

搭建 Hadoop+Spark 分布式集群

刘磊

2020 年 12 月

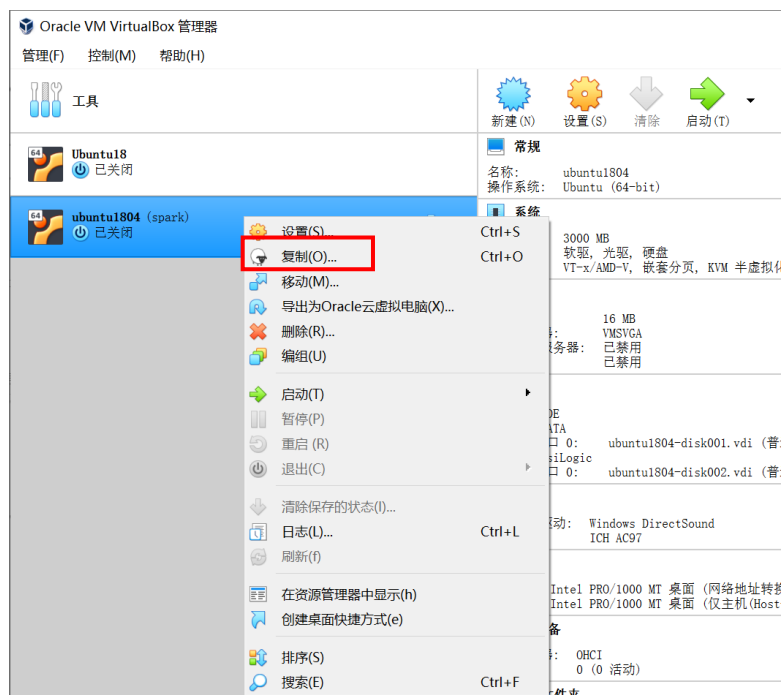
目录

1. 复制虚拟机.....	1
2. 设置网络	3
3. 修改 Hadoop 和 Spark 配置文件.....	13
4. 总结	21

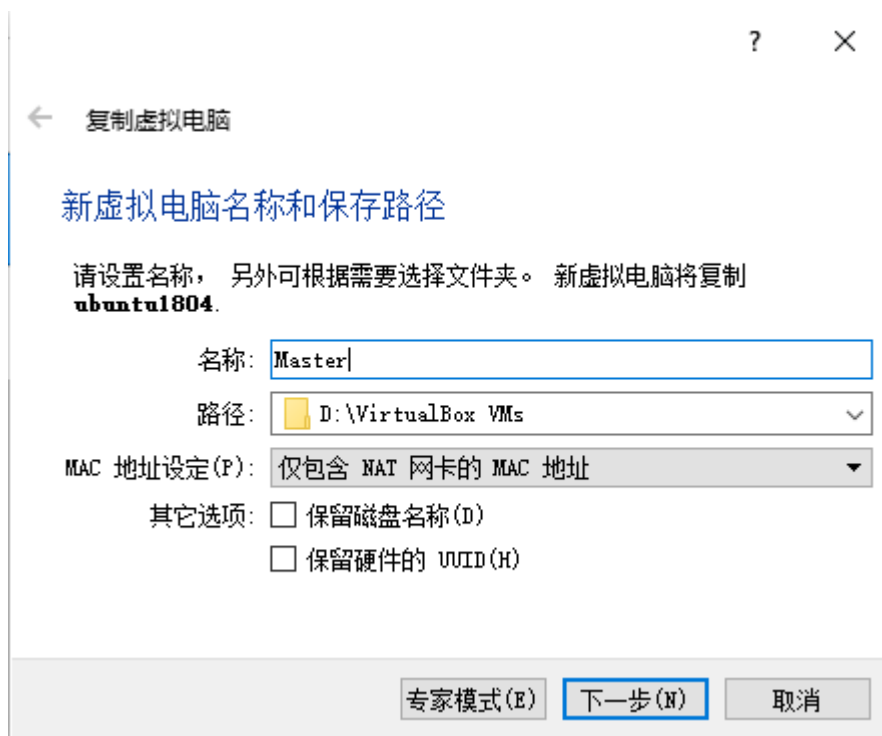
我们以安装 VirtualBox 的主机为 Windows10 系统为例，在 VirtualBox 中搭建一个包含三个节点的集群。

1. 复制虚拟机

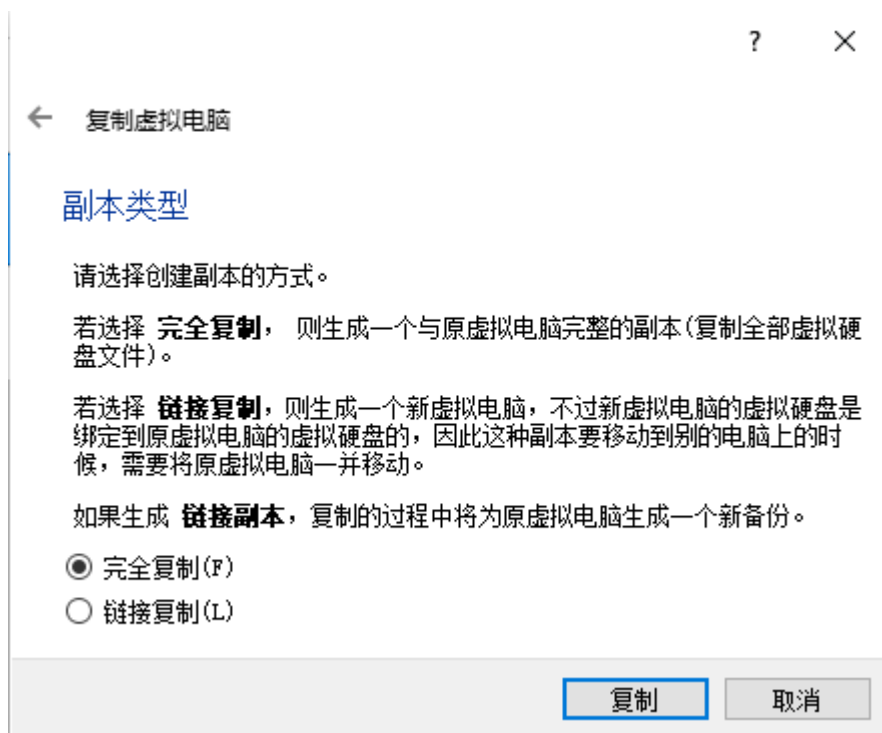
为了保留之前的伪分布式系统，首先复制一份安装伪分布式系统的虚拟机。打开 VirtualBox，在左边的虚拟机列表中右键要复制的虚拟机，选择【复制】。



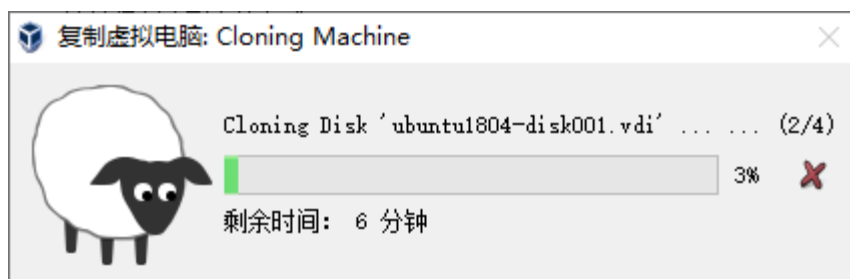
在弹出的复制虚拟机窗口中将名称改为 Master，路径为虚拟机文件保存位置，可以改为剩余空间较大的盘，建议路径名不要出现中文。点击【下一步】。



然后点击【复制】



等待复制完成，就可以在列表中看到复制得到的虚拟机。



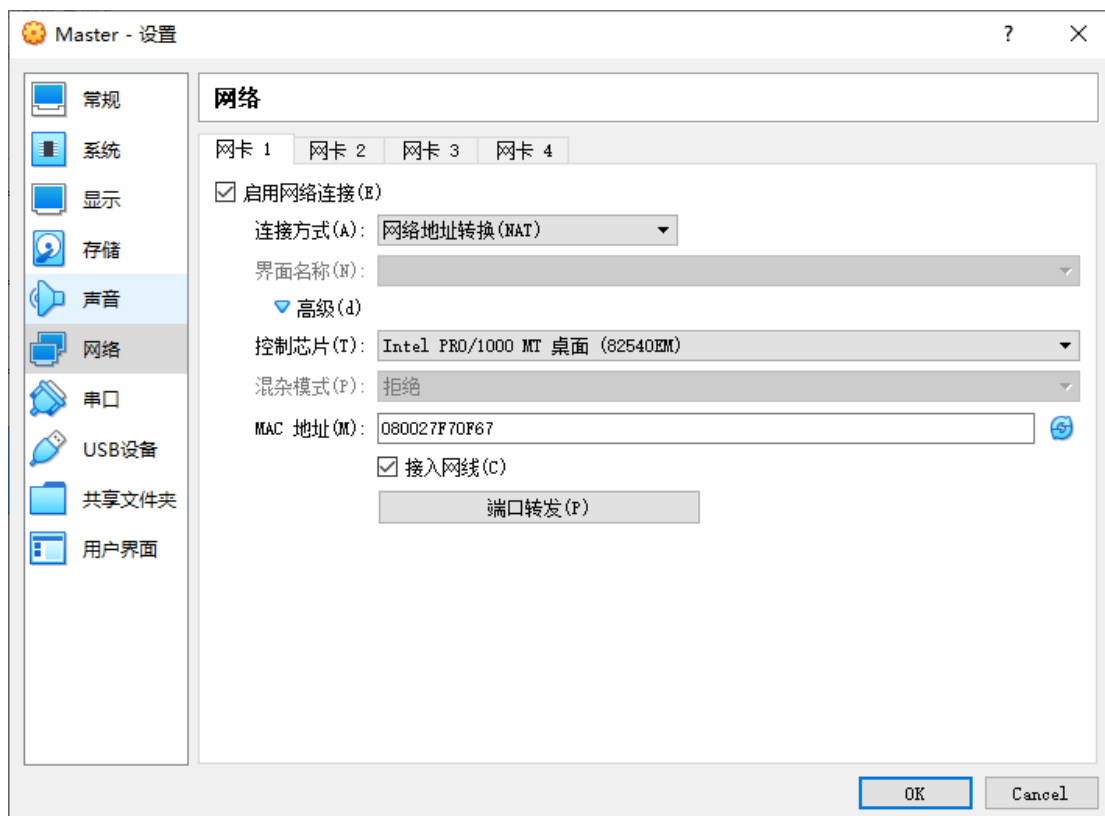
2. 设置网络

2.1 添加网卡

虚拟机启动前，在列表中选择虚拟机 Master，点击【设置】按钮，

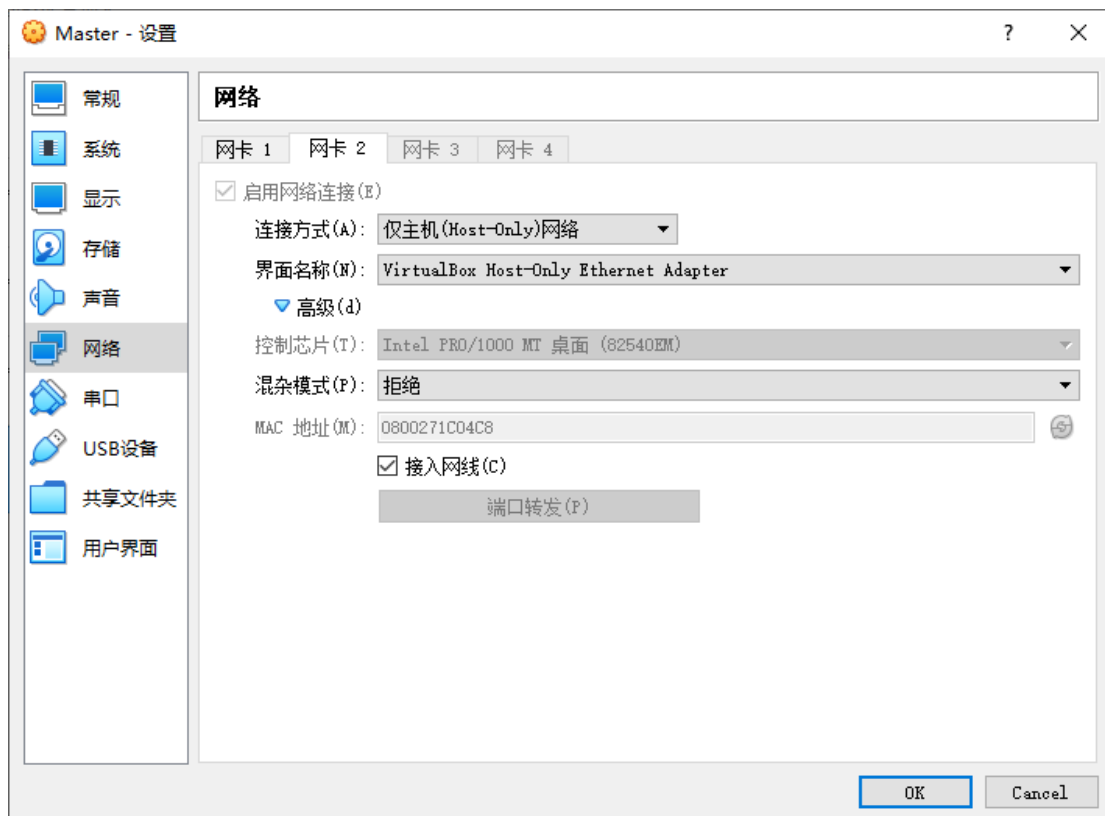


在弹出的设置窗口中选择【网络】。默认网卡 1 为访问外网的网卡，这个网卡可以用默认值，点击【高级】左边的蓝色三角形可以查看 MAC 地址等高级设置。



点击【网卡 2】标签，然后勾选【启动网络连接】，添加一个网卡使虚拟机可以与主机相互访问，连接方式选择【仅主机 (Host-Only) 网络】，其余选项不变，设置好以后点击

右下角【OK】，不点【OK】设置不会生效。



打开虚拟机，在终端中执行以下命令查看网卡信息

```
ifconfig
```

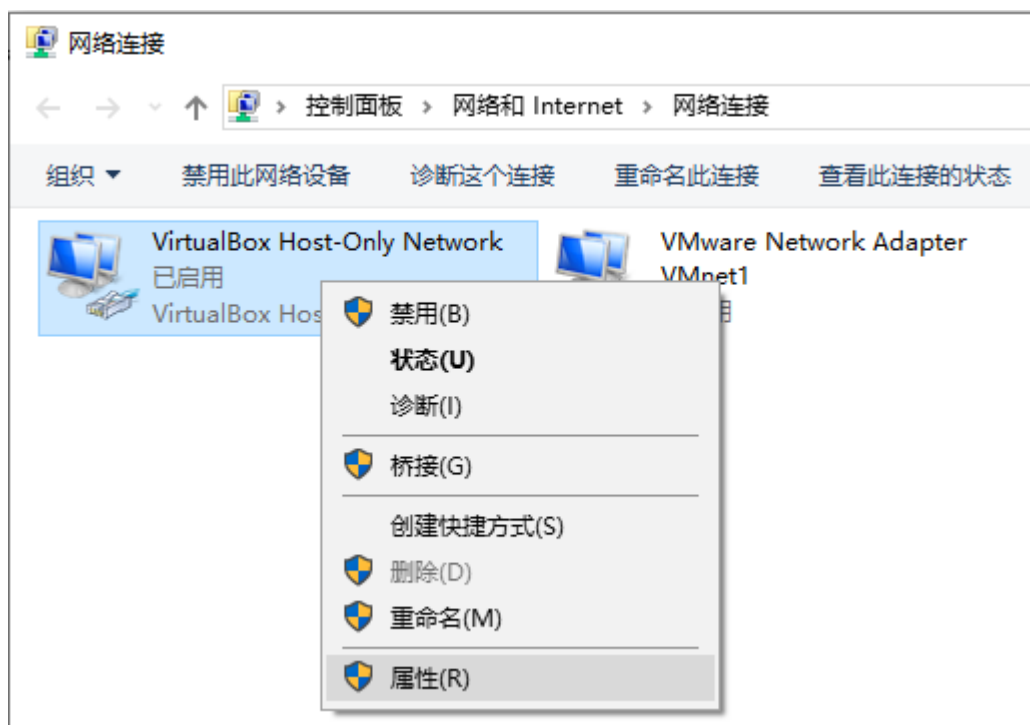
可以通过对比 MAC 地址知道 enp0s3 和 enp0s8 分别为网卡 1 和网卡 2。

```
lei@ubuntu:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.2.15 netmask 255.255.255.0 broadcast 10.0.2.255
    inet6 fe80::41c1:6985:4a8c:3cd0 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:f7:0f:67 txqueuelen 1000 (Ethernet)
    RX packets 19 bytes 2816 (2.8 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 69 bytes 7152 (7.1 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

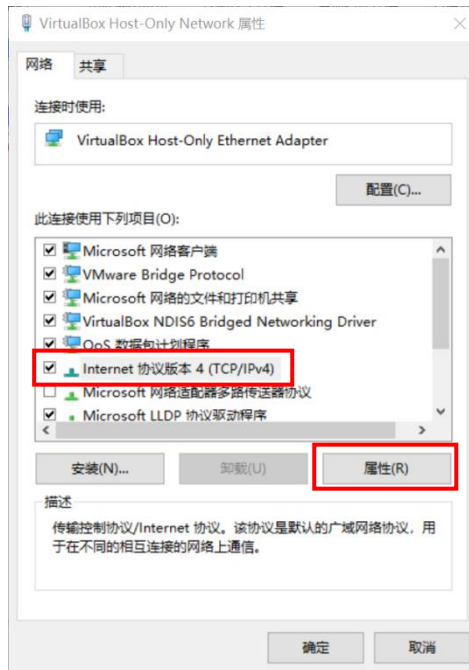
enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.56.102 netmask 255.255.255.0 broadcast 192.168.56.255
    inet6 fe80::1315:667d:99e4:26 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:1c:04:c8 txqueuelen 1000 (Ethernet)
    RX packets 9 bytes 1619 (1.6 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 50 bytes 5992 (5.9 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

2.2 设置虚拟机静态 IP

进入主机控制面板=>网络和 Internet=>网络连接中，在【VirtualBox Host-Only Network】图标上，点击右键选择【属性】



在弹出的属性窗口点击【Internet 协议版本 4(TCP/IPv4)】，然后点击【属性】



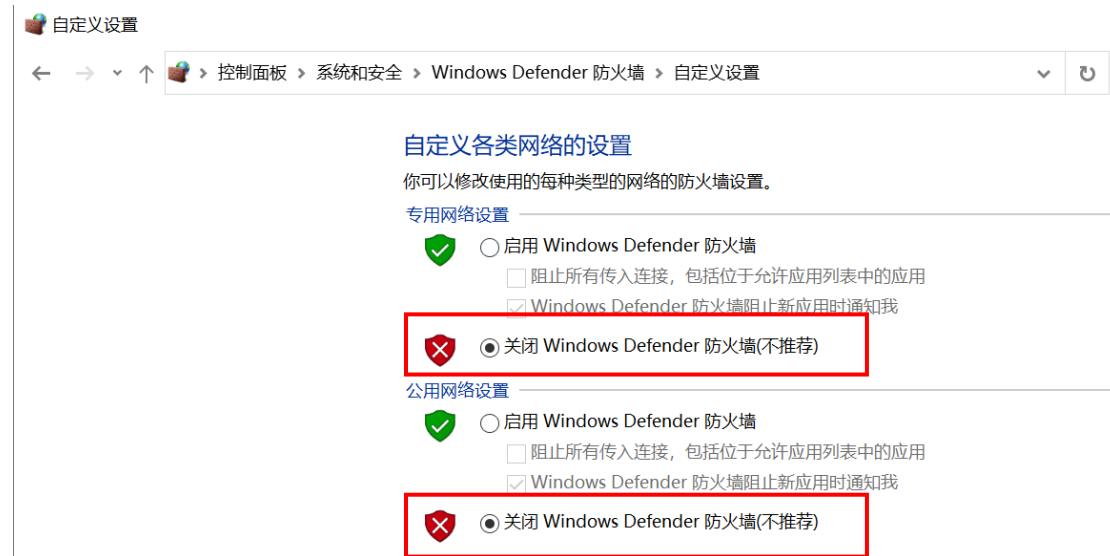
在弹出的窗口可以看到主机的 IP 地址为 192.168.56.1。



在虚拟机中使用 ifconfig 命令，看到 enp0s8 的 IP 地址为 192.168.56.101，与主机 IP 处于同一个网段，因此可以相互访问。

```
enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
inet 192.168.56.101 netmask 255.255.255.0 broadcast 192.168.56.255
inet6 fe80::e118:ccd0:3974:c527 prefixlen 64 scopeid 0x20<link>
ether 08:00:27:4b:40:a0 txqueuelen 1000 (Ethernet)
RX packets 11 bytes 2269 (2.2 KB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 58 bytes 6976 (6.9 KB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

关闭主机的防火墙,



在虚拟机中执行

```
ping 192.168.56.1
```

```
lei@ubuntu:~$ ping 192.168.56.1
PING 192.168.56.1 (192.168.56.1) 56(84) bytes of data:
64 bytes from 192.168.56.1: icmp_seq=1 ttl=128 time=0.212 ms
64 bytes from 192.168.56.1: icmp_seq=2 ttl=128 time=0.664 ms
64 bytes from 192.168.56.1: icmp_seq=3 ttl=128 time=0.637 ms
64 bytes from 192.168.56.1: icmp_seq=4 ttl=128 time=0.629 ms
64 bytes from 192.168.56.1: icmp_seq=5 ttl=128 time=0.651 ms
```

在主机打开命令提示符执行

```
ping 192.168.56.101
```

```
C:\Users\lei>ping 192.168.56.101

正在 Ping 192.168.56.101 具有 32 字节的数据:
来自 192.168.56.101 的回复: 字节=32 时间<1ms TTL=64
来自 192.168.56.101 的回复: 字节=32 时间<1ms TTL=64
来自 192.168.56.101 的回复: 字节=32 时间<1ms TTL=64
来自 192.168.56.101 的回复: 字节=32 时间<1ms TTL=64

192.168.56.101 的 Ping 统计信息:
    数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),
往返行程的估计时间(以毫秒为单位):
    最短 = 0ms, 最长 = 0ms, 平均 = 0ms
```

可以看到从主机 ping 虚拟机和从虚拟机 ping 主机都是成功的, 说明网络是相互连通的。

为了后面建立集群方便, 这里我们把虚拟机的 ip 地址修改一下, 打开网络接口设置文件


```
/etc/network/interfaces
```

```
sudo vim /etc/network/interfaces
```

添加以下内容，将网卡 2 的 IP 地址改为 192.168.56.111。

```
auto enp0s8
iface enp0s8 inet static
address 192.168.56.111
netmask 255.255.255.0

auto enp0s3
iface enp0s3 inet dhcp
```

```
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback

auto enp0s8
iface enp0s8 inet static
address 192.168.56.111
netmask 255.255.255.0

auto enp0s3
iface enp0s3 inet dhcp
```

重启一下虚拟机，再次执行 ifconfig 可以看到 enp0s8 的 IP 地址已经改为 192.168.56.111。

```
enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.56.111 netmask 255.255.255.0 broadcast 192.168.56.255
    inet6 fe80::a00:27ff:fe4b:40a0 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:4b:40:a0 txqueuelen 1000 (Ethernet)
    RX packets 6 bytes 379 (379.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 59 bytes 6590 (6.5 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

再次在主机 ping 虚拟机，可以看到依然可以 ping 通

```
ping 192.168.56.111
```

```
C:\Users\lei>ping 192.168.56.111

正在 Ping 192.168.56.111 具有 32 字节的数据:
来自 192.168.56.111 的回复: 字节=32 时间<1ms TTL=64
来自 192.168.56.111 的回复: 字节=32 时间<1ms TTL=64
来自 192.168.56.111 的回复: 字节=32 时间<1ms TTL=64
来自 192.168.56.111 的回复: 字节=32 时间<1ms TTL=64

192.168.56.111 的 Ping 统计信息:
    数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),
往返行程的估计时间(以毫秒为单位):
    最短 = 0ms, 最长 = 0ms, 平均 = 0ms
```

在虚拟机中，ping 百度也是可以 ping 通的，说明虚拟机可以访问外网。

```
ping www.baidu.com
```

```
lei@ubuntu:~$ ping www.baidu.com
PING www.a.shifen.com (180.101.49.12) 56(84) bytes of data.
64 bytes from 180.101.49.12 (180.101.49.12): icmp_seq=1 ttl=51 time=39.0 ms
64 bytes from 180.101.49.12 (180.101.49.12): icmp_seq=2 ttl=51 time=31.0 ms
64 bytes from 180.101.49.12 (180.101.49.12): icmp_seq=3 ttl=51 time=33.4 ms
64 bytes from 180.101.49.12 (180.101.49.12): icmp_seq=4 ttl=51 time=30.7 ms
64 bytes from 180.101.49.12 (180.101.49.12): icmp_seq=5 ttl=51 time=42.9 ms
```

2.3 修改主机名

将主机名修改为 master，重启使修改生效，可以在命令提示看到主机名已经修改为

master。

```
sudo vim /etc/hostname
```

配置节点 IP 对应的主机名

```
sudo vim /etc/hosts
```

添加以下内容，设置以后就可以通过主机名访问相应的节点。这里先把 work1 和 work2

的 IP 地址提前设置好，这两个节点后面会进行创建。

```
192.168.56.111 master
192.168.56.112 worker1
192.168.56.113 worker2
```

删掉含有 127.0.1.1 的那一行。修改后如下图所示

```
127.0.0.1    localhost

# The following lines are desirable for IPv6 capable hosts
::1          ip6-localhost ip6-loopback
fe00::0      ip6-localnet
ff00::0      ip6-mcastprefix
ff02::1      ip6-allnodes
ff02::2      ip6-allrouters

192.168.56.111 master
192.168.56.112 worker1
192.168.56.113 worker2
```

重启网络服务生效

```
systemctl restart networking
```

测试

```
ssh master
```

第一次连接需要输入 yes 确认。因为我们前面设置过免密码登录，所以这里不需要输入登录密码。

至此，集群的主节点就创建好了。将 Master 关机，从 Master 复制两个虚拟机，复制时分别命名为 Worker1 和 Worker2。

2.4 设置从节点

修改 ip

打开从节点虚拟机，在 `/etc/network/interfaces` 中将 Worker1 IP 改为 192.168.56.112，Worker2 IP 改为 192.168.56.113。例如 Work1，修改后，如下图所示

```
auto enp0s8
iface enp0s8 inet static
address 192.168.56.112
netmask 255.255.255.0

auto enp0s3
iface enp0s3 inet dhcp
```

修改主机名

打开 `/etc/hostname` 把主机名分别改为 worker1 和 worker2。

设置好以后重启。同时打开三台虚拟机。

测试

1. 测试三台主机是否网络是否连通，比如 master 到 worker1 和 worker2，在 master 的终端中执行 ping 命令，因为我们在 `/etc/hosts` 中设置了主机名与 IP 的对应关系，所以这里可以直接 ping 主机名

```
ping worker1
ping worker2
```

```
lei@master:~$ ping worker1
PING worker1 (192.168.56.112) 56(84) bytes of data.
64 bytes from worker1 (192.168.56.112): icmp_seq=1 ttl=64 time=0.372 ms
64 bytes from worker1 (192.168.56.112): icmp_seq=2 ttl=64 time=0.365 ms
64 bytes from worker1 (192.168.56.112): icmp_seq=3 ttl=64 time=0.972 ms
64 bytes from worker1 (192.168.56.112): icmp_seq=4 ttl=64 time=0.434 ms
64 bytes from worker1 (192.168.56.112): icmp_seq=5 ttl=64 time=0.804 ms
64 bytes from worker1 (192.168.56.112): icmp_seq=6 ttl=64 time=0.343 ms
```

2. 测试三台主机之间是否能免密码登录，比如主机到 worker1

```
ssh worker1
```

```
lei@master:~$ ssh worker1
The authenticity of host 'worker1 (192.168.56.112)' can't be established.
ECDSA key fingerprint is SHA256:f/x+0lSf7k2Vff2KRqvg15xGxbEiyW87PLrNWjxU0nw.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'worker1,192.168.56.112' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 5.4.0-56-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

12 packages can be updated.
8 updates are security updates.

New release '20.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Your Hardware Enablement Stack (HWE) is supported until April 2023.
Last login: Thu Dec 10 10:17:07 2020 from 192.168.56.111
```

第一次登录需要输入 yes 进行确认。

3. 测试是否能够访问外网。比如 worker1 ping 百度

```
ping www.baidu.com
```

```
lei@worker1:~$ ping www.baidu.com
PING www.a.shifen.com (180.101.49.12) 56(84) bytes of data.
64 bytes from 180.101.49.12 (180.101.49.12): icmp_seq=1 ttl=48 time=20.4 ms
64 bytes from 180.101.49.12 (180.101.49.12): icmp_seq=2 ttl=48 time=19.4 ms
64 bytes from 180.101.49.12 (180.101.49.12): icmp_seq=3 ttl=48 time=19.5 ms
64 bytes from 180.101.49.12 (180.101.49.12): icmp_seq=4 ttl=48 time=20.0 ms
64 bytes from 180.101.49.12 (180.101.49.12): icmp_seq=5 ttl=48 time=19.5 ms
64 bytes from 180.101.49.12 (180.101.49.12): icmp_seq=6 ttl=48 time=12.6 ms
```

测试无误后，三台主机的集群就搭建好了。

3. 修改 Hadoop 和 Spark 配置文件

接下来修改 Hadoop 和 Spark 配置文件，将伪分布式改为集群模式。

3.1 设置 Hadoop

首先将 Hadoop 改为集群模式。在 master 主机中修改下面四个文件，

1. 修改 core-site.xml

将配置文件 /apps/hadoop/etc/hadoop/core-site.xml 中 fs.defaultFS 的值由

hdfs://localhost:9000 改为 hdfs://master:9000，修改以后，如下图所示

```
<property>
  <name>fs.defaultFS</name>
  <value>hdfs://master:9000</value>
</property>
<property>
```

2. 修改 hdfs-site.xml 文件

将配置文件/apps/hadoop/etc/hadoop/hdfs-site.xml 中 dfs.replication 的值由 1 改为

3，修改以后如下图所示

```
<property>
  <name>dfs.replication</name>
  <value>3</value>
</property>
```

3. 修改 workers 文件

将配置文件/apps/hadoop/etc/hadoop/workers 中 localhost 改为

```
master
worker1
worker2
```

修改后如下图所示

```
master  
worker1  
worker2
```

4. 修改 yarn-site.xml 文件

将以下内容添加到配置文件/apps/hadoop/etc/hadoop/ yarn-site.xml 中

```
<property>  
  <name>yarn.resourcemanager.hostname</name>  
  <value>master</value>  
</property>
```

将默认的 ResourceManager 地址 0.0.0.0 改为 master, 不修改的话, 在 WebUI:

<http://master:8088/cluster/nodes> 中只显示一个 Active Node。修改以后的文件内容如

下图所示

```
<configuration>  
<!-- Site specific YARN configuration properties -->  
  <property>  
    <name>yarn.nodemanager.aux-services</name>  
    <value>mapreduce_shuffle</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.resourcemanager.hostname</name>  
    <value>master</value>  
  </property>  
</configuration>
```

将上面修改的四个文件复制到 worker1 和 worker2 两个节点, 覆盖原来的文件

```
cd /apps/hadoop/etc/hadoop/  
scp core-site.xml hdfs-site.xml workers yarn-site.xml \  
worker1:/apps/hadoop/etc/hadoop/  
scp core-site.xml hdfs-site.xml workers yarn-site.xml \  
worker2:/apps/hadoop/etc/hadoop/
```

删除伪分布式 namenode 文件

重新对分布式文件系统进行格式化前，需要删除三台主机中/data/tmp/hadoop/hdfs/目录下的文件和文件夹。首先删除 master 上/data/tmp/hadoop/hdfs/目录下的文件和文件夹

```
rm -rf /data/tmp/hadoop/hdfs/*
```

删除另外两台主机上相应的文件，不需要切换窗口，可以直接从 master 分别登录到 work1 和 work2 进行删除，比如，删除 worker1 上的文件

```
ssh worker1
rm -rf /data/tmp/hadoop/hdfs/*
```

格式化分布式文件系统

在主节点 master 执行以下命令

```
hadoop namenode -format
```

```
2020-12-10 12:07:38,766 INFO common.Storage: Storage directory /data/tmp/hadoop/hdfs/name has been successfully formatted.
2020-12-10 12:07:38,814 INFO namenode.FSImageFormatProtobuf: Saving image file /data/tmp/hadoop/hdfs/name/current/fsimage.ckpt_000000000000000000 using no compression
2020-12-10 12:07:38,977 INFO namenode.FSImageFormatProtobuf: Image file /data/tmp/hadoop/hdfs/name/current/fsimage.ckpt_000000000000000000 of size 388 bytes saved in 0 seconds.
2020-12-10 12:07:39,018 INFO namenode.NNStorageRetentionManager: Going to retain 1 images with txid >= 0
2020-12-10 12:07:39,057 INFO namenode.NameNode: SHUTDOWN_MSG:
/*****
SHUTDOWN_MSG: Shutting down NameNode at master/192.168.56.111
*****/
```

至此，Hadoop 分布式集群就设置好了，下面进行测试。

启动 Hadoop

在 master 节点执行

```
/apps/hadoop/sbin/start-all.sh
```

查看 Hadoop 进程

在主节点 master 执行

```
jps
```

```
lei@master:/apps/hadoop$ jps
2980 DataNode
3636 NodeManager
3476 ResourceManager
3209 SecondaryNameNode
2810 NameNode
3996 Jps
```

从节点 worker1

```
jps
```

```
lei@worker1:~$ jps
2983 Jps
2696 DataNode
2862 NodeManager
```

从节点 worker2

```
jps
```

```
lei@worker2:~$ jps
4993 NodeManager
4818 DataNode
5300 Jps
```

可以看到 HDFS 的 NameNode 和 SecondaryNameNode，以及 Yarn 的 ResourceManager 只运行在主节点；HDFS 的 DataNode 和 MapReduce 的 NodeManager 只运行在从节点。

测试 HDFS

在 HDFS 上创建目录/input

```
hadoop fs -mkdir /input
```

查看是否创建成功

```
hadoop fs -ls /
```

将文件传到 HDFS

```
hadoop fs -put /data/testfile /input
```

运行 wordcount


```
cd /apps/hadoop/share/hadoop/mapreduce/
hadoop jar hadoop-mapreduce-examples-3.0.0.jar wordcount \
/input/testfile /output
```


查看结果

```
hadoop fs -cat /output/*
```

```
lei@master:/apps/hadoop/share/hadoop/mapreduce$ hadoop fs -cat /output/*
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/apps/hadoop/share/hadoop/common/lib/slf4j-log
4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/apps/hive/lib/log4j-slf4j-impl-2.6.2.jar!/org
/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
big      1
data     1
hdfs     1
hello    3
world    1
```

Web UI

<http://master:8088/> 可以查看 Hadoop 集群，节点及任务相关信息。可以看到现在活跃的节点数是 3。



Logged in as: dr.who

About the Cluster

- Cluster
- About
- Nodes
- Node Labels
- Applications
- NEW
- NEW SAVING
- SUBMITTED
- ACCEPTED
- RUNNING
- FINISHED
- FAILED
- KILLED
- Scheduler
- Tools

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved
2	0	0	2	0	0 B	24 GB	0 B	0	24	0

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

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation	Maximum Cluster Application Priority
Capacity Scheduler	[memory-mb (unit=M), vcores]	<memory:1024, vCores:1>	<memory:8192, vCores:4>	0

HDFS Web 界面

在浏览器中访问 <http://master:9870/>，可以查看 HDFS 相关信息，浏览 HDFS 上的

文件系统

Hadoop

Overview

Datanodes

Datanode Volume Failures

Snapshot

Startup Progress

Utilities

Overview

'master:9000' (active)

Started:	Thu Dec 10 14:28:12 +0800 2020
Version:	3.0.0, rc25427ceca461ee979d30edd7a4b0f50718e6533
Compiled:	Sat Dec 09 03:16:00 +0800 2017 by andrew from branch-3.0.0
Cluster ID:	CID-4c1415c7-2d73-4cd6-944d-7c367f740865
Block Pool ID:	BP-645325260-192.168.56.111-1607581121408

Summary

Security is off.

Safemode is off.

20 files and directories, 8 blocks = 28 total filesystem object(s).

Heap Memory used 44.56 MB of 61.02 MB Heap Memory. Max Heap Memory is 953.19 MB.

Non Heap Memory used 53.2 MB of 54.3 MB Committed Non Heap Memory. Max Non Heap Memory is <unbounded>.

Hadoop

Overview

Datanodes

Datanode Volume Failures

Snapshot

Startup Progress

Utilities

Browse Directory

/output

Go!

Show

25

entries

Search:

<input type="checkbox"/>	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name	
<input type="checkbox"/>	-rw-r--r--	lei	supergroup	0 B	Dec 10 14:59	3	128 MB	._SUCCESS	
<input type="checkbox"/>	-rw-r--r--	lei	supergroup	36 B	Dec 10 14:59	3	128 MB	part-r-00000	

Showing 1 to 2 of 2 entries

Previous

1

Next

3.2 配置 Spark 集群

修改配置文件

在 master 节点修改下面三个文件

1. 修改 spark-env.sh, 将 SPARK_MASTER_IP 的值改为 master, 修改后, 如下图所示

```
HADOOP_CONF_DIR=/apps/hadoop/etc/hadoop
JAVA_HOME=/apps/java
SPARK_MASTER_IP=master
SPARK_MASTER_PORT=7077
SPARK_MASTER_WEBUI_PORT=8080
SPARK_WORKER_CORES=1
SPARK_WORKER_MEMORY=1g
SPARK_WORKER_PORT=7078
SPARK_WORKER_WEBUI_PORT=8081
SPARK_EXECUTOR_INSTANCES=1
```

2. 修改 slaves 文件, 将 localhost 改为

```
master
worker1
worker2
```

3. 修改 spark-defaults.conf, 将 spark.master 改为 spark://master:7077,

spark.eventLog.dir 改为 hdfs://master:9000/spark/eventLog。修改后 如下图所示

```
spark.master                spark://master:7077
spark.eventLog.enabled      true
spark.eventLog.dir          hdfs://master:9000/spark/eventLog
spark.serializer            org.apache.spark.serializer.KryoSerializer
spark.driver.memory         1g
spark.jars.package          Azure:mmlspark:0.12
```

eventLog 用来存放日志, 需要手动创建

```
hadoop fs -mkdir -p /spark/eventLog
```

将修改的三个文件复制到 worker1 和 worker2 两个节点, 覆盖原来的文件

```
cd /apps/spark/conf
scp spark-env.sh slaves spark-defaults.conf worker1:/apps/spark/conf
scp spark-env.sh slaves spark-defaults.conf worker2:/apps/spark/conf
```

至此, 配置文件就修改好了, 下面进行测试。

启动 spark 集群

```
/apps/spark/sbin/start-all.sh
```

查看进程

使用 jps 命令查看 spark 进程，主节点多了两个进程 Master 和 Worker。

```
lei@master:/apps/spark$ jps
14370 Worker
14438 Jps
12026 ResourceManager
11755 SecondaryNameNode
14204 Master
12191 NodeManager
11519 DataNode
11343 NameNode
```

从节点多了一个进程 Worker

```
lei@worker1:/apps/spark/conf$ jps
4519 NodeManager
4344 DataNode
6586 Worker
6638 Jps
```

Web UI

查看 spark 管理界面，在浏览器中输入 <http://master:8080>，可以看到 Worker 有三个。



Spark Master at spark://master:7077

URL: spark://master:7077
Alive Workers: 3
Cores in use: 3 Total, 0 Used
Memory in use: 3.0 GB Total, 0.0 B Used
Applications: 0 Running, 0 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers (3)

Worker Id	Address	State	Cores	Memory
worker-20201210155733-192.168.56.112-7078	192.168.56.112:7078	ALIVE	1 (0 Used)	1024.0 MB (0.0 B Used)
worker-20201210155733-192.168.56.113-7078	192.168.56.113:7078	ALIVE	1 (0 Used)	1024.0 MB (0.0 B Used)
worker-20201210155735-192.168.56.111-7078	192.168.56.111:7078	ALIVE	1 (0 Used)	1024.0 MB (0.0 B Used)

Running Applications (0)

Application ID	Name	Cores	Memory per Executor	Submitted Time	User	State	Duration
----------------	------	-------	---------------------	----------------	------	-------	----------

Completed Applications (0)

Application ID	Name	Cores	Memory per Executor	Submitted Time	User	State	Duration
----------------	------	-------	---------------------	----------------	------	-------	----------

运行演示实例

计算 pi 的值

```
/apps/spark/bin/run-example SparkPi  
lei@master:/apps/spark/conf$ /apps/spark/bin/run-example SparkPi  
20/12/10 16:19:10 WARN NativeCodeLoader: Unable to load native-hadoop library fo  
r your platform... using builtin-java classes where applicable  
Pi is roughly 3.1431757158785794
```

4. 总结

本次实验练习了如何将 Hadoop, Spark 伪分布式系统修改为分布式系统。首先根据
需要创建多台虚拟机，然后设置虚拟机的 IP，使得虚拟机之间的网络是连通的，并且相互
之间可以免密码登录。然后对大数据系统系统进行配置，主要是根据节点数和主机名修改
配置文件。最后，对分布式系统进行了测试。