

1 虚拟机硬件情况和网络配置信息

宿主机处理器: Intel® Core™ i5-8250U

宿主机操作系统: Windows 10, 64 位

宿主机内存: 8G

虚拟机软件: VMware Workstation Pro 12

虚拟机操作系统: ubuntu 18.04, 2G 内存

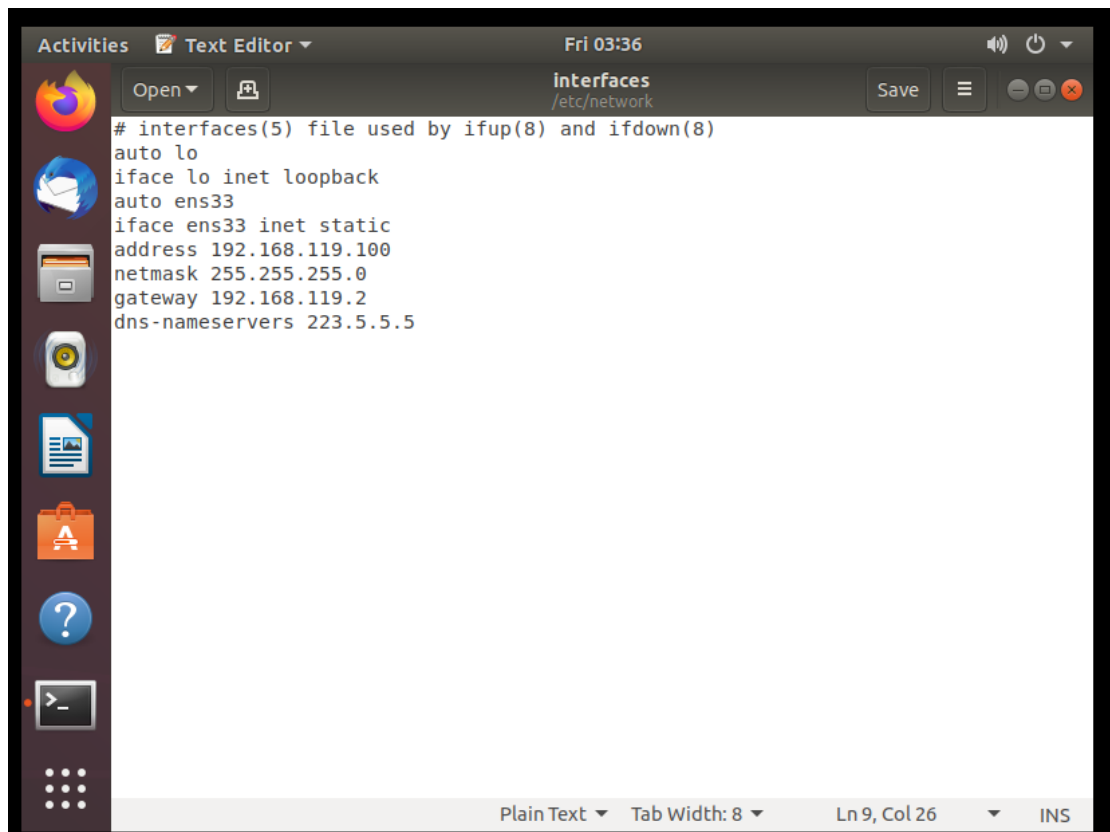
虚拟机硬件:

| 设备 | 摘要 |
|---------------|-------------|
| 内存 | 2 GB |
| 处理器 | 1 |
| 硬盘(SCSI) | 20 GB |
| CD/DVD (SATA) | 自动检测 |
| 网络适配器 | 自定义(VMnet8) |
| USB 控制器 | 存在 |
| 声卡 | 自动检测 |
| 打印机 | 存在 |
| 显示器 | 自动检测 |

虚拟机网络环境:

| 序号 | IP 地址 | 主机名 | 系统 | User |
|----|-----------------|---------|-------------|---------|
| 1 | 192.168.119.100 | master | Ubuntu18.04 | master |
| 2 | 192.168.119.101 | slaver1 | Ubuntu18.04 | slaver1 |
| 3 | 192.168.119.102 | slaver2 | Ubuntu18.04 | slaver2 |

master 主机上的网络环境截图:



```
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback
auto ens33
iface ens33 inet static
address 192.168.119.100
netmask 255.255.255.0
gateway 192.168.119.2
dns-nameservers 223.5.5.5
```

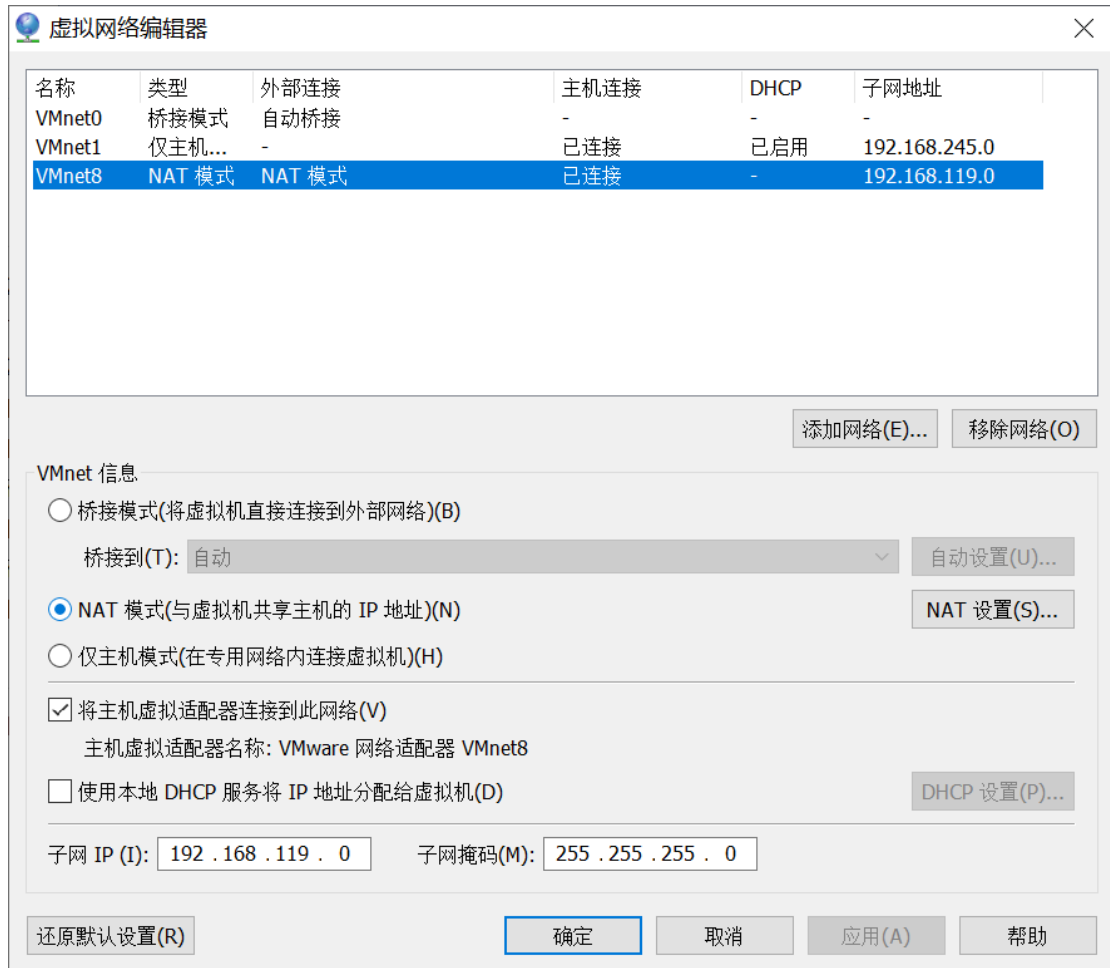
```
# This file is managed by man:systemd-resolved(8). Do not edit.
#
# This is a dynamic resolv.conf file for connecting local clients to the
# internal DNS stub resolver of systemd-resolved. This file lists all
# configured search domains.
#
# Run "systemd-resolve --status" to see details about the uplink DNS servers
# currently in use.
#
# Third party programs must not access this file directly, but only through the
# symlink at /etc/resolv.conf. To manage man:resolv.conf(5) in a different way,
# replace this symlink by a static file or a different symlink.
#
# See man:systemd-resolved.service(8) for details about the supported modes of
# operation for /etc/resolv.conf.
nameserver 223.5.5.5]
options edns0
```

```
127.0.0.1 localhost
127.0.1.1 ubuntu
192.168.119.100 master
192.168.119.101 slaver1
192.168.119.102 slaver2
# 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
```

2 集群搭建操作记录

2.1 Vmware 网络配置以及 Ubuntu 虚拟机网络配置

打开 Vmware 中的虚拟网络编辑器，我们需要实现静态 IP 地址，所以在 VMnet8 网络中，去掉勾选“使用本地 DHCP 服务将 IP 地址分配给虚拟机”。子网 IP 为：192.168.119.0，子网掩码为：255.255.255.0。设置三个虚拟机的 IP 地址分别为 192.168.119.100(master)，192.168.119.101(slaver1)，192.168.119.102(slaver2)。

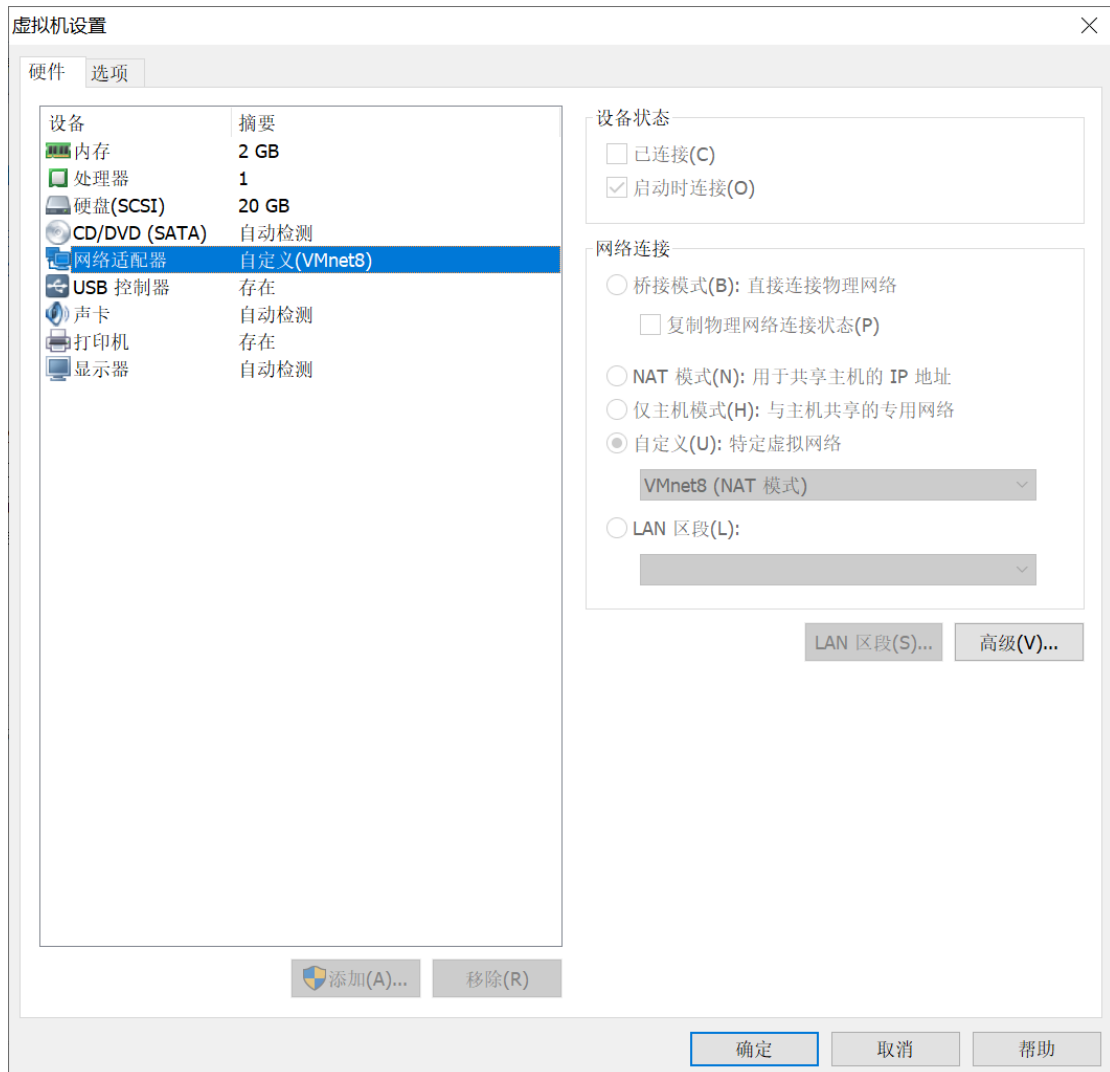


点击 NAT 设置，查看网关为 192.168.119.2。

NAT 设置

| | |
|-----------|---------------|
| 网络: | vmnet8 |
| 子网 IP: | 192.168.119.0 |
| 子网掩码: | 255.255.255.0 |
| 网关 IP(G): | 192.168.119.2 |

右键点击自己建立的虚拟机，点击“设置”，查看如下界面，选择网络适配器，确定网络连接选中的是“自定义”中的 VMnet8(NAT 模式)。最后点击确定，开启虚拟机。



打开 Ubuntu 的终端，输入：`sudo gedit /etc/network/interfaces`，进行如下编辑并保存。

```
auto lo
iface lo inet loopback
auto ens33
iface ens33 inet static
address 192.168.119.100
netmask 255.255.255.0
gateway 192.168.153.2
dns-nameservers 223.5.5.5
```

配置 DNS 服务器，在终端中输入：`sudo gedit /etc/resolv.conf`，进行如下编辑并保存。

```
nameserver 223.5.5.5
```

然后，在终端中输入：`sudo /etc/init.d/networking restart`，重启网络。

2.2 分布式环境搭建

修改 hosts 文件，将 IP 与主机名的映射添加到 hosts 文件中。在终端中输入：`sudo gedit /etc/hosts`，进行如下编辑并保存。

```
192.168.119.100 master
192.168.119.101 slaver1
192.168.119.102 slaver2
```

修改完成后保存执行如下命令：

```
source /etc/hosts
```

关闭虚拟机，在 Vmware 界面对 master 进行克隆，克隆出两个节点 slaver1 和 slaver2。

在每个机器上使用以下命令修改本机名称，分别改成 master, slaver1, slaver2。

```
sudo gedit /etc/hostname
```

在每个机器的/etc/network/interfaces 中，将机器的 ip 设置为 static，并分别分配固定 ip。分别为：192.168.119.100，192.168.119.101，192.168.119.102。

对虚拟机进行重启，使相关配置生效。

2.3 免密 ssh 配置

在三台虚拟机中，使用以下命令安装 ssh：

```
sudo apt-get install openssh-server
```

在 master 节点上进行 ssh 配置，在 master 节点上执行以下命令。

```
su root
```

```
cd /root/.ssh
```

```
ssh-keygen -t rsa
```

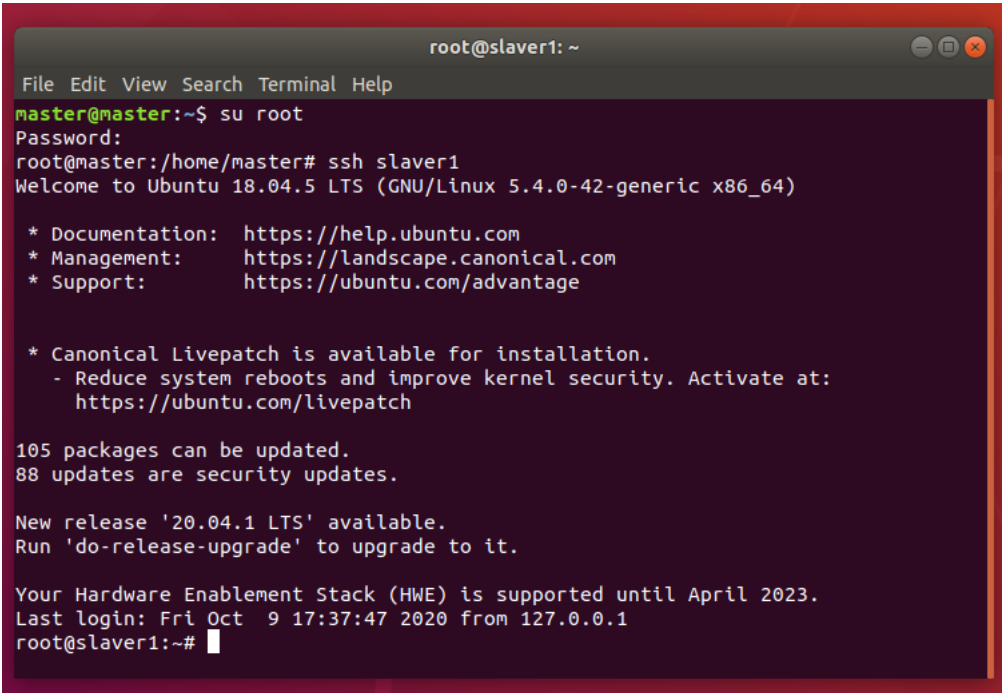
```
ssh-copy-id -i /root/.ssh/id_rsa.pub master
```

```
ssh-copy-id -i /root/.ssh/id_rsa.pub slaver1
```

```
ssh-copy-id -i /root/.ssh/id_rsa.pub slaver2
```

在 master 上对每一个节点进行测试，看是否能进行免密登录：

```
ssh slaver1
```

A terminal window titled 'root@slaver1: ~' showing a successful SSH connection. The user 'master' on the 'master' machine runs 'su root' and then 'ssh slaver1'. The terminal output shows the Ubuntu login banner for 'root@slaver1' and the prompt 'root@slaver1:~#'.

```
root@slaver1: ~
File Edit View Search Terminal Help
master@master:~$ su root
Password:
root@master:/home/master# ssh slaver1
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 5.4.0-42-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

105 packages can be updated.
88 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: Fri Oct 9 17:37:47 2020 from 127.0.0.1
root@slaver1:~#
```

2.4 Java 环境搭建

在/usr 目录下新建 java 文件夹，然后将 jdk 压缩包复制到 java 文件夹下并进行解压。执行：

```
cd /usr
```

```
mkdir java
```

```
sudo tar -zxvf jdk-8u201-linux-x64.tar.gz
```

在/etc/profile 内配置环境变量：gedit /etc/profile。添加如下信息：

```
export JAVA_HOME=/usr/java/jdk1.8.0_201
```

```
export CLASSPATH=$JAVA_HOME/lib:$JAVA_HOME/jre/lib:$CLASSPATH
```

```
export PATH=$JAVA_HOME/bin:$JAVA_HOME/jre/bin:$PATH
```

刷新环境配置。然后检测 Java 版本：

```
source /etc/profile
```

```
java -version
```

2.5 scala 环境配置

将 scala 压缩包复制到 java 文件夹下并进行解压。执行：

```
sudo tar -zxvf scala-2.11.8.tgz
```

在/etc/profile 内配置环境变量：gedit /etc/profile。添加如下信息：

```
export SCALA_HOME=/usr/scala-2.11.8
```

```
export PATH=$PATH:$SCALA_HOME/bin
```

刷新环境配置。然后检测 scala 版本：

```
source /etc/profile
```

```
scala -version
```

2.6 hadoop 环境配置

解压并移动到/usr/java 目录

```
sudo tar -zxvf hadoop-2.7.7.tar.gz
```

修改相应的配置文件。修改/etc/profile，增加如下内容：

```
export HADOOP_HOME=/usr/java/hadoop-2.7.7
```

```
export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin
```

在 hadoop-2.7.7 目录下添加目录：

```
mkdir tmp
```

```
mkdir hdfs
```

```
mkdir hdfs/name
```

```
mkdir hdfs/data
```

修改\$HADOOP_HOME/etc/hadoop/hadoop-env.sh，修改 JAVA_HOME 如下：

```
export JAVA_HOME=/usr/java/jdk1.8.0_201
```

修改\$HADOOP_HOME/etc/hadoop/slaves，将原来的 localhost 删除，添加如下内容：

```
slaver1
```

```
slaver2
```

修改\$HADOOP_HOME/etc/hadoop/core-site.xml，修改为如下内容：

```
<configuration>
  <property>
    <name>fs.defaultFS</name>
    <value>hdfs://Master:9000</value>
  </property>
  <property>
    <name>io.file.buffer.size</name>
    <value>131072</value>
  </property>
  <property>
    <name>hadoop.tmp.dir</name>
    <value>/usr/java/hadoop-2.7.7/tmp</value>
  </property>
```

```
</configuration>
```

修改\$HADOOP_HOME/etc/hadoop/hdfs-site.xml。

```
<configuration>
```

```
  <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.namenode.name.dir</name>
```

```
    <value>file:/usr/java/hadoop-2.7.7/hdfs/name</value>
```

```
  </property>
```

```
  <property>
```

```
    <name>dfs.datanode.data.dir</name>
```

```
    <value>file:/usr/java/hadoop-2.7.7/hdfs/data</value>
```

```
  </property>
```

```
</configuration>
```

在\$HADOOP_HOME/etc/hadoop 目录下复制 template，生成 xml，命令如下：

```
cp mapred-site.xml.template mapred-site.xml
```

修改\$HADOOP_HOME/etc/hadoop/mapred-site.xml

```
<configuration>
```

```
  <property>
```

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

```
    <value>yarn</value>
```

```
  </property>
```

```
  <property>
```

```
    <name>mapreduce.jobhistory.address</name>
```

```
    <value>Master:10020</value>
```

```
  </property>
```

```
  <property>
```

```
    <name>mapreduce.jobhistory.address</name>
```

```
    <value>Master:19888</value>
```

```
  </property>
```

```
</configuration>
```

修改\$HADOOP_HOME/etc/hadoop/yarn-site.xml。

```
<configuration>
```

```
  <property>
```

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

```
    <value>mapreduce_shuffle</value>
```

```
  </property>
```

```
</property>
```

```

        <name>yarn.resourcemanager.address</name>
        <value>Master:8032</value>
    </property>
    <property>
        <name>yarn.resourcemanager.scheduler.address</name>
        <value>Master:8030</value>
    </property>
    <property>
        <name>yarn.resourcemanager.resource-tracker.address</name>
        <value>Master:8031</value>
    </property>
    <property>
        <name>yarn.resourcemanager.admin.address</name>
        <value>Master:8033</value>
    </property>
    <property>
        <name>yarn.resourcemanager.webapp.address</name>
        <value>Master:8088</value>
    </property>
</configuration>

```

2.7 spark 环境配置

解压并移动到相应目录，命令如下：

```
sudo tar -zxvf spark-2.4.7-bin-hadoop2.7.tgz
```

修改/etc/profile，增加如下内容：

```
export SPARK_HOME=/usr/java/spark-2.4.7-bin-hadoop2.7/
```

```
export PATH=$PATH:$SPARK_HOME/bin
```

在\$SPARK_HOME/conf/目录下复制 spark-env.sh.template 成 spark-env.sh

```
cp spark-env.sh.template spark-env.sh
```

修改\$SPARK_HOME/conf/spark-env.sh，添加如下内容：

```
export SCALA_HOME=/usr/java/scala-2.11.8
```

```
export JAVA_HOME=/usr/java/jdk1.8.0_201
```

```
export HADOOP_HOME=/usr/java/hadoop-2.7.7
```

```
export SPARK_WORKER_MEMORY=1g
```

```
export HADOOP_CONF_DIR=/usr/java/hadoop-2.7.7/etc/Hadoop
```

在\$SPARK_HOME/conf/目录下复制 slaves.template 成 slaves

```
cp slaves.template slaves
```

修改\$SPARK_HOME/conf/slaves，添加如下内容：

```
master
```

```
slaver1
```

```
slaver2
```

2.8 节点环境配置及测试

将配置好的环境拷贝到 Slaver1 和 Slaver2 节点。

```
scp -r /usr/java root@slaver1:/usr
```

```
scp -r /etc/profile root@slaver1:/etc/profile
```



```
scp -r /usr/java root@slaver2:/usr
scp -r /etc/profile root@slaver2:/etc/profile
在每个节点上刷新环境配置: source /etc/profile。
ssh slaver1
source /etc/profile
exit
ssh slaver2
source /etc/profile
exit
```

在 master 节点启动 Hadoop, 启动之前格式化一下 namenode:

```
hadoop namenode -format
```

启动:

```
/usr/java/hadoop-2.7.7/sbin/start-all.sh
```

查看 Hadoop 是否启动成功, 输入命令: jps

Master 显示: SecondaryNameNode, ResourceManager, NameNode。

Slaver 显示: NodeManager, DataNode。

```
root@master:/home/master# jps
6993 NameNode
7235 SecondaryNameNode
7731 Master
7380 ResourceManager
11373 Jps
7854 Worker
root@slaver1:/usr/java/hadoop-2.7.7# jps
3475 NodeManager
3316 DataNode
5243 Jps
3695 Worker
```

在 master 节点启动 Spark: /usr/java/spark-2.4.7-bin-hadoop2.7/sbin/start-all.sh

查看 Spark 是否启动成功, 输入命令: jps

Master 在 Hadoop 的基础上新增了: Master。

Slaver 在 Hadoop 的基础上新增了: Worker。

```
root@master:/home/master# jps
6993 NameNode
7235 SecondaryNameNode
7731 Master
7380 ResourceManager
11373 Jps
7854 Worker
root@slaver1:/usr/java/hadoop-2.7.7# jps
3475 NodeManager
3316 DataNode
5243 Jps
3695 Worker
```

另外我们根据官网, 运行一下代码:

```
/usr/java/spark-2.4.7-bin-hadoop2.7/bin/run-example SparkPi 10
```

结果如下, 计算出 pi:

```

20/10/09 18:09:14 INFO executor.Executor: Adding file:/tmp/spark-f8d5cd3c-8b2d-483e-9b13-ec9ee3191c78/userFiles-deebefca-ebbd-4dd5-bbd0-3c2bec893477/spark-examples_2.11-2.4.7.jar to class loader
20/10/09 18:09:15 INFO executor.Executor: Finished task 0.0 in stage 0.0 (TID 0). 910 bytes result sent to driver
20/10/09 18:09:15 INFO scheduler.TaskSetManager: Starting task 1.0 in stage 0.0 (TID 1), localhost, executor driver, partition 1, PROCESS_LOCAL, 7866 bytes)
20/10/09 18:09:15 INFO executor.Executor: Running task 1.0 in stage 0.0 (TID 1)
20/10/09 18:09:16 INFO scheduler.TaskSetManager: Finished task 0.0 in stage 0.0 (TID 0) in 1653 ms on localhost (executor driver) (1/10)
20/10/09 18:09:16 INFO executor.Executor: Finished task 1.0 in stage 0.0 (TID 1). 824 bytes result sent to driver
20/10/09 18:09:16 INFO scheduler.TaskSetManager: Starting task 2.0 in stage 0.0 (TID 2), localhost, executor driver, partition 2, PROCESS_LOCAL, 7866 bytes)
20/10/09 18:09:16 INFO executor.Executor: Running task 2.0 in stage 0.0 (TID 2)
20/10/09 18:09:17 INFO executor.Executor: Finished task 2.0 in stage 0.0 (TID 2). 867 bytes result sent to driver
20/10/09 18:09:17 INFO scheduler.TaskSetManager: Starting task 3.0 in stage 0.0 (TID 3), localhost, executor driver, partition 3, PROCESS_LOCAL, 7866 bytes)
20/10/09 18:09:17 INFO executor.Executor: Running task 3.0 in stage 0.0 (TID 3)
20/10/09 18:09:17 INFO scheduler.TaskSetManager: Finished task 2.0 in stage 0.0 (TID 2) in 848 ms on localhost (executor driver) (3/10)
20/10/09 18:09:17 INFO executor.Executor: Running task 4.0 in stage 0.0 (TID 4)
20/10/09 18:09:17 INFO scheduler.TaskSetManager: Starting task 4.0 in stage 0.0 (TID 4), localhost, executor driver, partition 4, PROCESS_LOCAL, 7866 bytes)
20/10/09 18:09:17 INFO executor.Executor: Running task 4.0 in stage 0.0 (TID 4)
20/10/09 18:09:17 INFO scheduler.TaskSetManager: Finished task 3.0 in stage 0.0 (TID 3) in 588 ms on localhost (executor driver) (4/10)
20/10/09 18:09:18 INFO executor.Executor: Running task 5.0 in stage 0.0 (TID 5)
20/10/09 18:09:18 INFO scheduler.TaskSetManager: Starting task 5.0 in stage 0.0 (TID 5), localhost, executor driver, partition 5, PROCESS_LOCAL, 7866 bytes)
20/10/09 18:09:18 INFO executor.Executor: Running task 5.0 in stage 0.0 (TID 5)
20/10/09 18:09:18 INFO scheduler.TaskSetManager: Finished task 4.0 in stage 0.0 (TID 4) in 345 ms on localhost (executor driver) (5/10)
20/10/09 18:09:18 INFO executor.Executor: Running task 6.0 in stage 0.0 (TID 6)
20/10/09 18:09:18 INFO scheduler.TaskSetManager: Starting task 6.0 in stage 0.0 (TID 6), localhost, executor driver, partition 6, PROCESS_LOCAL, 7866 bytes)
20/10/09 18:09:18 INFO executor.Executor: Running task 6.0 in stage 0.0 (TID 6)
20/10/09 18:09:18 INFO scheduler.TaskSetManager: Finished task 5.0 in stage 0.0 (TID 5) in 355 ms on localhost (executor driver) (6/10)
20/10/09 18:09:18 INFO executor.Executor: Running task 7.0 in stage 0.0 (TID 7)
20/10/09 18:09:18 INFO scheduler.TaskSetManager: Starting task 7.0 in stage 0.0 (TID 7), localhost, executor driver, partition 7, PROCESS_LOCAL, 7866 bytes)
20/10/09 18:09:18 INFO executor.Executor: Running task 7.0 in stage 0.0 (TID 7)
20/10/09 18:09:18 INFO scheduler.TaskSetManager: Finished task 6.0 in stage 0.0 (TID 6) in 432 ms on localhost (executor driver) (7/10)
20/10/09 18:09:19 INFO executor.Executor: Running task 8.0 in stage 0.0 (TID 8)
20/10/09 18:09:19 INFO scheduler.TaskSetManager: Starting task 8.0 in stage 0.0 (TID 8), localhost, executor driver, partition 8, PROCESS_LOCAL, 7866 bytes)
20/10/09 18:09:19 INFO executor.Executor: Running task 8.0 in stage 0.0 (TID 8)
20/10/09 18:09:19 INFO scheduler.TaskSetManager: Finished task 7.0 in stage 0.0 (TID 7) in 395 ms on localhost (executor driver) (8/10)
20/10/09 18:09:19 INFO executor.Executor: Running task 9.0 in stage 0.0 (TID 9)
20/10/09 18:09:19 INFO scheduler.TaskSetManager: Starting task 9.0 in stage 0.0 (TID 9), localhost, executor driver, partition 9, PROCESS_LOCAL, 7866 bytes)
20/10/09 18:09:19 INFO executor.Executor: Running task 9.0 in stage 0.0 (TID 9)
20/10/09 18:09:19 INFO scheduler.TaskSetManager: Finished task 8.0 in stage 0.0 (TID 8) in 384 ms on localhost (executor driver) (9/10)
20/10/09 18:09:19 INFO executor.Executor: Running task 10.0 in stage 0.0 (TID 10)
20/10/09 18:09:19 INFO scheduler.TaskSetManager: Starting task 10.0 in stage 0.0 (TID 10), localhost, executor driver, partition 0, PROCESS_LOCAL, 7866 bytes)
20/10/09 18:09:19 INFO executor.Executor: Running task 10.0 in stage 0.0 (TID 10)
20/10/09 18:09:19 INFO scheduler.TaskSetManager: Finished task 9.0 in stage 0.0 (TID 9) in 824 bytes result sent to driver
20/10/09 18:09:20 INFO scheduler.DAGScheduler: ResultStage 0 (reduce at sparkPL.scala:38) finished in 17.422 s
20/10/09 18:09:20 INFO scheduler.DAGScheduler: Job 0 finished: reduce at sparkPL.scala:38, took 10.565479 s
PL to roughly 3.541971141971142
20/10/09 18:09:20 INFO server.AbstractConnector: Stopped sparkIe@0ceff(HTTP/1.1,[http://1.1.1.1:0.0.0:4040])
20/10/09 18:09:20 INFO ui.SparkUI: Stopped Spark web UI at http://master:4040
20/10/09 18:09:20 INFO spark.MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
20/10/09 18:09:20 INFO memory.MemoryStore: MemoryStore cleared
20/10/09 18:09:20 INFO storage.BlockManager: BlockManager stopped
20/10/09 18:09:20 INFO scheduler.OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
20/10/09 18:09:20 INFO spark.SparkContext: Successfully stopped SparkContext
20/10/09 18:09:20 INFO util.ShutdownHookManager: Shutdown hook called
20/10/09 18:09:20 INFO util.ShutdownHookManager: Deleting directory /tmp/spark-4311b1b5-228e-4f63-a228-ede57a3e2e25
20/10/09 18:09:21 INFO util.ShutdownHookManager: Deleting directory /tmp/spark-f8d5cd3c-8b2d-483e-9b13-ec9ee3191c78
root@master:~/usr/java/spark-2.4.7-bin-hadoop2.7/bin#

```

在输入以下代码后，可以在 WebUI 看到运行应用：

```

$SPARK_HOME/bin/spark-submit --class org.apache.spark.examples.SparkPi --master
spark://master:7077 /usr/java/spark-2.4.7-bin-hadoop2.7/examples/jars/spark-
examples_2.11-2.4.7.jar

```

The screenshot shows the Spark Master WebUI at `spark://master:7077`. The page displays the following information:

- 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, 1 Completed
- Drivers:** 0 Running, 0 Completed
- Status:** ALIVE

Under the **Workers (3)** section, there is a table with the following data:

| Worker ID | Address | State | Cores | Memory |
|---------------------------------------------|-----------------------|-------|------------|----------|
| worker-20201010064944-192.168.119.101-38331 | 192.168.119.101:38331 | ALIVE | 1 (0 Used) | 1024.0 M |
| worker-20201010064945-192.168.119.102-46745 | 192.168.119.102:46745 | ALIVE | 1 (0 Used) | 1024.0 M |
| worker-20201010064955-192.168.119.100-46267 | 192.168.119.100:46267 | ALIVE | 1 (0 Used) | 1024.0 M |

Under the **Running Applications (0)** section, there is an empty table with the following headers:

| Application ID | Name | Cores | Memory per Executor | Submitted Time | User |
|----------------|------|-------|---------------------|----------------|------|
|----------------|------|-------|---------------------|----------------|------|

Under the **Completed Applications (1)** section, there is a table with the following data:

| Application ID | Name | Cores | Memory per Executor | Submitted Time | User | St |
|-------------------------|----------|-------|---------------------|---------------------|------|----|
| app-20201010071554-0000 | Spark Pi | 3 | 1024.0 MB | 2020/10/10 07:15:54 | root | Fi |