Day01_Hadoop简介及集群安装

大数据-张军锋 Day01 hadoop简介 集群安装

概论

1. 起源于nutch项目(是一个搜索引擎)
   基于nutch又研发了其他的搜索引擎和延伸到数据的处理
   搜索:Lucene solr elasticsearch
   数据处理:Hadoop Avro
2. **分布式的产生**:传统服务器对新型业务的处理达到了瓶颈,主要处理大数据和高并发的问题,数据进行分布在多台电脑上.
   分布式的处理:利用网络把多台机器连接起来,使用消息系统来保持通信,以一个整体对外提供统一的服务
3. **集群**:可以实现分布式
   集群里面是一个一个的物理计算机,也就是节点
4. **分布式通信**:
   序列化:把文件序列化为二进制,进行传输
   再经过IPC协议(通信调用协议),分为两种:LPC(本地过程调用),RPC(远程过程调用).

克隆虚拟机

1. 关闭虚拟机,选择右键管理,选择克隆
2. 更改网卡地址以及IP

> `vi /etc/sysconfig/network-scripts/ifcfg-eth0` 编辑对应的配置文件，注意修改一下 ip地址以及网卡地址(网卡地址可以通过ifconfig命令查看)

配置服务器

> 1个主节点：master(192.168.89.200)，2个（从）子节点，slaver1(192.168.89.201)，slaver2(192.168.89.202)

配置主节点名

`vi /etc/sysconfig/network`
添加内容：

```
NETWORKING=yes
HOSTNAME=master
```

配置两台子节点名

`vi /etc/sysconfig/network`
添加内容：

```
NETWORKING=yes
HOSTNAME=slaver1
```

`vi /etc/sysconfig/network`
添加内容：

```
NETWORKING=yes
HOSTNAME=slaver2
```

配置hosts

> 打开主节点的hosts文件，要将文件的前两行注释掉 (注释当前主机的信息)并在文件中添加所有hadoop集群的主机信息。

`vi /etc/hosts`

```
192.168.15.128   master
192.168.15.129   slaver1
192.168.15.130   slaver2
```

保存之后，将主节点的hosts分别拷贝到其他两个子节点
`scp /etc/hosts root@192.168.15.129:/etc/`
`scp /etc/hosts root@192.168.15.130:/etc/`

然后分别执行(重启服务器也可以不执行下面的语句): `/bin/hostsname hostsname`

> 例如：master上执行 `/bin/hostsname master` ，使之生效。

配置ssh无密码访问

生成公钥密钥对，在每个节点上分别执行：`ssh-keygen -t rsa`
一直按回车直到生成结束
执行结束之后每个节点上的/root/.ssh/目录下生成了两个文件 id_rsa 和 id_rsa.pub
其中前者为私钥，后者为公钥

在主节点上执行： `cp id_rsa.pub authorized_keys`
将子节点的公钥拷贝到主节点并添加进authorized_keys
将两个子节点的公钥拷贝到主节点上，**分别在两个子节点上执行：**
`scp ~/.ssh/ id_rsa.pub root@master:/root/.ssh/id_rsa_slaver1.pub`
`scp ~/.ssh/ id_rsa.pub root@master:/root/.ssh/id_rsa_slaver2.pub`
然后在主节点上，将拷贝过来的两个公钥合并到authorized_keys文件中去
主节点上执行：
`cat id_rsa_slaver1.pub>> authorized_keys`
`cat id_rsa_slaver2.pub>> authorized_keys`

最后测试是否配置成功
在master上分别执行
`ssh slaver1`
`ssh slaver2`

> 能正确跳转到两台子节点的操作界面即可，同样在每个子节点通过相同的方式登录主节点和其他子节点也能无密码正常登录就表示配置成功。
>
> 这里的配置方式可以有多种操作步骤，最终目的是每个节点上的/root/.ssh/authorized_keys文件中都包含所有的节点生成的公钥内容。

将主节点的authorized_keys文件分别替换子节点的authorized_keys文件
主节点上用scp命令将authorized_keys文件拷贝到子节点的相应位置
`scp authorized_keys root@slaver1:/root/.ssh/`
`scp authorized_keys root@slaver2:/root/.ssh/`

安装jdk

查看系统已经装的jdk： `rpm -qa|grep jdk`
卸载jdk： `rpm -e --nodeps java-1.6.0-openjdk-javadoc-1.6.0.0-1.66.1.13.0.el6.x86_64`
安装JDK（三台机器都要安装）
安装在同一位置/opt/java/jdk1.7.0_72
下载JDK
解压JDK ： `tar -zxvf /opt/java/jdk-7u72-linux-x64.gz`
配置环境变量, 编辑profile文件：`vi /etc/profile`
在profile文件末尾添加以下代码:

```
export JAVA_HOME=/opt/java/jdk1.7.0_72
export JRE_HOME=$JAVA_HOME/jre
export PATH=$JAVA_HOME/bin:$PATH
export CLASSPATH=.:$JRE_HOME/lib/rt.jar:$JAVA_HOME/lib/dt.jar:$JAVA_HOME/lib/tools.jar
```

保存后，使刚才编辑的文件生效： `source /etc/profile`
测试是否安装成功： `java -version`

安装hadoop

> 下载hadoop，推荐官网上下载 http://hadoop.apache.org/releases.html

在master主机上安装hadoop
安装位置自定，例如安装在/usr目录下面
下载hadoop包，放在/usr目录下
解压hadoop： `tar -zxvf /opt/hadoop/hadoop-2.6.4.tar.gz`

在usr下面生成hadoop-2.6.4目录
配置环境变量： `vi /etc/profile`
在末尾添加：

```
export HADOOP_HOME=/usr/hadoop-2.6.4
export PATH=$PATH:$HADOOP_HOME/bin
```

保存后使新编辑的profile生效： `source /etc/profile`

配置hadoop

配置hadoop配置文件
需要配置的文件的位置为/hadoop-2.6.4/etc/hadoop，需要修改的有以下几个
hadoop-env.sh
yarn-env.sh
core-site.xml
hdfs-site.xml
mapred-site.xml
yarn-site.xml
slaves
其中hadoop-env.sh和yarn-env.sh里面都要添加jdk的环境变量：
hadoop-env.sh中

```
# The java implementation to use.
export JAVA_HOME=/opt/java/jdk1.7.0_72

# The jsvc implementation to use. Jsvc is required to run secure da
tanodes
# that bind to privileged ports to provide authentication of data t
ransfer
# protocol.  Jsvc is not required if SASL is configured for authent
ication of
# data transfer protocol using non-privileged ports.
#export JSVC_HOME=${JSVC_HOME}
```

yarn-env.sh中

```
# User for YARN daemons
export HADOOP_YARN_USER=${HADOOP_YARN_USER:-yarn}

# resolve links - $0 may be a softlink
export YARN_CONF_DIR="${YARN_CONF_DIR:-$HADOOP_YARN_HOME/conf}"

# some Java parameters
export JAVA_HOME=/opt/java/jdk1.7.0_72
```

core-site.xml中

```
<configuration>
    <property>
        <name>fs.defaultFS</name>
        <value>hdfs://master:9000</value>
    </property>
    <property>
        <name>io.file.buffer.size</name>
        <value>131072</value>
    </property>
    <property>
        <name>hadoop.tmp.dir</name>
        <value>file:/usr/temp</value>
    </property>
    <property>
        <name>hadoop.proxyuser.root.hosts</name>
        <value>*</value>
    </property>
    <property>
        <name>hadoop.proxyuser.root.groups</name>
        <value>*</value>
    </property>
</configuration>
```

hdfs-site.xml中

```
<configuration>
    <property>
        <name>dfs.namenode.secondary.http-address</name>
        <value>master:9001</value>
    </property>
    <property>
        <name>dfs.namenode.name.dir</name>
        <value>file:/usr/dfs/name</value>
    </property>
    <property>
        <name>dfs.datanode.data.dir</name>
        <value>file:/usr/dfs/data</value>
    </property>
    <property>
        <name>dfs.replication</name>
        <value>2</value>
    </property>
    <property>
        <name>dfs.webhdfs.enabled</name>
        <value>true</value>
    </property>
    <property>
        <name>dfs.permissions</name>
        <value>false</value>
    </property>
    <property>
        <name>dfs.web.ugi</name>
        <value>supergroup</value>
    </property>
</configuration>
```

mapred-site.xml中

```
<configuration>
    <property>
        <name>mapreduce.framework.name</name>
        <value>yarn</value>
    </property>
    <property>
        <name>mapreduce.jobhistory.address</name>
        <value>master:10020</value>
    </property>
    <property>
        <name>mapreduce.jobhistory.webapp.address</name>
        <value>master:19888</value>
    </property>
</configuration>
```

yarn-site.xml中

```
<configuration>
    <property>
        <name>yarn.nodemanager.aux-services</name>
        <value>mapreduce_shuffle</value>
    </property>
    <property>
        <name>yarn.nodemanager.aux-services.mapreduce.shuffle.clas
s</name>
        <value>org.apache.hadoop.mapred.ShuffleHandler</value>
    </property>
    <property>
        <name>yarn.resourcemanager.address</name>
        <value>master:8032</value>
    </property>
    <property>
        <name>yarn.resourcemanager.scheduler.address</name>
        <value>master:8030</value>
    </property>
    <property>
        <name>yarn.resourcemanager.resource-tracker.address</name>
        <value>master:8031</value>
    </property>
    <property>
        <name>yarn.resourcemanager.admin.address</name>
        <value>master:8033</value>
    </property>
    <property>
        <name>yarn.resourcemanager.webapp.address</name>
        <value>master:8088</value>
    </property>
</configuration>
```

拷贝hadoop安装文件到子节点
主节点上执行：
`scp -r /usr/hadoop-2.6.4 root@slaver1:/usr`
`scp -r /usr/hadoop-2.6.4 root@slaver2:/usr`

拷贝profile到子节点
主节点上执行：
`scp /etc/profile root@slaver1:/etc/`
`scp /etc/profile root@slaver2:/etc/`
在两个子节点上分别使新的profile生效： `source /etc/profile`

格式化主节点的namenode
主节点上进入hadoop目录，然后执行：`./bin/hadoop namenode -format`
新版本用下面的语句不用hadoop命令了 `./bin/hdfs namenode -format`

> 提示：successfully formatted表示格式化成功

启动hadoop，主节点上在hadoop目录下执行： `./sbin/start-all.sh`

主节点上jps进程有：
NameNode
SecondaryNameNode
ResourceManager
每个子节点上的jps进程有：
DataNode
NodeManager
如果这样表示hadoop集群配置成功

要想在浏览器中进行访问，必须开放端口，这里我们直接将防火墙进行关闭

```
service iptables stop 关闭防火墙
chkconfig ipdatables off 开机不启动
```

```
enter code here
```

异常信息

1. 如果出现 doesn't satisfy minimum 异常信息，需要指定内存信息，在yarn-site.xml中配置如下配置文件

```
<property>
<!--NodeManager总的可用物理内存。注意，该参数是不可修改的，一旦设置，整个运行过程中不 可动态修改。另外，该参数的默认值是8192MB，因此，这个值通过一 定要配置。不过，Apache已经正在尝试将该参数做成可动态修改的。-->
        <name>yarn.nodemanager.resource.memory-mb</name>
        <value>1024</value>
    </property>
    <property>
        <name>yarn.nodemanager.resource.cpu-vcores</name>
        <value>1</value>
 </property>
 nodemanager要求的内存较低1024MB
```

2. core-site.xml文件中配置的缓冲区大小详见官方文档http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-

common/ClusterSetup.html

```
<property>
    <name>io.file.buffer.size</name>
    <value>131072</value>
</property>
```