

```
./bin/spark-shell --master spark://spark-master:7077 \  
--conf spark.hadoop.fs.s3a.endpoint=http://172.26.0.6:9000 \  
--conf spark.hadoop.fs.s3a.access.key=minio \  
--conf spark.hadoop.fs.s3a.secret.key=minio123 \  
--conf spark.hadoop.fs.s3a.path.style.access=true \  
--conf  
spark.hadoop.fs.s3a.impl=org.apache.hadoop.fs.s3a.S3AFileSystem
```

```
./bin/spark-shell --master spark://spark-master:7077 --jars examples/  
delta-core_2.11-0.4.0.jar \  
--conf  
spark.delta.logStore.class=org.apache.spark.sql.delta.storage.S3SingleDri  
verLogStore \  
--conf spark.hadoop.fs.s3a.endpoint=http://172.26.0.6:9000 \  
--conf spark.hadoop.fs.s3a.access.key=minio \  
--conf spark.hadoop.fs.s3a.secret.key=minio123 \  
--conf spark.hadoop.fs.s3a.path.style.access=true \  
--conf  
spark.hadoop.fs.s3a.impl=org.apache.hadoop.fs.s3a.S3AFileSystem
```

```
./bin/spark-shell --master spark://spark-master:7077 \  
--conf  
spark.delta.logStore.class=org.apache.spark.sql.delta.storage.S3SingleDri  
verLogStore \  
--conf spark.hadoop.fs.s3a.endpoint=http://172.26.0.6:9000 \  
--conf spark.hadoop.fs.s3a.access.key=minio \  
--conf spark.hadoop.fs.s3a.secret.key=minio123 \  
--conf spark.hadoop.fs.s3a.path.style.access=true \  
--conf  
spark.hadoop.fs.s3a.impl=org.apache.hadoop.fs.s3a.S3AFileSystem
```

```
./bin/spark-submit --master spark://spark-master:7077 \  
--conf  
spark.delta.logStore.class=org.apache.spark.sql.delta.storage.S3SingleDri  
verLogStore \  
--conf spark.hadoop.fs.s3a.endpoint=http://172.22.0.4:9000 \  
--conf spark.hadoop.fs.s3a.access.key=minio \  
--conf spark.hadoop.fs.s3a.secret.key=minio123 \  
--conf spark.hadoop.fs.s3a.path.style.access=true \  
--conf
```

```
spark.hadoop.fs.s3a.impl=org.apache.hadoop.fs.s3a.S3AFileSystem \  
--jars /spark/examples/delta-core_2.11-0.5.0.jar \  
--class com.delta.Run examples/original-deltaLake2-1.0-SNAPSHOT.jar  
s3a://spark-test/ delta21 schemaCheck21
```

```
./bin/spark-shell --master spark://spark-master:7077 \  
--conf  
spark.delta.logStore.class=org.apache.spark.sql.delta.storage.S3SingleDriverLogStore \  
--conf spark.hadoop.fs.s3a.endpoint=http://172.22.0.4:9000 \  
--conf spark.hadoop.fs.s3a.access.key=minio \  
--conf spark.hadoop.fs.s3a.secret.key=minio123 \  
--conf spark.hadoop.fs.s3a.path.style.access=true \  
--conf  
spark.hadoop.fs.s3a.impl=org.apache.hadoop.fs.s3a.S3AFileSystem \  
--jars /spark/examples/delta-core_2.11-0.5.0.jar
```

```
val loans = spark.sql("""  
    SELECT addr_state, CAST(rand(10)*count as bigint) AS count,  
    CAST(rand(10) * 10000 * count AS double) AS amount  
    FROM loan_by_state_delta  
    """)
```

```
./bin/spark-submit --master spark://spark-master:7077 --jars examples/  
delta-core_2.11-0.4.0.jar --class com.delta.Run examples/original-  
deltaLake2-1.0-SNAPSHOT.jar s3a://spark-test/
```

Book [stack.cn/read/angel-v3.0](http://stack.cn/read/angel-v3.0)

## Spark 集群执行命令

1. 在 spark 的 conf 目录下的 aprk.env.sh(aprk.env.sh 是复制来于 aprk.env.sh.template) 里面有很多的相关的配置，目前初始阶段只涉及到  
spark\_master\_host = node-1  
spark\_master\_port=7077
2. 还是在 conf 目录下，这里定义了三个子节点，node-2, node-3, node-4
3. Centos7 防火墙的操作  
systemctl status firewalld 相对于的 status -> stop -> start 来查看，关

闭，开启防火墙

Centos7 安装

<https://www.bilibili.com/video/av76793551/>

<https://www.youtube.com/watch?v=ACypx1rwm6g&t=4s>

4. 查看 master 的管理页面

<http://192.168.31.100:8080/>

<http://192.168.31.100:7077/> master 和

目录所在地方: [/opt/spark](#)

5. 开启 spark 到 spark 文件的 sbin 文件下, 执行  
[start-all.sh](#) 开启, [stop-all.sh](#) 关闭

6. 开启 hdfs 文件系统

在 hadoop 的 sbin 目标下执行 [./start-dfs.sh](#) 命令

7. 查看 hdfs 文件

hdfs 的命令都在 [/opt/spark/hadoop/hadoop-2.6.5/bin](#) 目录中  
[./hdfs dfs -ls /](#)

在 web 页面可以看见 <http://192.168.31.100:50070>

8. 启动 spark 的 shell

在 [/opt/spark/spark/spark-2.2.0-bin-hadoop2.7/bin](#) 路径下

[./spark-shell](#) 如果直接执行这个启动 spark, 那么这是本地模式启动 spark

启动的时候指定 master 启动集群模式 [./spark-shell --master spark://node-1:7077](#)

并在 master 中会启动 SparkSubmit 进程, 这个进程主要是提交任务到每个 worker 的 Executor 中去执行

然后通过 master 的管理页面 <http://192.168.31.100:8080/> 可以查看到 Spark shell 正在运行的 spark application

启动之后, 再所有的节点 node 中都会出现一个 CoarseGrainedExecutorBackend 进程 这就是真正的跑集群任务

9. <http://192.168.31.100:8088/cluster> ResourceManager UI 界面

1. 启动 yarn [\\${HADOOP\\_HOME}/sbin/yarn.sh](#)

## 9. 计算第一个任务

```
sc.textFile("hdfs://node-1:9000/wordcount.txt").flatMap(_._split("
")).map((_,
    1)).reduceByKey(_+_).sortBy(_._2).collect
```

注意关闭 所有集群上的防火墙，否则计算的时候，集群之间无法通信

## 10. Master 负责资源调度，也就是决定在哪些 worker 中启动 Executor 和监控 worker，这和 Yarn 是很相似的 这就相当于 yarn 中的 ResourceManager，都是去完成资源调度。

worker 负责启动对应的执行任务的进程 (Excutor)，并且监控这个进程 (Excutor)，并且将当前集群的信息通过心跳汇报给

master，worker 相对于 yarn 而言就相当于 NodeManager，Executor 进程负责执行计算任务，

spark-submit 负责向 master 提交任务并申请资源，然后该任务下的 Executor 和 sparkSubmit 进行通信，监控 Executor

而这个 spark-submit 就相当于 yarn 中的 AppMater

## 11 提交任务到集群

打 jar 包并上传到集群机器 scp learning-1.0-SNAPSHOT.jar  
root@192.168.31.100:/opt/spark

在任何一台 node 中都可以提交任务

执行

```
./spark-submit --master spark://node-1:7077 --executor-memory
512m --total-executor-cores 2
--class com.bigData.spark.App /opt/spark/spark-code-jars/
learning-1.0-SNAPSHOT.jar hdfs://node-1:9000/wordcount.txt
hdfs://node-1:9000/wordcountResult121.txt
```

```
./spark-submit --class org.apache.spark.examples.SparkPi --master yarn
--deploy-mode cluster --driver-memory 1G --executor-memory 1G --
executor-cores 1 lib/spark-exampl
```

最后面几个参数分别是：jar 路径 执行的那个 class 需要的 2 个参数

安装 Centos 虚拟机，关键的步骤是设置好网络，这里这是 2 个网络，一个 NAT 网络，用于连接网络，一个 HOST-ONLY，用于 物理机 ssh 连接到虚拟

机，如果不使用物理机连接，那么就需要在虚拟机上安装增强功能，才可以用那些 复制粘贴等功能。

The screenshot shows the 'Network Adapter' configuration window for 'Adapter 1'. The 'Enable Network Adapter' checkbox is checked. The 'Attached to' dropdown is set to 'NAT'. The 'Name' field is empty. Under the 'Advanced' section, the 'Adapter Type' is 'Intel PRO/1000 MT Desktop (82540EM)', 'Promiscuous Mode' is 'Deny', and the 'MAC Address' is '080027B8D40D'. The 'Cable Connected' checkbox is checked, and the 'Port Forwarding' button is visible. At the bottom right are 'Cancel' and 'OK' buttons.

Adapter 1   Adapter 2   Adapter 3   Adapter 4

☒ Enable Network Adapter

Attached to: NAT

Name:

Advanced

Adapter Type: Intel PRO/1000 MT Desktop (82540EM)

Promiscuous Mode: Deny

MAC Address: 080027B8D40D

☒ Cable Connected

Port Forwarding

Cancel   OK

## 主机网络

The screenshot shows the 'Network Adapter' configuration window for 'Adapter 2'. The 'Enable Network Adapter' checkbox is checked. The 'Attached to' dropdown is set to 'Host-only Adapter'. The 'Name' field is 'vboxnet0'. Under the 'Advanced' section, the 'Adapter Type' is 'Intel PRO/1000 MT Desktop (82540EM)', 'Promiscuous Mode' is 'Deny', and the 'MAC Address' is '08002767B303'. The 'Cable Connected' checkbox is checked, and the 'Port Forwarding' button is visible. At the bottom right are 'Cancel' and 'OK' buttons.

General   System   Display   Storage   Audio   Network   Ports   Shared Folders   User Interface

Adapter 1   Adapter 2   Adapter 3   Adapter 4

☒ Enable Network Adapter

Attached to: Host-only Adapter

Name: vboxnet0

Advanced

Adapter Type: Intel PRO/1000 MT Desktop (82540EM)

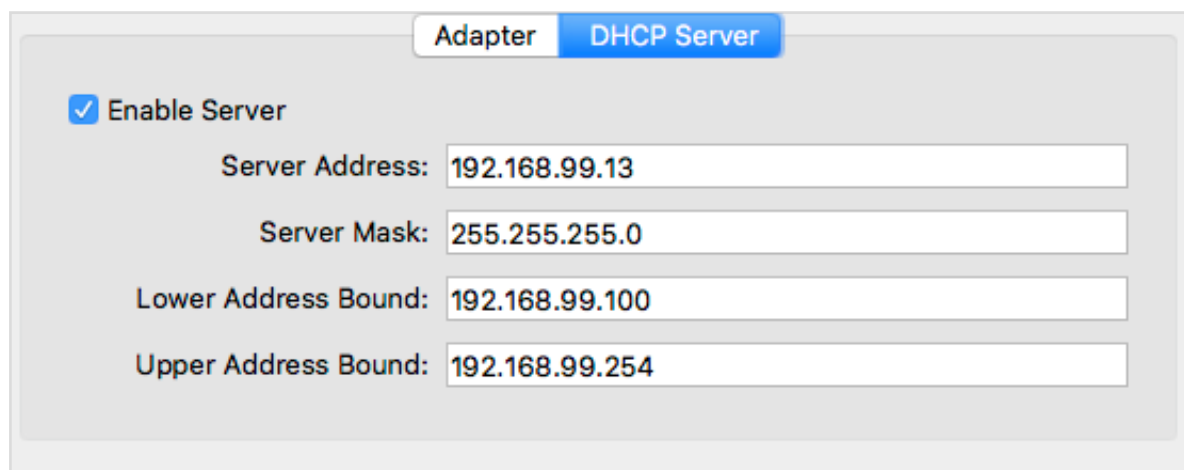
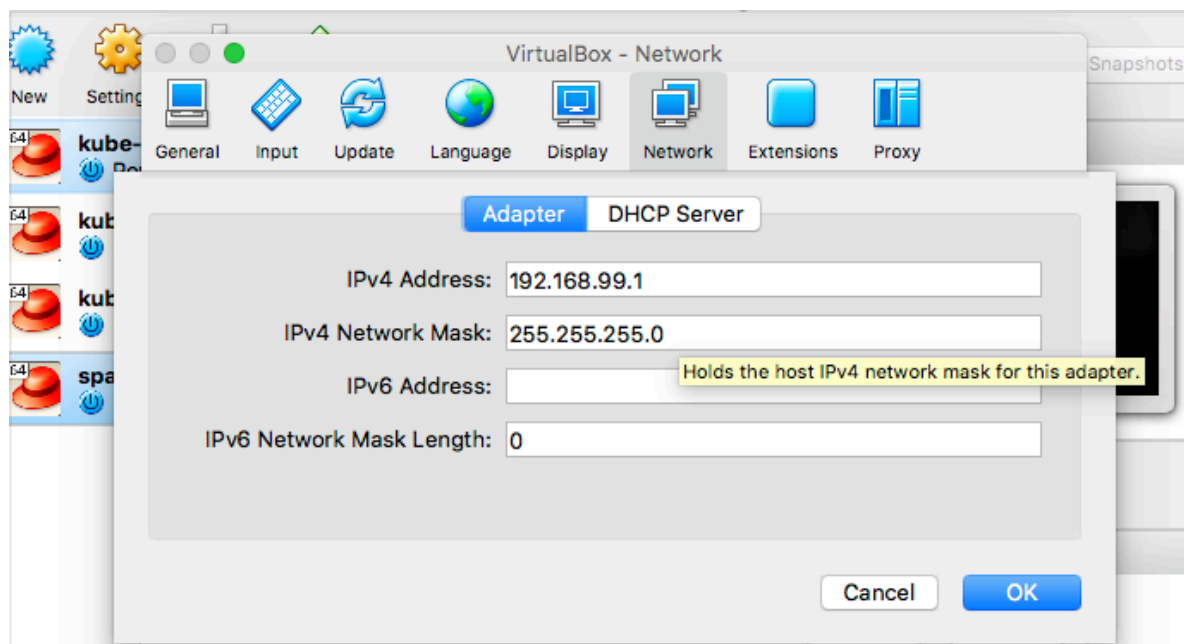
Promiscuous Mode: Deny

MAC Address: 08002767B303

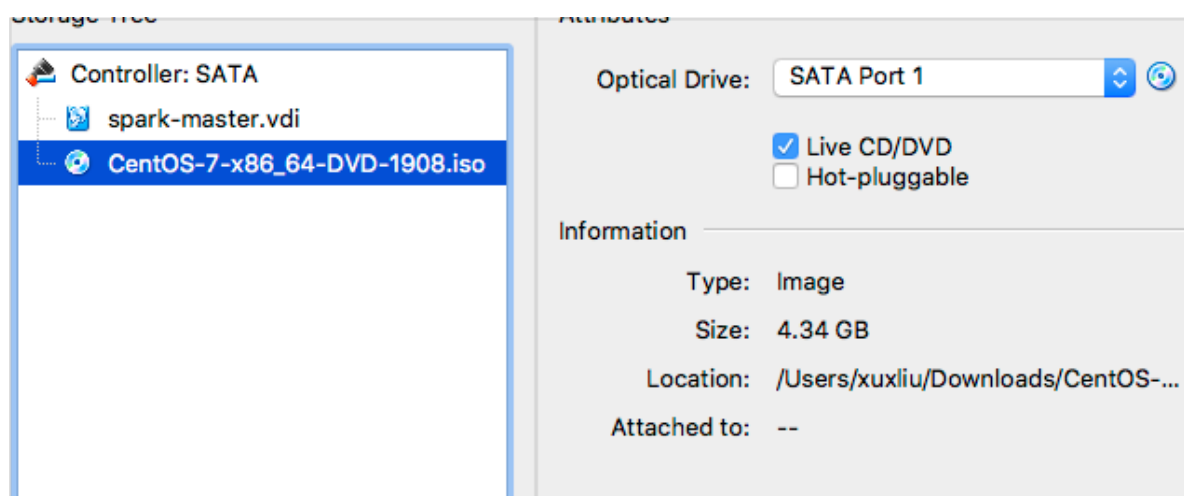
☒ Cable Connected

Port Forwarding

Cancel   OK

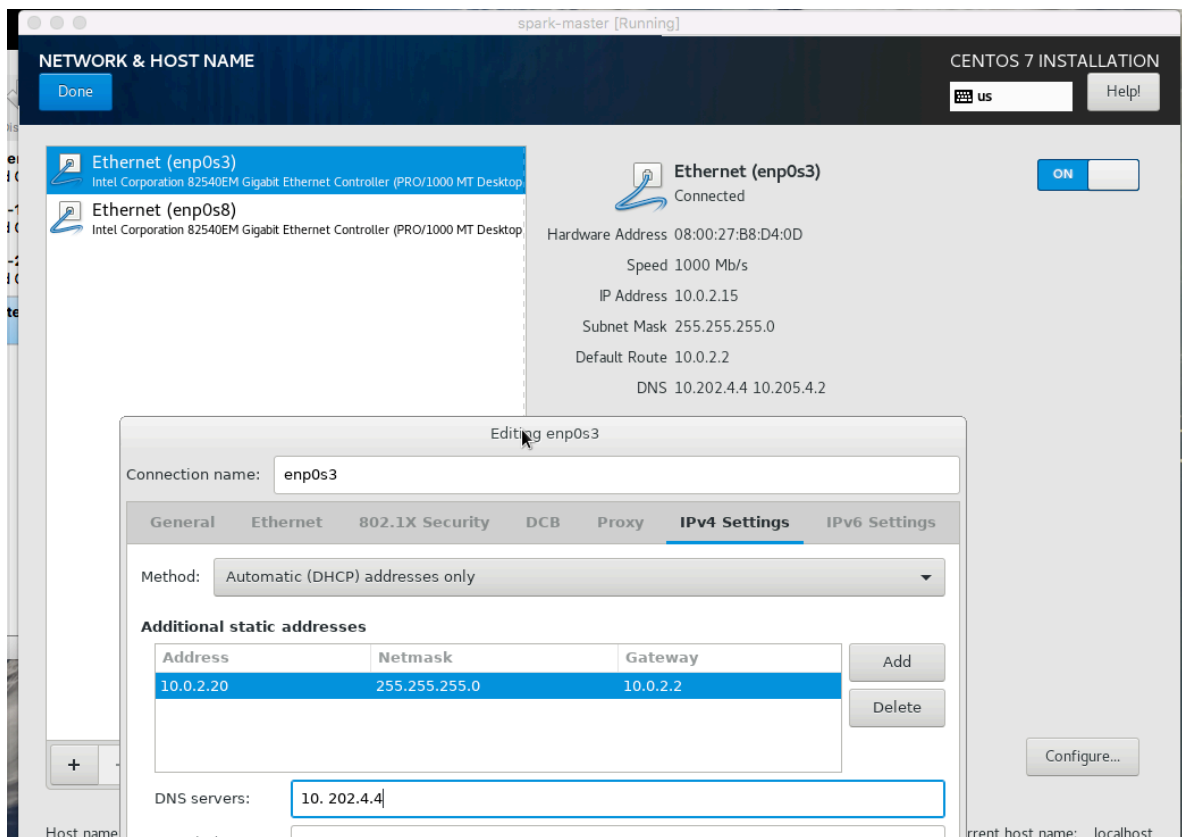


铁架 iso 光驱，去掉 IDE

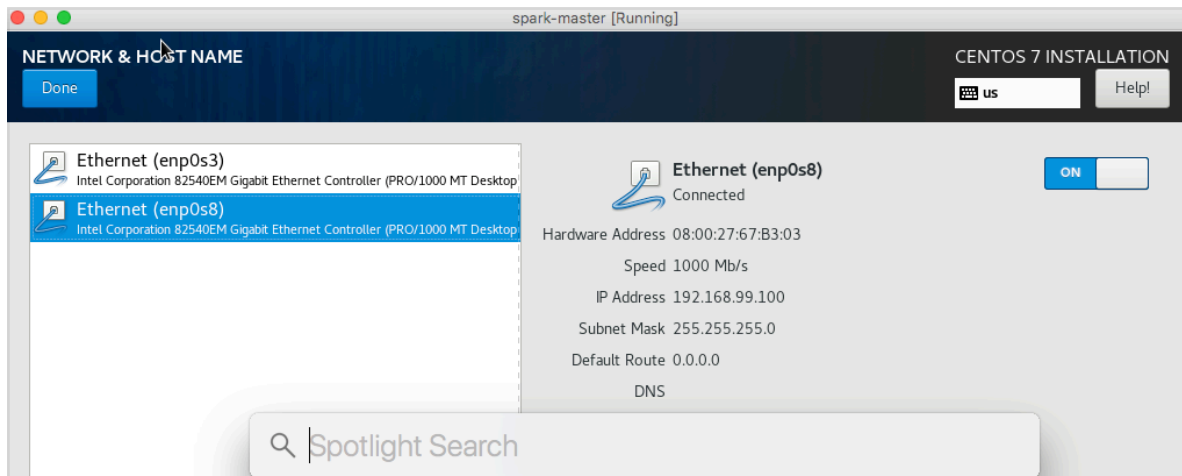


确定后，启动系统，开始安装

安装过程中配置网卡 NAT，其他步骤省略，基本就是下一步下一步 NAT 模式进入 configure.. 配置静态网络。



HOST-ONLY，这里没有配置静态IP，在安装完成之后，安装过几次有时候有IP，有时候没有的，如果没有的，那么安装完整之后配置网卡配置文件。



然后设置用户名密码即可  
如果配置之后，发现主机模式下面没有IP地址那么进行如下配置

```

1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:6c:41:1e brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.20/24 brd 10.0.2.255 scope global noprefixroute enp0s3
        valid_lft forever preferred_lft forever
    inet6 fe80::32a:5cc3:5632:4c30/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:6c:41:1e brd ff:ff:ff:ff:ff:ff
    inet 192.168.56.20/24 brd 192.168.56.255 scope global noprefixroute enp0s8
        valid_lft forever preferred_lft forever
    inet6 fe80::181c:792f:4b13:d64/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
root@k8s-master ~# vim /etc/sysconfig/network-scripts/if
ifcfg-enp0s3    ifdown-eth    ifdown-ppp    ifdown-tunnel    ifup-ippv6    ifup-post    ifup-TeamPort
ifcfg-enp0s8    ifdown-ippv6    ifdown-routes    ifup    ifup-ipv6    ifup-ppp    ifup-tunnel
ifcfg-lo        ifdown-ipv6    ifdown-sit    ifup-aliases    ifup-isdn    ifup-routes    ifup-wireless
ifdown        ifdown-isdn    ifdown-Team    ifup-bnep    ifup-plip    ifup-sit
ifdown-bnep    ifdown-post    ifdown-TeamPort    ifup-eth    ifup-plusb    ifup-Team
root@k8s-master ~# vim /etc/sysconfig/network-scripts/if
ifcfg-enp0s3    ifdown-eth    ifdown-ppp    ifdown-tunnel    ifup-ippv6    ifup-post    ifup-TeamPort
ifcfg-enp0s8    ifdown-ippv6    ifdown-routes    ifup    ifup-ipv6    ifup-ppp    ifup-tunnel
ifcfg-lo        ifdown-ipv6    ifdown-sit    ifup-aliases    ifup-isdn    ifup-routes    ifup-wireless
ifdown        ifdown-isdn    ifdown-Team    ifup-bnep    ifup-plip    ifup-sit
ifdown-bnep    ifdown-post    ifdown-TeamPort    ifup-eth    ifup-plusb    ifup-Team
root@k8s-master ~# vim /etc/sysconfig/network-scripts/ifcfg-enp0s_

```

确认 onboot 是 yes

```

1 TYPE="Ethernet"
2 PROXY_METHOD="none"
3 BROWSER_ONLY="no"
4 BOOTPROTO="none"
5 DEFROUTE="yes"
6 IPV4_FAILURE_FATAL="no"
7 IPV6INIT="yes"
8 IPV6_AUTOCONF="yes"
9 IPV6_DEFROUTE="yes"
10 IPV6_FAILURE_FATAL="no"
11 IPV6_ADDR_GEN_MODE="stable-privacy"
12 NAME="enp0s3"
13 UUID="10f170f2-6401-4fdf-9427-2b9700593bf2"
14 DEVICE="enp0s3"
15 ONBOOT="yes"
16 IPADDR="10.0.2.20"
17 PREFIX="24"
18 GATEWAY="10.0.2.2"
19 DNS1="10.0.2.3"
20 DNS2="114.114.114.114"
21 IPV6_PRIVACY="no"

```

然后重启网络,  
我的配置



```
spark-master [running]
TYPE="Ethernet"
PROXY_METHOD="none"
BROWSER_ONLY="no"
BOOTPROTO="none"
DEFROUTE="yes"
IPV4_FAILURE_FATAL="no"
IPV6INIT="yes"
IPV6_AUTOCONF="yes"
IPV6_DEFROUTE="yes"
IPV6_FAILURE_FATAL="no"
IPV6_ADDR_GEN_MODE="stable-privacy"
NAME="enp0s8"
UUID="26c3f2e2-2604-4e0d-b5fc-b57852495e2d"
DEVICE="enp0s8"
ONBOOT="yes"
IPADDR="192.168.99.100"
PREFIX="24"
GATEWAY="0.0.0.0"
IPV6_PRIVACY="no"
```

网络重启后 service network restart 然后 ip addr 就能看到 网卡 8 上的 ip 了

## 更改国内镜像源

```
mv /etc/yum.repos.d/CentOS-Base.repo /etc/yum.repos.d/CentOS-Base.repo.backup
wget -O /etc/yum.repos.d/CentOS-Base.repo http://mirrors.aliyun.com/repo/Centos-7.repo
然后跟下 yum 安装包
yum -y update
大于出来使用 aliyun 的镜像源信息
```

下载 spark 2.4.5，此次安装时要使用 Delta lake，它需要 spark 在 2.4.2 以上的版本

<https://mirrors.tuna.tsinghua.edu.cn/apache/spark/spark-2.4.5/spark-2.4.5-bin-hadoop2.7.tgz>

下载 hadoop，这里使用国内镜像下载

<https://mirrors.tuna.tsinghua.edu.cn/apache/hadoop/common/hadoop-2.7.7/>

在 spark 安装的过程中不需要安装 scala，原因是 spark 依赖中以及有 scala 了。

## 关闭防火墙

```
systemctl stop firewalld  
systemctl disable firewalld
```

我是在本地下载好 spark, \_hadoop\_jdk 等然后 Scp 到虚拟机  
scp spark-2.4.5-bin-hadoop2.7.tgz root@192.168.99.100:/

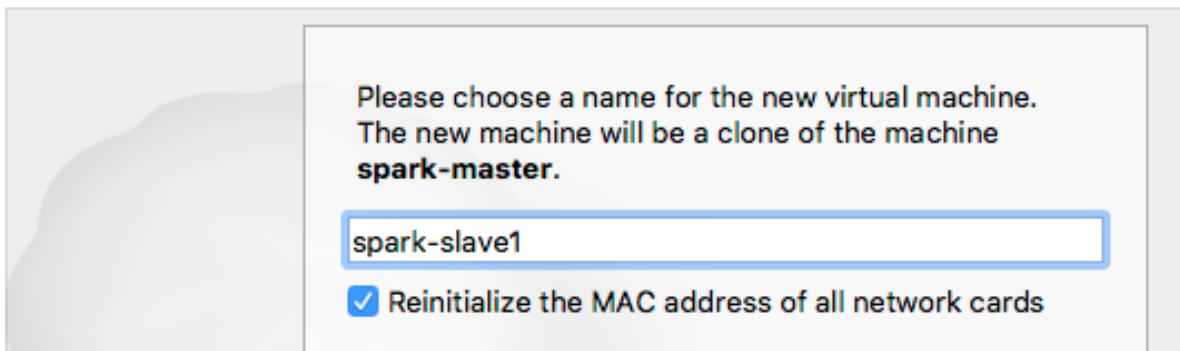
## 安装 Java, 配置环境变量

```
export JAVA_HOME=/opt/bigData/othersoftware/jdk1.8.0_221  
export JRE_HOME=$JAVA_HOME/jre  
export CLASSPATH=.:${JAVA_HOME}/lib:${JRE_HOME}/lib  
export PATH=${JAVA_HOME}/bin:$PATH  
source ~/.bachrc
```

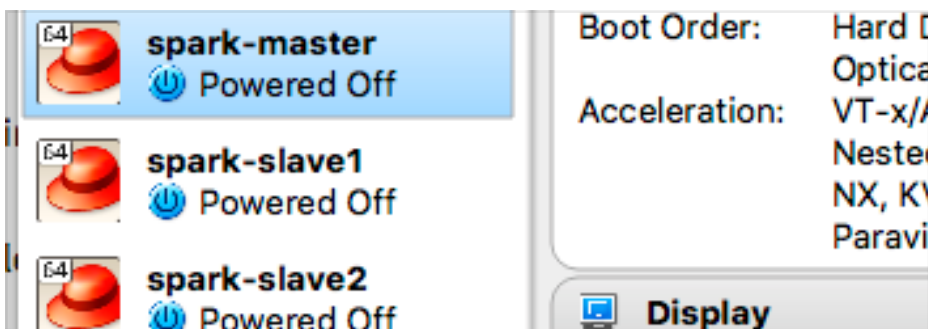
## 安装 slave 节点, \_

然后在 master 安装 hadoop 以及 spark, 之间 scp 到 slave 节点中, 就省去很多一一的配置

克隆 虚拟机, slave1, 注意在克隆的时候 MAC 地址需要重新初始化



这样最终复制出来 2 个虚拟机 slave1, slave2,



克隆过来的虚拟机, 系统修改 **hostname**

```
[root@spark-master ~]# cat /etc/hostname  
spark-slave1
```

**Master** 使用的是静态 IP，所以 **master** 和 **slave** 之间的 IP 也是冲突的，需要修改  
这是 master 的网卡

```
root@spark-master ~/# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:b8:d4:0d brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.20/24 brd 10.0.2.255 scope global noprefixroute enp0s3
        valid_lft forever preferred_lft forever
    inet6 fe80::abee:8e05:6203:bc49/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:67:b3:03 brd ff:ff:ff:ff:ff:ff
    inet 192.168.99.100/24 brd 192.168.99.255 scope global noprefixroute enp0s8
        valid_lft forever preferred_lft forever
    inet6 fe80::c15e:47f1:e6a4:53ba/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
```

查看下 slave 的网卡和这里的 ip 都是一样的

进入目录将 2 张网卡的配置文件全部都重新配置，这里贴出来修改网卡 enp0s8 的，将 192.168.99.100 改为 92.168.99.101，同样，在网卡 enp0s3 中将，10.0.2.20 改为 10.0.2.21

```
BOOTPROTO="none"
DEFROUTE="yes"
IPV4_FAILURE_FATAL="no"
IPV6INIT="yes"
IPV6_AUTOCONF="yes"
IPV6_DEFROUTE="yes"
IPV6_FAILURE_FATAL="no"
IPV6_ADDR_GEN_MODE="stable-privacy"
NAME="enp0s8"
UUID="26c3f2e2-2604-4e0d-b5fc-b57852495e2d"
DEVICE="enp0s8"
ONBOOT="yes"
IPADDR="192.168.99.101"
PREFIX="24"
GATEWAY="0.0.0.0"
IPV6_PRIVACY="no"

~
~
~

"ifcfg-enp0s8" 20L, 386C written
root@spark-master network-scripts# pwd
/etc/sysconfig/network-scripts
root@spark-master network-scripts#
```

service network restart 重启网络，然后在查看 ip addr，修改已全部生效

```
spark-slave1 [Running]
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:04:73:fd brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.21/24 brd 10.0.2.255 scope global noprefixroute enp0s3
        valid_lft forever preferred_lft forever
    inet6 fe80::abee:8e05:6203:bc49/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:af:45:ed brd ff:ff:ff:ff:ff:ff
    inet 192.168.99.101/24 brd 192.168.99.255 scope global noprefixroute enp0s8
        valid_lft forever preferred_lft forever
    inet6 fe80::c15e:47f1:e6a4:53ba/64 scope link tentative noprefixroute dadfailed
        valid_lft forever preferred_lft forever
    inet6 fe80::c3c1:b083:cdc3:2487/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
[root@spark-master network-scripts]#
```

在2台 slave 机器上执行相同的操作。

## 修改文件

```
[root@spark-master ~]# cat /etc/hosts
127.0.0.1    localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6

192.168.99.100 spark-master
192.168.99.101 spark-slave1
192.168.99.102 spark-slave2
```

重新启动 1、reboot; 2、shutdown -r now 上文明配置的 hosts 以及 hostname 生效

## 设置免密登录,

spark master 需要给 slave 节点分配任务, 需要确保 master 可以免密登录到其他机器上

ssh-keygen -t rsa 一路回车

cat ~/.ssh/id\_rsa.pub >> ~/.ssh/authorized\_keys 密匙是默认放在 ~/.ssh/id\_rsa.pub, 目录中,

然后将master 的 id\_rsa.pub scp 到 slave 目录中。并写入到

cat id\_rsa.pub >> ~/.ssh/authorized\_keys

然后 master 就可以免密登录到 slave1 和 slave2, 中, 如果需要 slave 登录到 master, 同样将 slave 的该文件写到 master 的 authorized\_keys 文件中。

```
[root@spark-master .ssh]# scp id_rsa.pub root@spark-slave2:/
The authenticity of host 'spark-slave2 (192.168.99.102)' can't be established.
ECDSA key fingerprint is SHA256:JW+Vbp1PDP7mtfWY6BSpuEx0VjCGoCDx4jPtxYVdTWo.
ECDSA key fingerprint is MD5:58:f6:f2:71:92:6f:0c:96:59:b4:31:ae:89:92:60:4a.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'spark-slave2,192.168.99.102' (ECDSA) to the list of known hosts.
root@spark-slave2's password:
id_rsa.pub
service network restart 重启网络，然后在查看 ip addr，修改已全部生效
[root@spark-master .ssh]# ssh spark-slave1
Last login: Wed Apr 15 10:40:43 2020 from 192.168.99.1
[root@spark-slave1 ~]#
```

执行 exit，退出登录。

## 安装及配置 hadoop

tar -zxvf hadoop-2.7.7.tar.gz

export HADOOP\_HOME=/opt/bigData/hadoop/hadoop-2.7.7

export PATH=\$PATH:\${HADOOP\_HOME}/bin:\${HADOOP\_HOME}/sbin

进入 /opt/bigData/hadoop/hadoop-2.7.7/etc/hadoop

配置 core-site

<configuration>

<property>

<name>fs.defaultFS</name>

<value>hdfs://spark-master:9000</value>

</property>

<property>

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

<value>/opt/bigData/hadoop/tmpDir</value>

</property>

</configuration>

配置 hdfs-site

<configuration>

<property>

<name>dfs.replication</name>

<value>2</value>

</property>

</configuration>

配置 mr 以及 secondaryNameNode

<configuration>

<property>

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

<value>yarn</value> // 在 yarn 上跑 mapreduce 计算

```
</property>
<property>
  <name>dfs.namenode.secondary.http-address</name>
  <value>spark-slave1:50090</value>
</property>
</configuration>
```

配置 yarn

```
<property>
  <name>yarn.resourcemanager.hostname</name>
  <value>spark-master</value>
</property>
<property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce_shuffle</value>
</property>
</configuration>
```

配置 **slaves**(感觉这里应该不写 **spark-master** 的)

```
[root@spark-master hadoop]# cat slaves
spark-master
spark-slave1
spark-slave2
```

注意一定要配置 JAVA\_HOME 环境变量，因为 `hadoop-env.sh` 中会去读环境变量

```
# The java implementation to use.
export JAVA_HOME=${JAVA_HOME}
```

至此配置完 `hadoop` 将 `hadoop` 包 `scp` 到其他的 `slave` 机器。  
`scp -r hadoop-2.7.7 root@spark-slave2:/opt/bigData/hadoop`

然后就可以启动 **hadoop** 了。

首次启动需要执行格式化 `nameNode`  
`hdfs namenode -format`  
启动  
`start-dfs.sh`

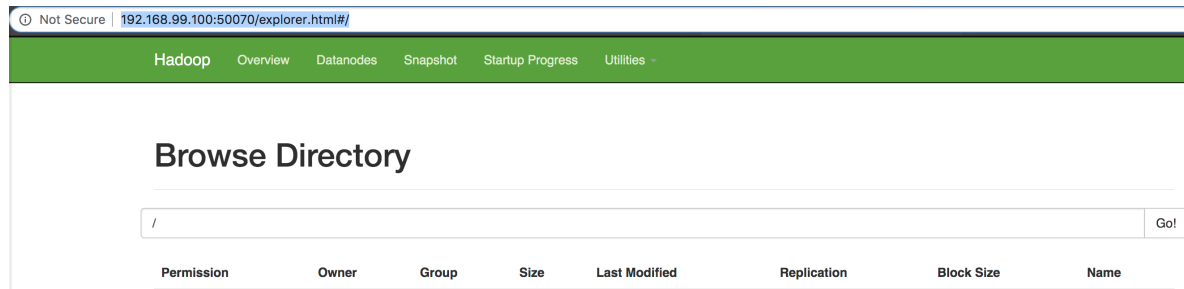
```

root@spark-master hadoop]# start-dfs.sh
Incorrect configuration: namenode address dfs.namenode.servicerpc-address or dfs.namenode.rpc-address is not configured.
Starting namenodes on [ ]
spark-slave2: starting namenode, logging to /opt/bigData/hadoop/hadoop-2.7.7/logs/hadoop-root-namenode-spark-slave2.out
spark-master: starting namenode, logging to /opt/bigData/hadoop/hadoop-2.7.7/logs/hadoop-root-namenode-spark-master.out
spark-slave1: starting namenode, logging to /opt/bigData/hadoop/hadoop-2.7.7/logs/hadoop-root-namenode-spark-slave1.out
spark-slave1: starting datanode, logging to /opt/bigData/hadoop/hadoop-2.7.7/logs/hadoop-root-datanode-spark-slave1.out
spark-slave2: starting datanode, logging to /opt/bigData/hadoop/hadoop-2.7.7/logs/hadoop-root-datanode-spark-slave2.out
spark-master: starting datanode, logging to /opt/bigData/hadoop/hadoop-2.7.7/logs/hadoop-root-datanode-spark-master.out
Starting secondary namenodes [spark-slave1]
spark-slave1: starting secondarynamenode, logging to /opt/bigData/hadoop/hadoop-2.7.7/logs/hadoop-root-secondarynamenode-spark-slave1.out

```

可以查看 hdfs 文件系统了

<http://192.168.99.100:50070/explorer.html#/>



## 安装配置 spark

减压，添加环境变量

```

export SPARK_HOME=/opt/bigData/spark/spark-2.4.5-bin-hadoop2.7
export PATH=$PATH:${HADOOP_HOME}/bin:${HADOOP_HOME}/sbin:${SPARK_HOME}/bin:${SPARK_HOME}/sbin

```

配置：

slaves

spark-slave1

spark-slave2

spark-env.sh

```

export JAVA_HOME=/opt/bigData/othersoftware/jdk1.8.0_221
export SPARK_MASTER_IP=192.168.99.100
export SPARK_WORKER_CORES=2
export SPARK_EXECUTOR_MEMORY=1g
export SPARK_WORKER_MEMORY=1g
export HADOOP_CONF_DIR=/opt/bigData/hadoop/hadoop-2.7.7/etc/hadoop

```

启动 spark start-all.sh

Spark-UI

<http://192.168.31.100:8080/>



```
[root@spark-master sbin]# jps
4196 NameNode
4331 DataNode
4795 Jps
4716 Master
```

```
[root@spark-slave1 spark]# jps
4227 SecondaryNameNode
4123 DataNode
4507 Worker
4559 Jps
```

启动 yarn

[./start-yarn.sh](#)

—

spark-shell(不加 --master **spark://spark-master:7077** 是 local 启动的)

spark-shell --master **spark://spark-master:7077**

```
spark.range(0,
50).select($"id".as("id")).repartition(1).write.mode(SaveMode.Overwrite)
.format("delta").save(path + table)
```