<value>10.211.55.12:19888</value>

</property>

<property>

<name>mapreduce.jobtracker.http.address</name>

<value>10.211.55.12:50030</value>

</property>

<property>

<name>mapreduce.jobtracker.address</name>

<value>10.211.55.12:9001</value>

</property>

</configuration>

E.配置yarn-env.sh

export JAVA\_HOME=/usr/jdk1.7

F.配置yarn-site.xml

<configuration>

<!-- Site specific YARN configuration properties -->

<property>

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

<value>10.211.55.12:18040</value>

</property>

<property>

<name>yarn.resourcemanager.scheduler.address</name>

<value>10.211.55.12:18030</value>

</property>

<property>

<name>yarn.resourcemanager.webapp.address</name>

<value>10.211.55.12:18088</value>

</property>

<property>

<name>yarn.resourcemanager.resource-tracker.address</name>

<value>10.211.55.12:18025</value>

</property>

<property>

<name>yarn.resourcemanager.admin.address</name>

<value>10.211.55.12:18141</value>

</property>

<property>

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

<value>mapreduce\_shuffle</value>

</property>

<property>

<name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>

<value>org.apache.hadoop.mapred.ShuffleHandler</value>

</property>

</configuration>

G.复制mapred-queues.xml文件

cp mapred-queues.xml.template mapred-queues.xml

H.设置master和slaver

master文件：cloud001

slaver文件：cloud002，cloud003

### 2.5配置hosts

10.211.55.12 cloud001

10.211.55.15 cloud002

10.211.55.16 cloud003

### 2.6分发hadoop文件

将配置好的hadoop文件夹和环境变量文件（profile文件）用scp命令分发到集群的其他服务器。

注：文件夹要使用scp -r，文件只需使用scp即可。

### 2.7启动hadoop集群

首次启动需要格式化节点：hadoop namenode –format

启动hadoop：start-dfs.sh start-yarn.sh

查看hadoop集群的状态：hadoop dfsadmin –report

### 2.8添加新的节点

在namenode节点slaver文件中添加ip，然后用scp分发到各个集群中去，然后在新节点下启动：start-dfs.sh start-balancer.sh。在namenode中刷新一下节点：hadoop dfsadmin –refreshNodes

注：如果不balance，那么cluster会把新的数据存放在新的node上，这样会降低mapered的工作效率；设置平衡阀值，默认是10%，值越低各节点越平衡，但消耗时间也越长。

## 3.hadoop中的HDFS

### 3.1 DFS的定义与来源

（1）数据量越来越多，在一个操作系统管辖的范围存不下，那么就分配到更多的操作系统管理的磁盘中，但是不方便管理与维护，因此迫切需要一种系统来管理多台机器上的文件，这就是分布式文件管理系统。

（2）是一种允许文件通过网络在多台主机上分享的文件系统，可让多机器上的多用户分享文件和存储空间。

（3）通透性。让实际上是通过网络来访问文件的动作，由程序与用户来看，就像是访问本地的磁盘一般。

（4）容错。即使系统中有某些节点脱机，整体来说系统仍然可以持续运作而不会有数据丢失。

（5）分布式文件管理系统很多，hdfs只是其中一种。适合于一次写入多次读取，不支持并发写的情况。小文件不合适。

### 3.2 HDFS的shell

对hdfs操作的命令格式是hadoop dfs

|  |  |  |
| --- | --- | --- |
| 命令 | 路径 | 说明 |
| -ls | <path> | 查看hdfs下一级目录 |
| -lsr | <path> | 递归查看hdfs目录 |
| -mkdir | <path> | 创建目录 |
| -put | <input> <output> | 上传到hdfs |
| -get | <input> <output> | 从hdfs上下载 |
| -cat/-text | <path> | 查看文件内容 |
| -rm | <path> | 删除文件 |
| -rmr | <path> | 递归删除 |

## 4.Nameode

### 4.1 作用

是整个文件系统的管理节点。它维护着整个文件系统的文件目录树，文件/目录的元数据信息和每个文件对应的数据块列表。接收用户的操作请求（如：对hdfs的shell命令操作都属于用户操作范围）。

### 4.2 文件包含

（1）fsimage：元数据镜像文件，存储某一时段namenode内存元数据信息

（2）edits：操作的日志文件

（4）fstime：保存最近一次checkpoint的时间

注：namenode对应的配置文件是hdfs-site.xml的dfs.name.dir属性

## 5.DataNode

### 5.1 作用

提供真实数据文件的存储服务

### 5.2 描述

（1）文件块（Block）：最基本的存储单位，对于文件内容而言，一个文件的长度大小就是一个size，从文件的0偏移开始，按照固定的大小，顺序对文件进行划分编号，划分好每一个块的Block，hdfs的Block的默认大小是64M。如一个512M的文件，可以分为512/64=8个Block

（2）如果一个文件大的大小不足块的大小（如小于默认的64M），则会按照该文件的实际大小存储，不会占用整个块的大小，但是此时的Block只有一个

（3）replication的副本数默认是三个

注：replication的设置属性在hdfs-site.xml的dfs.replication属性中设置

## 6. SecondaryNameNode

来源：如果木有SecondaryNameNode，若namenode更新频繁（如启动次数过多）导致edits文件无比大，这样会导致下次启动namenode的时候非常的慢，这里先说明下，namenode的启动顺序，fsimage—edits，如果edits无比大，启动的时间会很长，甚至不可控，这样就需要有SecondaryNameNode。SecondaryNameNode会定时合并fsimage和edits，防止edits变大，导致启动namenode时间过长，这里SecondaryNameNode依赖两个配置：

fs.checkpoint.period：单位秒，默认值3600，检查点的间隔时间，当距离上次检查点执行超过该时间后启动检查点

fs.checkpoint.size：单位字节，默认值67108864，