集群安装部署

## 安装前准备

### 准备服务器

10.10.60.204 master

10.10.60.205 slave1

10.10.60.206 slave2

10.10.60.207 slave3

分别在四台服务器上创建hadoop、spark、utils三个用户

### 配置ip与hostname

用root用户修改每台机器的hosts

Vi /etc/hosts

添加以下内容：

10.10.60.204 master

10.10.60.205 slave1

10.10.60.206 slave2

10.10.60.207 slave3

应用配置source /etc/hosts

### 关闭防火墙

切换到管理员 su root

连接设备，键入命令“service iptables status”查看防火墙状态

关闭命令“chkconfig iptables off”，重启后生效。

### ssh免密码登录

* **SSH检查**

首先确认系统已经安装SSH，切换到管理员 su root

键入命令:

rpm -qa | grep openssh

rpm -qa | grep rsync

出现如下图信息表示已安装。

假设没有安装ssh和rsync，可以通过下面命令进行安装。

安装SSH协议--> yum install ssh

安装rsync工具--> yum install rsync

启动服务--> service sshd restart

* **SSH免密码登录的设置，每台设备的操作都是一样的，以10.10.60.204为例：**

切换用户 su - hadoop，执行ssh-keygen -t rsa生成密钥，一直按回车就行

进入.ssh目录 cd ~/.ssh，执行：

cp id\_rsa.pub authorized\_keys

本机器测试，执行：

ssh master

如果没有提示输入密码表示免密登陆成功

打包 .ssh 下的所有文件分发到各子节点

tar -czvf ssh.tar.gz .ssh/

scp ssh.tar.gz slave1:~/

scp ssh.tar.gz slave2:~/

scp ssh.tar.gz slave3:~/

在各节点上执行 tar -xzvf ssh.tar.gz

* **常见问题**

如果以上操作做完后，ssh登录还需要密码，请检查文件和文件夹权限

.ssh目录和用户目录/hadoop的权限必须是700

修改权限命令--> chmod 700 hadoop

.ssh目录下的authorized\_keys文件的权限必须是600

修改权限命令--> chmod 600 authorized\_keys

### 1.5 软件准备

所有软件已在在服务器/root/package目录下

hadoop-2.7.4.tar.gz

jdk-8u144-linux-x64.tar.gz

scala-2.11.11.tgz

spark-2.2.0-bin-hadoop2.7.tgz

zookeeper-3.4.10.tar.gz

安装软件并配置环境变量

用root将hadoop解压到/home/hadoop目录下

tar -xzvf hadoop-2.7.4.tar.gz

用root将zookeeper解压到/home/zookeeper目录下

tar -xzvf zookeeper-3.4.10.tar.gz

用root将spark解压到/home/spark目录下

tar -xzvf spark-2.2.0-bin-hadoop2.7.tgz

用root将java 和scala解压到/home/utils目录下

tar -xzvf jdk-8u144-linux-x64.tar.gz

tar -xzvf scala-2.11.11.tgz

### 1.6 配置环境/etc/profile添加以下内容

export MAVEN\_HOME=/home/utils/maven

export JAVA\_HOEM=/home/utils/java

export SCALA\_HOME=/home/utils/scala

export SPARK\_HOME=/home/spark/spark

export HADOOP\_HOME=/home/hadoop/hadoop

export ZK\_HOME=/home/zookeeper/zookeeper

export PATH=${JAVA\_HOME}/bin:${SCALA\_HOME}/bin:${SPARK\_HOME}/sbin:${HADOOP\_HOME}/sbin:${SPARK\_HOME}/bin:${HADOOP\_HOME}/bin:${ZK\_HOME}/bin:$PATH

## 2 配置zookeeper

### 2.1 配置zoo.cfg

tickTime=2000

initLimit=10

syncLimit=5

dataDir=/home/zookeeper/data

clientPort=2181

server.1=slave1:2888:3888

server.2=slave2:2888:3888

server.3=slave3:2888:3888

## 3 配置hadoop

### 修改core-site.xml

<configuration>

<!-- 指定hdfs的nameservice为 bigdata -->

<property>

<name>fs.defaultFS</name>

<value>hdfs://bigdata/</value>

</property>

<!-- 指定hadoop临时目录 -->

<property>

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

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

</property>

<property>

<name>hadoop.data.dir</name>

<value>/home/hadoop/data/data</value>

</property>

<!-- 指定zookeeper地址 -->

<property>

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

<value>slave1:2181,slave2:2181,slave3:2181</value>

</property>

</configuration>

### 修改hadoo-env.sh

export JAVA\_HOME=/home/utils/java

### 修改hdfs-site.xml

<configuration>

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

<property>

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

<value>false</value>

</property>

<property>

<name>dfs.nameservices</name>

<value>bigdata</value>

</property>

<!-- bigdata（逻辑名）下面有两个NameNode，分别是nn1，nn2 -->

<property>

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

<value>nn1,nn2</value>

</property>

<!-- nn1的RPC通信地址 -->

<property>

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

<value>master:9000</value>

</property>

<!-- nn1的http通信地址 -->

<property>

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

<value>master:50070</value>

</property>

<!-- nn2的RPC通信地址 -->

<property>

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

<value>slave1:9000</value>

</property>

<!-- nn2的http通信地址 -->

<property>

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

<value>slave1:50070</value>

</property>

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

<property>

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

<value>qjournal://slave1:8485;slave2:8485;slave3:8485/bigdata</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>/home/hadoop/data/journaldata</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.bigdata</name>

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

</property>

<!-- 配置隔离机制方法，多个机制用换行分割，即每个机制暂用一行-->

<property>

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

<value>

sshfence

shell(/bin/true)

</value>

</property>

<!-- 使用sshfence隔离机制时需要ssh免登陆 -->

<property>

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

<value>/usr/local/.ssh/id\_rsa</value>

</property>

<!-- 配置sshfence隔离机制超时时间 -->

<property>

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

<value>30000</value>

</property>

</configuration>

### 修改yarn-site.xml

<configuration>

<!-- 开启RM高可用 -->

<property>

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

<value>true</value>

</property>

<!-- 指定RM的cluster id -->

<property>

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

<value>yrc</value>

</property>

<!-- 指定RM的名字 -->

<property>

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

<value>rm1,rm2</value>

</property>

<!-- 分别指定RM的地址 -->

<property>

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

<value>master</value>

</property>

<property>

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

<value>slave1</value>

</property>

<!-- 指定zk集群地址 -->

<property>

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

<value>slave1:2181,slave2:2181,slave3:2181</value>

</property>

<property>

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

<value>mapreduce\_shuffle</value>

</property>

<property>

<name>yarn.scheduler.minimum-allocation-mb</name>

<value>1024</value>

</property>

<property>

<name>yarn.scheduler.maximum-allocation-mb</name>

<value>102400</value>

</property>

<property>

<name>yarn.scheduler.minimum-allocation-vcores</name>

<value>8</value>

</property>

<property>

<name>yarn.scheduler.maximum-allocation-vcores</name>

<value>32</value>

</property>

<property>

<name>yarn.nodemanager.pmem-check-enabled</name>

<value>false</value>

</property>

<property>

<name>yarn.nodemanager.vmem-check-enabled</name>

<value>false</value>

</property>

<property>

<name>yarn.nodemanager.resource.memory-mb</name>

<value>204800</value>

</property>

<property>

<name>yarn.nodemanager.resource.cpu-vcores</name>

<value>32</value>

</property>

</configuration>

### 修改yarn-env.sh

export JAVA\_HOME=/home/utils/java

### 修改mapred-site.xml

<configuration>

<!-- 指定mr框架为yarn方式 -->

<property>

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

<value>yarn</value>

</property>

</configuration>

### 修改slave

slave1

slave2

slave3

## 4 配置spark

### 4.1 配置spark-default.conf

spark.master spark://master:7077

spark.eventLog.enabled true

spark.eventLog.dir hdfs://master:9000/sparklog

spark.driver.memory 5g

### 4.2 配置spark-env.sh

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

**以上操作只在10.10.60.204上完成，完成后将对应文件同步到 slave1 slave2 slave3 三个节点**

## 5启动服务

### 启动Zookeeper集群

分别在slave1，slave2，slave3上启动Zookeeper·

zkServer.sh start

启动成功后slave1，slave2，slave3会有**QuorumPeerMain**进程

### 创建命名空间（格式化bigdata）

在master上执行：

hdfs zkfc -formatZK

### 格式化主NameNode节点（namenode1）

在master上执行命令：

hdfs namenode -format

### 启动主NameNode节点

在master上执行：

hadoop-daemon.sh start namenode

启动成功后master会有**NameNode**进程

### 格式备NameNode节点（namenode2）

在slave1上执行命令：

hdfs namenode -bootstrapStandby

### 启动备NameNode节点（namenode2）

在slave1上执行：

hadoop-daemon.sh start namenode

启动成功后slave1会有**NameNode**进程

### 启动集群

在master上执行

start-dfs.sh

启动成功后master，slave1上会有**DFSZKFailoverController**进程

slave1，slave2，slave3会有**JournalNode**进程

master,slave1，slave2，slave3会有**DataNode**进程

### 启动YARN（namenode1）

在master,slave1上执行：

start-yarn.sh

mr-jobhistory-daemon.sh start historyserver

启动成功后master,slave1其中一台上会有resourcemanager进程

master,slave1，slave2，slave3上会有nodemanager进程

## 查看hdfs

hadoop fs -ls hdfs://bigdata/