

hadoop完全分布式集群

搭建完全分布模式

单机模式和伪分布模式一般用于简单的学习测试环境，在生产环境采用的是完全分布模式环境运行，接下来我们搭建一个5节点完全分布式环境。

主机规划如下：

IP地址	机器名称	作用	完整域名	运行帐号
192.168.182.11	Master	NameNode SecondaryNameNode	master.lab.hwadee.com	hadoop
192.168.182.12	RM01	ResourceManager (yarn)	rm01.lab.hwadee.com	hadoop
192.168.182.101	datanode01	DataNode NodeManager	datanode01.lab.hwadee.com	hadoop
192.168.182.102	datanode02	DataNode NodeManager	datanode02.lab.hwadee.com	hadoop
192.168.182.103	datanode03	DataNode NodeManager	datanode03.lab.hwadee.com	hadoop

*实际部署时请根据环境修改IP网段

实施步骤如下：

1. **hadoop完全分布式模式所需的环境与伪分布模式相同，只是会将不同的功能模块运行到不同主机上，这里先准备5台虚拟机，参考并配置好相应的IP和机器名。**
2. **参考单机模式环境配置1-4步骤完成java安装和hadoop帐号环境配置。**
(没有特殊需要，请把firewalld和selinux关闭)
3. **编辑/etc/hosts文件，维护完整域名。**

```
1 | [hadoop@master ~]$ sudo vi /etc/hosts
```

加入以下内容：

```
1 | 192.168.182.11 master master.lab.hwadee.com
2 | 192.168.182.12 rm01 rm01.lab.hwadee.com
3 |
4 | 192.168.182.101 datanode01 datanode01.lab.hwadee.com
5 | 192.168.182.102 datanode02 datanode02.lab.hwadee.com
6 | 192.168.182.103 datanode03 datanode03.lab.hwadee.com
```

说明：

```
1 | 在实际部署生产环境时，一般我们不会直接修改hosts文件，而是通过专用的DNS服务器来维护域名系
2 | 统。
```

4. **因为5个节点的hosts文件内容是相同内容，故可以直接将此文件分发到各个节点，当然也可以手工在各个节点上进行维护。**

```
1 [hadoop@master ~]$ sudo scp /etc/hosts root@192.168.182.12:/etc/hosts
2 [hadoop@master ~]$ sudo scp /etc/hosts root@192.168.182.101:/etc/hosts
3 [hadoop@master ~]$ sudo scp /etc/hosts root@192.168.182.102:/etc/hosts
4 [hadoop@master ~]$ sudo scp /etc/hosts root@192.168.182.103:/etc/hosts
```

完成后用ping命令验证结果是否正确，需要每个节点都能ping通主机名和完整域名，且返回IP正确无误。

5. 配置SSH免密登录

类似于伪分布式环境，完全分布式环境中需要进行ssh远程通讯，在这里我们维护一份公私钥，然后分发到所有节点，让各个节点使用相同的公私钥进行通讯，减少部署难度。

- 1) 在master节点上使用ssh-keygen生成一份没有密码的公钥和私钥

```
1 [hadoop@master ~]$ ssh-keygen
```

- 2) 使用ssh-copy-id分发公钥或cat 分发

```
1 [hadoop@master ~]$ ssh-copy-id hadoop@localhost
```

或者(注意检查 是否关闭selinux, ls -lZ检查一下)

```
1 [hadoop@master ~]$ cd .ssh
2 [hadoop@master .ssh]$cat id_rsa.pub >> authorized.keys
3 [hadoop@master .ssh]$chmod 600 authorized.keys
```

3) 使用 ssh进行第一次远程节点访问时，需要确认公钥指纹，默认情况下用户必须输入yes才能通过，当节点很多时，手工交叉确认操作十分繁琐，工作量巨大，这里我们使用ssh_keyscan提前将各个节点的指纹加入到known_hosts文件，然后再分发到各个节点。

操作过程如下：

- a.) 编辑节点清单，将各节点机器名加入其中，包括机器短名和完整域名

```
1 vi nodelist.txt
```

- b.) 使用以下命令将RSA公钥指纹加入到known_hosts文件中

```
1 [hadoop@master ~]$ ssh-keyscan -t rsa -f node-list.txt > .ssh/known_hosts
2
```

说明：

- 1 另一种方法是修改ssh客户端配置文件/etc/ssh/ssh_config文件中的StrictHostKeyChecking参数，由默认值ask改为no，让其直接跳过公钥指纹确认动作。公钥指纹作用是防止中间人攻击，对于系统安全来说还是有存在的必要。

4) 将.ssh目录复制到其它节

```
1 [hadoop@master ~]$ scp -r .ssh hadoop@rm01:~/
2 [hadoop@master ~]$ scp -r .ssh hadoop@datanode01:~/
3 [hadoop@master ~]$ scp -r .ssh hadoop@datanode02:~/
4 [hadoop@master ~]$ scp -r .ssh hadoop@datanode03:~/
```

5) 验证：在5个节点中任意一个ssh另一个节点，确认可以不需要密码直接登录访问

6. 在master主机上下载并解压hadoop程序。

```
1 [hadoop@Master ~]$ sudo yum install wget -y
2 [hadoop@Master ~]$ wget URL(下载地址)
3 [hadoop@Master ~]$ sudo chmod 777 /opt
4 [hadoop@master ~]$ tar zxvf hadoop-3.2.1.tar.gz -C /opt
5 [hadoop@master ~]$ ln -sf /opt/hadoop-3.2.1 /opt/hadoop
6 [hadoop@master ~]$ ln -sf /opt/hadoop /usr/local/hadoop
```

7. 为方便后续直接运行hadoop程序，将hadoop安装目录加入到帐号hadoop的PATH环境变量中。

```
1 [hadoop@master ~]$ vi .bash_profile
```

```
1 export HADOOP_HOME=/opt/hadoop
2 PATH=$PATH:$HOME/bin:$JAVA_HOME/bin:$JRE_HOME/bin:$HADOOP_HOME/bin:$HADO
  OP_HOME/sbin
3
```

以上参数，在root的登录脚本.bash_profile中也加入一份

8. 将环境配置文件分发到各个节点 (hadoop和root都更新)

```
1 [hadoop@master ~]$ scp .bash_profile hadoop@rm01:~/
2 [hadoop@master ~]$ scp .bash_profile hadoop@datanode01:~/
3 [hadoop@master ~]$ scp .bash_profile hadoop@datanode02:~/
4 [hadoop@master ~]$ scp .bash_profile hadoop@datanode03:~/
```

```
1 [hadoop@master ~]# scp .bash_profile hadoop@rm01:~/
2 [hadoop@master ~]# scp .bash_profile hadoop@datanode01:~/
3 [hadoop@master ~]# scp .bash_profile hadoop@datanode02:~/
4 [hadoop@master ~]# scp .bash_profile hadoop@datanode03:~/
5
```

9. 将hadoop帐号设置为免密码sudo执行管理命令(可选)。

默认安装时，如果将hadoop设置为了管理员，此时会把 hadoop帐号加入到wheel组，而wheel组成员可以执行所有管理员命令，只是在使用sudo时需要输入hadoop自身密码。如果需要以hadoop帐号自动以管理员身份执行一些管理任务，我们需要让hadoop不需密码即可执行。

```
1 [hadoop@master ~]$ sudo vi /etc/sudoer
```

增加以下一行内容：

```
1 | hadoop ALL=(ALL) NOPASSWD: ALL
```

也可以在/etc/sudoers.d/目录下新创建一个与用户名相同的文件，在基中加入此内容

```
1 | cat /etc/sudoers.d/hadoop
2 | hadoop ALL=(ALL) NOPASSWD: ALL
```

完成后等待几分钟(一般为上一次输入密码5分钟后,因为之前加入wheel组的帐号在使用sudo执行操作后,很多系统设置的是在5分钟内不用再次输入密码,可直接sudo操作),即可用hadoop帐号执行sudo操作,验证是否可以需要密码执行。其它各节点做相同操作。

相关配置文件及加载顺序

```
1 | HDFS: hadoop-env.sh -> core-default.xml -> core-site.xml -> hdfs-
    default.xml --> hdfs-site.xml
2 |
3 | Mapred: hadoop-env.sh -> core-default.xml -> core-site.xml ->
    mapred.default.xml --> mapred.site.xml
```

10. 修改 /opt/hadoop/etc/hadoop/hadoop-env.sh

增加java环境变量设置：

```
1 | export JAVA_HOME=/usr/lib/jvm/java
2 | export HADOOP_HOME=/usr/local/hadoop
3 |
4 | export HDFS_NAMENODE_USER=hadoop
5 | export HDFS_DATANODE_USER=hadoop
6 | export HDFS_SECONDARYNAMENODE_USER=hadoop
7 |
8 | export YARN_RESOURCEMANAGER_USER=root
9 | export YARN_NODEMANAGER_USER=root
10 |
11 | export HADOOP_PID_DIR=/var/lib/hadoop
12 | export HADOOP_LOG_DIR=/var/log/hadoop
13 |
```

Hadoop默认运行pid文件将会放到/tmp目录下,可以自行指定一个单独目录。

```
1 | [hadoop@master ~]$ mkdir /var/lib/hadoop
2 | [hadoop@master ~]$ mkdir /var/log/hadoop
3 | [hadoop@master ~]$ chown hadoop:hadoop /var/lib/hadoop
4 | [hadoop@master ~]$ chown hadoop:hadoop /var/log/hadoop
5 | [hadoop@master ~]$ ls -ld /var/lib/hadoop /var/log/hadoop
```

11. 同上修改mapred-env.sh文件, 修改JAVA_HOME变量为实际路径

```
1 vi mapred-env.sh
2 export JAVA_HOME=/usr/lib/jvm/java
```

```
1 vi yarn-env.sh
2 export JAVA_HOME=/usr/lib/jvm/java
```

12. 修改core-site.xml配置文件参数 (全局配置)

a.) 类似于伪分布式环境, 设置namenode的URI参数

```
1 <property>
2     <name>fs.defaultFS</name>
3     <value>hdfs://master.lab.hwadee.com</value>
4     <description>name: node URI Param</description>
5 </property>
6
```

b.) 根据情况设置hadoop临时目录位置, 这里指到/data/hadoop/tmp

```
1 <property>
2     <name>hadoop.tmp.dir</name>
3     <value>file:///data/hadoop/tmp</value>
4     <description>namenode tmp data direcrory</description>
5 </property>
6
```

```
1 mkdir -p /data/hadoop/tmp
```

13. 修改分布式文件系统配置文件hdfs-site.xml

a.) 指定namenode元数据目录, 此数据十分重要

```
1 <property>
2     <name>dfs.namenode.name.dir</name>
3     <value>file:///data/hadoop/hdfs/name</value>
4 </property>
5
```

```
1 mkdir -p /data/hadoop/hdfs/name
```

这里需要说明的是, dfs.namenode.name.dir可以是多个目录, 以逗号分隔, 多个目录下的内容完全一样, 相当于起到备份冗余作用, 建议在设置多个时, 将数据指定到不同硬盘。

b.) 指定hdfs数据目录

```

1 <property>
2   <name>dfs.datanode.data.dir</name>
3   <value>file:///data/hadoop/hdfs/data</value>
4 </property>
5

```

```
1 mkdir -p /data/hadoop/hdfs/data
```

c.) 指定hdfs文件块副本数，如果文件块副本数大于1，即可表文件有冗余功能
默认值为3，这里刚好有3个数据节点，故可以使用默认值。

```

1 <property>
2   <name>dfs.replication</name>
3   <value>3</value>
4 </property>
5

```

参考：

两个replication参数的意义： **

dfs.namenode.replication.min：数据块副本的最小份数；

dfs.namenode.safemode.threshold-pct:数据块满足最小副本数比例。

- 1 当HDFS上报的数据块比例值 小于dfs.namenode.safemode.threshold-pct时，HDFS将进入安全模式，此时hdfs对数据块进行复制操作，不允外界对数据块进行修改和删除等操作。如果threshold-pct为0，表示不进入安全模式，如果为1表示记远为安全模式，默认值为0.999f。
- 2 另外还有一个参数dfs.namenode.safemode.replication.min，此参数设置时请不要小于dfs.namenode.replication.min，否则数据可能会处于危险状态 。

d.) 设置 secondary节点位置信息,这里放到datanode01或者 rm01节点上，尽量不要跟namenode放在一起，可以相对提高一点安全性.

```

1 <property>
2   <name>dfs.namenode.secondary.http-address</name>
3   <value>datanode01.lab.hwadee.com:9868</value>
4 </property>
5
6 <!-- 同时指定namesecondary的工作目录 -->
7 <property>
8   <name>dfs.namenode.checkpoint.dir</name>
9   <value>file:///data/hadoop/hdfs/namesecondary</value>
10 </property>

```

hadoop 2.9.x与3.x在默认端口上有一些不同：

hadoop 2.9. x:

dfs.namenode.secondary.http-address	0.0.0.0:50090
dfs.namenode.secondary.https-address	0.0.0.0:50091
dfs.datanode.address	0.0.0.0:50010
dfs.datanode.http.address	0.0.0.0:50075
dfs.datanode.ipc.address	0.0.0.0:50020

hadoop 3.2.1:

dfs.namenode.secondary.http-address	0.0.0.0:9868
dfs.namenode.secondary.https-address	0.0.0.0:9869
dfs.datanode.address	0.0.0.0:9866
dfs.datanode.http.address	0.0.0.0:9864
dfs.datanode.ipc.address	0.0.0.0:9867

14. 修改MapReduce配置文件mapred-site.xml文件

在其中指定资源调度器yarn,运算临时目录, 以及相关class路径

```
1 [hadoop@master hadoop]$ cd $HADOOP_HOME/etc/hadoop
2 [hadoop@master hadoop]$ cp mapred-site.xml.template mapred-site.xml
3 [hadoop@master hadoop]$ vi mapred-site.xml
```

```
1 <property>
2   <name>mapreduce.framework.name</name>
3   <value>yarn</value>
4 </property>
5 <property>
6   <name>yarn.app.mapreduce.am.staging-dir</name>
7   <value>/data/hadoop/mapred/staging</value>
8 </property>
9
10 <property>
11   <name>mapreduce.application.classpath</name>
12   <value>$HADOOP_MAPRED_HOME/share/hadoop/mapreduce/*:$HADOOP_MAPRED_HOME/sh
13   are/hadoop/mapreduce/lib/*</value>
14 </property>
```

15. 修改资源调度配置文件yarn-site.xml

a.) 设置yarn运行节点

```
1 <property>
2   <name>yarn.resourcemanager.hostname</name>
3   <value>rm01.lab.hwadee.com</value>
4 </property>
```

b.) 设置yarn的数据使用方法, 这里默认为mapreduce_shuffle

```
1 <property>
2   <name>yarn.nodemanager.aux-services</name>
3   <value>mapreduce_shuffle</value>
4 </property>
5
```

c.) 设置是关CLASS环境 变量位置

```
1 <property>
2   <name>yarn.nodemanager.env-whitelist</name>
3   <value>JAVA_HOME,HADOOP_COMMON_HOME,HADOOP_HDFS_HOME,HADOOP_CONF_DIR,CLASSPA
4   TH_PREPEND_DISTCACHE,HADOOP_YARN_HOME,HADOOP_MAPRED_HOME</value>
5 </property>
```

d.) 设置yarn日志聚合功能

日志聚合是YARN提供的日志中央化管理功能，它可将运行完成的Container/任务日志上传到HDFS上，从而减轻NodeManager负载，且提供一个中央化存储和分析机制。

```
1 <!-- log setting -->
2 <property>
3   <name>yarn.log-aggregation-enable</name>
4   <value>true</value>
5   <description>开启日志聚合功能</description>
6 </property>
7
8 <property>
9   <name>yarn.log-aggregation.retain-seconds</name>
10  <value>864000</value>
11  <description>日志保留时间,单位秒</description>
12 </property>
13
14 <property>
15   <name>yarn.log-aggregation.retain-check-interval-seconds</name>
16   <value>86400</value>
17   <description>每隔多久检查一次日志,超过保留时间的将被删除</description>
18 </property>
19
20 <property>
21   <name>yarn.nodemanager.remote-app-log-dir</name>
22   <value>/logs</value>
23   <description>hdfs系统上保存日志的目录位置</description>
24 </property>
25
```

16. 配置workers文件，修改为数据节点域名(默认情况下，文件中只有localhost):

(在原2.9.x版本中，对应的文件名为slaves)

将内容修改为:

```
1 datanode01.lab.hwadee.com
2 datanode02.lab.hwadee.com
3 datanode03.lab.hwadee.com
```


17. 将软件和配置分发到各个节点。

为快速分发软件，可参考以下步骤：

a.) 创建一个需要分发的节点清单文件。

```
1 [hadoop@master ~]$ echo rm01 datanode01 datanode02 datanode03 > node-  
list.txt
```

b.) 对需要分发的每个节点的/opt和/data/hadoop设置相应权限

```
1 [hadoop@master ~]# for i in `cat node-list.txt`  
2 do  
3 ssh hadoop@$i "sudo mkdir -p /data/hadoop;  
4     sudo chown hadoop /data/hadoop;  
5     sudo chmod 777 /opt";  
6 done
```

c.) 分发hadoop软件及配置文件

```
1 [hadoop@master ~]# for i in `cat node-list.txt`  
2 do scp -r /opt/hadoop-2.9.2 hadoop@$i:/opt/  
3 done
```

hadoop软件及配置分发：

```
1 分发方式还可以使用rsync方式进行，rsync方式最大的特点是可以增量更新，而不必每次都全部复制  
一次，这在更新部分文件但需要分发到大量主机时特别有用。
```

18. 格式化hdfs分布式文件系统

```
1 [hadoop@master ~]# hdfs namenode -format
```

查看格式化成功后的文件系统样式：

```
1 find /data/hadoop
```

19. 启动分布式文件系统。

```
1 [hadoop@master ~]# start-dfs.sh
```

20. 验证hdfs是否正常启动

<http://master:9870>或<http://master.lab.hwadee.com:9870>

Overview 'master.lab.hwadee.com:9000' (active)

Started:	Wed Jun 03 13:22:44 +0800 2020
Version:	3.2.1, rb3cbbb467e22ea829b3808f4b7b01d07e0bf3842
Compiled:	Tue Sep 10 23:56:00 +0800 2019 by rohithsharmaks from branch-3.2.1
Cluster ID:	CID-684dd023-e373-40e6-b3e5-e4e67e4da9b5
Block Pool ID:	BP-1593563980-192.168.183.1-1591157120855

Summary

在各节点上查看进程(rm01暂时没有进程):

```

1 [hadoop@master ~]# jps
2 [hadoop@datanode01 ~]# jps
3 [hadoop@datanode02 ~]# jps
4 [hadoop@datanode03 ~]# jps

```


21. 启动yarn资源管理器(切记需要ssh到配置的运行resourcemanager的节点启动)

```

1 [hadoop@master ~]# ssh rm01
2 [hadoop@rm01 ~]# start-yarn.sh

```

22. 访问<http://rm01:8088>或<http://rm01.lab.hwadee.com:8088>，验证yarn正确启动。



All Applications

Cluster

About

Nodes

Node Labels

Applications

NEW

NEW SAVING

SUBMITTED

ACCEPTED

RUNNING

FINISHED

FAILED

KILLED

Scheduler

Tools

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved
0	0	0	0	0 B	24 GB	0 B	

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes
3	0	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation
Capacity Scheduler	[memory-mb (unit=Mi), vcores]	<memory:1024, vCores:1>	<memory:8192, vCores:4>

Show 20 ▾ entries

ID	User	Name	Application Type	Queue	Application Priority	StartTime	LaunchTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU VCoers	Allocated Memory MB	Reserved CPU VCoers	Re M
No data available in table															

Showing 0 to 0 of 0 entries

分别在datanode01， datanode02， datanode03上使用jps命令查看运行进程，会发现多出一个NodeManager 的进程。

rm01的进程情况：

```

1 [root@rm01 ~]# jps
2 8450 DataNode
3 8563 SecondaryNameNode
4 9064 ResourceManager
5 9225 NodeManager
6 9627 Jps
7

```

23. 运行example程序进行测试:

计算Pi值:

```

1 [hadoop@master ~]$
2 [hadoop@master ~]$ id
3 uid=1000(hadoop) gid=1000(hadoop) groups=1000(hadoop),10(wheel)
4 [hadoop@master ~]$ cd /opt/hadoop/
5 [hadoop@master ~]$ bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-
  examples-3.2.1.jar pi 10 10
6

```

运行任务后，HDFS上产生的目录情况，包括聚合日志目录

Hadoop
Overview
Datanodes
Datanode Volume Failures
Snapshot
Startup Progress
Utilities

Browse Directory

/
Go!

Show 25 entries
Search:

<input type="checkbox"/>	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name	
<input type="checkbox"/>	drwx-----	hadoop	supergroup	0 B	Jun 03 13:32	0	0 B	data	
<input type="checkbox"/>	drwxrwxrwt	hadoop	hadoop	0 B	Jun 03 13:32	0	0 B	logs	
<input type="checkbox"/>	drwxr-xr-x	hadoop	supergroup	0 B	Jun 03 13:32	0	0 B	user	

Showing 1 to 3 of 3 entries
Previous
1
Next