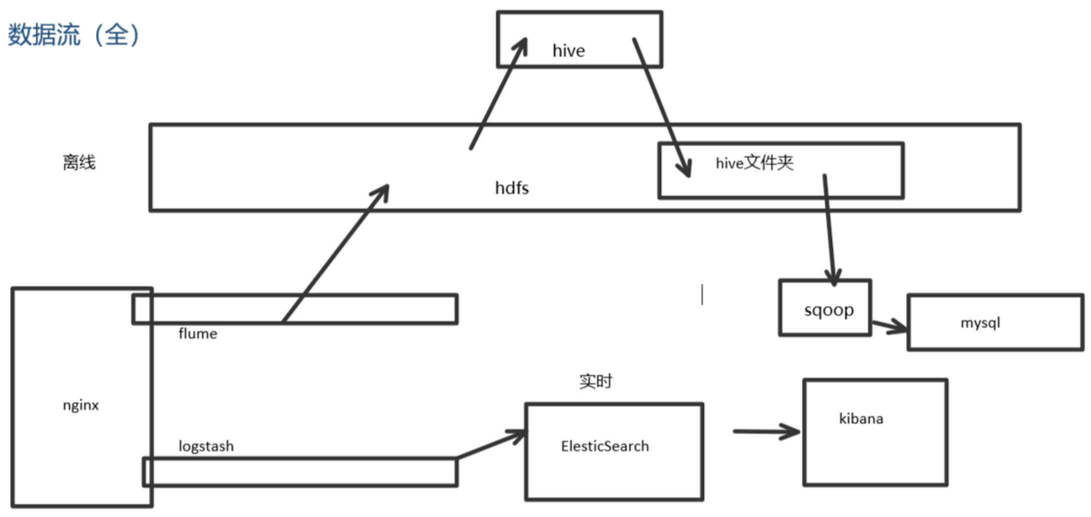
整体架构



一、基础环境配置

Linux虚拟机的安装

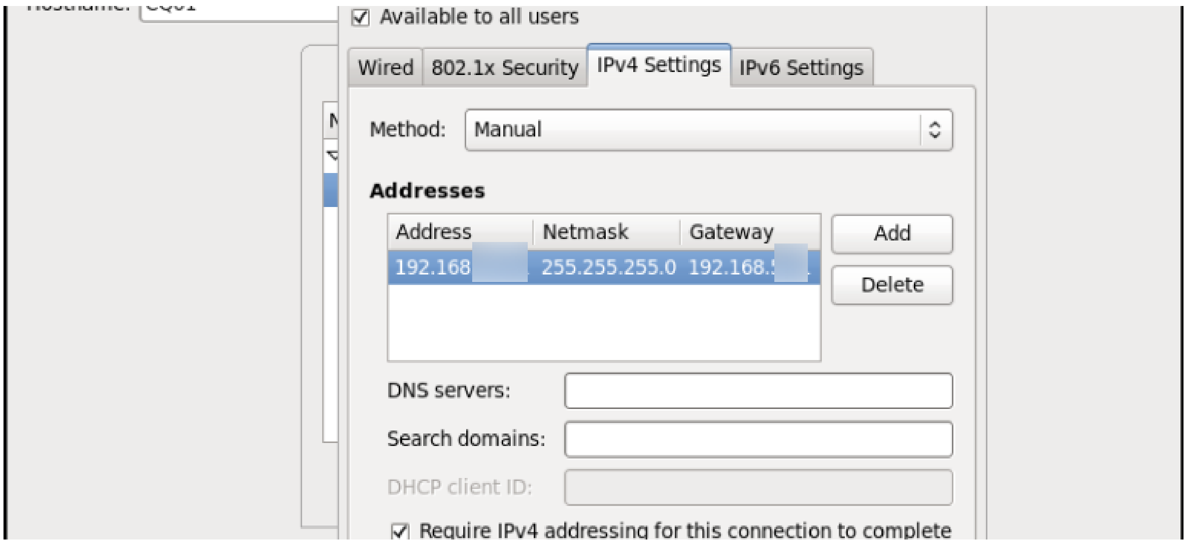
三台名命

hadoop01

hadoop02

hadoop03

网络配置



☑ Available to all users

Wired | 802.1x Security | IPv4 Settings | IPv6 Settings

Method: Manual

Addresses

Address	Netmask	Gateway
192.168[redacted]	255.255.255.0	192.168.[redacted]

Add

Delete

DNS servers:

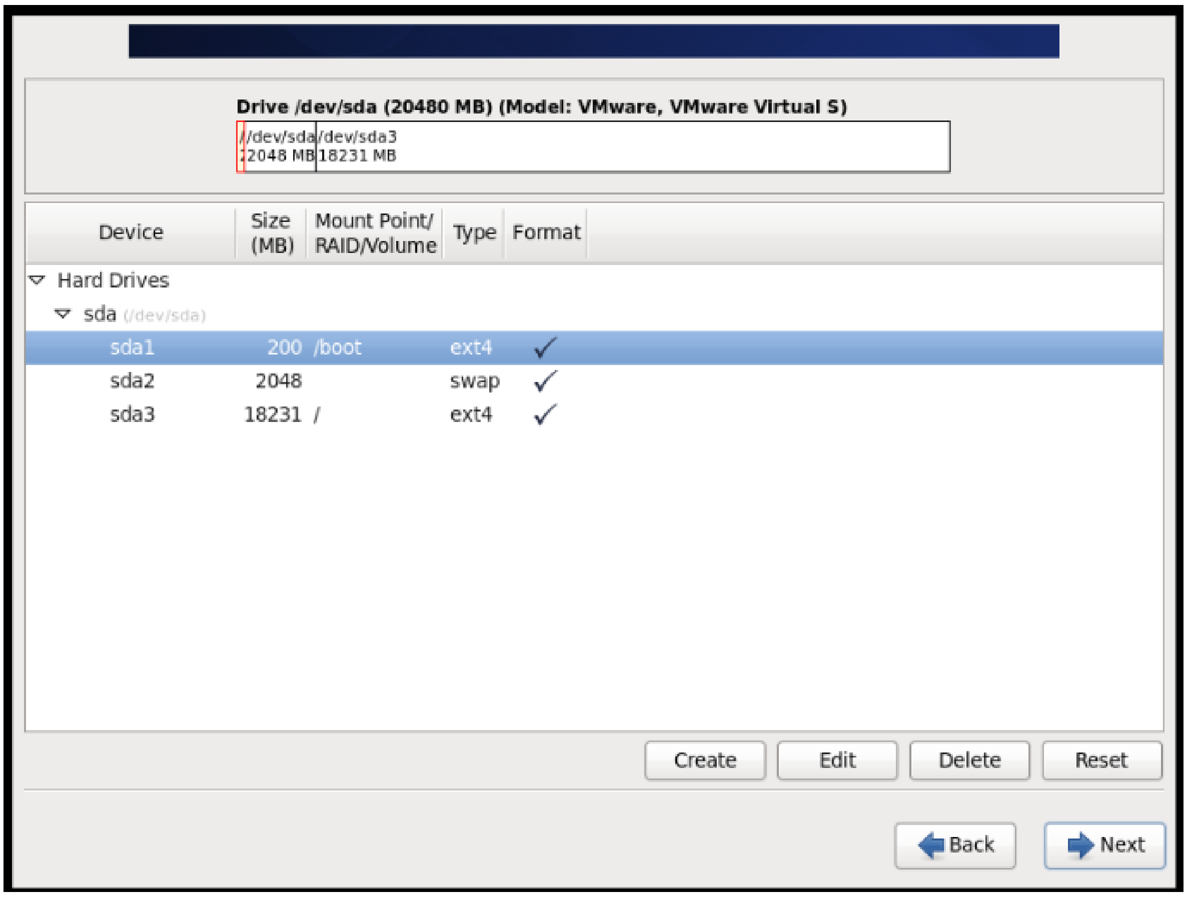
Search domains:

DHCP client ID:

☑ Require IPv4 addressing for this connection to complete

```
Address:通过ipconfig进行查询，第三位是什么，第四位自定义。
Netwask: 255.255.255.0
Gateway: 192.168.x.254   //x 表示你自己网段的数值，根Address第三位一样。

//打码处主要是为了要你自己去看你主机的第三位是什么，不要盲目与我相同。
```

虚拟内存配置



Drive /dev/sda (20480 MB) (Model: VMware, VMware Virtual S)

/dev/sda 2048 MB | /dev/sda3 18231 MB

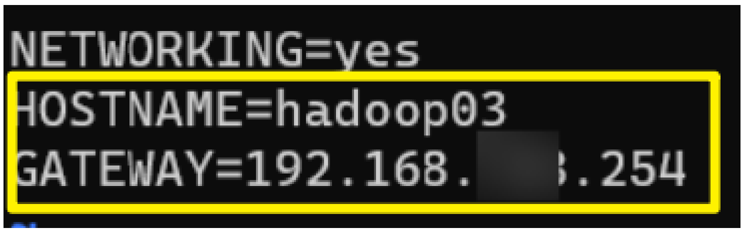
Device	Size (MB)	Mount Point/ RAID/Volume	Type	Format
Hard Drives				
sda (/dev/sda)				
sda1	200	/boot	ext4	✓
sda2	2048		swap	✓
sda3	18231	/	ext4	✓

```
sda1 启动盘分区 /boot   //引导分区
sda2 交换区  swap //一般大小为你分配内存大小，当内存的不够用时，可以去读取一部分磁盘空间作为内存使用
sda3 剩下的磁盘空间分配 /    //分配全部剩余空间
```

网络配置

修改 network（3台）

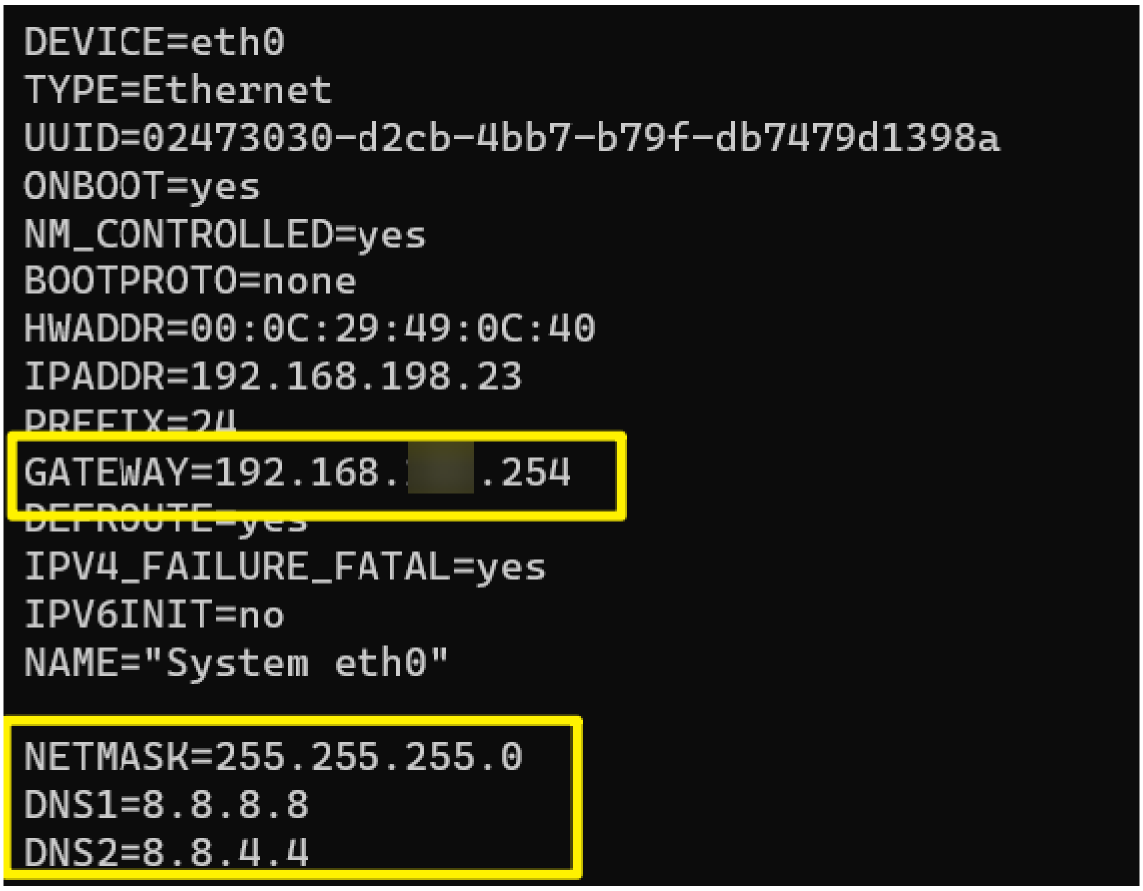
```
vi /etc/sysconfig/network
```



```
NETWORKING=yes
HOSTNAME=hadoop03
GATEWAY=192.168.[redacted].254
```

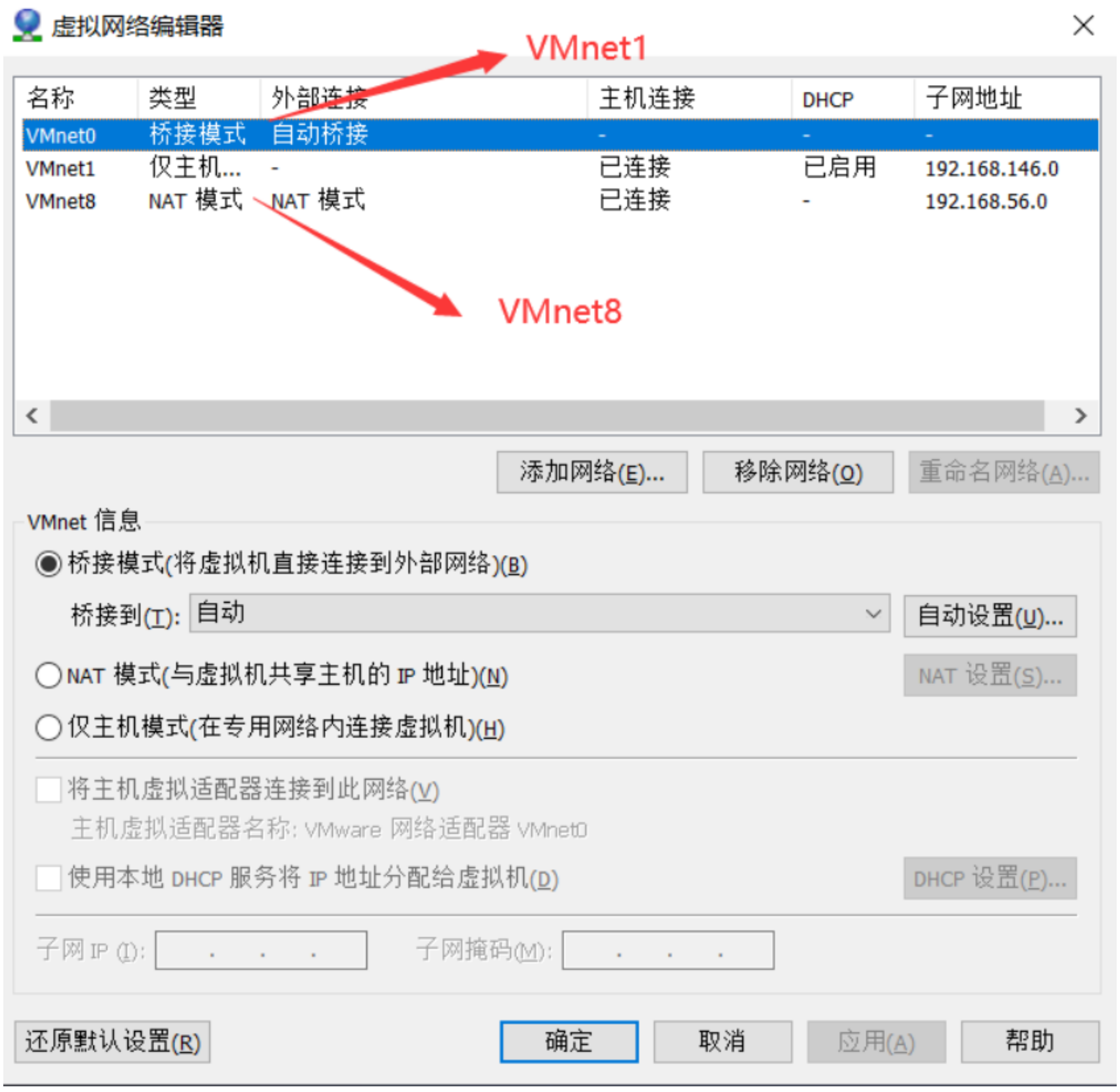
修改 ifcfg-eth0（3台）

```
vi /etc/sysconfig/network-scripts/ifcfg-eth0
```



```
DEVICE=eth0
TYPE=Ethernet
UUID=02473030-d2cb-4bb7-b79f-db7479d1398a
ONBOOT=yes
NM_CONTROLLED=yes
BOOTPROTO=none
HWADDR=00:0C:29:49:0C:40
IPADDR=192.168.198.23
PREFIX=24
GATEWAY=192.168.[redacted].254
DEFROUTE=yes
IPV4_FAILURE_FATAL=yes
IPV6INIT=no
NAME="System eth0"

NETMASK=255.255.255.0
DNS1=8.8.8.8
DNS2=8.8.4.4
```

- **VM配置（1次）**



名称	类型	外部连接	主机连接	DHCP	子网地址
VMnet0	桥接模式	自动桥接	-	-	-
VMnet1	仅主机...	-	已连接	已启用	192.168.146.0
VMnet8	NAT 模式	NAT 模式	已连接	-	192.168.56.0

VMnet 信息

(●) 桥接模式(将虚拟机直接连接到外部网络)(B)

桥接到(T): 自动

( ) NAT 模式(与虚拟机共享主机的 IP 地址)(N)

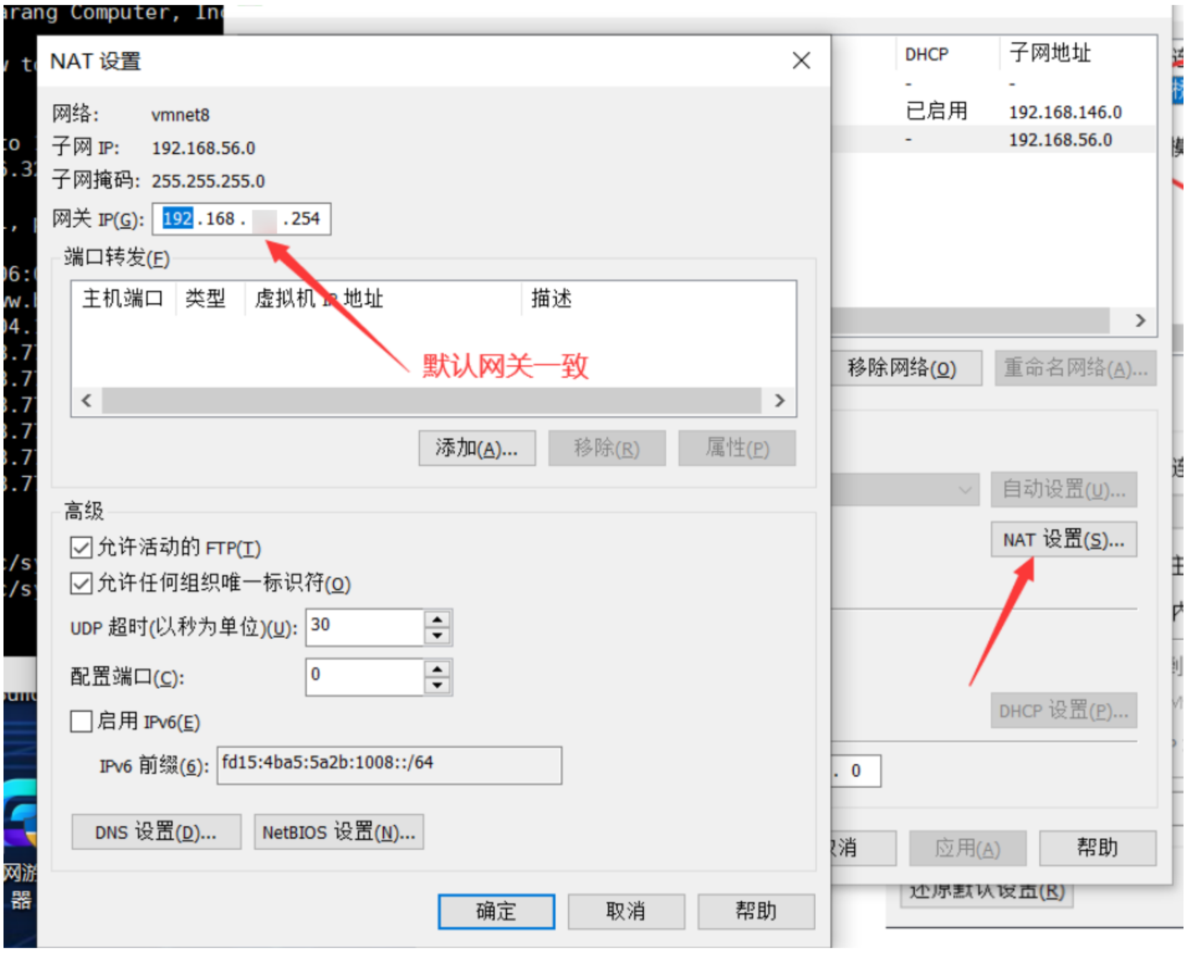
( ) 仅主机模式(在专用网络内连接虚拟机)(H)

☐ 将主机虚拟适配器连接到此网络(V)

主机虚拟适配器名称: VMware 网络适配器 VMnet0

☐ 使用本地 DHCP 服务将 IP 地址分配给虚拟机(D)

子网 IP (I): . . .   子网掩码(M): . . .


NAT 设置

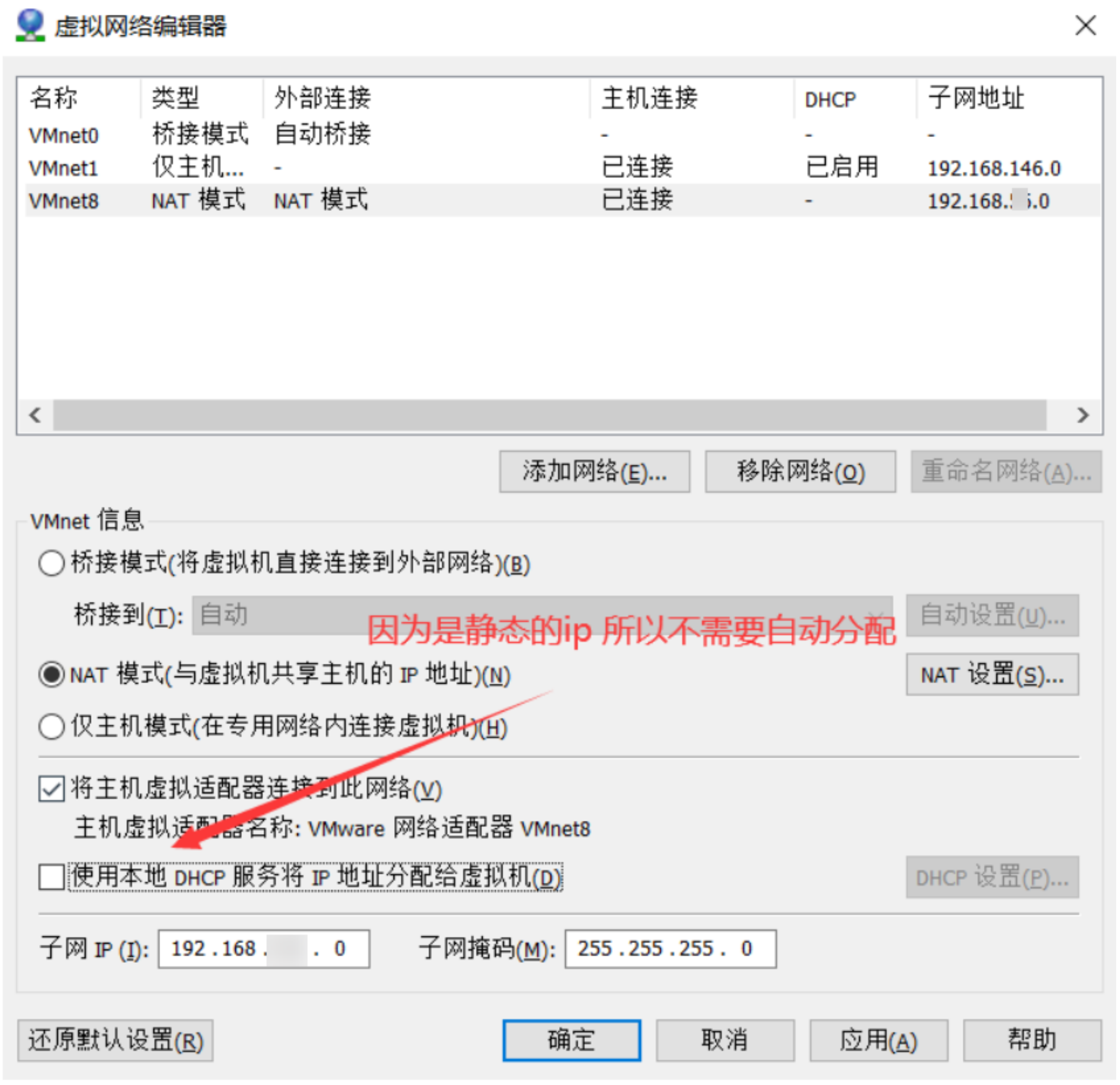
网络: vmnet8
子网 IP: 192.168.56.0
子网掩码: 255.255.255.0
网关 IP(G): 192.168. .254

端口转发(F)

主机端口	类型	虚拟机 IP 地址	描述

高级
☑ 允许活动的 FTP(T)
☑ 允许任何组织唯一标识符(O)
UDP 超时(以秒为单位)(U): 30
配置端口(C): 0
☐ 启用 IPv6(E)
IPv6 前缀(6): fd15:4ba5:5a2b:1008::/64

DHCP	子网地址
-	-
已启用	192.168.146.0
-	192.168.56.0


虚拟网络编辑器

名称	类型	外部连接	主机连接	DHCP	子网地址
VMnet0	桥接模式	自动桥接	-	-	-
VMnet1	仅主机...	-	已连接	已启用	192.168.146.0
VMnet8	NAT 模式	NAT 模式	已连接	-	192.168.[redacted].0

添加网络(E)... 移除网络(O) 重命名网络(A)...

VMnet 信息

○ 桥接模式(将虚拟机直接连接到外部网络)(B)

桥接到(T): 自动 自动设置(U)...

◉ NAT 模式(与虚拟机共享主机的 IP 地址)(N) NAT 设置(S)...

○ 仅主机模式(在专用网络内连接虚拟机)(H)

☑ 将主机虚拟适配器连接到此网络(V)

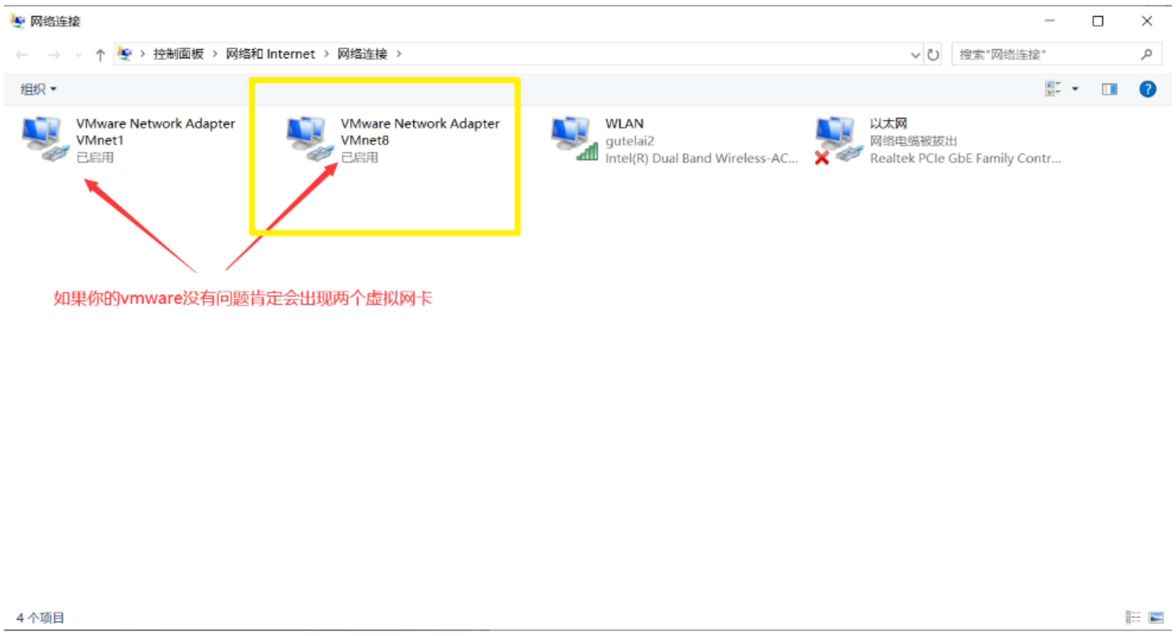
主机虚拟适配器名称: VMware 网络适配器 VMnet8

☐ 使用本地 DHCP 服务将 IP 地址分配给虚拟机(D) DHCP 设置(P)...

子网 IP (I): 192 . 168 . [redacted] . 0 子网掩码(M): 255 . 255 . 255 . 0

还原默认设置(R) 确定 取消 应用(A) 帮助

主机适配器配置（1次）


网络连接

控制面板 › 网络和 Internet › 网络连接 › 搜索"网络连接"

组织 ▾

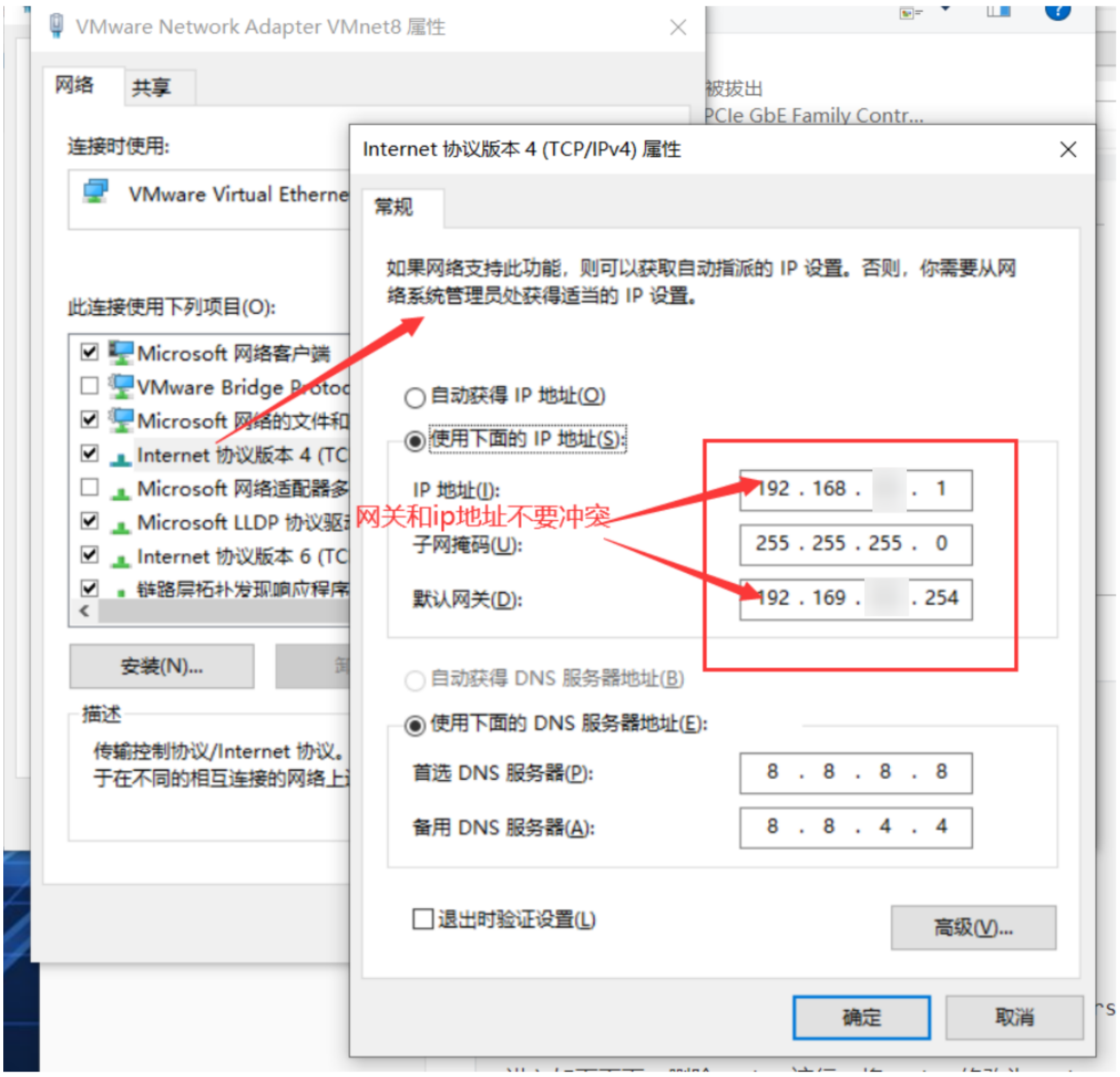
VMware Network Adapter VMnet1 已启用

VMware Network Adapter VMnet8 已启用

WLAN gutelai2 Intel(R) Dual Band Wireless-AC...

以太网 网络电缆被拔出 Realtek PCIe GbE Family Contr...

4 个项目


VMware Network Adapter VMnet8 属性

网络 共享

连接时使用:

VMware Virtual Etherne

此连接使用下列项目(O):

- [x] Microsoft 网络客户端
- [ ] VMware Bridge Protoco
- [x] Microsoft 网络的文件和
- [x] Internet 协议版本 4 (TC
- [ ] Microsoft 网络适配器多
- [x] Microsoft LLDP 协议驱
- [x] Internet 协议版本 6 (TC
- [x] 链路层拓扑发现响应程序

安装(N)...

描述

传输控制协议/Internet 协议。于在不同的相互连接的网络上

Internet 协议版本 4 (TCP/IPv4) 属性

常规

如果网络支持此功能，则可以获取自动指派的 IP 设置。否则，你需要从网络系统管理员处获得适当的 IP 设置。

( ) 自动获得 IP 地址(O)
(●) 使用下面的 IP 地址(S):

IP 地址(I):	192 . 168 . [redacted] . 1
子网掩码(U):	255 . 255 . 255 . 0
默认网关(D):	192 . 169 . [redacted] . 254

网关和ip地址不要冲突

( ) 自动获得 DNS 服务器地址(B)
(●) 使用下面的 DNS 服务器地址(E):

首选 DNS 服务器(P):	8 . 8 . 8 . 8
备用 DNS 服务器(A):	8 . 8 . 4 . 4

- [ ] 退出时验证设置(L)

高级(V)... 确定 取消

重启网卡

```
service network restart
```

防火墙设置

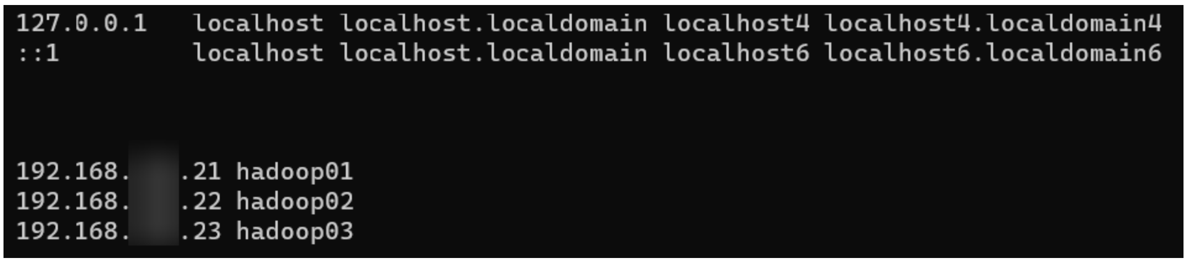
```
建议永久关闭

service iptables status （功能描述：查看防火墙状态）

chkconfig iptables –list （功能描述：查看防火墙开机启动状态）

service iptables stop （功能描述：临时关闭防火墙）

chkconfig iptables off （功能描述：关闭防火墙开机启动）

chkconfig iptables on （功能描述：开启防火墙开机启动）
```

二、配置ssh免密登录

host映射配置

虚拟机中配置

```
配置hosts映射文件
##1.编辑映射文件
vi /etc/hosts
```

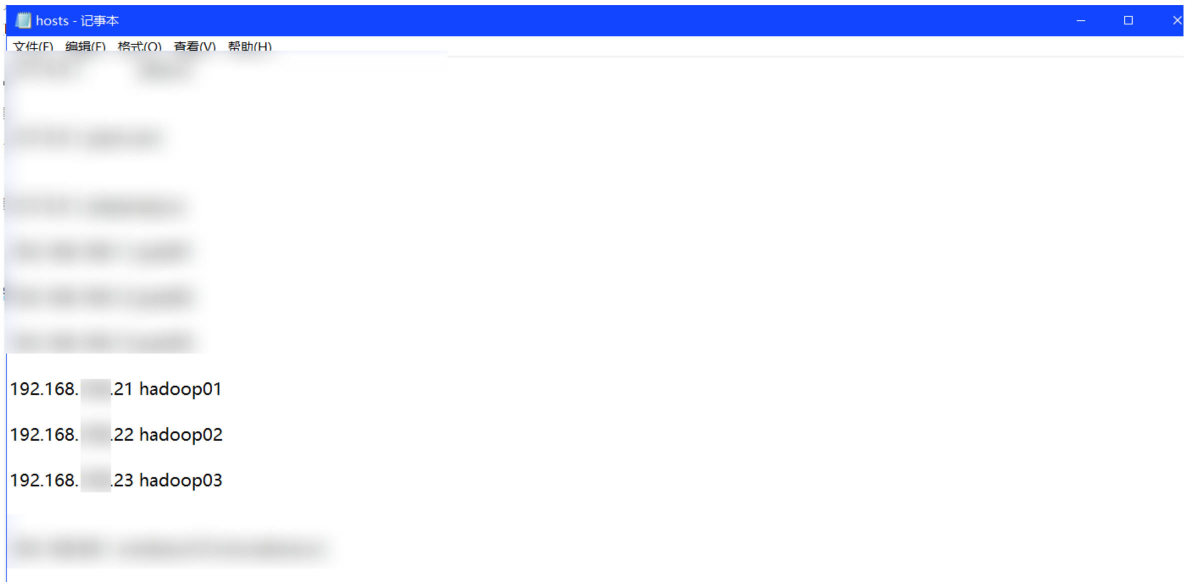


```
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6


192.168.[redacted].21 hadoop01
192.168.[redacted].22 hadoop02
192.168.[redacted].23 hadoop03
```

```
##2.将编辑好的文件发送给其他虚拟机(默认会覆盖文件，我在3号机操作，所以发往1、2机)
scp /etc/hosts hadoop02:/etc/hosts
scp /etc/hosts hadoop01:/etc/hosts
```

主机配置映射

```
C:\Windows\System32\drivers\etc\hosts
```



```
[redacted]
192.168.[redacted].21 hadoop01
192.168.[redacted].22 hadoop02
192.168.[redacted].23 hadoop03
[redacted]
```

创建钥匙

```
##创建两个钥匙（私钥）（公钥）
ssh-keygen -t rsa
##发送公钥 接受公钥的主机（hadoop01 hadoop01 hadoop03 都需要发）
ssh-copy-id 主机名
```

三、linux相关软件安装

JDK的安装

卸载原来虚拟机安装的jdk

```
查看安装的jdk
rpm -qa | grep jdk

卸载jdk
rpm -e --nodeps java-1.7.0-openjdk-1.7.0.79-2.5.5.4.el6.x86_64
rpm -e --nodeps java-1.6.0-openjdk-1.6.0.35-1.13.7.1.el6_6.x86_64
```

配置环境变量

```
export JAVA_HOME=/opt/apps/jdk1.8.0_181
export PATH=$PATH:$JAVA_HOME/bin

刷新环境变量
source /etc/profile
```

mysql的安装

卸载原来安装的mysql

```
## 查看原来安装了哪些mysql
rpm -qa | grep mysql

## 卸载原来的mysql
rpm -e --nodeps mysql-libs-5.1.73-5.el6_6.x86_64

## 安装自己的mysql
rpm -ivh ./MySQL-client-5.5.47-1.linux2.6.x86_64.rpm

rpm -ivh ./MySQL-server-5.5.47-1.linux2.6.x86_64.rpm

## 启动mysql服务
service mysql start
```

配置mysql

```
## 添加root用户密码
mysqladmin -u root password "root"

## 添加除服务器之外任何主机的使用root用户登录
grant all privileges on *.* to 'root'@'%' identified by 'root';

## 刷新权限
flush privileges;

##使用navcat访问要记得关闭防火墙
service iptables stop
```

四、hadoop集群HA(高可用的搭建)

配置JPS结果

hadoop03	hadoop02	hadoop01
QuorumPeerMain	QuorumPeerMain	QuorumPeerMain
journalnode	journalnode	journalnode
namenode(active)	namenode(standby)	datanode
datanode	datanode	Resourcemanager(active)
nodemanager	Resourcemanager(standby)	nodemanager
DFSZKFailoverController	nodemanager	
	DFSZKFailoverController	

修改Hadoop-env.sh 中 JAVA_HOME路径

```
export JAVA_HOME=/opt/app/jdk1.8.0_181
```

zookeeper配置

解压到你对应的文件目录

然后修改配置文件，所有配置文件我建议使用cp进行复制了在修改。

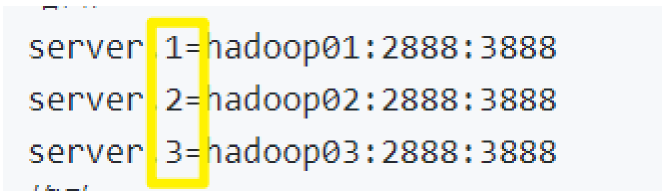
```
cp conf/zoo_ sample.cfg ./zoo.cfg
```

在zookeeper根目录创建zkData文件夹，然后对zoo.cfg进行修改。

```
增加
server.1=hadoop01:2888:3888
server.2=hadoop02:2888:3888
server.3=hadoop03:2888:3888
修改
dataDir=/opt/apps/zookeeper-3.4. 10/zkData/
```

//一定要注意你的zkData的文件路径

在zkData目录下创建myid文件，文件内容为zoo.cfg文件中对应的数字（就是你主机的编号），然后发送到2、3号机器，更改myid为1、 2。



```
server.1=hadoop01:2888:3888
server.2=hadoop02:2888:3888
server.3=hadoop03:2888:3888
```

namenode HA环境搭建

在配置HA环境之前，首先包正zookeeper已经安装完毕

首先在core.site.xml配置文件中，配置如下信息

```
<configuration>
        <property>
                <name>fs.defaultFS</name>
                <value>hdfs://yzsen</value>   // 你集群的名字 我的是yzsen
        </property>
        <property>
                <name>hadoop.tmp.dir</name>
                <value>/opt/apps/hadoop-2.7.2/data/tmp</value>  //hadoop路径
        </property>
        <property>
                <name>ha.zookeeper.quorum</name>
                <value>hadoop01:2181,hadoop02:2181,hadoop03:2181</value>
        </property>
</configuration>
```

hdfs配置

把一个hdfs的namenode路径修改为集群HA对外的名称，在hdfs-site.xml 文件中配置如下信息。

```
<configuration>
# 如果无法访问50070的端口就添加这个配置
    <property>
        <name>dfs.http.address</name>
        <value>0.0.0.0:50070</value>
    </property>
    <property>   #集群的服务名称
        <name>dfs.nameservices</name>
        <value>yzsen</value>
    </property>
    <property>    #在集群服务名称下配置两个namenode nn1 nn2
        <name>dfs.ha.namenodes.yzsen</name>
        <value>nn1,nn2</value>
    </property>
    <property>
        <name>dfs.namenode.rpc-address.yzsen.nn1</name>
        <value>hadoop03:8020</value>  #配置nn1namenode的路径
    </property>
    <property>
        <name>dfs.namenode.rpc-address.yzsen.nn2</name>
        <value>hadoop02:8020</value>  #配置nn2 namenode的路劲
    </property>
    <property>
        <name>dfs.namenode.http-address.yzsen.nn1</name>
        <value>hadoop03:50070</value>  #配置namenode1 web接口的地址
    </property>
    <property>
        <name>dfs.namenode.http-address.yzsen.nn2</name>
        <value>hadoop02:50070</value> #配置namenode2 web接口的地址
    </property>
    <property>  # 配置日志服务器的地址
        <name>dfs.namenode.shared.edits.dir</name>
        <value>qjournal://hadoop01:8485;hadoop02:8485;hadoop03:8485/ns1</value>
    </property>
    <property>  #配置日志服务器存储日志在磁盘上的位置
        <name>dfs.journalnode.edits.dir</name>
        <value>/opt/apps/hadoop-2.7.2/journalnode</value>
    </property>
    <property>  #配置HA下客户端代理服务器
        <name>dfs.client.failover.proxy.provider.yzsen</name>
        <value>org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider</value>
    </property>
    <property>  # 配置ssh 免密登录用于stand by kill 掉 active
        <name>dfs.ha.fencing.methods</name>
        <value>sshfence</value>
    </property>
    <property> # 钥匙路径
        <name>dfs.ha.fencing.ssh.private-key-files</name>
        <value>/root/.ssh/id_rsa</value>
    </property>
    <property>  #ssh kill 时限
```

```
        <name>dfs.ha.fencing.ssh.connect-timeout</name>
        <value>30000</value>
    </property>
    <property>
        <name>dfs.permissions.enabled</name>
        <value>false</value>
    </property>
    <property>   #开启自动故障转移的功能
        <name>dfs.ha.automatic-failover.enabled</name>
        <value>true</value>
    </property>
</configuration>
```

配置slaves

把从节点的主机名称添加进去

```
hadoop01
hadoop02
hadoop03
```

发送到其它主机

配置完以上的内容之后，需要把节点分发到各个节点。

注意： 在分发之前，需要把原来文件data/tmp 目录下的文件，以及logs 下面的文件 还有根目录/tmp下面的文件全部清空。

初始化启动

启动的顺序：

1、先启动zookeeper

```
bin/zkServer.sh start  //依次启动3台zookeeper   （建议顺序是1、2、3）
```

2、启动journalnode

```
sbin/hadoop-daemon.sh start journalnode   //3台
```

必须先启动zookeeper 才能启动journalnode

要确保你的journalnode启动没有问题 可以通过查看hadoop中logs目录下的日志来查看

确保三台虚拟机的防火墙是关闭状态

3、格式化hdfs namenode（在两台配置了namenode的虚拟机的3号机上格式化namenode即可即可）

```
bin/hdfs namenode -format
```

4、格式化完成在格式化了namenode的虚拟机上先启动namenode 服务

```
sbin/hadoop-daemon.sh start namenode
```

5、在2号机配置了namenode的虚拟机上同步刚刚启动了namenode active的元数据

```
bin/hdfs namenode -bootstrapStandby  //同步元数据
```

```
About to bootstrap Standby ID nn2 from:
           Nameservice ID: yzsen
        Other Namenode ID: nn1
  Other NN's HTTP address: http://hadoop03:50070
  Other NN's IPC  address: hadoop01/192.168.x.23:8020
             Namespace ID: 1400592920
            Block pool ID: BP-206752366-10.36.143.140-1544684131154
               Cluster ID: CID-b5588c4f-b9e4-4e29-b39f-e33d3edd2ed1
           Layout version: -63
       isUpgradeFinalized: true
```

看到以上的信息代表同步成功，类型相同即可。

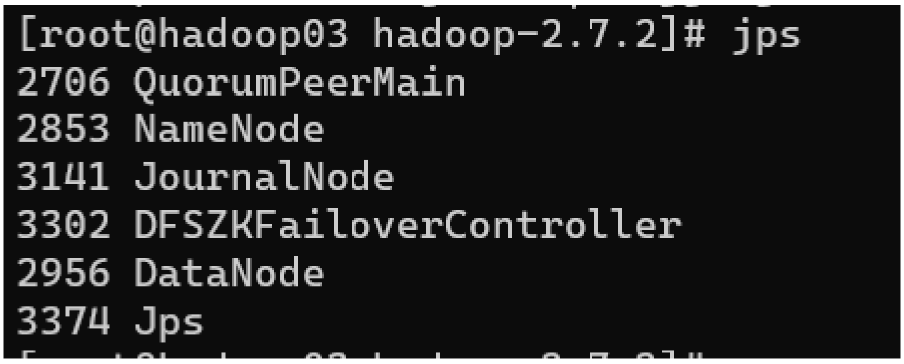
6、然后关闭namenode进程，初始化zkfc 在zookeeper中初始化hadoopHA的信息（在3号机初始化即可）

```
bin/hdfs zkfc -formatZK
```

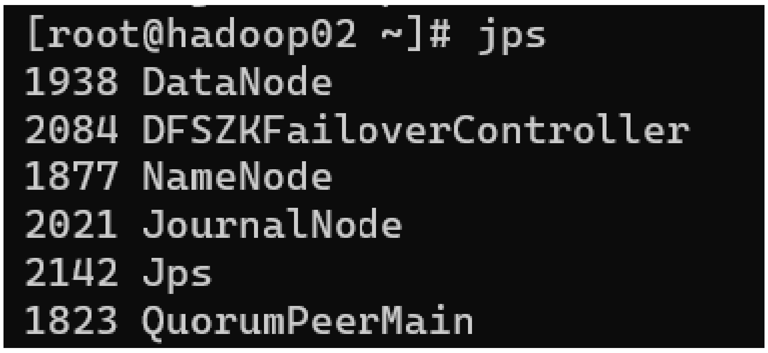
7、直接启动hdfs

```
sbin/start-dfs.sh  //启动全部的hdfs组件
```

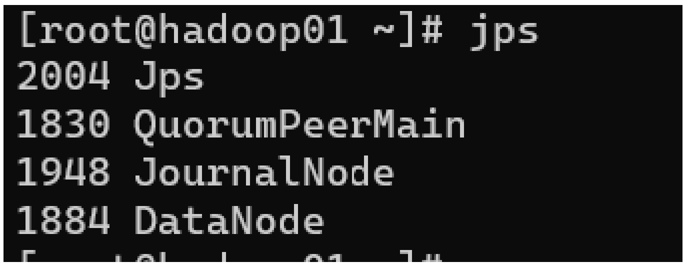
启动结果



```
[root@hadoop03 hadoop-2.7.2]# jps
2706 QuorumPeerMain
2853 NameNode
3141 JournalNode
3302 DFSZKFailoverController
2956 DataNode
3374 Jps
```



```
[root@hadoop02 ~]# jps
1938 DataNode
2084 DFSZKFailoverController
1877 NameNode
2021 JournalNode
2142 Jps
1823 QuorumPeerMain
```



```
[root@hadoop01 ~]# jps
2004 Jps
1830 QuorumPeerMain
1948 JournalNode
1884 DataNode
```

配置yarn

RM HA机制主要是为了解决RM 出现单点故障，切换机制比较简单，主要是在zookeeper中记录RM的状态 当activeRM出现故障，则通过zookeeper切换到standby。

配置mapred-site.xml配置文件

```
<configuration>
    <property>
        <name>mapreduce.framework.name</name>
        <value>yarn</value>
    </property>
    <property>
        <name>mapreduce.jobhistory.address</name>
        <value>hadoop03:10020</value>
    </property>
    <property>
        <name>mapreduce.jobhistory.webapp.address</name>
        <value>hadoop03:19888</value>
    </property>
</configuration>
```

修改yarn.-site.xml配置文件

```
<configuration>
    <property>
        <name>yarn.resourcemanager.ha.enabled</name>
        <value>true</value>
    </property>
    <property>
        <name>yarn.resourcemanager.cluster-id</name>
        <value>cluster</value>
    </property>
    <property>
        <name>yarn.resourcemanager.ha.rm-ids</name>
        <value>rm1,rm2</value>
    </property>
    <property>
        <name>yarn.resourcemanager.hostname.rm1</name>
        <value>hadoop02</value>
    </property>
    <property>
        <name>yarn.resourcemanager.hostname.rm2</name>
        <value>hadoop01</value>
    </property>
    <property>
        <name>yarn.resourcemanager.webapp.address.rm1</name>
        <value>hadoop02:8088</value>
    </property>
    <property>
        <name>yarn.resourcemanager.webapp.address.rm2</name>
        <value>hadoop01:8088</value>
    </property>
    <property>
        <name>yarn.resourcemanager.zk-address</name>
        <value>hadoop01:2181,hadoop02:2181,hadoop03:2181</value>
    </property>
    <property>
    <name>yarn.nodemanager.recovery.enabled</name>
    <value>true</value>
    </property>
    <property>
        <name>yarn.resourcemanager.store.class</name>
    <value>org.apache.hadoop.yarn.server.resourcemanager.recovery.FileSystemRMStateStore</value>
    </property>
    <property>
        <name>yarn.nodemanager.aux-services</name>
        <value>mapreduce_shuffle</value>
    </property>
    <property>
        <name>yarn.log-aggregation-enable</name>
        <value>true</value>
    </property>
    <property>
        <name>yarn.log-aggregation.retain-seconds</name>
        <value>604800</value>
    </property>
    #设置nodemanager的端口
    <property>
        <name>yarn.nodemanager.address</name>
        <value>${yarn.nodemanager.hostname}:45454</value>
    </property>
    <property>
        <name>yarn.resourcemanager.address</name>
        <value>${yarn.nodemanager.hostname}:8032</value>
    </property>
    <property>
        <name>yarn.resourcemanager.resource-tracker.address</name>
        <value>${yarn.nodemanager.hostname}:8031</value>
    </property>
</configuration>
```

配置完成之后，需要分发到其他两台节点 （3号机发送到其它）

```
scp etc/hadoop/yarn-site.xml hadoop02:/opt/apps/hadoop-2.7.2/etc/hadoop/
scp etc/hadoop/yarn-site.xml hadoop01:/opt/apps/hadoop-2.7.2/etc/hadoop/
```

1号机启动yarn

```
sbin/start-yarn.sh
```

启动 resourcemanager 需要单独启动

在hadoop01节点上启动（单独启动）

```
sbin/yarn-daemon.sh start resourcemanager
```

在进行web访问的时候当RM为active的时候hadoop02访问的时候会自动转换到hadoop01的节点上。

五、nginx的安装部署和使用

nginx的简介

nginx的三大主要功能（静态web服务器、反向代理、负载均衡）

nginx安装

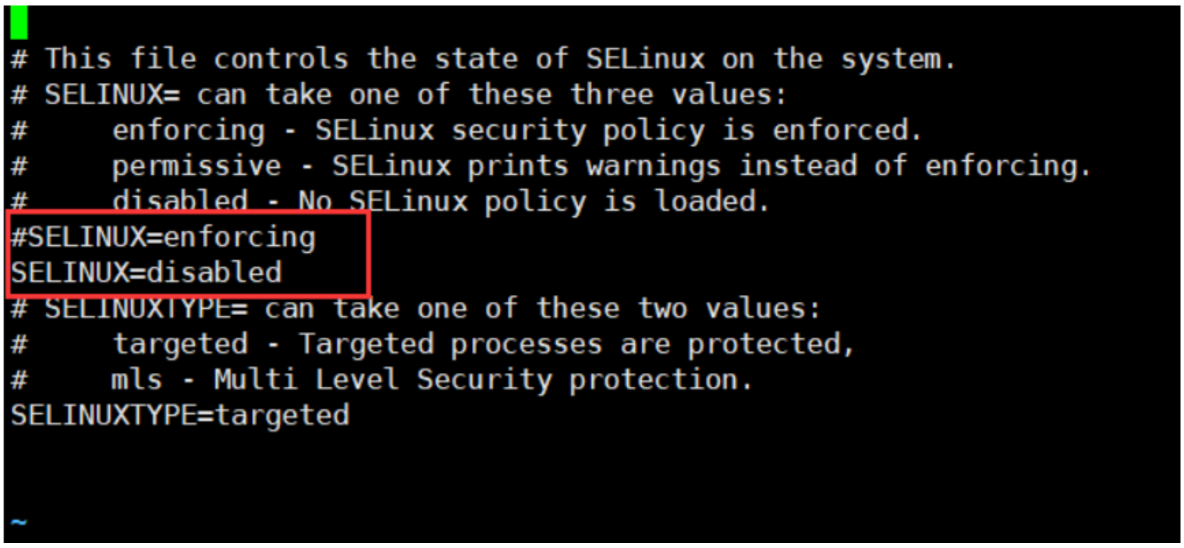
```
安装nginx 安装rpm包
rpm -ivh ./nginx-release-centos-6-0.el6.ngx.noarch.rpm

yum install nginx
```

nginx.conf的配置

- **虚拟机配置**

修改，如下配置，然后重启虚拟机。

```
vi /etc/selinux/config
```



```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
#SELINUX=enforcing
SELINUX=disabled
# SELINUXTYPE= can take one of these two values:
#     targeted - Targeted processes are protected,
#     mls - Multi Level Security protection.
SELINUXTYPE=targeted
~
```

- **3号机配置**

```
user root;
worker_processes auto;
error_log  /www/wwwlogs/nginx_error.log  crit;
pid        /www/server/nginx/logs/nginx.pid;
worker_rlimit_nofile 51200;

events
    {
        use epoll;
        worker_connections 51200;
        multi_accept on;
    }

http
    {
        include       mime.types;
                #include luawaf.conf;

                include proxy.conf;

        default_type  application/octet-stream;
        log_format  my_format  '$remote_addr - $remote_user [$time_local] "$request" '
                      '$status $body_bytes_sent "$http_referer" '
                      '"$http_user_agent" "$http_x_forwarded_for"';
        /***
        省略内容
        *****
        省略内容
        ****/
        upstream hadoopweb{   //负载均衡  核心内容
                server hadoop02:80;
                server hadoop01:80;
        }
        server{ //负载均衡  核心内容  hadoop
                listen 80;
                server_name hadoop03;
                location / {
                        proxy_pass http://hadoopweb;
                }
        }
        upstream hadoopwebR{ //负载均衡  核心内容
                server hadoop02:81;
                server hadoop01:81;
        }
        server{ //负载均衡  核心内容   Elasticsearch
                listen 81;
                server_name hadoop03;
                location / {
                        proxy_pass http://hadoopwebR;
                }
        }
}
```

2号机配置

```
user  root;
worker_processes  1;

error_log  /var/log/nginx/error.log warn;
pid        /var/run/nginx.pid;


events {
    worker_connections  1024;
}


http {
    include       /etc/nginx/mime.types;
    default_type  application/octet-stream;

    log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                      '$status $body_bytes_sent "$http_referer" '
                      '"$http_user_agent" "$http_x_forwarded_for"';

    log_format  hive_log   '"$time_iso8601",'
                           '"1",'
                           '"$remote_addr",'
                           '"$uri",'
                           '"$status",'
                           '"$host",'
                           '"$server_addr",'
                           '$body_bytes_sent,'
                           '$request_time,'
                           '"$http_referer",'
                           '"$http_user_agent"';

    log_format es_log '{"@timestamp":"$time_iso8601",'
                           '"@version":"1",'
                           '"client":"$remote_addr",'
                           '"url":"$uri",'
                           '"status":"$status",'
                           '"domain":"$host",'
                           '"host":"$server_addr",'
                            '"size":$body_bytes_sent,'
                           '"responsetime":$request_time,'
                           '"referer": "$http_referer",'
                           '"ua": "$http_user_agent"'
               '}';

    access_log  /var/log/nginx/access.log  main;

    sendfile        on;
    #tcp_nopush     on;

    keepalive_timeout  65;

    #gzip  on;

    include /etc/nginx/conf.d/*.conf;
        server {  // hadoop
                        #nginx监听的端口
                listen 80;
                #nginx发访问时的域名
                server_name hadoop02;
                        #本地文件
                location / {
                                #日志监听linux上的目录
                        access_log /var/log/nginx/access.log hive_log;
                        root /home/wwwroot;
                        index index.html;
                }
        }
        server {  //Elasticsearch
                        #nginx监听的端口
                listen 81;
                #nginx发访问时的域名
                server_name hadoop02;
                        #本地文件
                location / {
                                #日志监听linux上的目录
                        access_log /var/log/nginx/access_es.log es_log;
                        root /home/wwwroot;
                        index index.html;
                }
        }

}
```

1号机配置

```
user  root;
worker_processes  1;

error_log  /var/log/nginx/error.log warn;
pid        /var/run/nginx.pid;
events {
    worker_connections  1024;
}
http {
    include       /etc/nginx/mime.types;
    default_type  application/octet-stream;

    log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                      '$status $body_bytes_sent "$http_referer" '
                      '"$http_user_agent" "$http_x_forwarded_for"';
    log_format  hive_log   '"$time_iso8601",'
                           '"1",'
                           '"$remote_addr",'
                           '"$uri",'
                           '"$status",'
                           '"$host",'
                           '"$server_addr",'
                           '$body_bytes_sent,'
                           '$request_time,'
                           '"$http_referer",'
                           '"$http_user_agent"';

    log_format es_log '{"@timestamp":"$time_iso8601",'
                           '"@version":"1",'
                           '"client":"$remote_addr",'
                           '"url":"$uri",'
                           '"status":"$status",'
                           '"domain":"$host",
                           '"host":"$server_addr",'
                           '"size":$body_bytes_sent,'
                           '"responsetime":$request_time,'
                           '"referer": "$http_referer",'
                           '"ua": "$http_user_agent"'
               '}';

    access_log  /var/log/nginx/access.log  main;
    sendfile        on;
    #tcp_nopush     on;
    keepalive_timeout  65;
    #gzip  on;

    include /etc/nginx/conf.d/*.conf;

        server { //hadoop
                        #nginx监听的端口
                listen 80;
                #nginx发访问时的域名
                server_name hadoop01;
                        #本地文件
                location / {
                                #日志监听linux上的目录
                        access_log /var/log/nginx/access.log hive_log;
                        #root /usr/share/nginx/html;
                        root /home/wwwroot;
                        index xiaomiFK.html index.html index.htm;
                        #proxy_pass http://jzsstweb;
                }
        }

        server { //Elasticsearch
                        #nginx监听的端口
                listen 81;
                #nginx发访问时的域名
                server_name hadoop01;
                        #本地文件
                location / {
                                #日志监听linux上的目录
                        access_log /var/log/nginx/access_es.log es_log;
                        root /home/wwwroot;
                        index index.html;
                }
        }
}
```

六、flume的安装和使用

flume的安装

将按转包上传到linux上然后解压

```
tar -zxvf ./apache-flume-1.7.0-bin.tar.gz -C /opt/apps/
```

执行以下命令

```
bin/flume-ng version
```

出现相应的flume版本号就表示安装成功了

驱动复制

```
cp /opt/apps/hadoop-2.7.2/share/hadoop/common/*.jar /opt/apps/apache-flume-1.7.0-bin/lib/
cp /opt/apps/hadoop-2.7.2/share/hadoop/hdfs/*.jar /opt/apps/apache-flume-1.7.0-bin/lib/
cp /opt/apps/hadoop-2.7.2/share/hadoop/common/lib/*.jar /opt/apps/apache-flume-1.7.0-bin/lib/
cp /opt/apps/hadoop-2.7.2/share/hadoop/hdfs/lib/*.jar /opt/apps/apache-flume-1.7.0-bin/lib/
```

识别高可用

因为识别不了hdfs高可用的路径

那么让flume读取hadoop的配置文件就可以了

```
cp /opt/apps/hadoop-2.7.2/etc/hadoop/hdfs-site.xml /opt/apps/apache-flume-1.7.0-bin/conf/
```

2号机配置

主要实现监听nginx产生的日志文件。

在conf下创建log2hdfs.conf文件

```
a1.sources = r1
a1.sinks = k1
a1.channels = c1

a1.sources.r1.type = exec  //配置数据录入
a1.sources.r1.command = tail -f /var/log/nginx/access.log
a1.sources.r1.bind = hadoop02

a1.sinks.k1.type= hdfs   //配置数据存储位置
a1.sinks.k1.hdfs.path= hdfs://yzsen/hadoop02_log/%Y-%m-%d
a1.sinks.k1.hdfs.useLocalTimeStamp = true
a1.sinks.k1.hdfs.fileType=DataStream
a1.sinks.k1.hdfs.filePrefix =%Y-%m-%d %H-%M
a1.sinks.k1.hdfs.writeFormat=Text
a1.sinks.k1.hdfs.fileSuffix = .log
a1.sinks.k1.hdfs.rollSize = 128000000
a1.sinks.k1.hdfs.rollCount = 0
a1.sinks.k1.hdfs.rollInterval = 60

a1.channels.c1.type = memory   //配置中间过程
a1.channels.c1.capacity = 1000
a1.channels.c1.transactionCapacity=80

a1.sources.r1.channels = c1
a1.sinks.k1.channel = c1
```

1号机配置

主要实现监听nginx产生的日志文件。

在conf下创建log2hdfs.conf文件

```
a1.sources = r1
a1.sinks = k1
a1.channels = c1

a1.sources.r1.type = exec //配置
a1.sources.r1.command = tail -f /var/log/nginx/access.log
a1.sources.r1.bind = hadoop01

a1.sinks.k1.type= hdfs
a1.sinks.k1.hdfs.path= hdfs://yzsen/hadoop01_log/%Y-%m-%d
a1.sinks.k1.hdfs.useLocalTimeStamp = true
a1.sinks.k1.hdfs.fileType=DataStream
a1.sinks.k1.hdfs.filePrefix =%Y-%m-%d %H-%M
a1.sinks.k1.hdfs.writeFormat=Text
a1.sinks.k1.hdfs.fileSuffix = .log
a1.sinks.k1.hdfs.rollSize = 128000000
a1.sinks.k1.hdfs.rollCount = 0
a1.sinks.k1.hdfs.rollInterval = 60

a1.channels.c1.type = memory
a1.channels.c1.capacity = 1000
a1.channels.c1.transactionCapacity=80

a1.sources.r1.channels = c1
a1.sinks.k1.channel = c1
```

启动flume

```
bin/flume-ng agent -n a1 -c ./conf/ -f conf/log2hdfs.conf -Dflume.root.logger=INFO,console
```

七、hive的安装和使用

hive安装

1、 解压hive的安装包

```
tar -zvxf apache-hive-1.2.1-bin.tar.gz -C /opt/apps/
```

2、进入hive的安装目录中/conf目录下修改hive-env.sh.template 修改为hive-env.sh

```
cp hive-env.sh.template  hive-env.sh
```

3、编辑hive-env.sh

```
HADOOP_HOME=/opt/app/hadoop-2.7.2
```

mysql配置

1、针对hive需要安装的主机名和用户设置密码

password **请进行查询你自己设置的什么**

```
update mysql.user set password = '*81F5E21E35407D884A6CD4A731AEBFB6AF209E1B' where host = 'hadoop01'
and user = 'root';
```

2、重启mysql服务

```
service mysql restart
```

编辑hive-site.xml配置

```
<configuration>
    ## 配置元数据管理服务
    <property>
        <name>hive.metastore.uris</name>
        <value>thrift://hadoop03:9083</value>
    </property>
    ## 配置hdfs上hive文件的位置
    <property>
        <name>hive.metastore.warehouse.dir</name>
        <value>/user/hive/warehouse</value>
    <description>location of default database for the warehouse</description>
    </property>
    ## 关闭hive元数据的验证
    <property>
        <name>hive.metastore.schema.verification</name>
        <value>false</value>
    </property>
    ## 设置MySQL的连接
    <property>
```

```
        <name>javax.jdo.option.ConnectionURL</name>
        <value>jdbc:mysql://hadoop03:3306/hive?
createDatabaseIfNotExist=true&amp;useUnicode=true&amp;characterEncoding=UTF-8</value>
    </property>
    ## 设置MySQL的连接驱动
    <property>
        <name>javax.jdo.option.ConnectionDriverName</name>
        <value>com.mysql.jdbc.Driver</value>
    </property>
    ## 设置MySQL的连接用户名
    <property>
        <name>javax.jdo.option.ConnectionUserName</name>
        <value>root</value>
    </property>
    ## 设置MySQL的连接密码
    <property>
        <name>javax.jdo.option.ConnectionPassword</name>
        <value>root</value>
    </property>
    ## 设置显示数据库名称
    <property>
        <name>hive.cli.print.current.db</name>
         <value>true</value>
    </property>
    ## 设置表头
    <property>
        <name>hive.cli.print.header</name>
        <value>false</value>
    </property>
</configuration>
```

驱动复制

完成以上的配置之后，需要在hive中lib目录下，拷贝进去一份jdbc 驱动jar包

```
cp ./mysql-connector-java-5.1.31.jar /opt/app/apache-hive-1.2.1-bin/lib/
```

启动hive

在hive的根目录中输入以下命令

```
bin/hive --service metastore
&
bin/hive
```

hive的操作

将本地文件导入hive

1) 首先需要准备待上传的数据 /data/stduent.txt

2) 创建带分隔符的student表

```
create table student1(id int,name string)
row format delimited fields terminated by '\t' ;
```

3) 通过加载的方式把student.txt 加载到数据表中

```
load data local inpath '/data/student.txt' into table stud;
```

以上的内容就是hive的基本操作命令。这些命令必须掌握

将hdfs上的文件导入hive

1) 在hdfs上创建文件

```
./hdfs dfs -mkdir /data
```

2) 从linux本地上传文件到hdfs中

```
./hdfs dfs -put /opt/apps/student.txt /data
```

3) 将文件导入hive当中

```
load data inpath '/data/student.txt' into table student;
```

八、sqoop安装与使用

sqoop的安装

```
tar -zxvf ./sqoop-1.4.7.bin__hadoop-2.6.0.tar.gz -C /opt/apps/
```

解压sqoop的安装包， 并且修改权限

配置sqoop-env.sh

```
export HADOOP_COMMON_HOME=/opt/apps/hadoop-2.7.2
export HADOOP_MAPRED_HOME=/opt/apps/hadoop-2.7.2
export HIVE_HOME=/opt/apps/apache-hive-1.2.1-bin
```

把mysql jar包拷贝到sqoop lib的目录下

```
cp /opt/softs/mysql-connector-java-5.1.31.jar /opt/apps/sqoop-1.4.7.bin__hadoop-2.6.0/lib/
```

测试配置是否成功

```
bin/sqoop list-databases --connect jdbc:mysql://hadoop01:3306 --username root --password root
```

连接成功之后，会显示数据库所有数据库列表

```
information_schema
metastore
mysql
performance_schema
test
```

导出到Mysql

```
bin/sqoop export --connect jdbc:mysql://hadoop03:3306/yzsen --table hadoop01_log --username root --
password root     --fields-terminated-by ','   --lines-terminated-by '\n' --export-dir
/user/hive/warehouse/yzsen.db/hadoop01_log/*
```

九、Elasticsearch安装与使用

Elasticsearch安装

预备

使用普通用户启动Elasticsearch，可以创建一个普通用户

1、新建用户：

```
# useradd -m es
# passwd es
```

2、为新用户添加sudo权限：

```
打开配置文件
vi /etc/sudoers

root    ALL=(ALL)       ALL
es   ALL=(ALLL)   ALL
```

解压Elasticsearch安装包（用普通用户操作）

```
tar -zxvf ./elasticsearch-5.2.2.tar.gz -C /opt/apps
```

在elasticsearch-5.2.2路径下创建data和logs文件夹

```
mkdir data
mkdir logs
```

修改elasticsearch-5.2.2文件夹所属用户组和所属于用户(**如果Elasticsearch报错，建议先运行这一条了，再启动**)

```
chown -R es:es ./elasticsearch-5.2.2/
```

修改配置文件/opt/apps/elasticsearch-5.2.2/config/elasticsearch.yml

```
//参考对应修改
cluster.name: my-es

node.name: hadoop-3

path.data: data
path.logs: logs

bootstrap.memory_lock: false
bootstrap.system_call_filter: false

network.host: hadoop03

discovery.zen.ping.unicast.hosts: ["hadoop01", "hadoop02", "hadoop03"]

http.cors.enabled: true
http.cors.allow-origin: "*"
```

注意

```
（1）cluster.name
```

```
如果要配置集群需要两个节点上的elasticsearch配置的cluster.name相同，都启动可以自动组成集群，这里如果不改
cluster.name则默认是cluster.name=my-application，

（2）nodename随意取但是集群内的各节点不能相同

（3）修改路径
    path.data: /opt/apps/elasticsearch-5.2.2/data
    path.logs: /opt/apps/elasticsearch-5.2.2/logs

（4）修改
    bootstrap.memory_lock: false
    增加
    bootstrap.system_call_filter: false

（5）修改成你当前的主机名IP
    network.host: 192.168.56.41
（6）修改主机名
    discovery.zen.ping.unicast.hosts: ["hadoop01"]
（7）修改后的每行前面不能有空格，修改后的“：”后面必须有一个空格
```

修改虚拟机配置文件

1、切换到root用户，编辑limits.conf 添加类似如下内容

```
[root@hadoop102 elasticsearch-5.2.2]# vi /etc/security/limits.conf

添加如下内容:

* soft nofile 65536
* hard nofile 131072
* soft nproc 2048
* hard nproc 4096
```

（2）切换到root用户，进入limits.d目录下修改配置文件。

```
[root@hadoop102 elasticsearch-5.2.2]# vi /etc/security/limits.d/90-nproc.conf

修改如下内容：

* soft nproc 1024

#修改为

* soft nproc 2048
```

（3）切换到root用户修改配置sysctl.conf

```
[root@hadoop102 elasticsearch-5.2.2]# vi /etc/sysctl.conf

添加下面配置：

vm.max_map_count = 655360
```

4、切换普通用户

```
bin/elasticsearch   //启动集群

如果报错
先在root下执行
chown -R es:es ./elasticsearch-5.2.2/
试试
```

5、测试集群

访问 hadoop01:9200

Elasticsearch head 插件的安装

只需要配置其中一台

首先要安装nodejs

```
tar -zxvf node-v6.9.2-linux-x64.tar.gz -C /opt/apps
```

为了使用命令，需要配置环境变量

```
vi /etc/profile
export NODE_HOME=/opt/apps/node-v6.9.2-linux-x64
export PATH=$PATH:$NODE_HOME/bin

刷新环境变量
source /etc/profile
```

查看安装的版本

```
[root@hadoop102 software]# node -v
v6.9.2
[root@hadoop102 software]# npm -v
3.10.9
```

解压head插件到/opt/apps目录下

```
unzip elasticsearch-head-master.zip -d /opt/apps/
```

查看当前head插件目录下有无node_modules/grunt目录：

没有：执行命令创建：(**注意：如果长时间不动，去掉**--registry=xxxxx...**试试**)

```
npm install grunt --save --registry=https://registry.npm.taobao.org
```

安装head插件：

```
npm install -g cnpm --registry=https://registry.npm.taobao.org
```

安装grunt：

```
npm install -g grunt-cli --registry=https://registry.npm.taobao.org
```

编辑Gruntfile.js

```
vim Gruntfile.js

options: {
        hostname: '0.0.0.0',
        port: 9100,
        base: '.',
        keepalive: true
      }
```

文件93行添加hostname:'0.0.0.0'

检查head根目录下是否存在base文件夹

没有：将 _site下的base文件夹及其内容复制到head根目录下(**这个目录是一些静态资源**)

```
cp -r ./base/ ../
```

启动grunt server：

```
grunt server -d
```

```
Running "connect:server" (connect) task

[D] Task source: /opt/module/elasticsearch-head-master/node_modules/grunt-contrib-
connect/tasks/connect.js

Waiting forever...

Started connect web server on http://localhost:9100
```

如果提示grunt的模块没有安装：

```
Local Npm module “grunt-contrib-clean” not found. Is it installed?

Local Npm module “grunt-contrib-concat” not found. Is it installed?

Local Npm module “grunt-contrib-watch” not found. Is it installed?

Local Npm module “grunt-contrib-connect” not found. Is it installed?

Local Npm module “grunt-contrib-copy” not found. Is it installed?

Local Npm module “grunt-contrib-jasmine” not found. Is it installed?

Warning: Task “connect:server” not found. Use –force to continue.
```

执行以下命令：

```
npm install grunt-contrib-clean -registry=https://registry.npm.taobao.org

npm install grunt-contrib-concat -registry=https://registry.npm.taobao.org

npm install grunt-contrib-watch -registry=https://registry.npm.taobao.org

npm install grunt-contrib-connect -registry=https://registry.npm.taobao.org

npm install grunt-contrib-copy -registry=https://registry.npm.taobao.org

npm install grunt-contrib-jasmine -registry=https://registry.npm.taobao.org
```

最后一个模块可能安装不成功，但是不影响使用。

测试连接

```
http://hadoop:9100    //请注意你访问的那一台机
```

十、Logstash的安装和使用

Logstash的安装

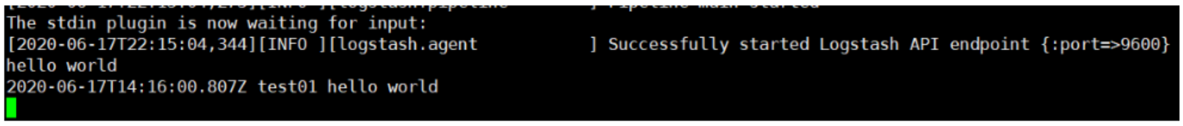
解压

```
tar -zxvf ./logstash-5.5.2.tar.gz -C /opt/apps/
```

来到bin目录下测试安装是否成功

```
bin/logstash -e 'input { stdin {} } output { stdout {} }'
```

类似于即可



```
The stdin plugin is now waiting for input:
[2020-06-17T22:15:04,344][INFO ][logstash.agent           ] Successfully started Logstash API endpoint {:port=>9600}
hello world
2020-06-17T14:16:00.807Z test01 hello world
```

Logstash的配置

2号机配置

在软件根目录文件夹config下创建nginx2es.conf文件

注意格式，一点不能错

```
input {
    file {
      path => "/var/log/nginx/access_es.log"
      type => "nginx-log"
      codec => json
      start_position => "beginning"
    }
}
output {
       if [type] == "nginx-log"{
        elasticsearch {
           hosts => ["hadoop02:9200"]
           index => "hadoop02-nginx-log-%{+YYYY.MM.dd}"
        }
    }
}
```

1号机配置

在软件根目录文件夹config下创建nginx2es.conf文件

```
input {
    file {
      path => "/var/log/nginx/access_es.log"
      type => "nginx-log"
      codec => json
      start_position => "beginning"
    }
}
output {

       if [type] == "nginx-log"{
        elasticsearch {
           hosts => ["hadoop01:9200"]
           index => "hadoop01-nginx-log-%{+YYYY.MM.dd}"
        }
    }
}
```

启动命令

```
bin/logstash -f config/nginx2es.conf
```

十一、kibana的安装和使用

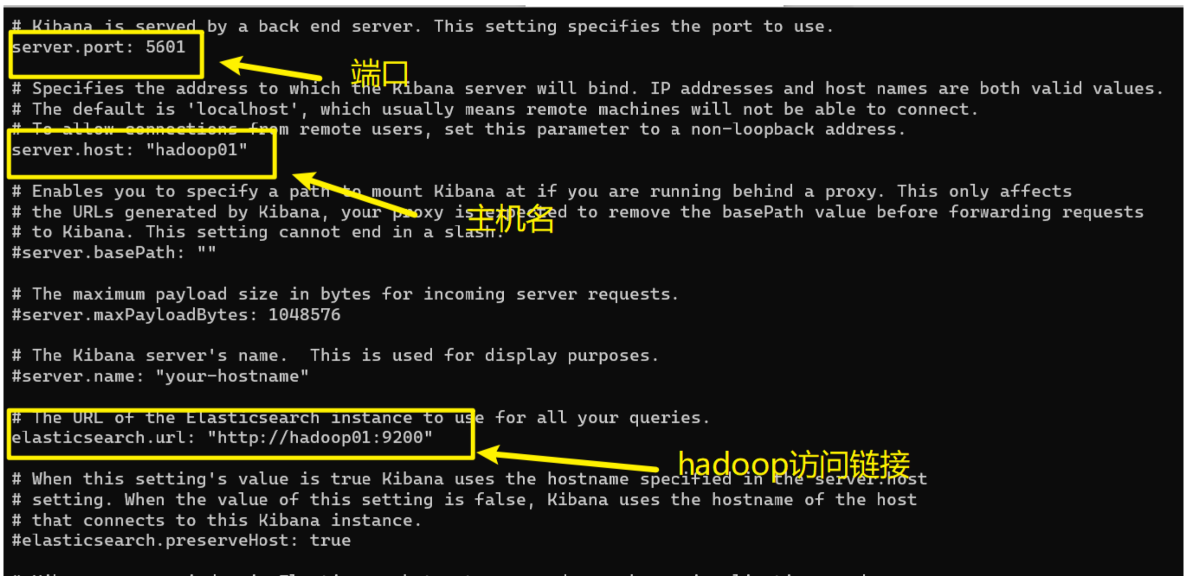
kibana安装

只需安装一台

解压

```
tar -zxvf ./kibana-5.2.2-linux-x86_64.tar.gz -C /opt/apps/
```

配置 kibana.ym文件



```
# Kibana is served by a back end server. This setting specifies the port to use.
server.port: 5601

# Specifies the address to which the Kibana server will bind. IP addresses and host names are both valid values.
# The default is 'localhost', which usually means remote machines will not be able to connect.
# To allow connections from remote users, set this parameter to a non-loopback address.
server.host: "hadoop01"

# Enables you to specify a path to mount Kibana at if you are running behind a proxy. This only affects
# the URLs generated by Kibana, your proxy is expected to remove the basePath value before forwarding requests
# to Kibana. This setting cannot end in a slash.
#server.basePath: ""

# The maximum payload size in bytes for incoming server requests.
#server.maxPayloadBytes: 1048576

# The Kibana server's name.  This is used for display purposes.
#server.name: "your-hostname"

# The URL of the Elasticsearch instance to use for all your queries.
elasticsearch.url: "http://hadoop01:9200"

# When this setting's value is true Kibana uses the hostname specified in the server.host
# setting. When the value of this setting is false, Kibana uses the hostname of the host
# that connects to this Kibana instance.
#elasticsearch.preserveHost: true
```

启动命令

```
bin/kibana
```

十二、环境启动命令

离线

- **启动zookeeper(3台)**

```
#顺序 1、2、3
/opt/apps/zookeeper-3.4.10/bin/zkServer.sh  start
```

- ## 3号机启动HDFS

```
/opt/apps/hadoop-2.7.2/sbin/start-dfs.sh
```

HDFS其它命令

```
#上传
bin/hdfs dfs -put 本机路径  hdfs路径
#删除
bin/hdfs dfs -rm -R /hadoop01_log
bin/hdfs dfs -rm -R /hadoop02_log
```

- ## 1号机启动yarn

```
sbin/start-yarn.sh
```

- ## 2号机启动ResourceManager

```
sbin/yarn-daemon.sh start resourcemanager
```

- ## 启动1、2号机flume

```
bin/flume-ng agent -n a1 -c ./conf/ -f conf/log2hdfs.conf -Dflume.root.logger=INFO,console
```

- ## 3号机启动hive

```
bin/hive --service metastore
bin/hive
```

其它hive命令

```
#hive表创建
create table hadoop01_log(
        time string,
        version string,
        client string,
        url string,
        status string ,
        domain  string,
        host string,
        size int,
        responsetime float ,
        referer string,
        ua string
)row format delimited fields terminated by ',' ;
#数据导入
load data inpath '/hadoop02_log' into table  yzsen.hadoop02_log;
load data inpath '/hadoop01_log' into table yzsen.hadoop01_log;
```

Mysql表创建

```
create table hadoop01_log(
        time varchar(50),
        version varchar(10),
        client varchar(30),
        url varchar(30),
        status varchar(10) ,
        domain  varchar(20),
        host varchar(30),
        size int,
        responsetime float ,
        referer varchar(10),
        ua varchar(150)
        ) ;
```

- ### sqoop命令

```
bin/sqoop export --connect jdbc:mysql://hadoop03:3306/yzsen --table hadoop01_log --username root --
password root     --fields-terminated-by ','   --lines-terminated-by '\n' --export-dir
/user/hive/warehouse/yzsen.db/hadoop01_log/*

bin/sqoop export --connect jdbc:mysql://hadoop03:3306/yzsen --table hadoop02_log --username root --
password root     --fields-terminated-by ','   --lines-terminated-by '\n' --export-dir
/user/hive/warehouse/yzsen.db/hadoop02_log/*
```

- ### 注意

```
注意路径
每次使用hive前需要清空hive文件夹 或者 mysql ，如果报错就清空一下hive
```

实时

- ### Elasticsearch启动(3台)

```
su es
bin/elasticsearch
```

- ### 2号机head启动

```
grunt server -d
```

- ### 1 2Logstash启动

```
bin/logstash -f config/nginx2es.conf
```

- ### 1号机启动kibana

```
bin/kibana
#hadoop01:5601
```