一、搭建准备

VMWare Workstation

Ubuntu 18.04 Server.iso

Server 版安装配置
新建三台虚拟机，安装时选择 OpenSSH Server 预安装环境

一台作为 master

另两台作为 slave，命名为 slave1 和 slave2

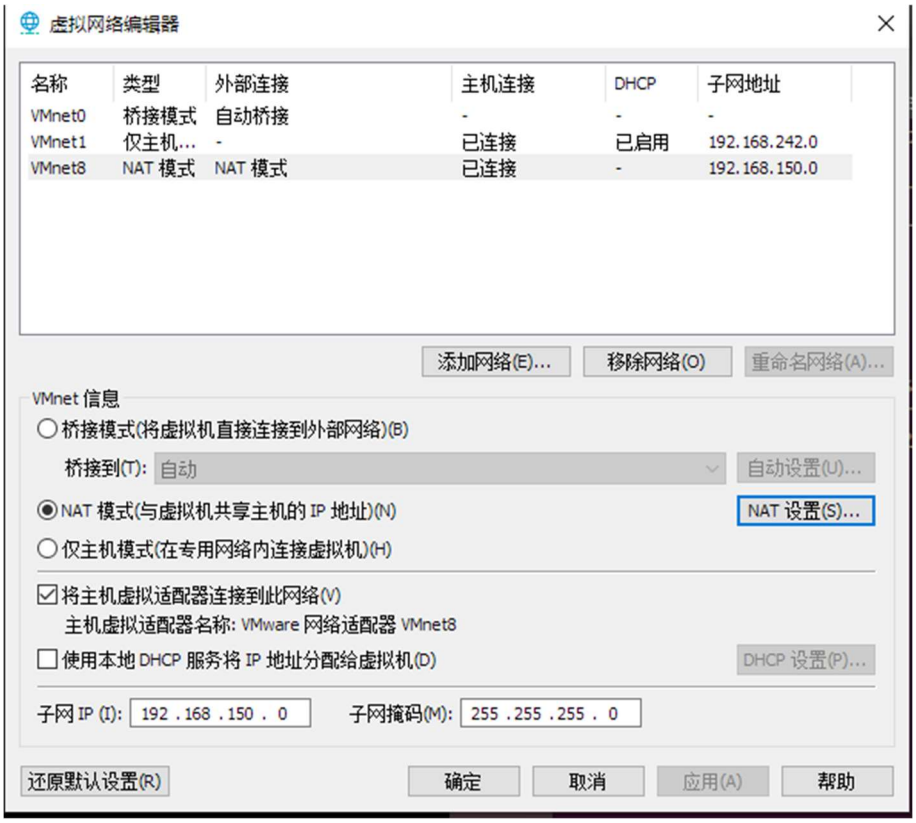
master 的配置比 slave 要稍微高一些

IP 设置

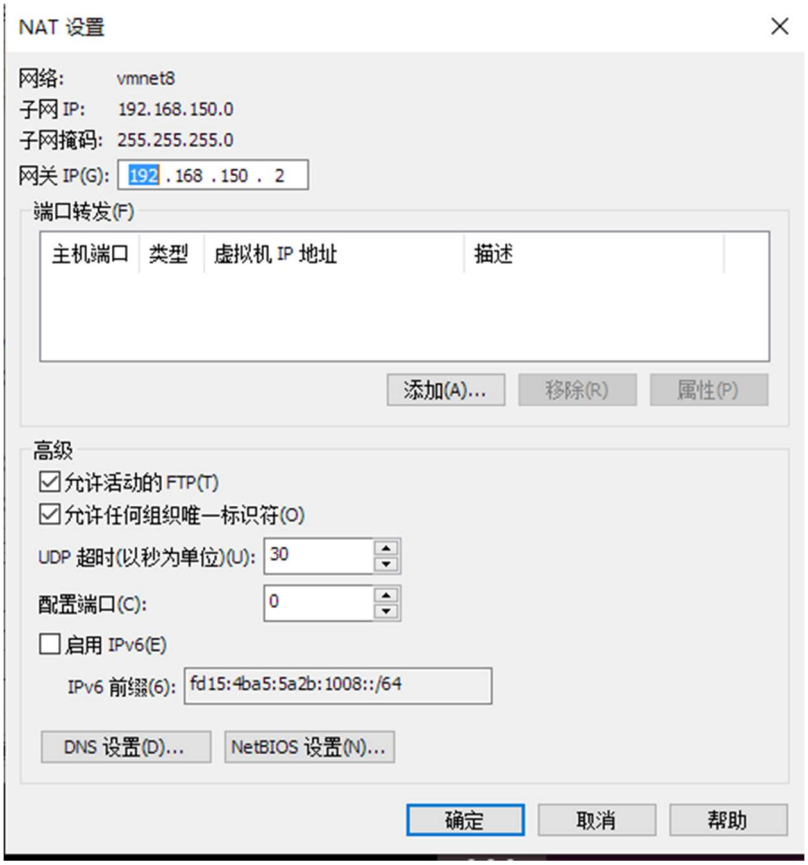
VMnet8 是 VM 配置的虚拟网卡

设置 VMnet8，选择「NAT 模式」

1） 编辑 > 虚拟网络编辑器。选择 VMnet8。配置子网 IP:192.168.150.0。去掉 Use local DHCP service to distribute IP address to VMs 选项。



2） 编辑 > 虚拟网络编辑器 > 选择 VMnet8 > Nat 设置。可查看网关的 IP。



Ubuntu 系统的登录和 IP 的固化

输入 ifconfig 回车查询当前的 ip 地址，准备将其长期绑定到当前虚拟机,为了方便后续实验。

master 的 ip：192.168.150.131

slave1 的 ip：192.168.150.129

slave2 的 ip：192.168.150.130

此时的 IP 地址是由 DHCP 服务器动态分配的，为了让这个 IP 地址能一直与这台虚拟机绑定，我们需要改变系统启动时获取 IP 的方式，从 DHCP 切换到静态 IP 地址，为此需要编辑 Linux 的网卡配置文件（/etc/network/interfaces），输入命令

sudo vi /etc/network/interfaces

回车，若能看到 eth0 的 IP 获取方式是 dhcp，则可在该文件中修改 ip 的获取方式从 DHCP 到 static，设置静态的 ip 地址、子网掩码和默认网关。

把这个文件更改为：

```
#This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
iface eth0 inet static
address 192.168.150.128
gateway 192.168.150.2
netmask 255.255.255.0
```

若 vim /etc/network/interfaces 修改网络配置时发现：
Ubuntu 18LTS ifupdown has been replaced by netplan(5) on this system。

```
# ifupdown has been replaced by netplan(5) on this system.    See
# /etc/netplan for current configuration.
# To re-enable ifupdown on this system, you can run:
#       sudo apt install ifupdown
```

则直接配置 netplan
vim /etc/netplan/50-cloud-init.yaml

配置如下：

```
network:
    ethernets:
        enp4s0:
            addresses: [192.168.150.131/24]    //IP 地址
            gateway4: 192.168.150.2    //  网关
            nameservers:
              addresses: [192.168.150.2] //DNS
            dhcp4: no
            optional: no
    version: 2
```

启用生效
sudo netplan apply

二、Hadoop 及相关环境的安装

开源分布式平台 Hadoop 可以聚合多个计算机形成集群，在各个节点上安装配置完 Hadoop 后可以直接提交分布式代码到集群计算。本次实验可以在个人电脑上用 VMware 完

成。

虚拟机 ip 配置测试

sudo vi /etc/hosts #编辑 /etc/hosts 文件，插入角色与 IP 映射

ping master -c 4 #尝试用角色名 ping 其它主机，一次 4 个包
hosts 文件修改为：

```
127.0.0.1           localhost

192.168.150.131 master
192.168.150.129 slave1
192.168.150.130 slave2

# The following lines are desirable for IPv6 capable hosts
::1         localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

三个虚拟机能够使用主机名（不是 ip）ping 通即配置正确

配置 SSH 无密码登录

保障了 Hadoop 可以通过角色名在局域网里找到各个节点，为了让 Hadoop 可以进一步读取、操作各个节点，需要赋予其登录的权限，意即让 Hadoop 拥有各个节点的普通用户账号，从而在需要操作各个节点时直接用对应的账号登录获取操作权限。SSH 协议可以为节点上的账户创建唯一的公私钥，然后利用这些公私钥实现无密码登录，从而让 Hadoop 直接绕开传统的账号密码登录过程，直接用公私钥访问节点。

生成各个节点的 SSH 公私钥：

```
cd ~/.ssh #  如果没有该目录，先执行一次 ssh localhost
rm ./id_rsa* #  删除之前生成的公匙（如果有）
ssh-keygen -t rsa #  一直按回车就可以
```

为了让每个节点都拥有其它节点的公钥，要先把所有公钥放进一个文件里

在 master 上，将 master 的公钥复制到 authorized_keys 文件里：
cat ./id_rsa.pub >> ./authorized_keys # cat 命令用于提取内容，>>输出重定向

将 slave1、slave2 的公钥文件发送给 master。Master 将接收到的 slave1 的公钥文件里的内容提取追加到 authorized_keys 文件里：

cat ~/id_rsa.pub >> ~/.ssh/authorized_keys

将 slave2 的公钥内容也放进 authorized_keys 文件，然后将 authorized_keys 文件分别发送到两个 slave 的~/.ssh/下：
设置 authorized_keys 的权限为 600

Sudo chmod 600 ~/.ssh/authorized_keys

搭建成功表现：每个节点尝试使用 ssh <角色名>的命令直接登录其它节点，直到每个节点都可成功免密码登录其它节点，则免密码登录配置成功！ 如在 master 上输入：ssh slave1 即可直接登陆 slave1 的虚拟机，不需要再输入密码登陆，便于后续实验操作。

安装 JDK

直接安装 openjdk1.8 即可
sudo apt install openjdk-8-jdk
sudo apt install openjdk-8-jdk-headless

安装 Hadoop

在各个节点上将 hadoop 解压到/usr/local/目录下，改变其所属用户和所属组（让 hadoop 软件登录时对 hadoop 文件夹拥有最高权限）：
此处用户组需要是设置 ssh 时使用的用户组

tar -zxvf hadoop-3.2.1.tar.gz -C /usr/local/
sudo mv /usr/loca/hadoop-3.2.1 /usr/local/hadoop #mv 实现重命名
sudo chown -R hadoop:hadoop /usr/local/Hadoop #此处用户以 hadoop 作示例，下同

将当前的 PATH 环境变量提取保存到.bashrc 和.profile 文件中

环境变量如下：

export HADOOP_HOME=/usr/local/hadoop
export HIVE_HOME=/usr/local/hive
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export JRE_HOME=${JAVA_HOME}/jre
export CLASSPATH=.:${JAVA_HOME}/lib:${JRE_HOME}/lib:$($HADOOP_HOME/bin/hadoop classpath):$CLASSPATH
export
PATH=$PATH:${JAVA_HOME}/bin:$HADOOP_HOME/bin:$HADOOP/sbin:$PATH:${HIVE_HOME}/bin

用 source 指令使环境变量生效：

source ~/.bashrc

查看 java 版本信息，如果出现版本信息则环境配置成功

java -version
javac -version

重点：hadoop 节点配置

修改 workers 文件，让 hadoop 知道自己可以聚合的节点名（保证与 hosts 里的角色名一致）

vi /usr/local/hadoop/etc/hadoop/workers
master
slave1
slave2

修改 core-site.xml 文件如下：
配置端口和路径

vi /usr/local/hadoop/etc/hadoop/core-site.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!-- Put site-specific property overrides in this file. -->

<configuration>
        <property>
                <name>fs.default.name</name>
                <value>hdfs://master:9000</value>
        </property>
        <property>
                <name>hadoop.tmp.dir</name>
                <value>/usr/local/hadoop/tmp</value>
        </property>
</configuration>
```

修改 hdfs-site.xml 文件如下（启用所有节点作为 DataNode，包括 master 故 replication_value=3）：
当节点增多时，需要更改配置文件，如主机名、IP 地址、节点信息等配置都要重新修改

vi /usr/local/hadoop/etc/hadoop/hdfs-site.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<configuration>
        <property>
                <name>dfs.replication</name>
                <value>3</value>
        </property>
        <property>
                <name>dfs.name.dir</name>
                <value>/usr/local/hadoop/hdfs/name</value>
        </property>
        <property>
                <name>dfs.data.dir</name>
                <value>/usr/local/hadoop/hdfs/data</value>
        </property>
</configuration>
```

修改 mapred-site.xml 文件如下:

vi /usr/local/hadoop/etc/hadoop/mapred-site.xml
yarn 为集群的表示

```
<configuration>

        <property>
                <name>mapreduce.framework.name</name>
                <value>yarn</value>
        </property>
</configuration>
```

修改 yarn-site.xml 文件如下（启用 yarn 资源管理器），为大数据计算分配计算、存储资源等

vi /usr/local/hadoop/etc/hadoop/yarn-site.xml

```
<configuration>

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

```
                <name>yarn.log-aggregation-enable</name>
                <value>true</value>
        </property>
</configuration>
```

修改 hadoop-env.sh 文件，将 25 行 JAVA_HOME 的值换成 jdk 所在的路径：

vi /usr/local/hadoop/etc/hadoop/hadoop-env.sh
当前 jdk 路径为：JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64

修改启动脚本，在文件开始空白处
在/usr/local/hadoop/sbin/start-dfs.sh 中：
(root 修改为 ssh 指定的用户名)
HDFS_DATANODE_USER=root
HADOOP_SECURE_DN_USER=hdfs
HDFS_NAMENODE_USER=root
HDFS_SECONDARYNAMENODE_USER=root
在/usr/local/hadoop/sbin/start-yarn.sh 中
(root 修改为 ssh 指定的用户名)
YARN_RESOURCEMANAGER_USER=root
HADOOP_SECURE_DN_USER=yarn
YARN_NODEMANAGER_USER=root

重点：hadoop 启动及验证

对 hadoop 进行 NameNode 的格式化：
/usr/local/hadoop/bin/hdfs namenode -format

启动 hdfs 和 yarn，并在各个节点上输入 jps 查看启动的服务：

只需在 master 上启动

/usr/local/hadoop/sbin/start-dfs.sh
/usr/local/hadoop/sbin/start-yarn.sh

或者
/usr/local/hadoop/sbin/start-all.sh

jps # 每个节点都查看一次

hadoop 搭建完成

参考链接：

1. https://www.jianshu.com/p/26c49450ba58
2. https://blog.csdn.net/weixx3/article/details/80782479
3. https://blog.csdn.net/CleverCode/article/details/50574695
4. https://blog.csdn.net/hongweigg/article/details/39995211
5. https://blog.csdn.net/qq_32635069/article/details/80859790