阿里云搭建集群

一 概述

1. 阿里云环境：

   共有 9 台机器，1 台作为 master，其余 8 台作为 slave

   每台机器 CPU：8 核，内存：32G

2. 集群使用软件：

   （1）基础软件：jdk，sdk

   （2）分布式集群：Hadoop，Flink，Spark

3. 集群具体要求：

   需要在 root 用户下安装基础软件和搭建分布式集群

   建立多个用户，每个用户只能使用软件，但是不能修改基础配置

4. 安装步骤概览：

   （1）修改 hostname，配置 hosts

   （2）进行磁盘挂载

   （3）配置 ssh 免密通信

   （4）安装 jdk

   （5）安装 sdk

   （6）安装 hadoop

   （7）安装 spark

   （8）安装 flink

   （9）对防火墙进行设置

   （10）集群用户管理

二 搭建集群具体步骤

1. 更改 hostname，配置 hosts 文件

（1）对每台服务器修改 hostname

进入/etc/sysconfig/network 将 HOSTNAME=NEWNAME

其中 NEWNAME 为即将设置的 hostname

（如果 linux 版本为 ubantu，则直接修改/etc 下的 hostname 即可）

（2）运行命令 `sudo vim /etc/hosts`

hosts 文件修改如下所示：

```
ip master

ip slave1

ip slave2
```

(3) 在 master 上修改 hosts 文件，运行命令

`scp /etc/hosts 用户名@ip:/etc/hosts`

将该文件 scp 到其他节点（可在 ssh 免密通信之后再做）

2. 进行磁盘挂载

对每个节点都需要进行磁盘挂载，具体步骤如下：

（1）运行 **`fidsk -l`** 命令查看数据盘，一般在/dev 下面，且此数据盘在挂载前通过 **`df -hl`** 命令无法查看到

（2） 假设数据盘为/dev/sdb,需要对进入该磁盘，对磁盘进行分区，运行命令 **`fdisk /dev/sdb`**

在运行此命令后会出现一些选择项，主要包括：

Command (m for help): n

（n 表示不需要帮助，m 表示需要帮助）

Command action

e　　extended

p　　primary partition (1-4)

（e 表示创建扩展分区，p 表示创建逻辑分区）

Partion number(1-4)：1

（输入 1 表示进入划分逻辑分区阶段）

First cylinder (51-125, default 51):

Last cylinder or +size or +sizeM or +sizeK (51-125, default 125):

（两处均输入回车，分别表示分区 start 值得默认值和分区大小的默认值，防止空间浪费）

（3）如果分区成功，运行 `fdisk -l` 命令，就可以看到/dev/sdb1

（4）格式化分区：运行命令 `mkfs.ext3 /dev/sdb1`

（共有四种格式化的方式，包括 ext1,ext2,ext3,ext4，不同格式功能不同，可根据自己的需求进行选择）

（5）创建 dataDisk 目录：运行 `mkdir /dataDisk`

（这样 dataDisk 文件夹就创建在/下面了）

（6）标记分区信息，保证下次重启时，磁盘依旧保持挂载状态

运行命令 `echo /dev/sdb1 /dataDisk ext3 defaults 0 0 >> /etc/fstab`

（8）进行分区挂载：运行命令 `mount /dev/sdb1/dataDisk`

（9）运行命令 `df -hl` 可以查看到/dev/sdb1 的大小和目前挂载在/dataDisk 下面

3. 配置 ssh 免密通信

因为有的机器不一定会有 known_keys 文件，为了确保万无一失，需要进行如下步骤：

（1）运行命令 `rpm -qa | grep sh` 查看是否存在 ssh，如果没有，则需要安装 ssh

（2）运行命令 `ls /root` 查看/root 下面有没有.ssh 文件夹，如果没有，运行命令 `mkdir /root/.ssh` 创建.ssh 文件夹

（3）运行命令 `ssh localhost` 确保.ssh 文件夹下存在 known_hosts 文件

（4）运行命令 `ssh-keygen -t rsa`

将会生成 id_rsa 和 id_rsa.pub 按照提示一直回车下去即可

（5）运行命令 `cat id_rsa.pub >> authorized_keys` 将公钥放入本地 autorized_keys 中

（6）修改.ssh 文件夹权限

运行命令

`chmod 700 ~/.ssh;chmod 600 ~/.ssh/authorized_keys`

（7）运行命令 `ssh localhost` 检查是否需要输入密码，如果需要输入密码，则输入后，运行命令 exit 再次 `ssh localhost` ssh 免密通信即可生效

每个服务器都需要先运行以上 6 个步骤，再做下面的步骤

（8）运行命令

`scp /root/.ssh/id_rsa.pub 用户名@ip:/root/.ssh/tmp`

将.pub 文件复制到其他节点的 tmp 文件中

（9）在对应的用户下面执行

`cat /root/.ssh/tmp >> /root/.ssh/authorized_keys`

将其他节点传来的公钥保存在 authorized_keys

（10）修改 authorized_keys 的权限，运行命令

`chmod 666 /root/.ssh/authorized_keys`

`chmod 700 /root/.ssh`

配置免密过程容易出现的问题及常见解决办法：

1.ping 目标服务器查看网络是否连通

2.检查密钥文件是否正确

3.查看防火墙配置，源和目标主机是否开通了 22 端口

4.sshd 服务器是否启动

5.查看/etc/ssh/sshd_config 文件配置，如是否允许 root 用户远程登录，是否启用了公钥认证等

6.ssh 不能连接还需查看/etc/ssh/sshd_config 文件配置，如是否允许 root 或普通用户远程登录，是否启用了公钥认证等，添加信任后，需重启 ssh 服务

7.运行命令 **tail /var/log/secure -n 20**，查看 debug 结果

8.运行命令 **sudo vi /etc/ssh/sshd_config** 找到#strictModes yes 修改其为 no

9.重启 ssh 服务，运行命令**/etc/init.d/sshd reload**

1. 出现 Authentication refused：

修改 authorized_keys 的权限，运行命令

chmod 666 /root/.ssh/authorized_keys

chmod 700 /root/.ssh

2. 如果出现 pam_unix(sshd:session): session closed for user root

执行命令

vim /etc/ssh/sshd_config

#PermitRootLogin no

修改成 PermitRootLogin yes

4. 安装 jdk

（1）从 oracle 官网上下载相应 jdk 版本,并解压到本地文件夹，这次 jdk 放在了/apps/jdk1.8.0 下

（2）配置 JAVA_HOME 环境变量：

运行命令 `vim ～/.bashrc` 对 bashrc 进行如下修改：

```
JAVA_HOME=/root/apps/jdk1.8.0_121
CLASSPATH=.:$JAVA_HOME/lib/tools.jar
PATH=$JAVA_HOME/bin:$PATH
export JAVA_HOME CLASSPATH PATH
（其中 JAVA_HOME 为 jdk 安装的目录）
```

（3）运行命令 `source ～/.bashrc` 让环境变量生效（每次修改环境变量以后都需要执行此步骤让其生效，下同）

（4）运行命令 `echo $JAVA_HOME` 查看 JAVA_HOME 变量显示是否正确

（5）运行命令 `java -version` 查看 java 版本是否正确，环境变量是否配置成功

（6）运行命令`$JAVA_HOME/bin/java -version` 如果显示和（4）步骤显示以及本身安装的 jdk 版本相同，则表明环境变量配置正确

环境变量一定要检查仔细，不然后期在安装其他软件时，会出现问题

5. 安装 sdk

从 scala 官网下载需要版本的 sdk，安装方式与安装 jdk 方式相同，也同样需要配置环境文件/.bashrc 在原有环境文件基础上，添加如下：

```
export SCALA_HOME=/root/apps/scala-2.12.0
export PATH=$SCALA_HOME/bin:$PATH
（其中 SCALA_HOME 是为 sdk 安装目录）
```

6. 安装 hadoop

（1）从官网上下载需要的 hadoop 版本，并解压到相应文件夹下，本次安装在/hadoop 下

（2）配置环境变量/.bashrc

```
export HADOOP_HOME=/root/hadoop/hadoop-2.7.1
export PATH=$HADOOP_HOME/sbin:$HADOOP_HOME/bin:$PATH
export HADOOP_CONF_DIR=$HADOOP_HOME/etc/hadoop
export CLASSPATH=$($HADOOP_HOME/bin/hadoop classpath):$CLASSPATH
（其中 HADOOP_HOME 是为 hadoop 安装目录）
```

（3）对 hadoop 相关 xml 文件进行配置

配置/etc/hadoop/slaves

配置/etc/hadoop/core-site.xml

配置/etc/hadoop/hdfs-site.xml

配置/etc/hadoop/mapred-site.xml

配置/etc/hadoop/yarn-sitexml

（4）运行命令 `scp -r /hadoop 用户名@slave*: /root`

将 hadoop 文件拷贝到各个节点上，其中*=1,2,3,4,5,6,7,8

（5）配置完成后，运行命令 `hdfs namenode -format`

如果出现 Exiting with status 0，则可以继续执行以下的步骤，如果 status 为 1，则需要重新查看*.xml 文件，查看配置是否正确

（6）运行命令`/hadoop/sbin/start-dfs.sh`

（7）运行命令`/hadoop/bin/hdfs dfsadmin -report`查看各个节点是否启动

（8）运行命令 **jps** 查看 master 节点上 namenode 和 secondary-namenode 是否启动，在其余 slave 节点上运行命令 jps 查看 datanode 是否启动

hadoop 相关问题和常见解决方法：

hadoop namenode 无法启动：

（1）运行命令**/hadoop/bin/hadoop dfsadmin -safemode leave**

（2）重启即可

hadoop datanode 无法启动：

（1）运行命令**/hadoop/bin/hdfs dfsadmin -refreshNodes**

（2）运行命令**/hadoop/sbin/stop-all.sh**

(3)将所有 slave 节点上的 tmp(即 hdfs-site.xml 中指定的 dfs.data.dir 文件夹，DataNode 存放数据块的位置)，log 文件夹删除，然后重新建立 tmp , logs 文件夹

（4）将所有 salve 节点上的/hadoop/conf 下的 core-site.xml 删除，将 master 节点的 core-site.xml 文件拷贝过来，到各个 salve 节点

（5）运行命令 **hadoop namenode -format**

（6）重启 hadoop

Error: Cannot find configuration directory:

终端执行 ./start-yarn.sh

starting yarn daemons

Error: Cannot find configuration directory: /etc/hadoop

Error: Cannot find configuration directory: /etc/hadoop

是找不到目录的原因，通过阅读相应的 shell 脚本可以找到解决方案~

解决方法：

在 hadoop-env.sh 配置一条 hadoop 配置文件所在目录：

export HADOOP_CONF_DIR=/opt/hadoop-2.7.1/etc/hadoop/

运行命令：**source hadoop-env.sh**

7. 配置防火墙白名单

正常情况下，我们需要关闭防火墙，以保证集群节点之间的正常访问，如果需要打开防火墙，那么则需要配置防火墙白名单

（1）运行命令 **vim /etc/sysconfig/iptables**

（2） 修改 iptables 的内容

Firewall configuration written by system-config-firewall

Manual customization of this file is not recommended.

*filter

:INPUT ACCEPT [0:0]

:FORWARD ACCEPT [0:0]

:OUTPUT ACCEPT [0:0]

-N whitelist

-A whitelist -s 133.133.61.112 -j ACCEPT

（此处 ip 地址为集群节点的 ip 地址和用户本地需要访问相关端口的主机）

-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

-A INPUT -p icmp -j ACCEPT

-A INPUT -i lo -j ACCEPT

-A INPUT -p tcp -j whitelist

-A INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j whitelist

（此处的 port number 为相关的与 hadoop 或者 spark 执行相关的端口，当然也可以将端口全部开放给白名单）

-A INPUT -j REJECT --reject-with icmp-host-prohibited

-A FORWARD -j REJECT --reject-with icmp-host-prohibited

COMMIT

（上边是白名单的 IP 列表，下边是针对白名单里的内容开启的一些端口，要把 ACCEPT 的写在上边，把 REJECT 的内容写在下边。）

（3）重启防火墙

运行命令 **service iptables stop**

运行命令 **service iptables start**

8. 安装 spark

（1）官网上下载需要的 spark 版本，并解压到相应文件夹下，本次安装在/spark 下

（2）对 spark 相关文件进行配置

配置/spark/conf/spark-env.sh 可以按照 spark-env.sh 内的模板按需配置，但是 JAVA_HOME 建议配置成绝对路径

*其中，SPARK_HISTORY_OPTS 必须要配置，需要在 hdfs 上创建目录，运行命令 **hdfs dfs -mkdir /log/spark_log** 创建存放 spark log 的目录，相应的配置中需要填入目录地址，如：

SPARK_HISTORY_OPTS="-Dspark.history.fs.logDirectory=hdfs://master:9000/log/spark_log"

配置/spark/conf/slaves 配置同/hadoop/etc/hadoop/slaves

配置/spark/conf/spark-defaults.conf

（3）运行命令 **scp -r /spark 用户名@slave*: /root**

将 spark 文件拷贝到各个节点上，其中*=1,2,3,4,5,6,7,8

（4）执行以下命令开启集群

/spark/sbin/start-master.sh

/spark/sbin/start-slaves.sh

/spark/sbin/start-historyserver.sh

9. 集群用户管理

集群用户管理包括：hdfs 文件目录管理和集群用户使用权限管理

（1）运行命令 `groupadd lab` 增加 lab 用户组

（2）运行命令 `adduser www` 增加用户名为 www 的用户

（3）运行命令 `passwd www` 为用户 www 修改登录密码

（4）运行命令 `gpasswd -a www lab` 将用户 www 增加到 lab 组内

集群用户使用权限管理：

（1）spark_log 对所用户开放：

运行命令 `hdfs dfs -chmod -R 777 /log/spark_log`

（2）hadoop 和 spark 集群只开放读和执行权限，不开放写权限，一是用户不可以修改相关的配置文件，二是集群在开启和关闭过程中需要写入 log 文件，不对用户开放写权限，相当于不允许用户私自开启和关闭集群

运行命令 `chmod -R a+x /hadoop`

`chmod -R a+x /spark`

hdfs 文件目录管理：

（1）运行命令 `hdfs dfs -mkdir /usr/www` 为 www 用户创建专属目录

（2）运行命令 `hdfs dfs -chown www:lab /usr/www` 将目录指定为用户 www 所属

（3）运行命令 `hdfs dfs -chmod -R 775 /usr/www` 更改目录权限

集群在使用过程中一些小 Tips:

1. 集群在长期使用过程中可能会出现磁盘空间占用过大等问题，出现这个问题需要考虑以下几个方面：

（1）数据盘是否挂载成功，是否将本该写入数据盘的数据写入磁盘空间

（2）是否存在系统中的程序依然在引用被删除的对象，需要通过命令查看 `lsof | grep '(deleted)'`

具体见：http://davidaq.com/technique/2015/07/18/linux-weird-disk-usage.html

2. Xshell 可以对多个集群节点同时进行相同操作。

3. 每次对集群进行操作以后，需要对各个节点查看进程情况（脚本 catJps.sh）

4. spark_home ， hdfs 和 spark 有相同的命令，环境变量中配置配置以后，会启动多个命令

配置/etc/hadoop/slaves

```
root@master:~/hadoop/hadoop-2.7.1/etc/hadoop# cat slaves
slave1
slave2
slave3
slave4
slave5
slave6
slave7
slave8
```

配置/etc/hadoop/core-site.xml

```
<configuration>
<property>
    <name>fs.defaultFS</name>
    <value>hdfs://master:9000</value>
    <description>NameNode URI</description>
</property>
<property>
    <name>hadoop.tmp.dir</name>
    <value>file:/dataDisk/hadoop/tmp</value>
    <description>Abase for other temporary directories.</description>
</property>
<property>
        <name>fs.hdfs.impl.disable.cache</name>
        <value>true</value>
</property>
</configuration>
```

配置/etc/hadoop/hdfs-site.xml

```
<configuration>
  <property>
    <name>dfs.namenode.name.dir</name>
    <value>file:/dataDisk/hadoop/hadoopdata/namenode</value>
  </property>
  <property>
    <name>dfs.datanode.data.dir</name>
    <value>file:/dataDisk/hadoop/hadoopdata/datanode</value>
  </property>
  <property>
    <name>dfs.namenode.secondary.http-address</name>
    <value>master:50090</value>
  </property>
  <property>
    <name>dfs.replication</name>
    <value>2</value>
  </property>
  <property>
    <name>dfs.datanode.max.xcievers</name>
    <value>4096</value>
  </property>
</configuration>
```

配置/etc/hadoop/mapred-site.xml

```
<configuration>
  <property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
    <description>Execution framework set to Hadoop YARN.</description>
  </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>
```

配置/etc/hadoop/yarn-site.xml

```
<property>
    <name>yarn.resourcemanager.hostname</name>
    <value>master</value>
    <description>ResourceManager host</description>
 </property>
 <property>
    <name>yarn.log-aggregation-enable</name>
    <value>true</value>
 </property>
 <property>
    <name>yarn.log.server.url</name>
    <value>http://master:19888/jobhistory/logs</value>
 </property>
 <property>
    <name>yarn.resourcemanager.scheduler.class</name>
    <value>org.apache.hadoop.yarn.server.resourcemanager.scheduler.capacity.CapacityScheduler</value>
 </property>
 <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle,spark_shuffle</value>
 </property>
 <property>
    <name>yarn.nodemanager.aux-services.spark_shuffle.class</name>
    <value>org.apache.spark.network.yarn.YarnShuffleService</value>
 </property>
```