

四、 虚拟机创建配置及连接..... 18

(一) 前置安装及配置..... 19

(二) 虚拟机 master 创建 19

(三) 虚拟机配置 23

1. 虚拟机系统安装..... 23

2. IP 配置 30

(四) XShell 连接 33

1. 创建会话 33

2. 连接成功界面 35

3. 注意..... 35

五、 个人本地环境配置 37

(一) Hadoop 配置..... 37

1. 克隆节点配置修改..... 37

2. 配置无密码登录.....	37
3. 配置时间同步服务.....	39
4. 安装 jdk.....	39
5. Hadoop3.14 配置.....	40
(二) Hive 安装.....	45
1. MySQL 安装.....	45
2. hive 配置.....	47
(三) 配置 Zookeeper.....	48
1. 解压缩.....	48
2. 进入安装路径.....	48
3. 复制 zoo_sample.cfg 重命名为 zoo.cfg.....	48
4. 各个子节点新建文件夹.....	48
5. 在 slave1 节点执行.....	48
6. 在各子节点的/etc/profile 中配置环境变量.....	49
7. 启动各个子节点 Zookeeper.....	49
8. 查看各个子节点的 zookeeper 是否启动.....	49
(四) 配置 HBase.....	49
1. 解压缩.....	49
2. 进入目录.....	49
3. 修改 hbase-site.xml 文件.....	49
4. 配置 hbase-env.sh.....	50
5. 配置 regionservers.....	50
6. 拷贝到各子节点.....	50
7. 配置环境变量.....	50
8. 主节点启动.....	50
9. 在浏览器查看.....	51
(五) 配置 Scala.....	51
1. 解压 scala-2.11.12 安装包.....	51
2. 修改解压后文件名.....	51

3. 修改环境变量	51
(六) 配置 Spark	51
1. 解压 spark-2.4.7-bin-hadoop2.7.tgz 安装包	51
2. 修改解压后 spark 文件名	51
3. 修改 spark-env.sh	51
4. 修改 spark-defaults.conf	52
5. 修改 slaves	52
6. 配置环境变量	52
7. 启动 saprk	52
8. 在浏览器查看	52
(七) 配置 Flume	53
1. 解压 apache-flume-1.9.0-bin.tar.gz 文件	53
2. 修改文件名	53
3. 修改 flume-env.sh 文件	53
(八) 配置 Kafka	53
1. 解压 Kafka 安装包	53
2. 配置环境变量	53
3. 修改 server.properties	53
4. 创建目录	53
5. 发送至其余节点	54
6. 修改子节点的 server.properties	54
7. 启动 Kafka	54
(九) 配置 Flink	54
1. 解压 flink-1.10.1-bin-scala_2.11.tgz 安装包	54
2. 配置环境变量	54
3. 修改配置文件\$FLINK_HOME/conf/flink-conf.yaml	54
4. 修改配置文件/conf/slaves	55
5. 修改配置文件/conf/masters	55
6. 拷贝安装文件到集群 slave 节点	55

一、简介

(一) 集群

配置的集群架构如下所示。

序号	主机名	IP
1	master	192.168.128.130
2	slave1	192.168.128.131
3	slave2	192.168.128.132
4	slave3	192.168.128.133

(二) 软件

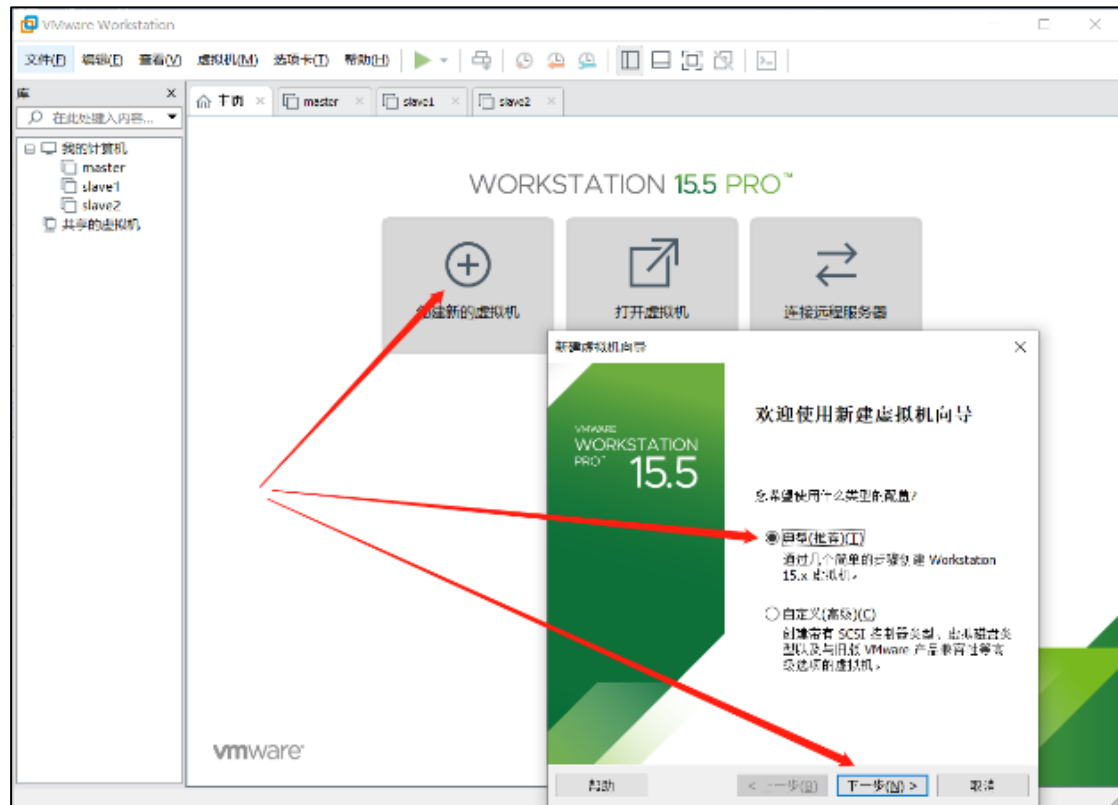
大数据实验环境涉及的相关软件如下所示。

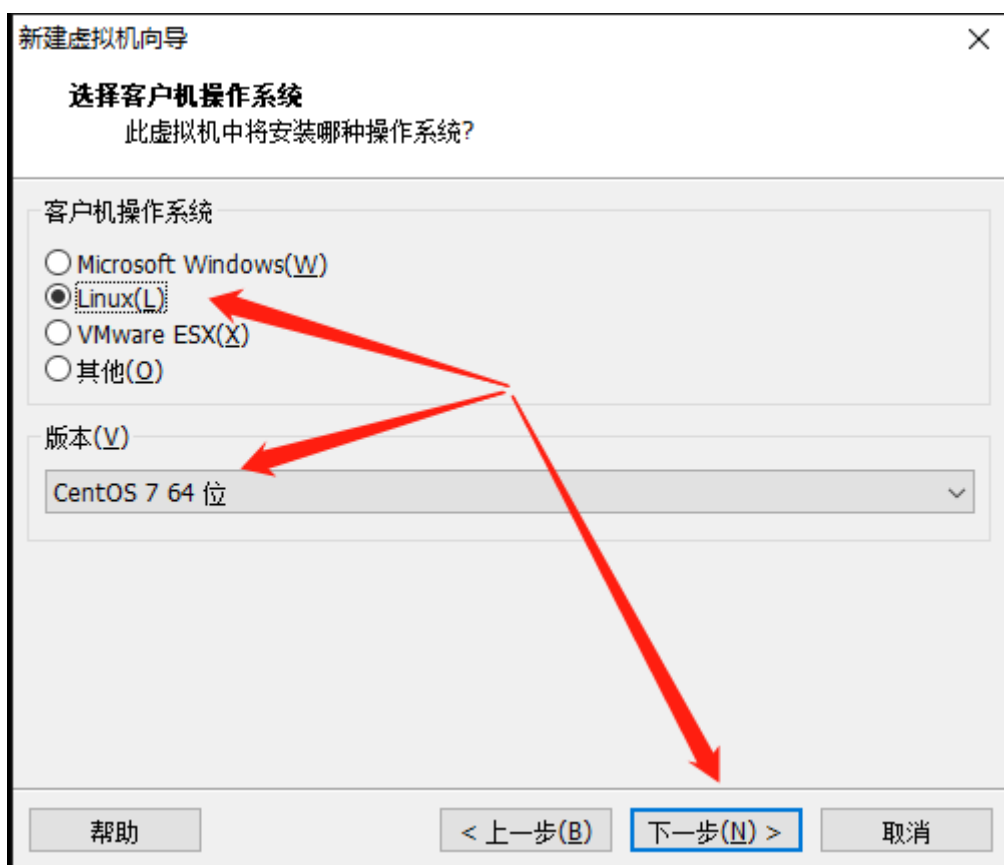
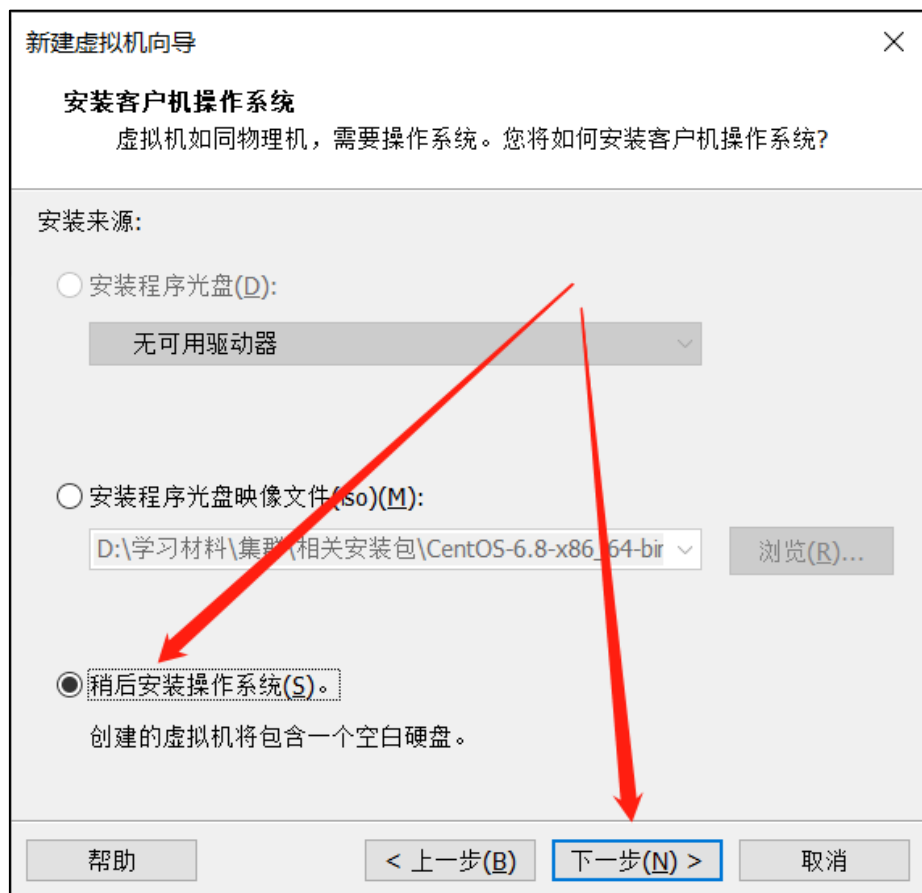
序号	软件	软件安装包名
1	VMware	VMware-workstation-full-16.1.0-17198959.exe
2	Centos	CentOS-7-x86_64-DVD-2009.iso
3	Xme	Xme5.exe
4	jdk	jdk-8u281-linux-x64.rpm
5	Hadoop	hadoop-3.1.4.tar.gz
6	Hive	apache-hive-2.3.8-bin.tar.gz
7	mysql 驱动	mysql-connector-java-8.0.20.jar
8	Zookeeper	zookeeper-3.4.6.tar.gz
9	HBase	hbase-2.2.6-bin.tar.gz
10	Spark	spark-2.4.7-bin-hadoop2.7.tgz
11	Scala	scala-2.11.12.tgz
12	Flume	apache-flume-1.9.0-bin.tar.gz
13	Kafka	kafka_2.13-2.4.0.tgz
14	Flink	flink-1.10.1-bin-scala_2.11.tgz

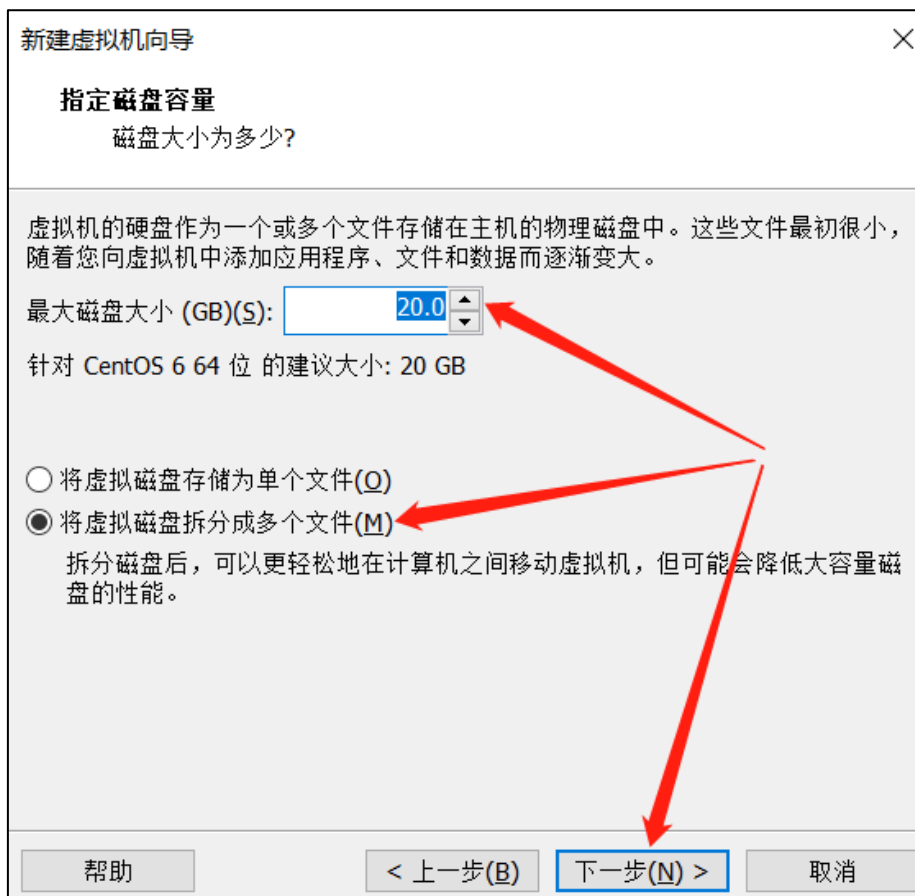
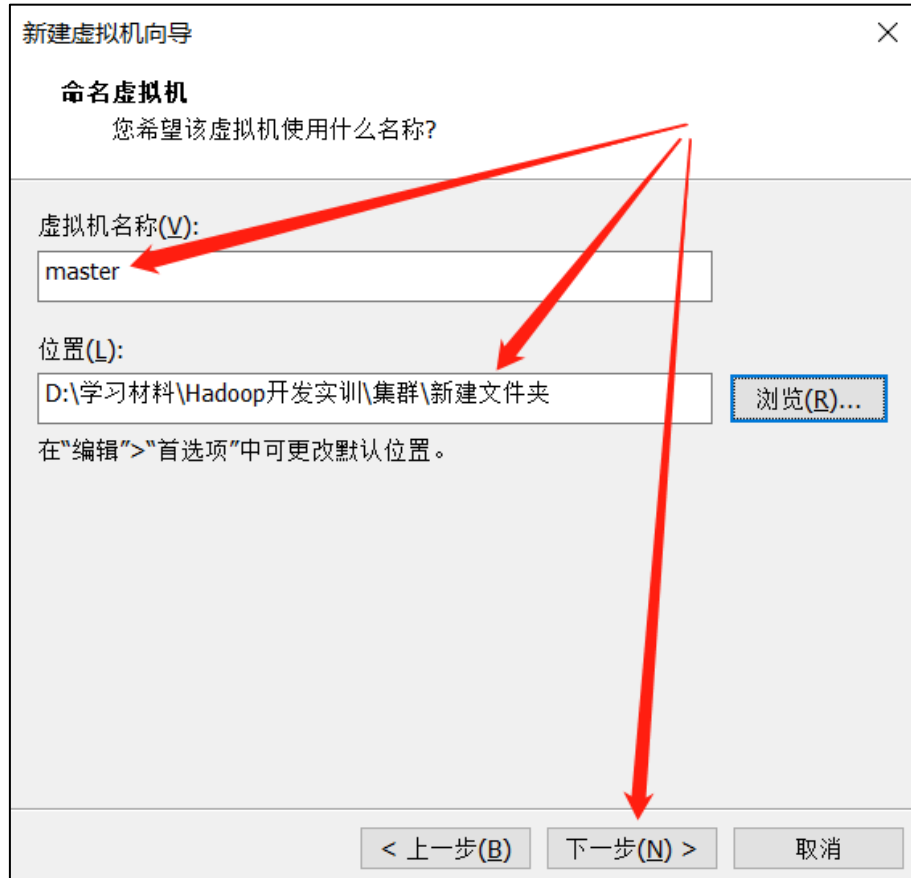
(一) 前置安装及配置

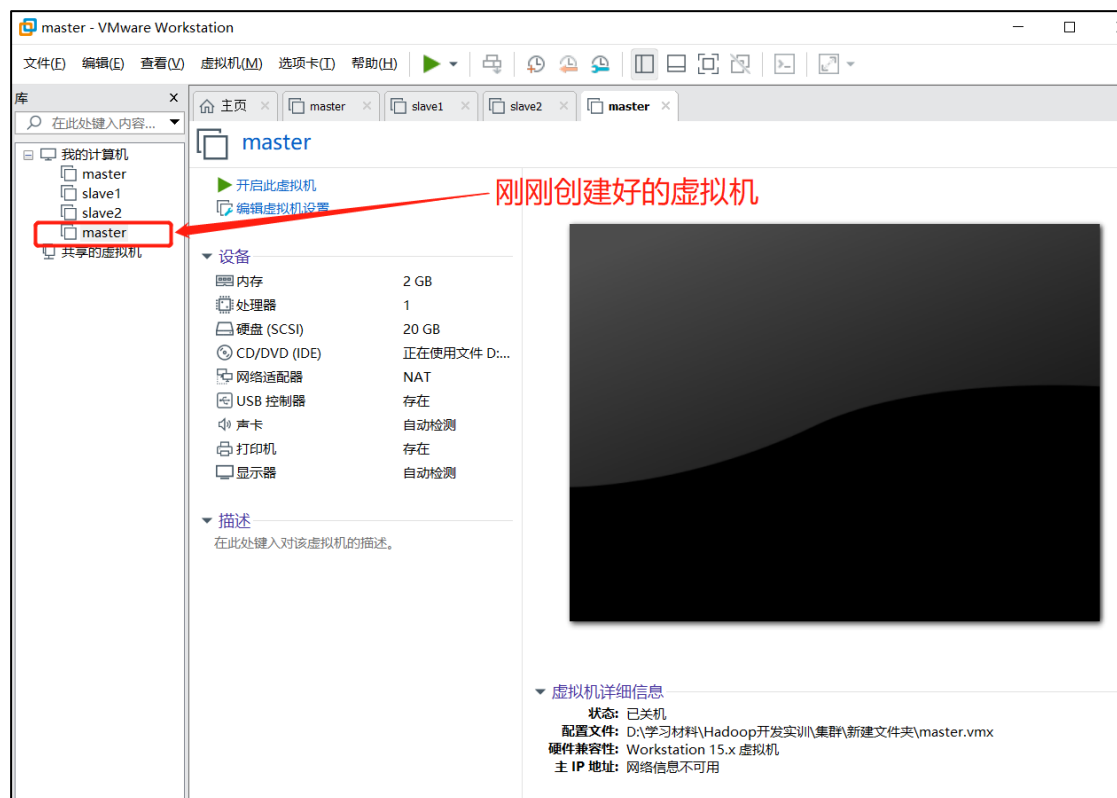
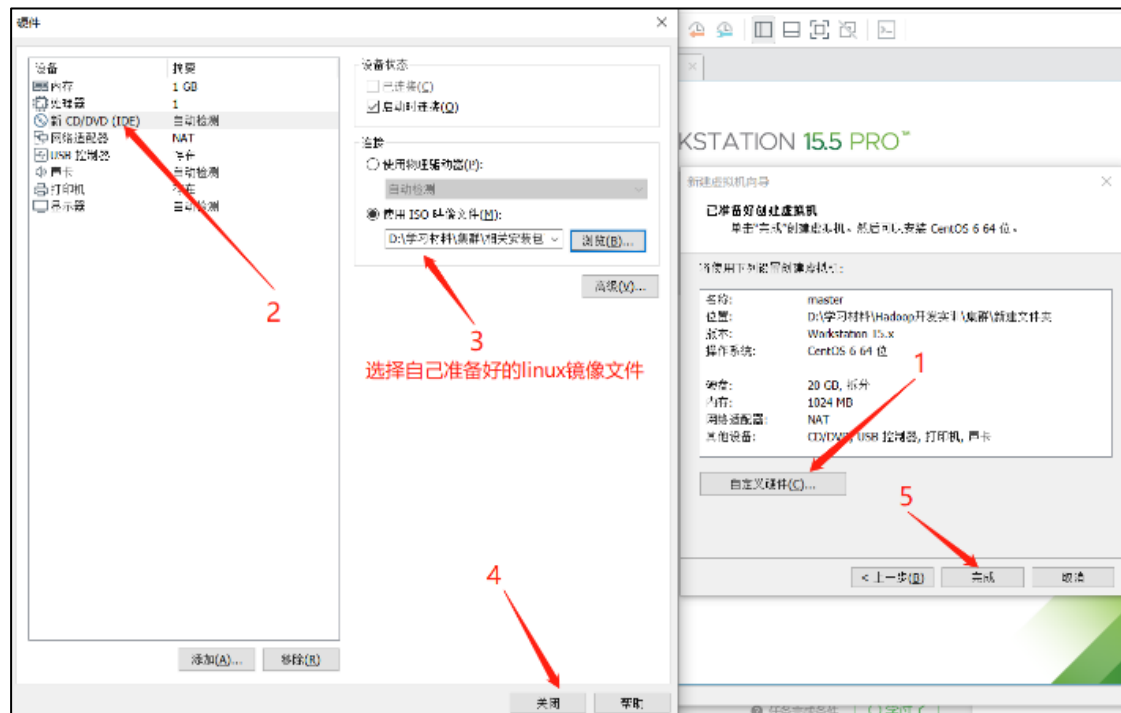
- ① 本地电脑需安装好 VMware，版本须为 11 或者以上。
- ② 本地电脑的环境变量中应先配置好 Java 环境变量。

(二) 虚拟机 master 创建









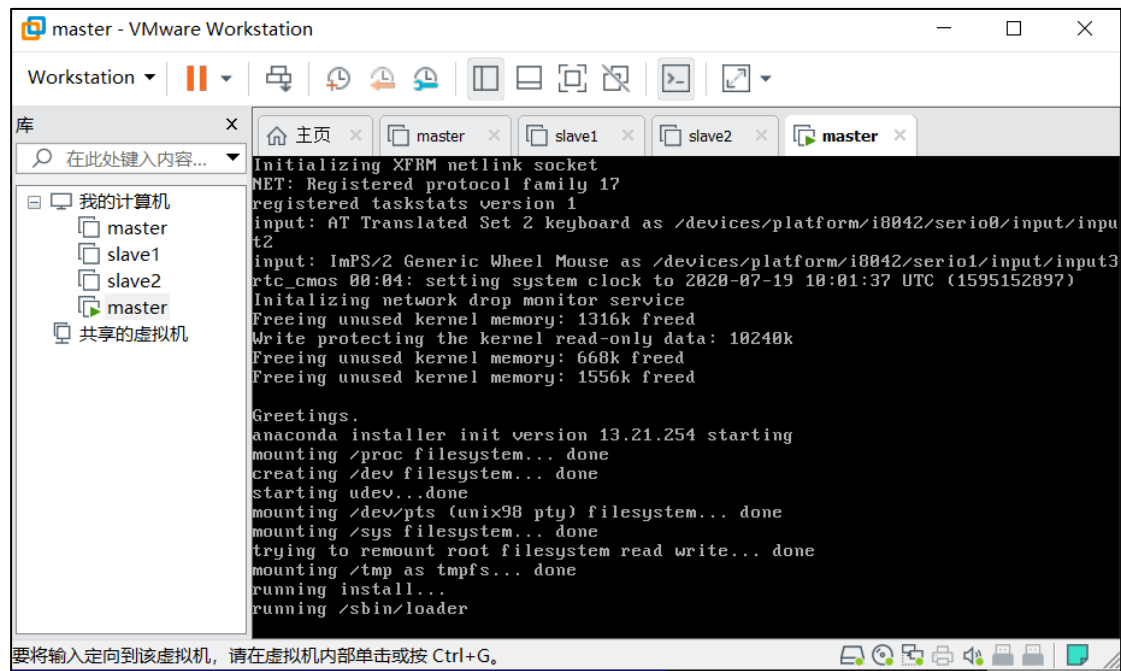
(三) 虚拟机配置

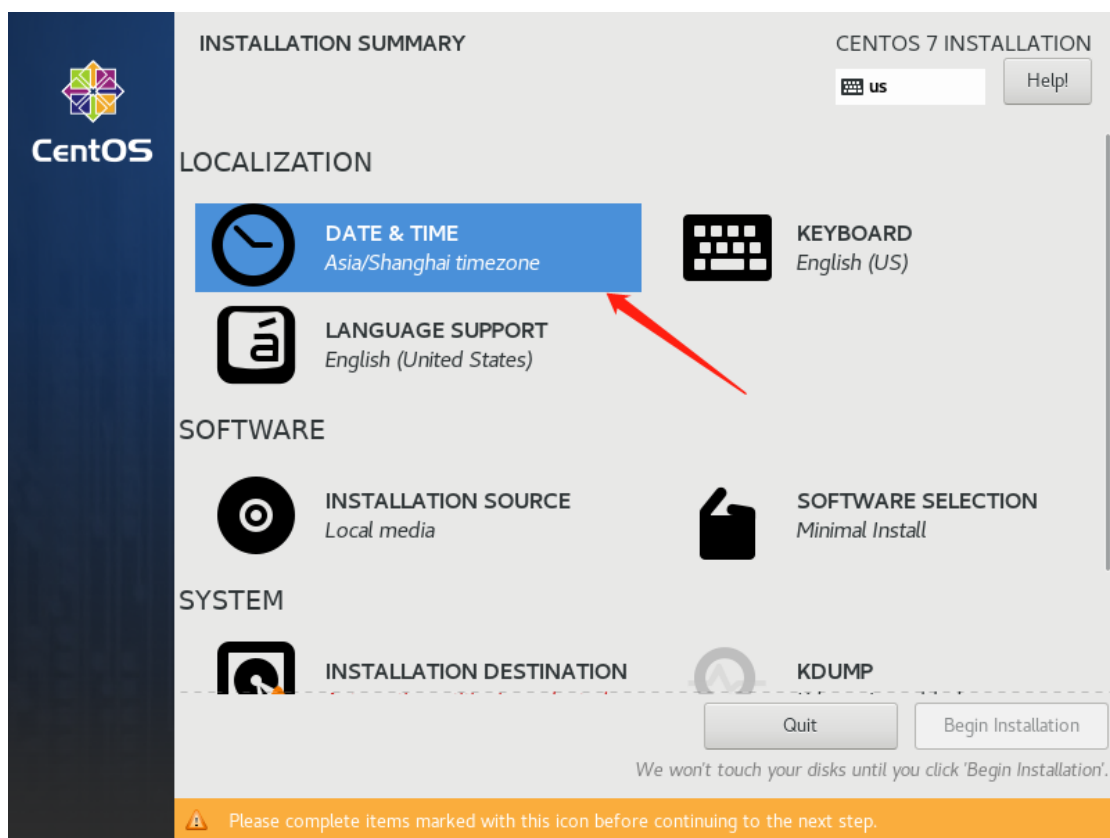
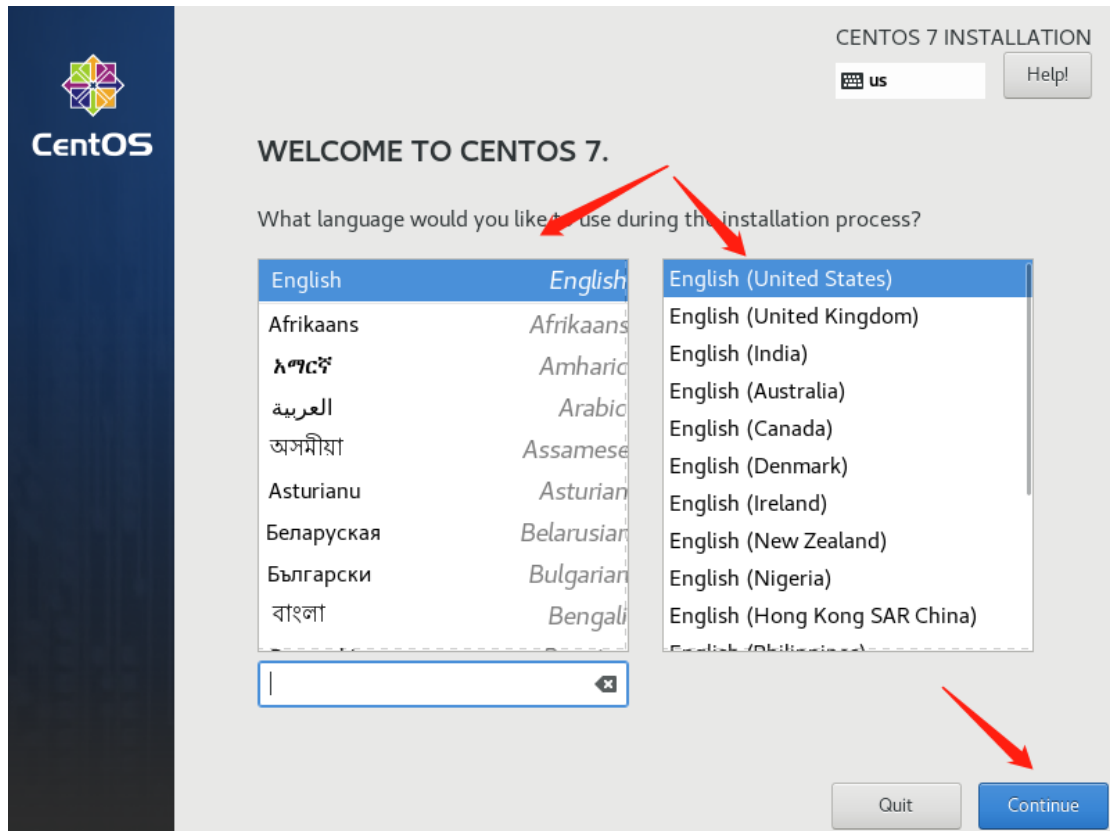
1. 虚拟机系统安装



- Press the <ENTER> key to begin the installation process.

按Enter键





DATE & TIME


Done

CENTOS 7 INSTALLATION

us

Help!

Region: AsiaCity: ShanghaiNetwork TimeOFF





11:43 PM

24-hour

AM/PM

05 / 13 / 2021

 You need to set up networking first if you want to use NTP



CentOS


INSTALLATION SUMMARY


CENTOS 7 INSTALLATION

us


Help!


DATE & TIME
Asia/Shanghai timezone

KEYBOARD
English (US)


LANGUAGE SUPPORT
English (United States)


SOFTWARE


INSTALLATION SOURCE
Local media


SOFTWARE SELECTION
Minimal Install

SYSTEM

INSTALLATION DESTINATION
Automatic partitioning selected

KDUMP
Kdump is enabled

NETWORK & HOST NAME
Not connected

SECURITY POLICY
No content found

Quit

Begin Installation

We won't touch your disks until you click 'Begin Installation'.

INSTALLATION DESTINATION

CENTOS 7 INSTALLATION

Done

us


Help!

Device Selection

Select the device(s) you'd like to install to. They will be left untouched until you click on the main menu's "Begin Installation" button.

Local Standard Disks

10 GiB



VMware, VMware Virtual S

sda / 10 GiB free

Disks left unselected here will not be touched.

Specialized & Network Disks

Add a disk...

Disks left unselected here will not be touched.

Other Storage Options


Partitioning

☒ Automatically configure partitioning. ☐ I will configure partitioning.

☐ I would like to make additional space available.

[Full disk summary and boot loader...](#)

1 disk selected; 10 GiB capacity; 10 GiB free [Refresh...](#)




CentOS

INSTALLATION SUMMARY

CENTOS 7 INSTALLATION


us

Help!




DATE & TIME

Asia/Shanghai timezone



KEYBOARD


English (US)



LANGUAGE SUPPORT


English (United States)

SOFTWARE



INSTALLATION SOURCE


Local media



SOFTWARE SELECTION


Minimal Install

SYSTEM




INSTALLATION DESTINATION

Automatic partitioning selected




KDUMP

Kdump is enabled



NETWORK & HOST NAME

Not connected



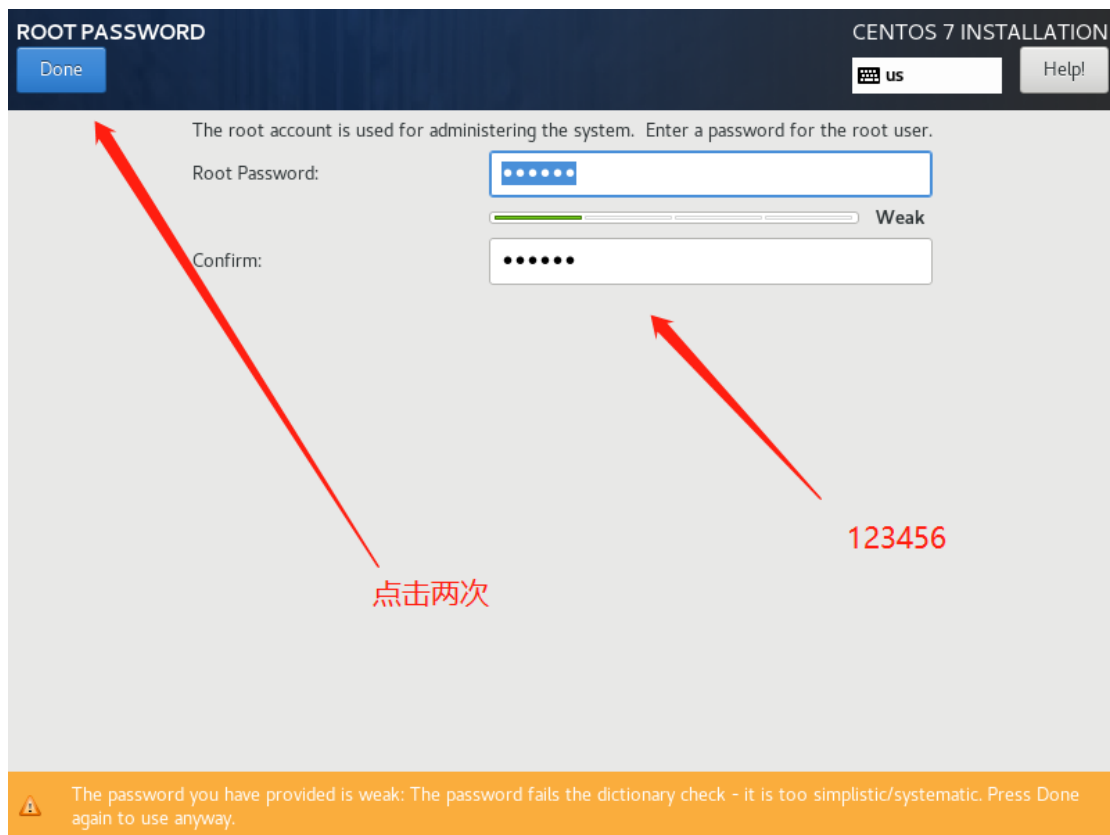
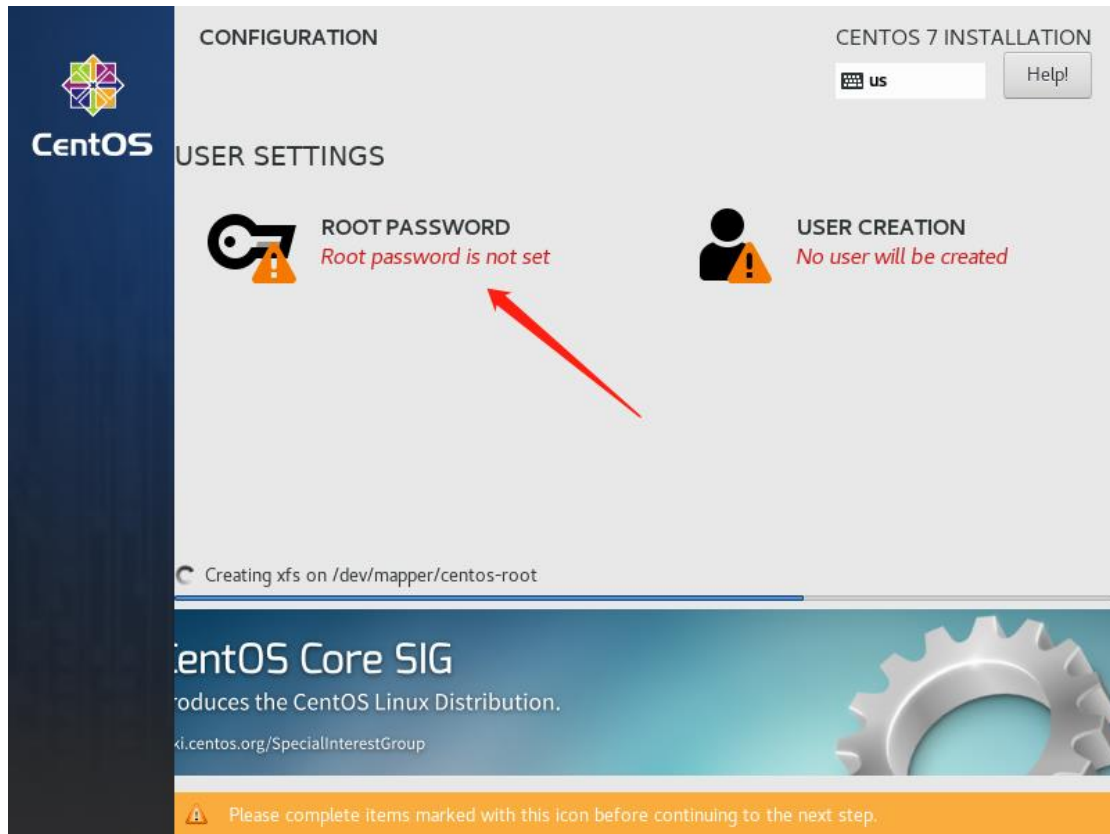
SECURITY POLICY

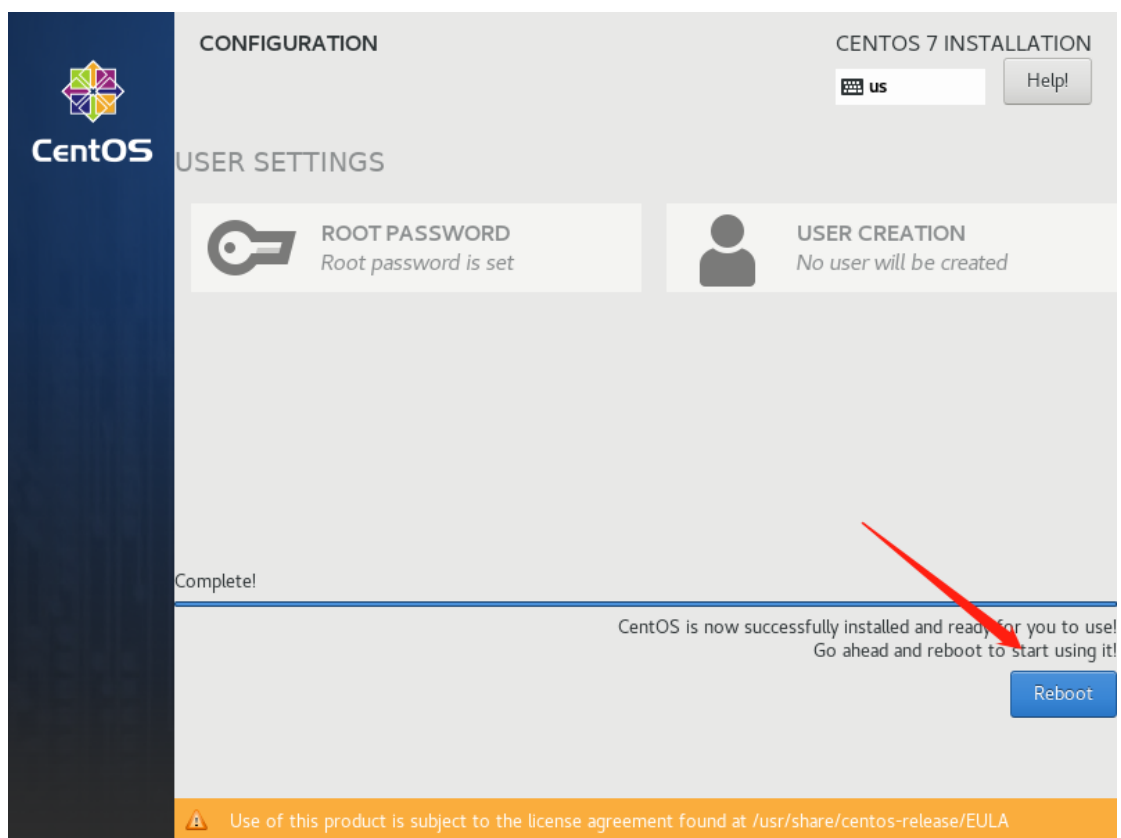
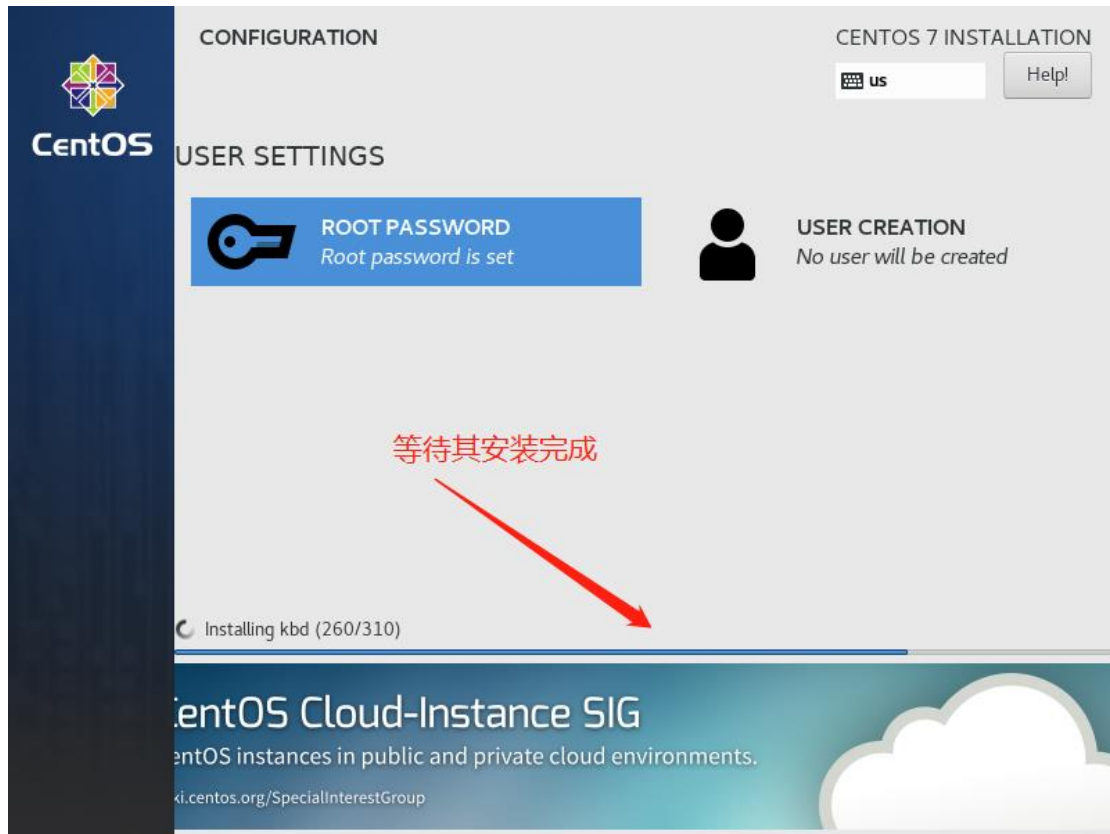
No content found

Quit

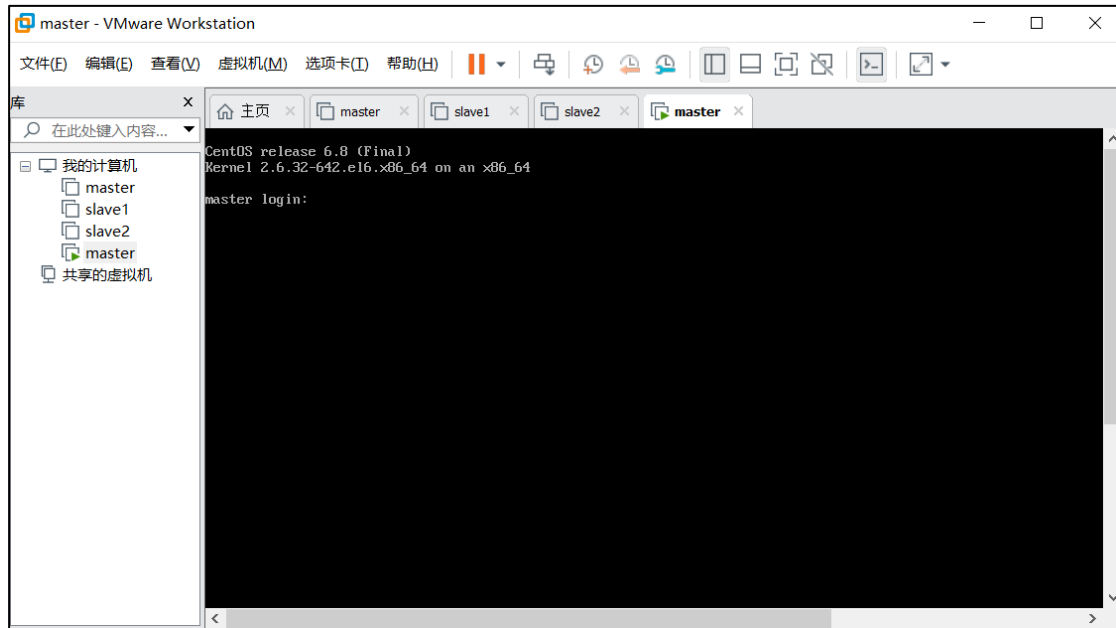
Begin Installation

We won't touch your disks until you click 'Begin Installation'.





输入用户名、密码进行登录



2. IP 配置

为了能用 xshell 连接主机，同时也为了集群中的节点可以相互通信，接下来对节点进行 IP 设置。

(1) 重启网络服务

进入刚刚创建好的虚拟机，输入：service network restart

```
CentOS release 6.8 (Final)
Kernel 2.6.32-642.el6.x86_64 on an x86_64

master login: root
Password:
Last login: Sun Jul 19 18:33:07 on tty1
[root@master ~]# service network restart
Shutting down loopback interface: [ OK ]
Bringing up loopback interface: [ OK ]
[root@master ~]# _
```

(2) 修改配置文件中的 IP 设置

通过命令 vi /etc/sysconfig/network-scripts/ifcfg-ens33 进入配置文件，并修改成如下形式：

```
#修改：
ONBOOT=yes
NM_CONTROLLED=yes
BOOTPROTO=static
#添加以下内容
IPADDR=192.168.128.130
```

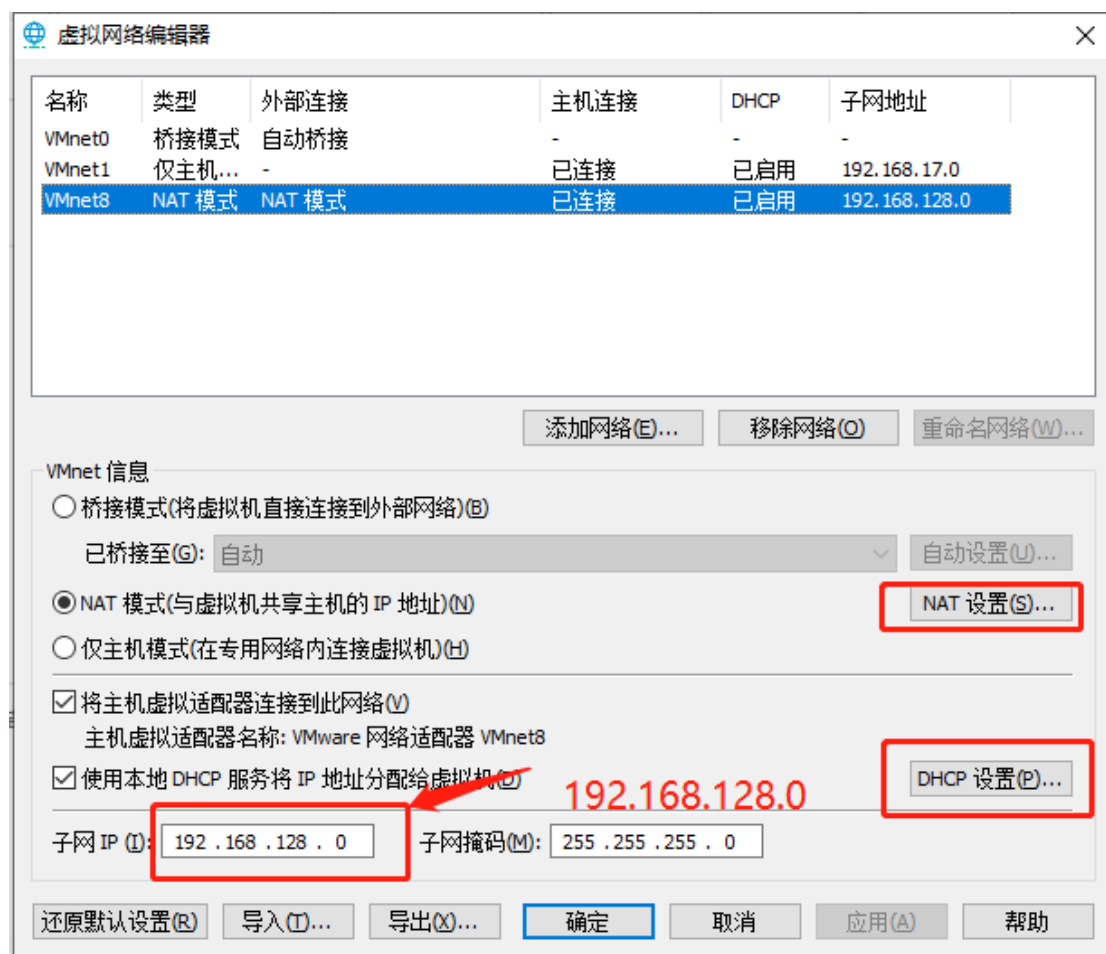


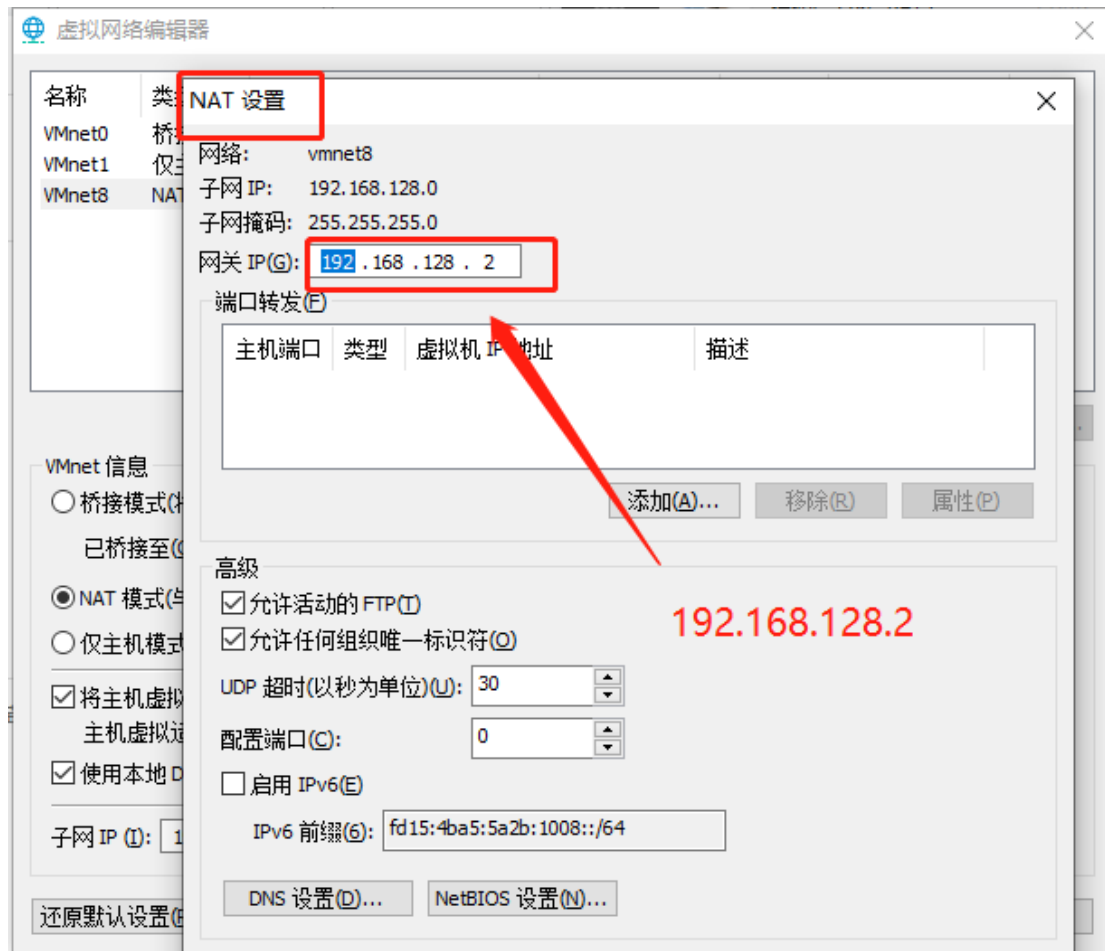
```
NETMASK=255.255.255.0
GATEWAY=192.168.128.2
DNS1=192.168.128.2
```

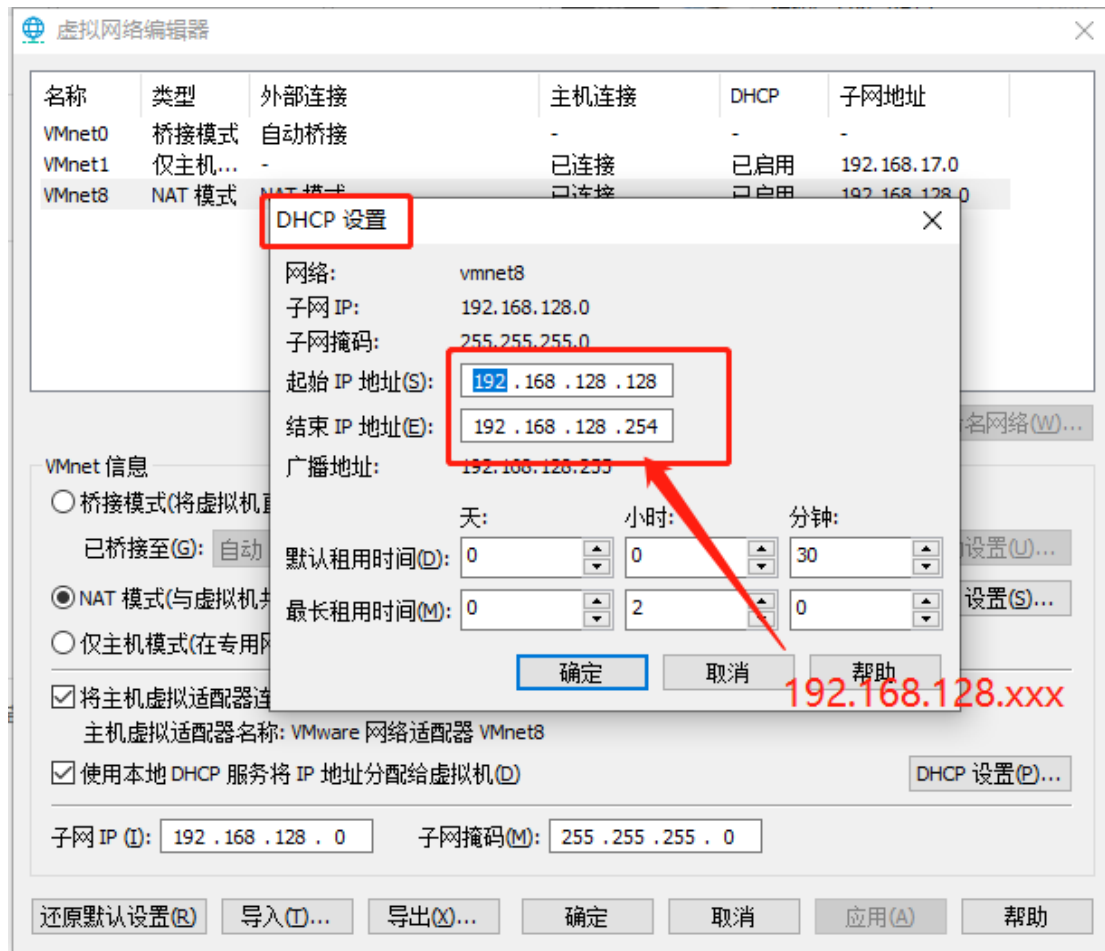
(3) 重启网络服务

输入“service network restart”，支持 IP 设置完成，接下来就可以使用 xshell 连接该节点。

点击编辑 VM 窗口栏处的“编辑”->“编辑虚拟网络”，按照下图对编辑虚拟网络进行修改

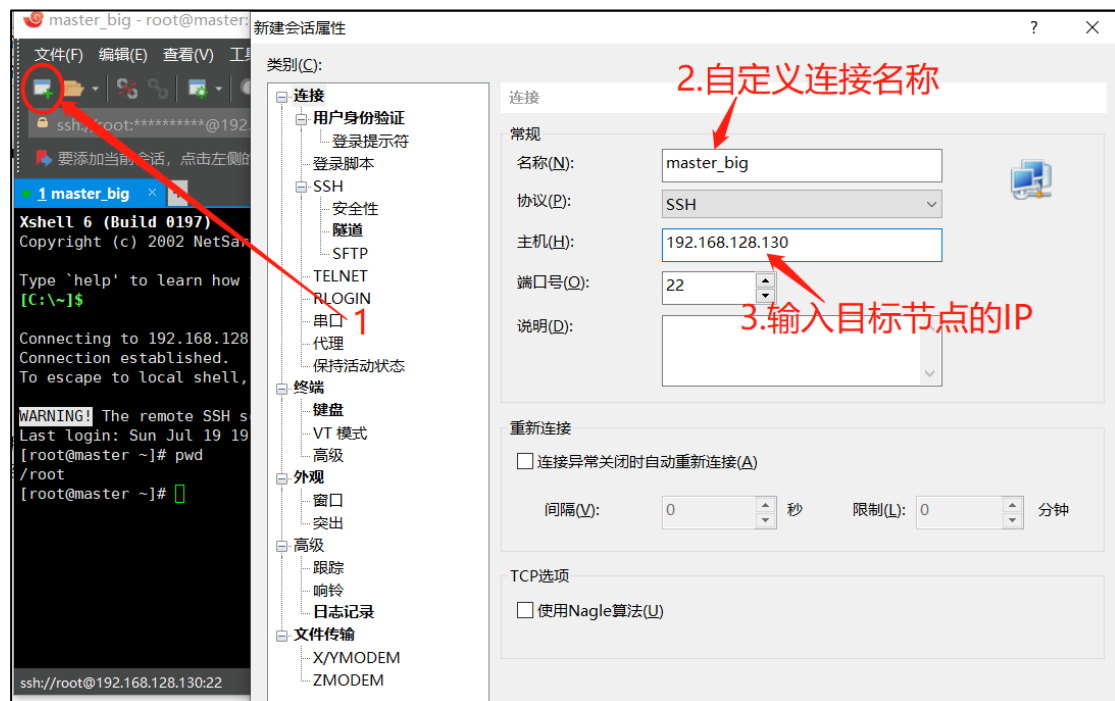


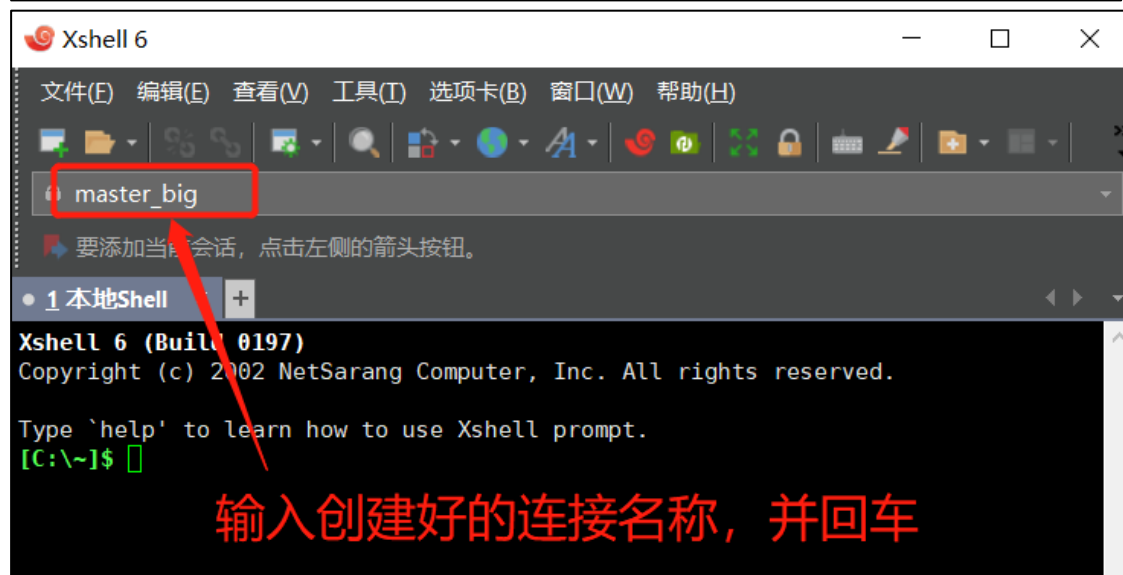
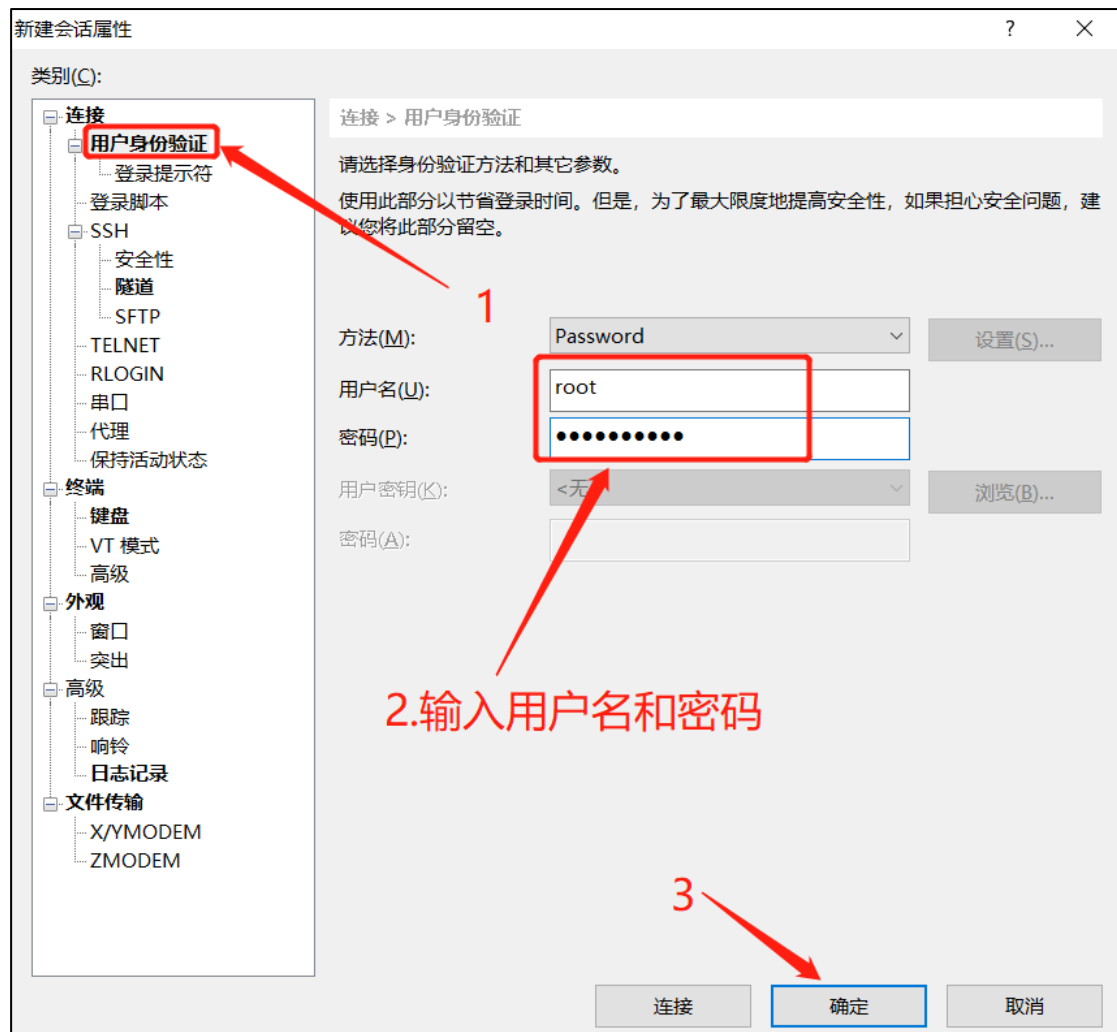




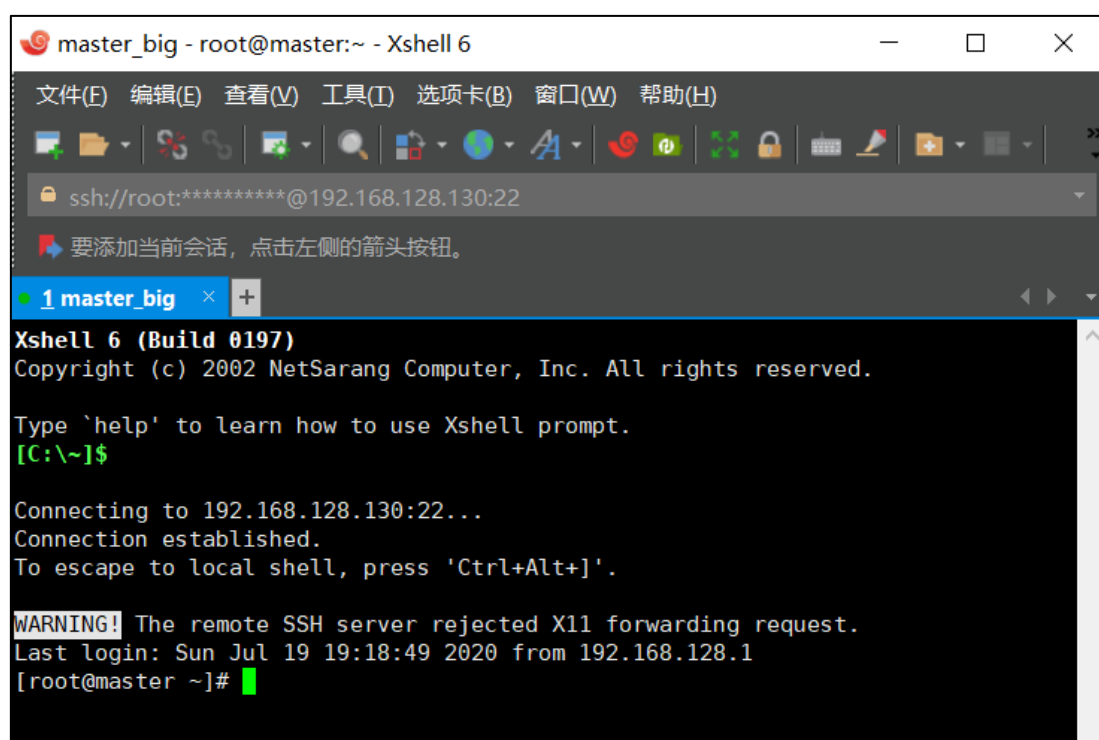
(四) XShell 连接

1. 创建会话



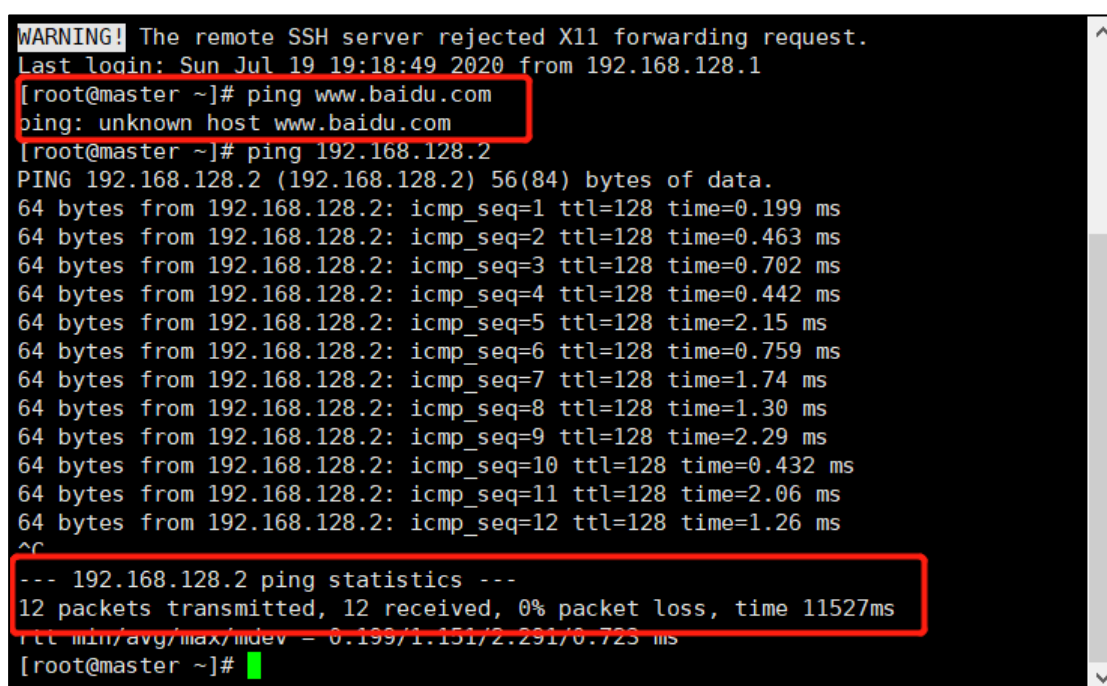


2. 连接成功界面



3. 注意

此时有一点需要注意，当前节点能 ping 通网关，当 ping 不通往外域名，如下图所示。



此时需修改域名解析服务器配置

输入 `vi /etc/resolv.conf`，新增一条信息如下：

```
nameserver 192.168.128.2
nameserver 8.8.8.8
```

```
~
~
~
~
```

然后 source 一下，再去 ping 百度则正常了。

```
[root@master ~]# source /etc/resolv.conf
-bash: nameserver: command not found
-bash: nameserver: command not found
[root@master ~]# ping www.baidu.com
PING www.a.shifen.com (183.232.231.174) 56(84) bytes of data.
64 bytes from 183.232.231.174: icmp_seq=1 ttl=128 time=12.2 ms
64 bytes from 183.232.231.174: icmp_seq=2 ttl=128 time=12.4 ms
64 bytes from 183.232.231.174: icmp_seq=3 ttl=128 time=12.7 ms
64 bytes from 183.232.231.174: icmp_seq=4 ttl=128 time=13.0 ms
64 bytes from 183.232.231.174: icmp_seq=5 ttl=128 time=12.9 ms
64 bytes from 183.232.231.174: icmp_seq=6 ttl=128 time=12.3 ms
64 bytes from 183.232.231.174: icmp_seq=7 ttl=128 time=12.2 ms
^C
--- www.a.shifen.com ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6452ms
rtt min/avg/max/mdev = 12.236/12.575/13.024/0.314 ms
[root@master ~]#
```

为了方便后续克隆操作，为该虚拟机安装一些软件

`yum -y install ntp openssh-clients openssh-server vim`

```
[root@master ~]# yum -y install ntp openssh-clients openssh-server vim
Loaded plugins: fastestmirror
Setting up Install Process
base | 3.7 kB | 00:00
base/primary_db 69% [=====] 558 kB/s | 3.2 MB | 00:02 ETA
ssh://root@192.168.128.130:22  SSH2  xterm  78x23  23,1  1 会话  CAP  NUM
```

安装完成

```
Dependency Installed:
gpm-libs.x86_64 0:1.20.6-12.el6
libedit.x86_64 0:2.11-4.20080712cvs.1.el6
ntpdate.x86_64 0:4.2.6p5-15.el6.centos
perl.x86_64 4:5.10.1-144.el6
perl-Module-Pluggable.x86_64 1:3.90-144.el6
perl-Pod-Escapes.x86_64 1:1.04-144.el6
perl-Pod-Simple.x86_64 1:3.13-144.el6
perl-libs.x86_64 4:5.10.1-144.el6
perl-version.x86_64 3:0.77-144.el6
vim-common.x86_64 2:7.4.629-5.el6_10.2
vim-filesystem.x86_64 2:7.4.629-5.el6_10.2

Updated:
openssh-server.x86_64 0:5.3p1-124.el6_10

Dependency Updated:
openssh.x86_64 0:5.3p1-124.el6_10

Complete!
[root@master ~]#
```

五、个人本地环境配置

该部分内容用于指导学员在自备电脑上进行大数据实验环境的集群搭建及配置。

(一) Hadoop 配置

1. 克隆节点配置修改

在 VMware 界面右击虚拟机 master--管理--克隆，每一个从 master 克隆后的节点都需要进行以下操作：

- (1) 开启虚拟机，输入命令 “ip addr”；
- (2) 修改 “/etc/sysconfig/network-scripts/ifcfg-ens33”，修改 IPADDR 后面的 IP；
- (3) 修改 “/etc/hostname”，修改主机名（主节点为 master，子节点分别为 slave1、slave2，slave3）；
- (4) 输入命令 “service network restart”；
- (5) 使用 “ip addr”，查看 IP、主机名是否修改；
- (6) reboot 重启虚拟机。

2. 配置无密码登录

后续操作都是连接 XShell 后在 XShell 界面操作。

(1) 配置 IP 映射（每个节点都需修改）

在 /etc/hosts 添加

192.168.128.130 master

```
192.168.128.131 slave1
192.168.128.132 slave2
192.168.128.133 slave3
```

(2) 配置 SSH 无密码登录

a) 使用 ssh-keygen 产生公钥与私钥对。输入命令 “ssh-keygen -t rsa”，接着按三次 Enter 键。执行后出现如下输出。

```
[root@master ~]# ssh-keygen -t rsa
```

```
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Created directory '/root/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
a6:13:5a:7b:54:eb:77:58:bd:56:ef:d0:64:90:66:d4 root@master.centos.com
The key's randomart image is:
+--[ RSA 2048]-----+
|           .. |
|          .E|
|         .  = |
|        .. o o |
|       o S .  .|=|
|      o * .   o ++|
|     . + . . o ooo|
|        o    ..o |
|                   .|
+-----+
```

生成私有密钥 id_rsa 和公有密钥 id_rsa.pub 两个文件。ssh-keygen 用来生成 RSA 类型的密钥以及管理该密钥，参数 “-t” 用于指定要创建的 SSH 密钥的类型为 RSA。

b) 用 ssh-copy-id 将公钥复制到远程机器中

```
ssh-copy-id -i /root/.ssh/id_rsa.pub master//依次输入 yes,123456 (root 用户的密码)
```

```
ssh-copy-id -i /root/.ssh/id_rsa.pub slave1
```

```
ssh-copy-id -i /root/.ssh/id_rsa.pub slave2
```

```
ssh-copy-id -i /root/.ssh/id_rsa.pub slave3
```

c) 验证是否设置无密码登录

依次输入

```
ssh slave1
```

```
exit;
```

```
ssh slave2
```

```
exit;
```

```
ssh slave3
```

```
exit;
```

3. 配置时间同步服务

(1) 安装 NTP 服务。在各节点：

```
yum -y install ntp
```

(2) 修改设置

假设 master 节点为 NTP 服务主节点,那么其配置如下。使用命令“vim /etc/ntp.conf”打开/etc/ntp.conf 文件,注释掉以 server 开头的行,并添加:

```
restrict 192.168.128.2 mask 255.255.255.0 nomodify notrap
server 127.127.1.0
fudge 127.127.1.0 stratum 10
```

(3) 在 slave 中配置 NTP

同样修改/etc/ntp.conf 文件,注释掉 server 开头的行,并添加:

```
server master
```

(4) 关闭防火墙

执行命令“systemctl stop firewalld.service & systemctl disable firewalld.service”

永久性关闭防火墙,主节点和从节点都要关闭。

(5) 启动 NTP 服务

① 在 master 节点执行命令 “service ntpd start & chkconfig ntpd on”

② 在 slave 上执行命令 “ntpdate master” 即可同步时间

③ 在 slave 上分别执行 “service ntpd start & chkconfig ntpd on” 即可启动并永久启动 NTP 服务。

4. 安装 jdk (每个节点都需安装)

(1) 上传并安装

上传 JDK 安装包到虚拟机/opt 目录,进入/opt 目录,执行命令 “rpm -ivh jdk-8u281-linux-x64.rpm” 安装 JDK。

(2) 修改环境变量

在/etc/profile 添加如下内容。

```
export JAVA_HOME=/usr/java/jdk1.8.0_281-amd64
export PATH=$PATH:$JAVA_HOME/bin
```

执行【source /etc/profile】使配置生效

(3) 验证

验证 JDK 是否配置成功，执行命令 “java -version”

5. Hadoop3.1.4 配置（主节点进行）

(1) 上传

通过 xmanager 的 Xftp 上传 hadoop-3.1.4.tar.gz 文件到/opt 目录

(2) 解压缩 hadoop-3.1.4.tar.gz 文件

```
tar -zxf hadoop-3.1.4.tar.gz -C /usr/local
```

解压后即可看到/usr/local/hadoop-3.1.4 文件夹。

(3) 配置 Hadoop

① 进入目录

```
cd /usr/local/hadoop-3.1.4/etc/hadoop/
```

依次修改以下配置文件（使用 vi 命令对文件进行修改 如 vi core-site.xml）。

注：文件中存在<configuration></configuration>，配置放入其中，且只保留一组的<configuration></configuration>。

② core-site.xml

```
<configuration>
  <property>
    <name>fs.defaultFS</name>
    <value>hdfs://master:8020</value>
  </property>
  <property>
    <name>hadoop.tmp.dir</name>
    <value>/var/log/hadoop/tmp</value>
  </property>
</configuration>
```

③ hadoop-env.sh

```
export JAVA_HOME=/usr/java/jdk1.8.0_281-amd64
```

④ hdfs-site.xml

```
<configuration>
<property>
```

```
<name>dfs.namenode.name.dir</name>
<value>file:///data/hadoop/hdfs/name</value>
</property>
<property>
  <name>dfs.datanode.data.dir</name>
  <value>file:///data/hadoop/hdfs/data</value>
</property>
<property>
  <name>dfs.namenode.secondary.http-address</name>
  <value>master:50090</value>
</property>
<property>
  <name>dfs.replication</name>
  <value>3</value>
</property>
</configuration>
```

⑤ mapred-site.xml

复制 `cp mapred-site.xml.template mapred-site.xml`（若无 `mapred-site.xml.template` 文件，则直接修改 `mapred-site.xml`）

```
<configuration>
<property>
  <name>mapreduce.framework.name</name>
  <value>yarn</value>
</property>
<!-- jobhistory properties -->
<property>
  <name>mapreduce.jobhistory.address</name>
  <value>master:10020</value>
</property>
<property>
  <name>mapreduce.jobhistory.webapp.address</name>
  <value>master:19888</value>
</property>
</configuration>
```

⑥ yarn-site.xml

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

```
<value>${yarn.resourcemanager.hostname}:8032</value>
</property>
<property>
  <name>yarn.resourcemanager.scheduler.address</name>
  <value>${yarn.resourcemanager.hostname}:8030</value>
</property>
<property>
  <name>yarn.resourcemanager.webapp.address</name>
  <value>${yarn.resourcemanager.hostname}:8088</value>
</property>
<property>
  <name>yarn.resourcemanager.webapp.https.address</name>
  <value>${yarn.resourcemanager.hostname}:8090</value>
</property>
<property>
  <name>yarn.resourcemanager.resource-tracker.address</name>
  <value>${yarn.resourcemanager.hostname}:8031</value>
</property>
<property>
  <name>yarn.resourcemanager.admin.address</name>
  <value>${yarn.resourcemanager.hostname}:8033</value>
</property>
<property>
  <name>yarn.nodemanager.local-dirs</name>
  <value>/data/hadoop/yarn/local</value>
</property>
<property>
  <name>yarn.log-aggregation-enable</name>
  <value>true</value>
</property>
<property>
  <name>yarn.nodemanager.remote-app-log-dir</name>
  <value>/data/tmp/logs</value>
</property>
<property>
  <name>yarn.log.server.url</name>
  <value>http://master:19888/jobhistory/logs</value>
  <description>URL for job history server</description>
</property>
<property>
  <name>yarn.nodemanager.vmem-check-enabled</name>
  <value>>false</value>
</property>
<property>
```

```

    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
  <property>
    <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
    <value>org.apache.hadoop.mapred.ShuffleHandler</value>
  </property>
</property>
  <property>
    <name>yarn.nodemanager.resource.memory-mb</name>
    <value>2048</value>
  </property>
  <property>
    <name>yarn.scheduler.minimum-allocation-mb</name>
    <value>512</value>
  </property>
  <property>
    <name>yarn.scheduler.maximum-allocation-mb</name>
    <value>4096</value>
  </property>
  <property>
    <name>mapreduce.map.memory.mb</name>
    <value>2048</value>
  </property>
  <property>
    <name>mapreduce.reduce.memory.mb</name>
    <value>2048</value>
  </property>
  <property>
    <name>yarn.nodemanager.resource.cpu-vcores</name>
    <value>1</value>
  </property>
  <property>
    <name>yarn.application.classpath </name>
    <value>
      /usr/local/hadoop-3.1.4/etc/hadoop:/usr/local/hadoop-
3.1.4/share/hadoop/common/lib/*:/usr/local/hadoop-
3.1.4/share/hadoop/common/*:/usr/local/hadoop-
3.1.4/share/hadoop/hdfs:/usr/local/hadoop-
3.1.4/share/hadoop/hdfs/lib/*:/usr/local/hadoop-
3.1.4/share/hadoop/hdfs/*:/usr/local/hadoop-
3.1.4/share/hadoop/mapreduce/lib/*:/usr/local/hadoop-
3.1.4/share/hadoop/mapreduce/*:/usr/local/hadoop-
3.1.4/share/hadoop/yarn:/usr/local/hadoop-
3.1.4/share/hadoop/yarn/lib/*:/usr/local/hadoop-3.1.4/share/hadoop/yarn/*
    </value>
  </property>

```

```
</value>
</property>

</configuration>
```

⑦ yarn-env.sh

```
export JAVA_HOME=/usr/java/jdk1.8.0_281-amd64
```

⑧ workers

删除原有 localhost，添加：

```
slave1
slave2
slave3
```

(4) 对启动命令赋予权限设置

进入到启动命令所在位置

```
cd $HADOOP_HOME/sbin
```

修改 start-dfs.sh 和 stop-dfs.sh(使用 vi 命令分别修改两个文件，添加以下内容)

```
HDFS_DATANODE_USER=root
HADOOP_SECURE_DN_USER=hdfs
HDFS_NAMENODE_USER=root
HDFS_SECONDARYNAMENODE_USER=root
```

修改 start-yarn.sh 和 stop-yarn.sh（使用 vi 命令分别修改两个文件，添加以下内容）

```
YARN_RESOURCEMANAGER_USER=root
HADOOP_SECURE_DN_USER=yarn
YARN_NODEMANAGER_USER=root
```

(5) 拷贝 hadoop 安装文件到集群 slave 节点

```
scp -r /usr/local/hadoop-3.1.4 slave1:/usr/local
scp -r /usr/local/hadoop-3.1.4 slave2:/usr/local
scp -r /usr/local/hadoop-3.1.4 slave3:/usr/local
```

(6) 在/etc/profile 添加 Hadoop 路径

```
export HADOOP_HOME=/usr/local/hadoop-3.1.4
export PATH=$HADOOP_HOME/bin:$PATH
```

修改后【source /etc/profile】使其生效

(7) 格式化 NameNode

```
cd /usr/local/hadoop-3.1.4/bin
```

./hdfs namenode -format

(8) 主节点启动

进入目录

cd /usr/local/hadoop-3.1.4/sbin

执行启动:

./start-dfs.sh

./start-yarn.sh

./mr-jobhistory-daemon.sh start historyserver

(9) 使用 jps, 查看进程

节点	jps 进程
master 节点	NameNode
	SecondaryNameNode
	JobHistoryServer
	Jps
	ResourceManager
slave 节点	Jps
	DataNode
	NodeManager

(10) 在本地电脑添加映射

在 Windows 下 C:\Windows\System32\drivers\etc\hosts 添加 IP 映射

```
192.168.128.130 master
192.168.128.131 slave1
192.168.128.132 slave2
192.168.128.133 slave3
```

(11) 浏览器查看

http://master:9870

http://master:8088

(二) Hive 安装

1. MySQL 安装

1. 设置 mysql8.x 的 yum 源

1.1 wget 下载 mysql8.x 源

wget https://repo.mysql.com//mysql80-community-release-el7-3.noarch.rpm

1.2 加载本地 yum 源

yum localinstall mysql80-community-release-el7-3.noarch.rpm

1.3 yum search mysql

查询是否存在 mysql-community-server.x86_64

下载 mysql 服务

yum install -y mysql-community-server.x86_64

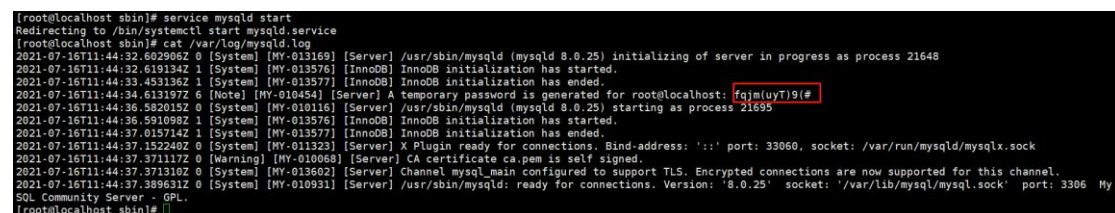
2.mysql 密码设置

2.1 查询 mysql 初始密码

service mysqld start

cat /var/log/mysqld.log

初始密码如下图红框处



```
[root@localhost sbin]# service mysqld start
Redirecting to /bin/systemctl start mysqld.service
[root@localhost sbin]# cat /var/log/mysqld.log
2021-07-16T11:44:32.602906Z 0 [System] [MY-013169] [Server] /usr/sbin/mysqld (mysqld 8.0.25) initializing of server in progress as process 21648
2021-07-16T11:44:32.619134Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
2021-07-16T11:44:33.453136Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
2021-07-16T11:44:34.613197Z 6 [Note] [MY-010454] [Server] A temporary password is generated for root@localhost: fqjm(uyT9(##
2021-07-16T11:44:36.582015Z 0 [System] [MY-010116] [Server] /usr/sbin/mysqld (mysqld 8.0.25) starting as process 21695
2021-07-16T11:44:36.591098Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
2021-07-16T11:44:37.015714Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
2021-07-16T11:44:37.152240Z 0 [System] [MY-011323] [Server] X Plugin ready for connections. Bind-address: '::' port: 33060, socket: /var/run/mysqld/mysqldx.sock
2021-07-16T11:44:37.371117Z 0 [Warning] [MY-010068] [Server] CA certificate ca.pem is self signed.
2021-07-16T11:44:37.371310Z 0 [System] [MY-013602] [Server] Channel mysql_main configured to support TLS. Encrypted connections are now supported for this channel.
2021-07-16T11:44:37.389631Z 0 [System] [MY-010931] [Server] /usr/sbin/mysqld: ready for connections. Version: '8.0.25' socket: '/var/lib/mysql/mysql.sock' port: 3306 MySQL Community Server - GPL.
[root@localhost sbin]#
```

2.2 使用初始密码登录

mysql -u root -p

2.3 修改 8.0 密码规则

set global validate_password.policy=0;

set global validate_password.length=1;

2.4 设置自定义密码

alter user 'root'@'localhost' identified by '123456';

3. 赋予外部连接权限

可能是你的帐号不允许从远程登陆，只能在 localhost。这个时候只要在 localhost 的那台电脑，登入 mysql 后，更改 "mysql" 数据库里的 "user" 表里的 "host" 项，从 "localhost" 改称 "%"

登录数据库：mysql -u root -pvmware

mysql>use mysql;

mysql>update user set host = '%' where user = 'root';

mysql>select host, user from user;

mysql>FLUSH PRIVILEGES

2. hive 配置

(1) 将安装包 **apache-hive-2.3.8-bin.tar.gz** 上传到/opt/目录下

(2) 解压安装包到/usr/local/目录下，解压命令：

```
tar -zxf apache-hive-2.3.8-bin.tar.gz -C /usr/local/
```

(3) 进入到 Hive 的安装目录的 **conf** 目录下，重命名 **hive-env.sh.template** 为 **hive-env.sh**

修改 hive-env.sh 文件，在末尾添加如下内容

```
export HADOOP_HOME=/usr/local/hadoop-3.1.4
```

(4) 在 MySQL 中新建 **hive** 数据库

```
mysql -uroot -p
```

```
create database hive;
```

(5) 将 **hive-site.xml** 文件上传至虚拟机的/opt 目录下

```
cp /opt/hive-site.xml /usr/local/apache-hive-2.3.8-bin/conf/
```

(6) 上传 **mysql 驱动 mysql-connector-java-8.0.20.jar** 到/usr/local/apache-hive-2.3.8-bin/lib 目录

```
cp /opt/mysql-connector-java-8.0.20.jar /usr/local/apache-hive-2.3.8-bin/lib/
```

(7) 在终端执行以下 2 句命令

```
rm -rf /usr/local/apache-hive-2.3.8-bin/lib/guava-14.0.1.jar
```

```
cp /usr/local/hadoop-3.1.4/share/hadoop/common/lib/guava-27.0-jre.jar
```

/usr/local/apache-hive-2.3.8-bin/lib/设置环境变量 vi /etc/profile，添加：

```
export HIVE_HOME=/usr/local/apache-hive-2.3.8-bin
export PATH=$HIVE_HOME/bin:$PATH
```

修改后【source /etc/profile】使其生效

(8) 初始化元数据库，进入 **hive** 安装包 **bin** 目录

```
./schematool -dbType mysql -initSchema
```

(9) 在 **hadoop** 集群已启动的情况下启动 **Hive** 服务

```
hive --service metastore &
```

根据个人需求开启 hiveserver2 服务。

```
nohup hive --service hiveserver2 &
```

(10) 进入 hive 命令窗口

```
/usr/local/apache-hive-2.3.8-bin/bin/hive
```

(三) 配置 Zookeeper

1. 解压缩

上传 zookeeper-3.4.6.tar.gz 安装包到 slave1 节点的/opt 目录下, 之后解压缩到 /usr/local/目录下。

```
tar -zxf /opt/zookeeper-3.4.6.tar.gz -C /usr/local/
```

2. 进入安装路径

```
cd /usr/local/zookeeper-3.4.6/conf
```

3. 复制 zoo_sample.cfg 重命名为 zoo.cfg

```
cp zoo_sample.cfg zoo.cfg
```

配置内容如下:

注: 原文件中部分参数同下表重复, 需要注释或删除原有参数。

<pre>dataDir=/usr/lib/zookeeper dataLogDir=/var/log/zookeeper clientPort=2181 tickTime=2000 initLimit=5 syncLimit=2 server.1=slave1:2888:3888 server.2=slave2:2888:3888 server.3=slave3:2888:3888</pre>

4. 各个子节点新建文件夹

```
mkdir /usr/lib/zookeeper
```

```
mkdir /var/log/zookeeper
```

```
vi /usr/lib/zookeeper/myid
```

在 slave1 的/usr/lib/zookeeper 目录下新建文件 myid, 内容为 1

在 slave2 的/usr/lib/zookeeper 目录下新建文件 myid, 内容为 2

在 slave3 的/usr/lib/zookeeper 目录下新建文件 myid, 内容为 3

5. 在 slave1 节点执行

```
scp -r /usr/local/zookeeper-3.4.6 slave2:/usr/local/
```

```
scp -r /usr/local/zookeeper-3.4.6 slave3:/usr/local/
```

6. 在各子节点的/etc/profile 中配置环境变量

```
export ZK_HOME=/usr/local/zookeeper-3.4.6
export PATH=$PATH:$ZK_HOME/bin
```

修改后【source /etc/profile】使其生效

7. 启动各个子节点 Zookeeper

```
/usr/local/zookeeper-3.4.6/bin/zkServer.sh start
```

8. 查看各个子节点的 zookeeper 是否启动

```
/usr/local/zookeeper-3.4.6/bin/zkServer.sh status
```

(四) 配置 HBase

1. 解压缩

通过 xmanager 的 Xftp 上传 hbase-2.2.6-bin.tar.gz 压缩包到/opt 目录，执行命令解压缩 hbase-2.2.6.tar.gz 文件。

```
tar -zxf /opt/hbase-2.2.6-bin.tar.gz -C /usr/local
```

2. 进入目录

```
cd /usr/local/hbase-2.2.6/conf
```

3. 修改 hbase-site.xml 文件

```
<configuration>
<property>
  <name>hbase.rootdir</name>
  <value>hdfs://master:8020/hbase</value>
</property>
<property>
  <name>hbase.master</name>
  <value>master</value>
</property>
<property>
  <name>hbase.cluster.distributed</name>
  <value>true</value>
</property>
<property>
  <name>hbase.zookeeper.property.clientPort</name>
  <value>2181</value>
</property>
<property>
  <name>hbase.zookeeper.quorum</name>
  <value>slave1,slave2,slave3</value>
</property>
```

```
<property>
  <name>zookeeper.session.timeout</name>
  <value>60000000</value>
</property>
<property>
  <name>dfs.support.append</name>
  <value>true</value>
</property>
<property>
  <name>hbase.unsafe.stream.capability.enforce</name>
  <value>false</value>
</property>
</configuration>
```

4. 配置 hbase-env.sh

注释下面两句

```
export HBASE_MASTER_OPTS="$HBASE_MASTER_OPTS -
XX:PermSize=128m -XX:MaxPermSize=128m"
export HBASE_REGIONSERVER_OPTS="$HBASE_REGIONSERVER_OPTS -
XX:PermSize=128m -XX:MaxPermSize=128m"
```

配置以下内容。

```
export HBASE_CLASSPATH=/usr/local/hadoop-3.1.4/etc/hadoop
export JAVA_HOME=/usr/java/jdk1.8.0_281-amd64
export HBASE_MANAGES_ZK=false
```

5. 配置 regionservers

```
slave1
slave2
slave3
```

6. 拷贝到各子节点

```
scp -r /usr/local/hbase-2.2.6/ slave1:/usr/local/
```

```
scp -r /usr/local/hbase-2.2.6/ slave2:/usr/local/
```

```
scp -r /usr/local/hbase-2.2.6/ slave3:/usr/local/
```

7. 配置环境变量

```
export HBASE_HOME=/usr/local/hbase-2.2.6
export PATH=$PATH:$HBASE_HOME/bin
```

修改后【source /etc/profile】使其生效

8. 主节点启动

启动前要先确保启动了 zookeeper 和 Hadoop 集群

```
cd /usr/local/hbase-2.2.6/bin/
```

./start-hbase.sh

9. 在浏览器查看

<http://192.168.128.130:16010>

(五) 配置 Scala

1. 解压 scala-2.11.12 安装包

```
tar -zxvf /opt/scala-2.11.12.tgz -C /usr/local
```

2. 修改解压后文件名

```
cd /usr/local
```

```
mv scala-2.11.12 scala
```

3. 修改环境变量

```
vi /etc/profile
```

添加：

```
export SCALA_HOME=/usr/local/scala
export PATH=$SCALA_HOME/bin
```

修改后【source /etc/profile】使其生效

(六) 配置 Spark

1. 解压 spark-2.4.7-bin-hadoop2.7.tgz 安装包

```
tar -zxvf /opt/spark-2.4.7-bin-hadoop2.7.tgz -C /usr/local/
```

2. 修改解压后 spark 文件名

```
cd /usr/local
```

```
mv spark-2.4.7-bin-hadoop2.7 spark
```

3. 修改 spark-env.sh

```
vim conf/spark-env.sh
```

添加以下内容

```
export JAVA_HOME=/usr/java/jdk1.8.0_281-amd64
export SCALA_HOME=/usr/local/scala
export HADOOP_HOME=/usr/local/hadoop-3.1.4
export HADOOP_CONF_DIR=/usr/local/hadoop-3.1.4/etc/hadoop
export SPARK_MASTER_IP=master
```

```
export SPARK_MASTER_PORT=7077
export SPARK_MASTER_WEBUI_PORT=8080
export SPARK_WORKER_CORES=1
export SPARK_WORKER_MEMORY=2g    #spark 里许多用到内存的地方默认
1g 2g 这里最好设置大与 1g
export SPARK_WORKER_PORT=7078
export SPARK_WORKER_WEBUI_PORT=8081
export SPARK_WORKER_INSTANCES=1
```

4. 修改 spark-defaults.conf

```
spark.master      spark://master:7077
```

5. 修改 slaves

```
slave1
slave2
salve3
```

6. 发送至其余节点

```
scp -r /usr/local/saprk slave1:/usr/local
scp -r /usr/local/saprk slave2:/usr/local
scp -r /usr/local/saprk slave3:/usr/local
```

7. 配置环境变量(所有节点)

```
vi /etc/profile
```

```
export SPARK_HOME=/usr/local/spark
export PATH=$SPARK_HOME/bin
```

修改后【source /etc/profile】使其生效。

8. 启动 saprk

执行【/usr/local/spark/sbin/start-all.sh】

9. 在浏览器查看

<http://192.168.128.130:8080/>

(七) 配置 Flume

1. 解压 apache-flume-1.9.0-bin.tar.gz 文件

```
tar -zxvf apache-flume-1.9.0-bin.tar.gz -C /usr/local
```

2. 修改文件名

```
cd /usr/local
```

```
mv apache-flume-1.9.0-bin/ flume/
```

3. 修改 flume-env.sh 文件

```
cd flume/conf
```

```
cp flume-env.sh.template flume-env.conf
```

添加 JDK 位置信息

```
export JAVA_HOME=/usr/java/jdk1.8.0_281-amd64/
```

(八) 配置 Kafka（Kafka 需要部署在有 zookeeper 的节点上）

1. 解压 Kafka 安装包

```
tar -zxvf /opt/kafka_2.13-2.4.0.tgz -C /usr/local
```

```
cd /usr/local
```

```
mv kafka_2.13-2.4.0 kafka
```

2. 配置环境变量

```
vim /etc/profile
```

```
export KAFKA_HOME=/usr/local/kafka
export PATH=$KAFKA_HOME/bin:$PATH:
```

修改后【source /etc/profile】使其生效。

3. 修改 server.properties

```
cd $KAFKA_HOME/config
```

```
broker.id=0

#此部分参数为设置节点的 zookeeper 连接
zookeeper.connect=slave1:2181,slave2:2181,slave3:2181

log.dirs=/log/kafka_logs
```

4. 创建目录

每个节点上创建日志目录

```
mkdir -p /log/kafka_logs
```

5. 发送至其余节点

```
scp -r /usr/local/kafka slave1:/usr/local
```

```
scp -r /usr/local/kafka slave2:/usr/local
```

```
scp -r /usr/local/kafka slave3:/usr/local
```

6. 修改子节点的 **server.properties**

slave1 和 slave2 节点的 broker.id 分别修改为 1、2

7. 启动 **Kafka**

在每个节点执行（注意启动 kafka 前需先正常启动 zookeeper）

【kafka-server-start.sh -daemon /usr/local/kafka/config/server.properties &】

(九) 配置 **Flink**（主节点进行）

1. 解压 **flink-1.10.1-bin-scala_2.11.tgz** 安装包

```
tar -zxvf /opt/flink-1.10.1-bin-scala_2.11.tgz -C /usr/local
```

```
cd /usr/local
```

```
mv flink-1.10.1/ flink/
```

2. 配置环境变量

```
vim /etc/profile
```

```
export FLINK_HOME=/usr/local/flink
export PATH=$PATH:$FLINK_HOME/bin
```

修改后【source /etc/profile】使其生效。

3. 修改配置文件**\$FLINK_HOME/conf/flink-conf.yaml**

```
jobmanager.rpc.address: 192.168.128.130
jobmanager.heap.size: 512m
#taskmanager.memory.process.size: 768
taskmanager.memory.flink.size: 1024m
taskmanager.numberOfTaskSlots: 3
parallelism.default: 1
rest.address: 192.168.128.130
rest.bind-address: 192.168.128.130
```



```
io.tmp.dirs: /var/log/flink/tmp
```

4. 修改配置文件/conf/slaves

```
slave1
```

```
slave2
```

```
slave3
```

5. 修改配置文件/conf/masters

```
master:8081
```

6. 拷贝安装文件到集群 slave 节点

```
scp -r /usr/local/flink slave1:/usr/local
```

```
scp -r /usr/local/flink slave2:/usr/local
```

```
scp -r /usr/local/flink slave3:/usr/local
```

7. 在每个节点上创建日志文件目录

```
mkdir -p /var/log/flink/tmp
```

8. 启动 Flink

```
/usr/local/flink/bin/start-cluster.sh
```

9. 查看 jps

主节点:

```
19489 jar
```

```
24595 StandaloneSessionClusterEntrypoint
```

子节点:

```
25189 Jps
```

```
25039 TaskManagerRunner
```