CentOS6.6上搭建hadoop-2.4.1集群

本文利用virtualbox虚拟机创建3个CentOS系统的虚拟机，以此来模拟分布式集群环境的搭建过程。以下是搭建的过程和配置：

一、安装虚拟机系统所需的软件：

1、virtualbox 或者vmvare

2、CentOS-6.6-x86_64.iso

安装过程中注意事项：

1、宿主机可以是windows也可以是linux的任何衍生版本，虚拟机也可以是vmvare，本实验的宿主机系统是kubuntu14.04，虚拟机是virtualbox，一般建议如果在windows下用vmvare。

2、虚拟机安装设置：安装虚拟机时可将显存调到最大，CPU模拟核心调到最大，不用勾选3D加速，其他默认；安装CentOS的时候，不用安装图形界面，因为可以不用它，可以增强虚拟机的性能。

3、虚拟机的网卡类型最好设置成桥接，关于网卡类型的选择读者可以到网上去选择一下，设置成桥接的好处有两个：方便宿主机访问的方便性、减少虚拟机与宿主机数据传输的路由代价消耗。

4、本实验一共虚拟了3太计算机，每台虚拟机配置如下：奔腾1虚拟内核2.1GHz，512M内存，10G虚拟硬盘。

宿主机ip：192.168.215.124，奔腾双核 2.1GHz 4G内存 100G硬盘

3个机器的主机和ip规划如下：

IP地址	主机名	用途
192.168.215.250	master	namenode
192.168.215.251	slave1	datanode
192.168.215.252	slave2	datanode

二、系统设置

（所有步骤都需要在所有节点执行）

1. 修改主机名及ip地址解析

1）修改主机名

```
[root@master ~]# vim /etc/hostname        #（有些版本已经没有hostname可不用写）
[root@master ~]# cat /etc/sysconfig/network
NETWORKING=yes
HOSTNAME=master
```

2）增加ip和主机映射

```
[root@master ~]# vi /etc/hosts
127.0.0.1   localhost
```

```
192.168.215.250   master
192.168.215.251   slave1
192.168.215.252   slave2
```
（以上过程在三台机器上都要配置）

3）验证是否成功

```
[root@master ~]# ping slave1
PING slave1 (192.168.215.251) 56(84) bytes of data.
64 bytes from slave1 (192.168.215.251): icmp_seq=1 ttl=63 time=1.55 ms

[root@master ~]# ping slave2
PING slave1 (192.168.215.252) 56(84) bytes of data.
64 bytes from slave1 (192.168.215.252): icmp_seq=1 ttl=61 time=1.33 ms
```
能 ping 通说明已经 OK。

2. 关闭防火墙

```
[root@master ~]# chkconfig iptables off
```

3. SSH 免密码登陆

SSH 是保证 namenode 和 datanode 之间无障碍通信的基础，如果没有 SSH，集群内部各个节点之间的相互访问以及数据传输将会变得异常困难。

1）生成密钥与公钥

centos 的 ssh 服务已经系统自带了，如果是系统未自带，须自行安装 ssh 服务
登陆到 master，把生成的 id_rsa.pub（公钥）内容 cat 到 authorized_keys 文件中。同时登陆到 slave1, slave2，生成 id_rsa.pub，并把 slave1, slave2 各自的 id_rsa.pub 的内容 copy 到 master 中的 authorzied_keys 中。最后从 master 中 scp 到 hd2, hd3 的.ssh 目录中。

```
[root@master ~]# ssh-keygen -t rsa
```
一直按回车就行
```
[root@master ~]# cat id_rsa.pub >> authorized_keys
```
同上
```
[root@slave1 ~]# scp id_rsa.pub master:/root/.ssh/
[root@master .ssh]# cat id_rsa.pub >> authorized_keys
[root@slave2 ~]# scp id_rsa.pub master:/root/.ssh/
[root@master .ssh]# cat id_rsa.pub >> authorized_keys
```
这样，authorized_keys 里有三个机器的公钥对

2）scp authorized_keys 到 slave1,slave2

```
[root@master ~]# scp authorized_keys slave1:/root/.ssh/
[root@master ~]# scp authorized_keys slave2:/root/.ssh/
```
3）验证 ssh 登陆是否是免密码

（第一次需要密码，若配置正确的话之后就不用密码了。）

```
[root@master ~]# ssh master
Last login: Sat Mar 05 03:13:12 2016 from master
[root@master ~]#ssh slave1
Last login: Sat Mar 05 03:13:18 2016 from slaver1
[root@master ~]# ssh slave2
Last login: Sat Mar 05 03:13:23 2016 from slaver2
```

表明 ssh 配置成功

二、安装 jdk、hadoop 及设置环境变量

1. 下载 jdk、hadoop 安装包

http://www.oracle.com/technetwork/cn/java/javase/downloads/jdk7-downloads-1880260.html

https://archive.apache.org/dist/hadoop/common/hadoop-2.4.1/hadoop-2.4.1.tar.gz

2. 解压

```
[root@master software]# tar -zxvf jdk-7u79-linux-x64.gz
[root@master software]# tar -zxvf hadoop-2.4.1.tar.gz
[root@master software]# mv hadoop-2.4.1 /root/hadoop-2.4.1
[root@master software]# mv jdk1.7.0_79  /usr/lib/jvm/jdk1.7.0
```

3. 设置 Java 环境变量

以 root 用户登陆编辑/etc/profile，加入以下内容：

```
[root@hd1 software]# vi /etc/profile
```

```
#java
export JAVA_HOME=/usr/lib/jvm/jdk1.7.0
export JRE_HOME=$JAVA_HOME/jre
export PATH=$PATH:$JAVA_HOME/bin
export CLASSPATH=./:$JAVA_HOME/lib:$JAVA_HOME/jre/lib

#hadoop
export HADOOP_HOME=/root/hadoop-2.4.1
export HADOOP_HDFS_HOME=$HADOOP_HOME
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_YARN_HOME=$HADOOP_HOME
export HADOOP_CONF_DIR=$HADOOP_HOME/etc/hadoop
export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin:$HADOOP_HOME/lib
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib"
export JAVA_LIBRARY_PATH=$HADOOP_HOME/lib/native
```

4. 验证环境变量

```
[root@master ~]# java -version
java version "1.7.0_79"
Java(TM) SE Runtime Environment (build 1.7.0_79-b15)
Java HotSpot(TM) Client VM (build 24.79-b02, mixed mode)
[root@master]# hadoop
Usage: hadoop [--config confdir] COMMAND
       where COMMAND is one of:
  fs                   run a generic filesystem user client
  version              print the version
  ....
```

```
  ....
Most commands print help when invoked w/o parameters.
```

三、hadoop 集群设置

1. 修改 hadoop 配置文件

```
[root@master ~]# cd hadoop-2.4.1/etc/hadoop
```

1）找到 hadoop-env.sh、yarn-env.sh ，设置 JAVA_HOME 环境变量,找到里面的 JAVA_HOME,修改为实际路径

```
[root@master hadoop]# vim hadoop-env.sh  yarn-env.sh
JAVA_HOME=/usr/lib/jvm/jdk1.7.0
HADOOP_PID_DIR=/root/hadoop/tmp
[root@master hadoop]# vim slaves
```

2）slaves
这个文件配置所有 datanode 节点，以便 namenode 搜索

```
[root@master ~]# vi slaves
slave1
slave2
```

3）core-site.xml

```
 <configuration>
  <property>
    <name>fs.defaultFS</name>
    <value>hdfs://master:9000</value>
    <final>true</final>
  </property>
   <property>
    <name>hadoop.tmp.dir</name>
    <value>/root/hadoop/tmp</value>
    <final>true</final>
  </property>
</configuration>
```

4）hdfs-site.xml

```
<configuration>
  <property>
     <name>dfs.name.dir</name>
     <value>/root/hadoop/dfs/name</value>
     <final>true</final>
  </property>
  <property>
     <name>dfs.data.dir</name>
     <value>/root/hadoop/dfs/data</value>
     <final>true</final>
  </property>
```

```
  <property>
     <name>dfs.replication</name>
          <value>1</value>
  </property>
  <property>
          <name>dfs.permissions</name>
          <value>false</value>
      </property>
</configuration>
```

5）mapred-site.xml

```
<configuration>
      <property>
          <name>mapred.job.tracker</name>
          <value>master:9001</value>
          <final>true</final>
     </property>
</configuration>
```

6）yarn-site.xml

```
<configuration>
        <property>
                <name>yarn.resourcemanager.address</name>
                <value>master:8080</value>
        </property>
        <property>
                <name>yarn.resourcemanager.scheduler.address</name>
                <value>master:8081</value>
        </property>
        <property>
                <name>yarn.resourcemanager.resource-tracker.address</name>
                <value>master:8082</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>
```

2. 把以下文件复制到其它节点

```
[root@master  ~]# scp -R hadoop-2.4.1/ slave1:~
[root@master ~]# scp -R hadoop-2.4.1/ slave2:~
[root@master ~]# scp -R /usr/lib/jvm/jdk1.7.0/ slave1:/usr/lib/jvm/
[root@master ~]# scp -R /usr/lib/jvm/jdk1.7.0/ slave2:/usr/lib/jvm/
[root@master ~]# scp /etc/profile slave1:/etc/profile
[root@master ~]# scp /etc/profile slave2:/etc/profile
```

```
[root@master ~]# scp /etc/hosts slave1:/etc/hosts
[root@master ~]# scp /etc/hosts slave2:/etc/hosts
```

配置完成之后需要重启电脑,主要是完成系统配置以及环境变量初始化

3. namenode 初始化

只需要第一次的时候初始化，之后就不需要了

```
[root@master ~]# hdfs namenode -format
```

如果“Exiting with status 0”，就说明 OK。

```
16/02/23 12:26:33 INFO util.ExitUtil: Exiting with status 0
```

4. 启动集群

```
[root@master ~]# start-all.sh
This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh
Starting namenodes on [master]
master: starting namenode, logging to /root/hadoop/logs/hadoop-root-namenode-master.out
slaver1: starting datanode, logging to /root/hadoop/logs/hadoop-root-datanode-slaver1.out
slaver2: starting datanode, logging to /root/hadoop/logs/hadoop-root-datanode-slaver2.out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /root/hadoop/logs/hadoop-root-
secondarynamenode-master.out
starting yarn daemons
starting resourcemanager, logging to /root/hadoop/logs/yarn-root-resourcemanager-master.out
slaver2: starting nodemanager, logging to /root/hadoop/logs/yarn-root-nodemanager-slaver2.out
slaver1: starting nodemanager, logging to /root/hadoop/logs/yarn-root-nodemanager-slaver1.out
```

5. 查看各节点的状态

```
[root@master ~]# jps
3761 NameNode
3939 SecondaryNameNode
4074 ResourceManager
4142 Jps
```

```
[root@slaver1 ~]# jps
2624 NodeManager
2519 DataNode
2710 Jps
```

```
[root@master ~]# hdfs dfsadmin -report
Configured Capacity: 16245424128 (15.13 GB)
Present Capacity: 5699403776 (5.31 GB)
DFS Remaining: 5699346432 (5.31 GB)
DFS Used: 57344 (56 KB)
DFS Used%: 0.00%
Under replicated blocks: 0
Blocks with corrupt replicas: 0
Missing blocks: 0

-------------------------------------------------
Datanodes available: 2 (2 total, 0 dead)

Live datanodes:
```

```
Name: 192.168.215.252:50010 (slaver2)
Hostname: slaver2
Decommission Status : Normal
Configured Capacity: 8122712064 (7.56 GB)
DFS Used: 28672 (28 KB)
Non DFS Used: 5348343808 (4.98 GB)
DFS Remaining: 2774339584 (2.58 GB)
DFS Used%: 0.00%
DFS Remaining%: 34.16%
Configured Cache Capacity: 0 (0 B)
Cache Used: 0 (0 B)
Cache Remaining: 0 (0 B)
Cache Used%: 100.00%
Cache Remaining%: 0.00%
Last contact: Sat Mar 12 07:32:37 EST 2016

Name: 192.168.215.251:50010 (slaver1)
Hostname: slaver1
Decommission Status : Normal
Configured Capacity: 8122712064 (7.56 GB)
DFS Used: 28672 (28 KB)
Non DFS Used: 5197676544 (4.84 GB)
DFS Remaining: 2925006848 (2.72 GB)
DFS Used%: 0.00%
DFS Remaining%: 36.01%
Configured Cache Capacity: 0 (0 B)
Cache Used: 0 (0 B)
Cache Remaining: 0 (0 B)
Cache Used%: 100.00%
Cache Remaining%: 0.00%
Last contact: Sat Mar 12 07:32:37 EST
```

另外也可查看 http://master:50070，可以查看相关的文件系统状态
2016 以上说明都 OK。

6.宿主机访问

linux、windows 添加快捷访问

为了方便访问，在 windows 下我们也可以编辑 %systemroot%\system32\drivers\etc\hosts 文件，加入以下的 ip 和主机映射，在 linux 下编辑文件 /etc/hosts

```
192.168.215.250   master
192.168.215.251   slave1
192.168.215.252   slave2
```

这样，我们在从节点上也可以通过 http://slave1:8042/node 方式查看某节点运行情况，而没必要用 http://192.168.215.251:8042

7. wordcount 测试

为了更进一步验证 hadoop 环境，我们运行 hadoop 自带的例子 wordcount 看看我们的集群能否正常工作。

我们进入到相应目录运行自带的 jar 包，来测试 hadoop 环境是否 OK。

具体步骤：

1) hdfs 上创建目录

```
[root@master ~]# hadoop fs -mkdir /user/input/wordcount
[root@master ~]# hadoop fs -mkdir /user/output/
```

2) 上传文件到 hdfs

```
[root@master ~]# cat in1.txt
Hello World , Hello China, Hello Shanghai
I love China
Wang Yuan Long
[root@master ~]# hadoop fs -put in1.txt /input/wordcount
```

3) 运行 wordcount

```
[root@master ~]# cd hadoop-2.4.1/share/hadoop/mapreduce/
[root@master mapreduce]# hadoop jar hadoop/share/hadoop/mapreduce/hadoop-
mapreduce-examples-2.4.1.jar wordcount /user/input/wordcount
/user/output/wordcount

16/03/06 10:42:36 INFO client.RMProxy: Connecting to ResourceManager at
hd1/192.168.0.101:18040
16/03/06 10:42:38 INFO input.FileInputFormat: Total input paths to process : 2
16/03/06 10:42:38 INFO mapreduce.JobSubmitter: number of splits:2
16/03/06 10:42:38 INFO mapreduce.JobSubmitter: Submitting tokens for job:
job_1406105556378_0003
16/03/06 10:42:38 INFO impl.YarnClientImpl: Submitted application
application_1406105556378_0003
16/03/06 10:42:38 INFO mapreduce.Job: The url to track the job:
http://hd1:8088/proxy/application_1406105556378_0003/
16/03/06 10:42:38 INFO mapreduce.Job: Running job: job_1406105556378_0003
16/03/06 10:42:46 INFO mapreduce.Job: Job job_1406105556378_0003 running in uber
mode : false
16/03/06 10:42:46 INFO mapreduce.Job: map 0% reduce 0%
16/03/06 10:42:55 INFO mapreduce.Job: map 100% reduce 0%
16/03/06 10:43:01 INFO mapreduce.Job: map 100% reduce 100%
```

4) 查看运行结果

```
[root@slave1 mapreduce]$ hadoop fs -cat /user/output/part-r-00000
, 1
China 1
China, 1
Hello 3
How 1
I 1
Shanghai 1
World 1
Wang 1
Yuan 1
You 1
```

到此，全部结束。整个 hadoop-2.4.1 集群搭建过程全部结束。