Kvm-cloud部署手册

目录

[Kvm-cloud部署手册 1](#_Toc130328470)

[1.系统需求 3](#_Toc130328471)

[2.网络检查 3](#_Toc130328472)

[3.基础配置 4](#_Toc130328473)

[3.1配置nginx的yum源 4](#_Toc130328474)

[3.2配置mysql的yum源 4](#_Toc130328475)

[3.3更新yum包并基础安装 5](#_Toc130328476)

[3.4修改主机名 5](#_Toc130328477)

[3.5关闭防火墙和selinux 6](#_Toc130328478)

[4.配置系统优化 6](#_Toc130328479)

[5.NFS配置 8](#_Toc130328480)

[5.1安装NFS软件包 8](#_Toc130328481)

[5.2设置开机自启以及启动服务 8](#_Toc130328482)

[5.3新建存储文件夹 8](#_Toc130328483)

[5.4编辑配置 8](#_Toc130328484)

[5.5重新加载exportfs文件 9](#_Toc130328485)

[5.6编辑nfs配置 9](#_Toc130328486)

[5.7重新服务 10](#_Toc130328487)

[5.8测试挂载是否正常 10](#_Toc130328488)

[6.KVM配置 10](#_Toc130328489)

[6.1验证主机是否支持虚拟化 10](#_Toc130328490)

[6.2安装KVM 11](#_Toc130328491)

[6.3配置KVM网桥，增加一个网桥（br0） 11](#_Toc130328492)

[6.4 VNC 配置 12](#_Toc130328493)

[6.5 Libvirtd配置 13](#_Toc130328494)

[6.6 配置mvn命令 14](#_Toc130328495)

[6.7 项目编译 14](#_Toc130328496)

[6.8 安装并导入mysql数据 15](#_Toc130328497)

[6.9 安装redis 16](#_Toc130328498)

[6.10修改项目的配置文件 17](#_Toc130328499)

[6.11启动管理端和客户端 20](#_Toc130328500)

[6.12配置nginx文件下载 20](#_Toc130328501)

[7.kvm-cloud平台基础操作 21](#_Toc130328502)

[7.1创建基础网络 22](#_Toc130328503)

[7.2创建主机 22](#_Toc130328504)

[7.3创建存储池(只支持nfs) 23](#_Toc130328505)

[7.4上传基础模板 23](#_Toc130328506)

[7.5创建虚拟机 23](#_Toc130328507)

[8.相关问题 24](#_Toc130328508)

[8.1找不到配置文件问题导致数据库连接问题 24](#_Toc130328509)

[8.2关于备份与恢复 24](#_Toc130328510)

[8.3关于网络隔离 25](#_Toc130328511)

[8.4 windows系统附加磁盘不识别问题 25](#_Toc130328512)

[8.5服务器掉电重启后处理 25](#_Toc130328513)

[8.6虚拟机虚拟化嵌套 25](#_Toc130328514)

[9项目一键启停脚本 26](#_Toc130328515)

KVM Cloud是一款基于KVM实现虚拟机管理系统，支持如下功能:

1、基于KVM的VM基础功能(创建、启动、停止、重装、webVNC等功能)

2、使用NFS作为磁盘存储池

3、支持磁盘动态添加取消

4、多主机管理

5、支持模版维护，用于快速创建VM

6、简单群组功能

7、虚拟机IP自动管理

8、多网卡支持

9、支持磁盘导入导出

10、支持 raw、qcow、qcow2、vdi、vmdk、vpc磁盘格式

11、磁盘快照支持

# 1.系统需求

操作系统：linux

部署资源需求：最低4核8G，磁盘300G

本次部署使用的时centos7的操作系统，其他linux系统也可进行部署。

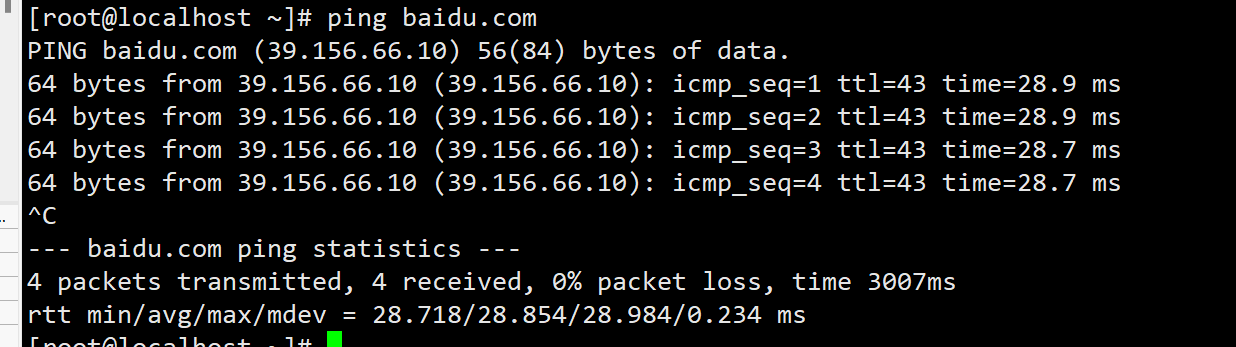
IP是192.168.190.9

# 2.网络检查

登录主机，检查是否可以通外网和更新yum源

|  |
| --- |
| ping baidu.com |

检查可以ping通即可



更新本地yum源，repo源可以通过阿里云开源镜像站下载国内的yum源地址：

|  |
| --- |
| <https://developer.aliyun.com/mirror/?spm=a2c6h.13651102.0.0.1f8a1b116W7S92&serviceType=mirror> |
| $ yum –y install wget  $ cd /etc/yum.repos.d/  $ mkdir bak  $ mv Cent\* bak/  $ wget -O /etc/yum.repos.d/CentOS-Base.repo <https://mirrors.aliyun.com/repo/Centos-7.repo>  $ wget -O /etc/yum.repos.d/epel.repo <https://mirrors.aliyun.com/repo/epel-7.repo> |

# 3.基础配置

## 3.1配置nginx的yum源

|  |
| --- |
| $ cd /etc/yum.repos.d/  $ touch nginx.repo  $ cat > nginx.repo <<EOF  [nginx-stable]  name=nginx stable repo  baseurl=http://nginx.org/packages/centos/$releasever/$basearch/  gpgcheck=1  enabled=1  gpgkey=https://nginx.org/keys/nginx\_signing.key  module\_hotfixes=true  [nginx-mainline]  name=nginx mainline repo  baseurl=http://nginx.org/packages/mainline/centos/$releasever/$basearch/  gpgcheck=1  enabled=0  gpgkey=https://nginx.org/keys/nginx\_signing.key  module\_hotfixes=true  EOF |

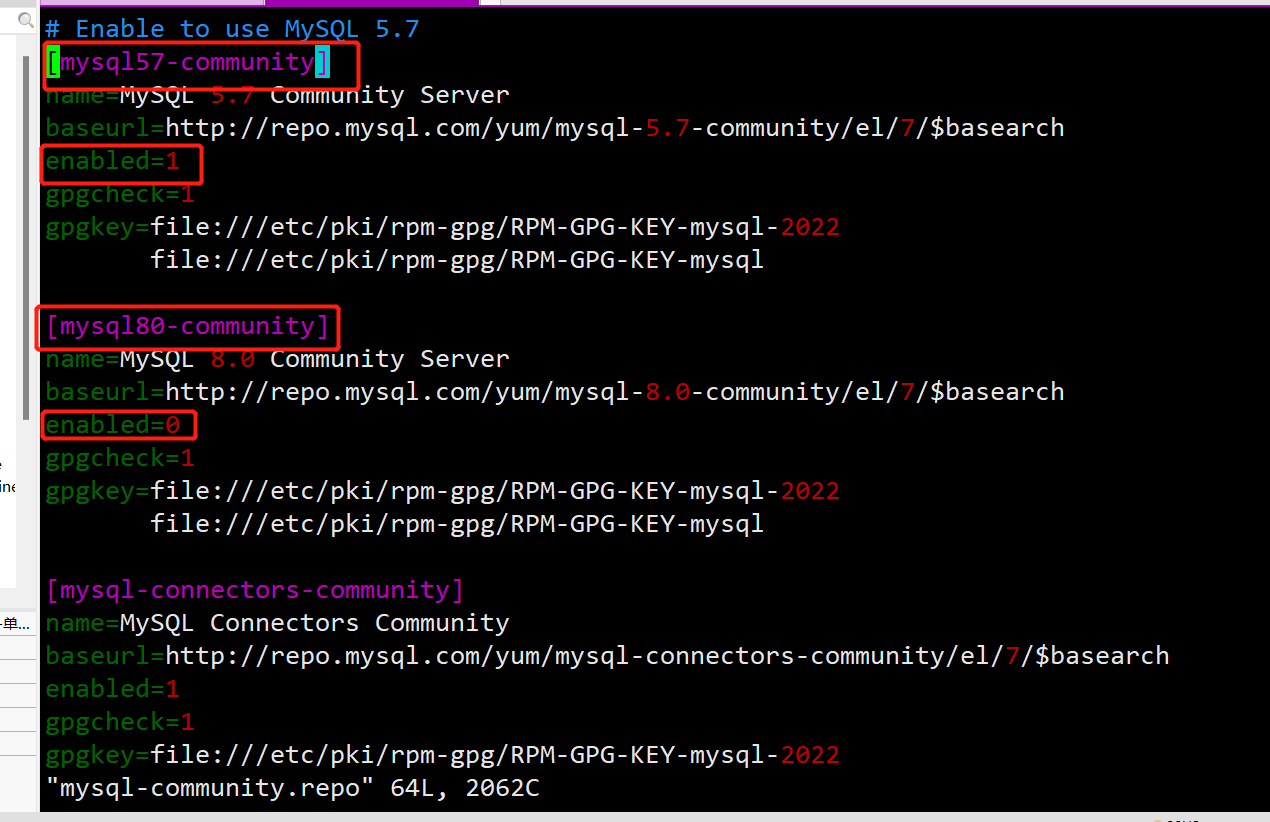
## 3.2配置mysql的yum源

将mysql80-community-release-el7-7.noarch.rpm包上传到主机的/root目录下。浏览器访问以下地址将包下载到本地：

|  |
| --- |
| http://192.168.190.3/download/mysql80-community-release-el7-7.noarch.rpm |

上传完成后，进行安装

|  |
| --- |
| $ rpm –ivh mysql80-community-release-el7-7.noarch.rpm  $ cd /etc/yum.repos.d/  $ vim mysql-community.repo  将mysql80-community的enable关闭，将mysql57-community的enable打开，保存退出（：wq） |

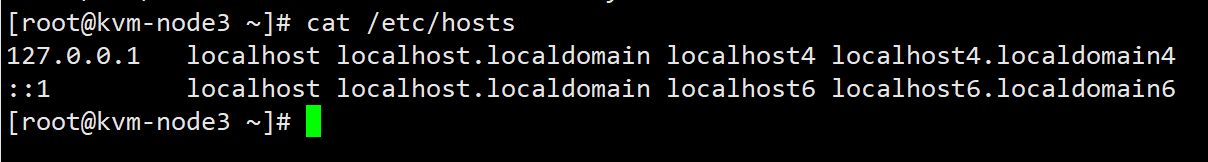


## 3.3 更新yum包并基础安装

|  |
| --- |
| $ yum clean all  $ yum makecache  $ yum –y upgrade (时间比较长)  $ yum –y install vim wget curl bash-completion net-tools lrzsz smarttools |

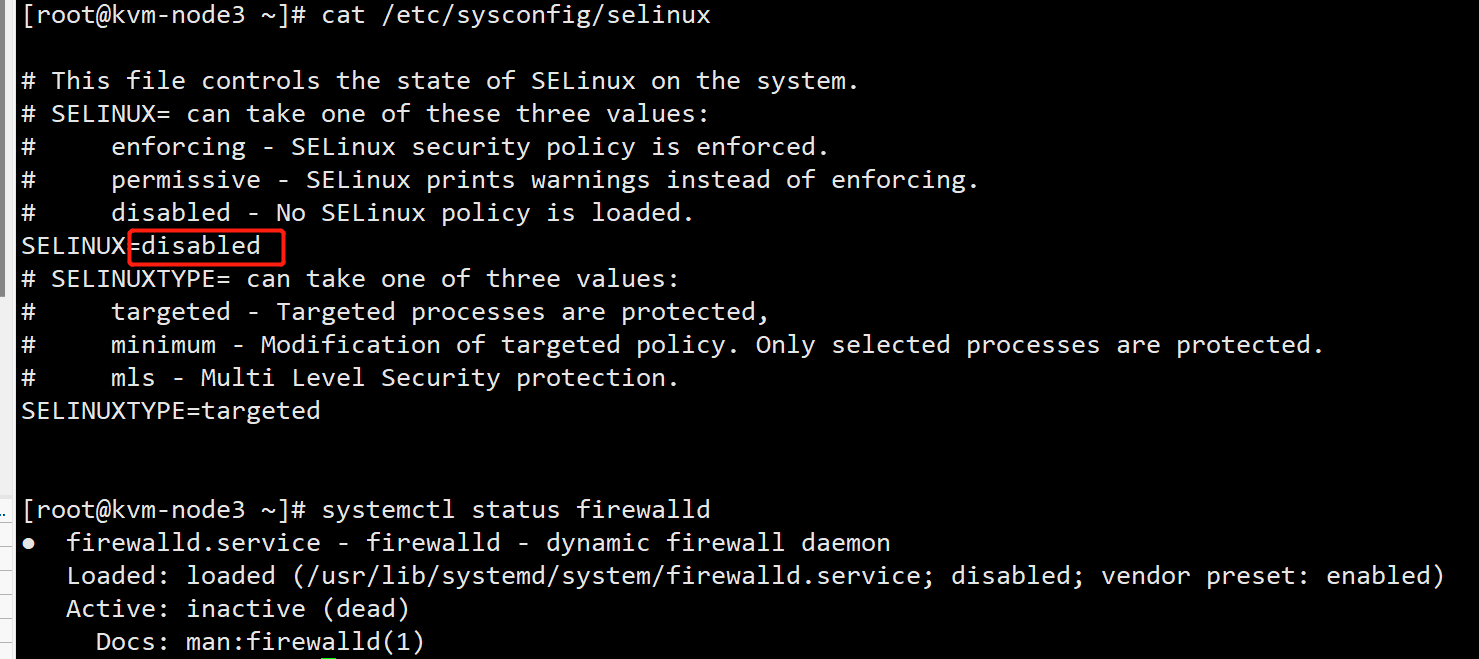
## 3.4 修改主机名

|  |
| --- |
| $ hostnamectl set-hostname kvm-cloud  $ echo "192.168.190.9 kvm-cloud" >> /etc/hosts |



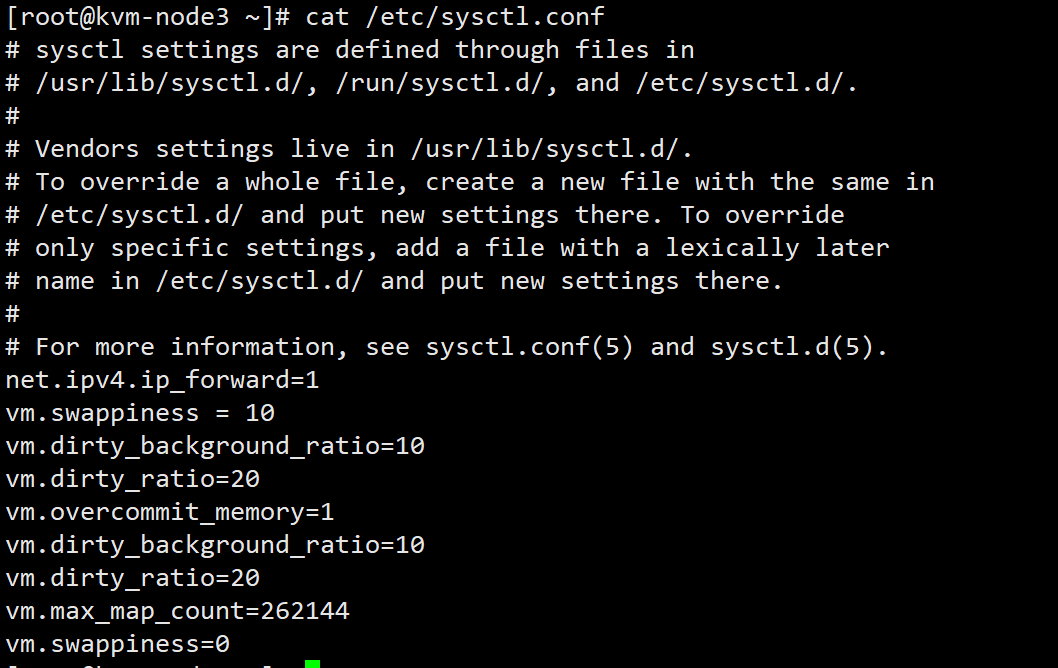
## 3.5 关闭防火墙和selinux

|  |
| --- |
| $ vi /etc/selinux/config  $ SELINUX=disabled  将SELINUX修改为disabled  $ setenforce 0  $ systemctl stop firewalld  $ systemctl disable firewalld  $ systemctl stop iptables  $ systemctl disable iptables |



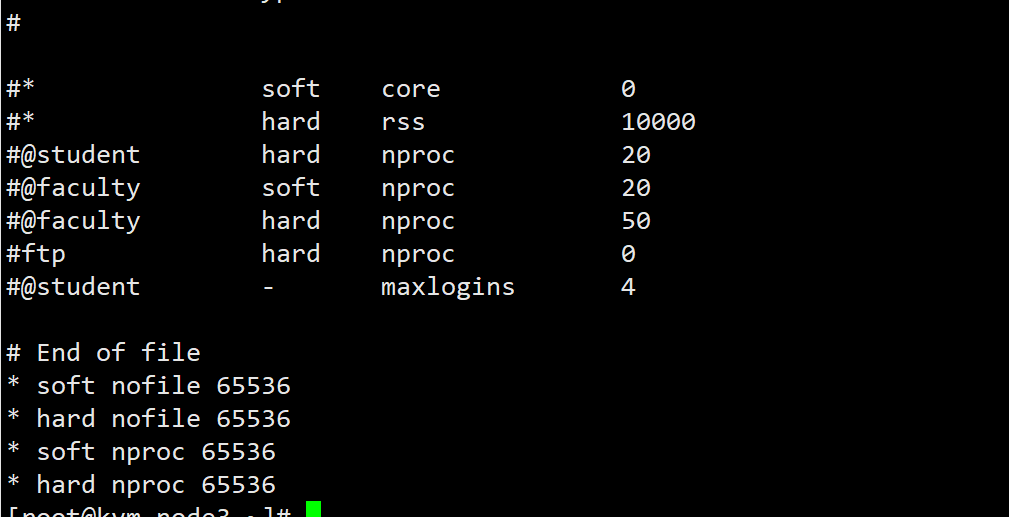
# 4.配置系统优化

|  |
| --- |
| $ cat >> /etc/sysctl.conf << EOF  net.ipv4.ip\_forward=1  vm.swappiness = 10  vm.dirty\_background\_ratio=10  vm.dirty\_ratio=20  vm.overcommit\_memory=1  vm.dirty\_background\_ratio=10  vm.dirty\_ratio=20  vm.max\_map\_count=262144  vm.swappiness=0EOF  $ sysctl -p |



配置linux打开文件的最大数和最大进程数

|  |
| --- |
| $ cat >> /etc/security/limits.conf << EOF  \* soft nofile 65536  \* hard nofile 65536  \* soft nproc 65536  \* hard nproc 65536  EOF |



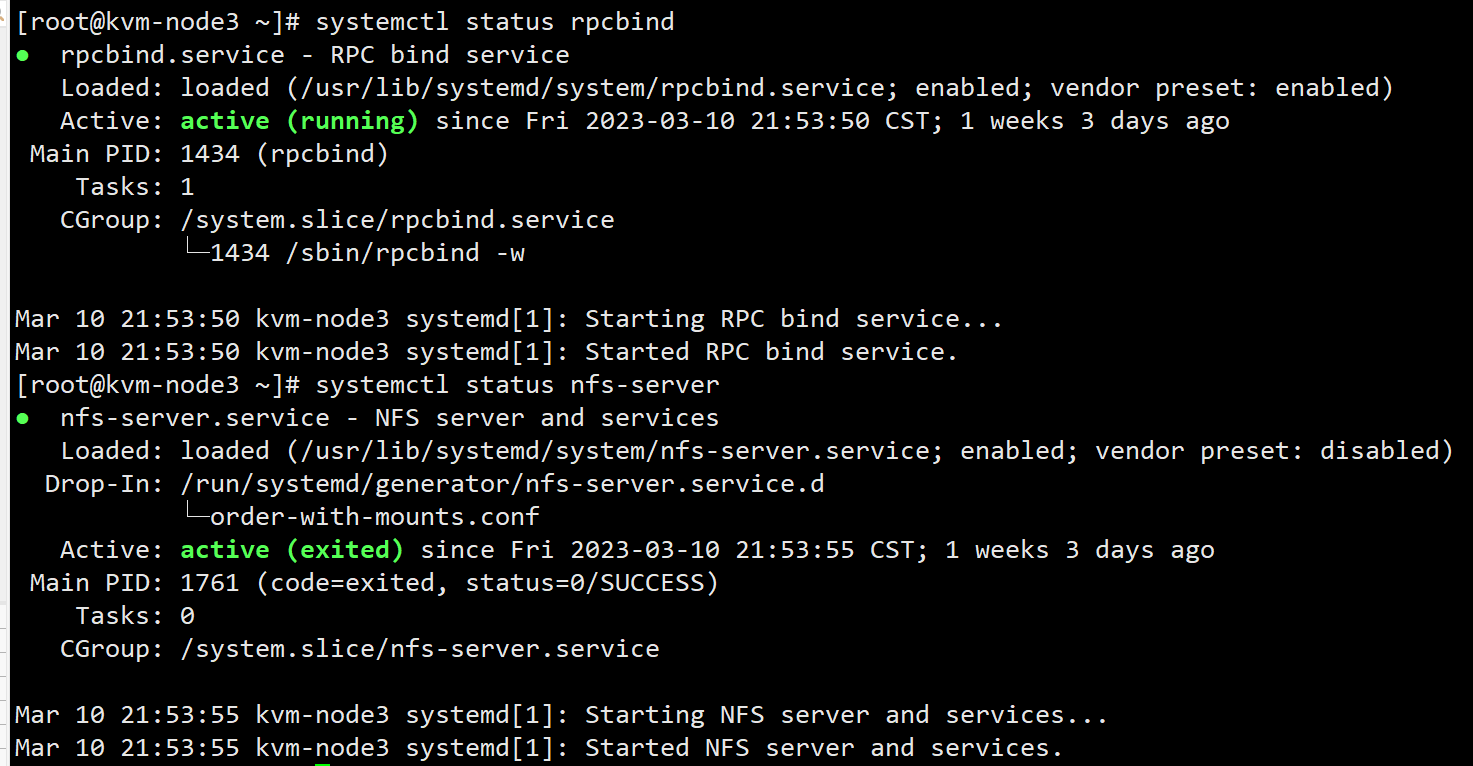
# 5.NFS配置

## 5.1 安装NFS软件包

|  |
| --- |
| yum -y install nfs-utils rpcbind |

## 5.2 设置开机自启以及启动服务

|  |
| --- |
| systemctl enable rpcbind  systemctl enable nfs-server  systemctl start rpcbind  systemctl start nfs-server |

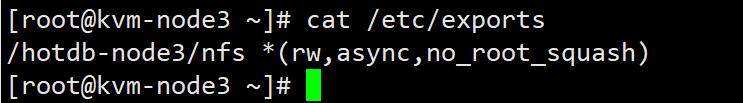


## 5.3 新建存储文件夹

|  |
| --- |
| mkdir -p /hotdb/nfs ##自定义 |

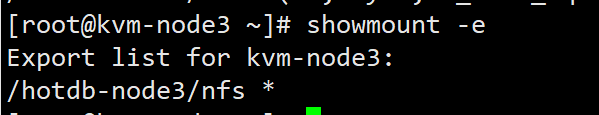
## 5.4 编辑配置

|  |
| --- |
| cat >> /etc/exports << EOF  /data/nfs \*(rw,async,no\_root\_squash)  EOF  cat /etc/exports  查看是否添加成功 |



## 5.5 重新加载exportfs文件

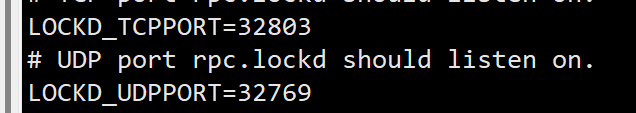
|  |
| --- |
| $ exportfs –a  测试nfs  $ showmount -e |

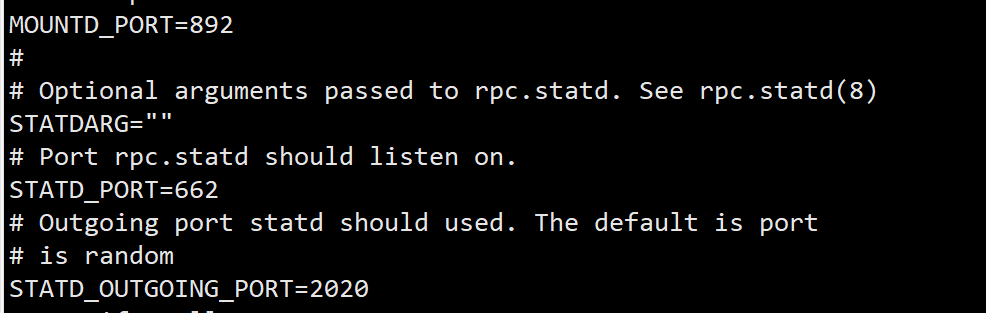


## 5.6 编辑nfs配置

将/etc/sysconfig/nfs配置文件的数据参数修改为如下所示，其他配置不变：

|  |
| --- |
| $ vi /etc/sysconfig/nfs  LOCKD\_TCPPORT=32803  LOCKD\_UDPPORT=32769  MOUNTD\_PORT=892  RQUOTAD\_PORT=875  STATD\_PORT=662  STATD\_OUTGOING\_PORT=2020 |





因nfs版本不同，以上参数可能有些没有或者有差异，没有的参数可不配置。

## 5.7 重新服务

|  |
| --- |
| $ systemctl restart rpcbind  $ systemctl restart nfs-server |
| $ systemctl status rpcbind |
| $ systemctl status nfs-server |

## 5.8 测试挂载是否正常

|  |
| --- |
| $ mount -t nfs 127.0.0.1:/hotdb/nfs /mnt  $ df -h ###查看有了代表成功  $ umount /mnt |

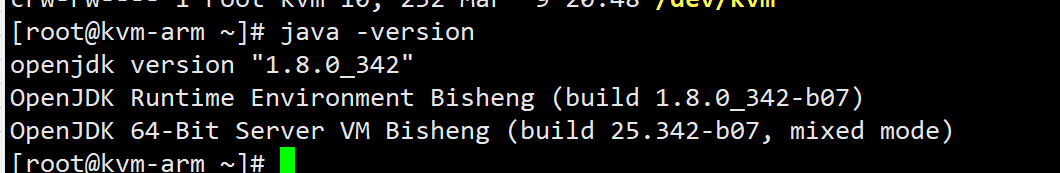
# 6.KVM配置

## 6.1 验证主机是否支持虚拟化

|  |
| --- |
| X86机器验证虚拟化：  $ lsmod | grep kvm #查看结果确认是否支持虚拟化    Arm机器验证虚拟化：  $ ls /sys/module/kvm  或者  $ ll /dev/kvm |

## 6.2 安装KVM

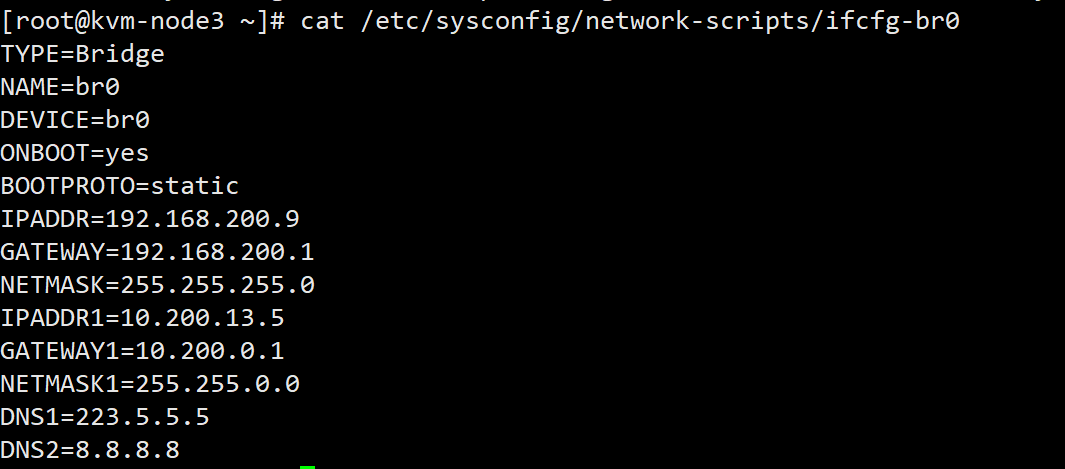
|  |
| --- |
| $ yum install qemu-kvm libvirt bridge-utils  $ yum install java-1.8.0-openjdk\* -y  验证java:  java –version |



## 6.3 配置KVM网桥，增加一个网桥（br0）

这一步一定注意：使用ip addr查看你的网卡名，在CentOS 7中网卡名可能不是eth0，错误的网卡名会导致后期配置的虚拟机无法正常被访问到！ 确认网卡名无误后配置网桥：

|  |
| --- |
| $ touch /etc/sysconfig/network-scripts/ifcfg-br0  $ cat > /etc/sysconfig/network-scripts/ifcfg-br0 <<EOF  DEVICE="br0"  TYPE="Bridge"  ONBOOT="yes"  BOOTPROTO=static  IPADDR=192.168.2.130  NATMASK=255.255.255.0  PREFIX=24  GATEWAY=192.168.2.1  DNS1=223.5.5.5  DNS2=8.8.8.8  EOF |



# 务必使用 ip addr 命令查看你的实际网卡名，他很可能不是eth0

# 下面这个编辑命令注意改成 ifcfg-实际网卡名

|  |
| --- |
| $ vi /etc/sysconfig/network-scripts/ifcfg-eth0  # 下面这个DEVICE注意改成 实际网卡名  DEVICE="eth0"  NM\_CONTROLLED="no"  ONBOOT="yes"  TYPE="Ethernet"  BOOTPROTO=none  BRIDGE="br0" |



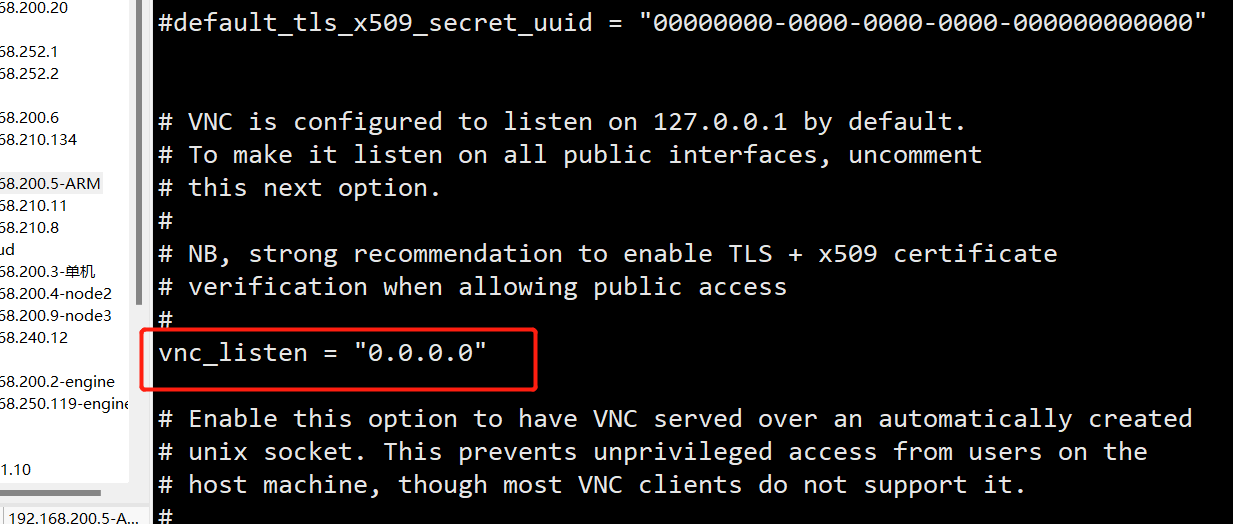
重启网卡：

|  |
| --- |
| Centos7:  $ systemctl restart network  Centos8:  $ nmcli c reload  $ nmcli c up 实际的物理网卡名 |

## 6.4 VNC 配置

修改/etc/libvirt/qemu.conf配置文件，将vnc\_listen参数修改为0.0.0.0

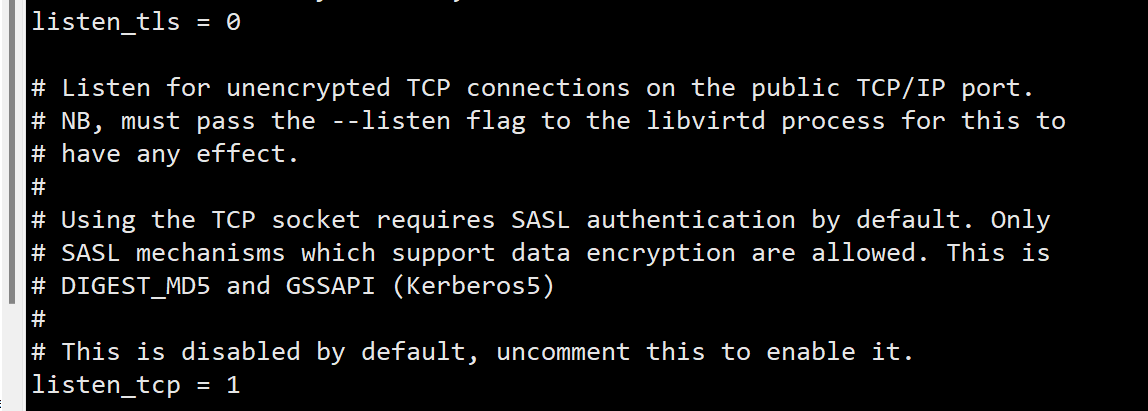
|  |
| --- |
| $ vi /etc/libvirt/qemu.conf  vnc\_listen="0.0.0.0" |



## 6.5 Libvirtd配置

修改/etc/libvirt/libvirtd.conf配置文件，将以下参数修改为对应的数据。

|  |
| --- |
| $ vi /etc/libvirt/libvirtd.conf  listen\_tls = 0  listen\_tcp = 1  tcp\_port = "16059"  auth\_tcp = "none"  mdns\_adv = 0 |



修改/etc/sysconfig/libvirtd配置文件，将LIBVIRTD\_ARGS修改为—listen(默认可能已经有了该配置，但是被注释掉，则将注释打开)

|  |
| --- |
| $ vi /etc/sysconfig/libvirtd  LIBVIRTD\_ARGS="--listen"    ##重启libvirted服务  $ systemctl restart libvirtd |

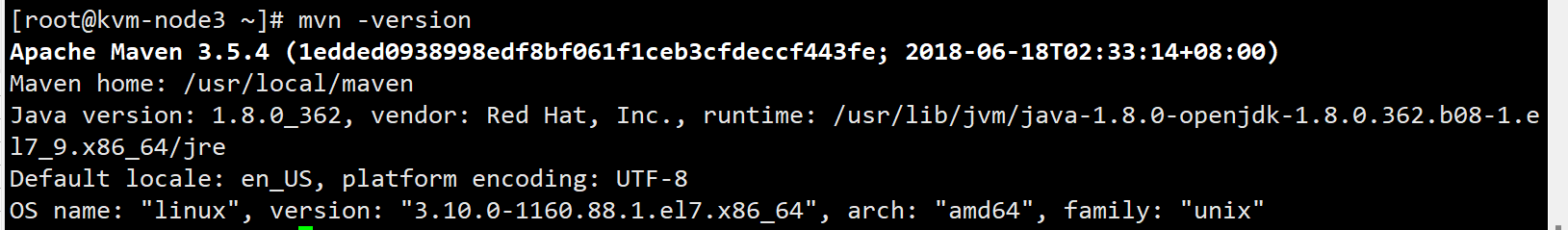
## 6.6 配置mvn命令

上传apache-maven-3.5.4-bin.tar.gz包至/root目录下。源码包可从以下地址获取：

|  |
| --- |
| http://192.168.190.3/download/apache-maven-3.5.4-bin.tar.gz |

解压安装包至/usr/local目录下，并配置/etc/profile

|  |
| --- |
| $ tar xf /root/apache-maven-3.5.4-bin.tar.gz -C /usr/local/  $ cd /usr/local  $ mv apache-maven-3.5.4 maven  $ cat >> /etc/profile << EOF  export MAVEN\_HOME=/usr/local/maven  export PATH=$PATH:$MAVEN\_HOME/bin  EOF  $ source /etc/profile  $ mvn -version |



## 6.7 项目编译

从gitee上拉去项目源码包，并编译（本次项目源码包拉取到了机器的/opt目录下）

|  |
| --- |
| $ cd /opt  $ git clone <https://gitee.com/cj520120/kvm-cloud.git>  $ cd kvm-cloud  $ mvn clean package |

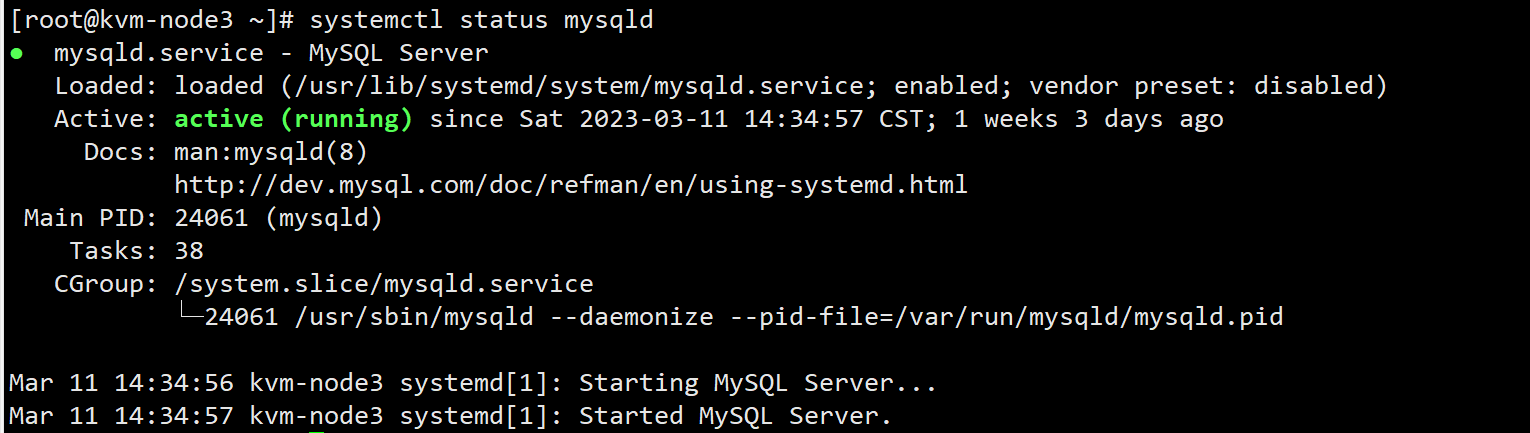
## 编译时间较长，请耐心等待，若编译错误，请重新执行mvn clean package

命令进行编译

## 6.8 安装并导入mysql数据

本次安装mysql5.7版本，并设置root远程登录

|  |
| --- |
| $ yum -y install mysql-community-server.x86\_64  $ cd /etc  $ mkdir /var/log/mysql/  $ cat >> /etc/my.cnf <<EOF  slow\_query\_log = 1  slow-query-log-file = /var/log/mysql/mysql-slow.log  long\_query\_time = 5  lower\_case\_table\_names=1  character\_set\_server = utf8  max\_connections = 3000  max\_allowed\_packet = 100M  innodb\_buffer\_pool\_size = 8G  EOF  $ systemctl start mysqld  $ systemctl enable mysqld |



查看mysql的初始密码，并使用文本文档保存，修改密码时会用到

|  |
| --- |
| $ grep password /var/log/mysqld.log |

修改mysql的root用户密码

|  |
| --- |
| $ mysqladmin -u root -p'augdgC!Xo6j.' password '\*\*\*\*\*\*' |

导入项目的mysql表及相关数据，**脚本位于/opt/kvm-cloud/scripts下**

|  |
| --- |
| $ cd /opt/kvm-cloud/scripts  $ mysql -uroot -p"\*\*\*\*\*\*\*"  Mysql > source mysql.sql  查看执行无报错即可  Mysql > grant all on \*.\* to root@'%' identified by " \*\*\*\*\*\*";  Mysql > flush privileges; |

## 6.9 安装redis

项目会使用redis作为缓存节点，存储缓存数据。

下载redis安装包，并上传至/root目录下，下载地址如下：

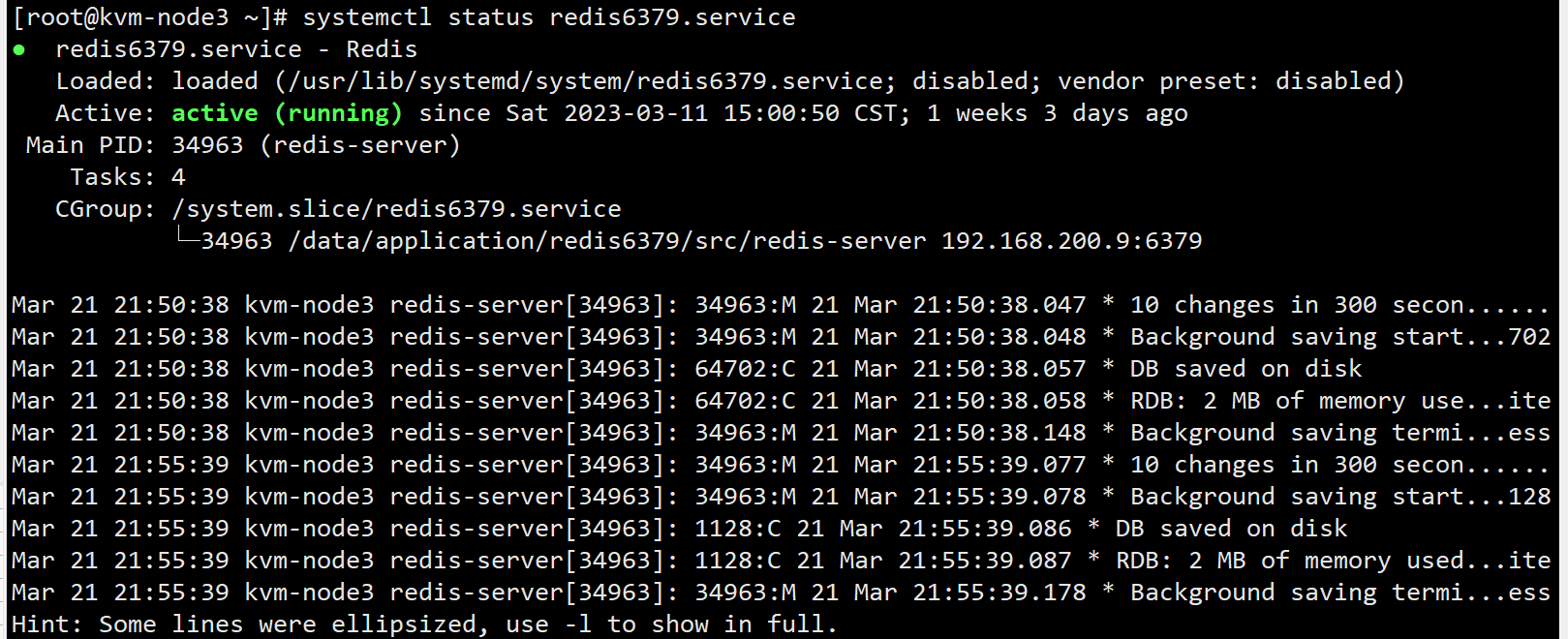
|  |
| --- |
| $ wget http://192.168.190.3/download/redis-4.0.9.tar.gz |

安装redis

|  |
| --- |
| $ mkdir -p /data/application  $ tar xzf redis-4.0.9.tar.gz -C /data/application/ ---解压  $ cd /data/application/  $ mv redis-4.0.9/ redis  $ cd redis/  $ yum install -y gcc make #安装编译工具  $ make  注：如果报错请将刚才解压的安装包删除掉，再次重新解压并进行make安装即可。  $ cp redis.conf redis.conf.bak  $ vim redis.conf ---修改如下  bind 192.168.190.9　　#只监听内网IP  daemonize yes　　　　　#开启后台模式将on改为yes  port 6379 #端口号  dir /data/application/redis/data　　#本地数据库存放持久化数据的目录该目录-----需要存在,创建存放数据的目录  $ mkdir /data/application/redis/data  $ cd /lib/systemd/system  $ touch redis.service  $ cat >> redis.service << EOF  [Unit]  Description=Redis  After=network.target  [Service]  ExecStart=/data/application/redis/src/redis-server /data/application/redis/redis.conf --daemonize no  ExecStop=/data/application/redis/src/redis-cli -h 127.0.0.1 -p 6379 shutdown  [Install]  WantedBy=multi-user.target  EOF  参数详解:  • [Unit] 表示这是基础信息  • Description 是描述  • After 是在那个服务后面启动，一般是网络服务启动后启动  • [Service] 表示这里是服务信息  • ExecStart 是启动服务的命令  • ExecStop 是停止服务的指令  • [Install] 表示这是是安装相关信息  • WantedBy 是以哪种方式启动：multi-user.target表明当系统以多用户方式（默认的运行级别）启动时，这个服务需要被自动运行。  $ systemctl daemon-reload  $ systemctl start redis.service |

测试redis是否联通正常

|  |
| --- |
| $ cd /data/application/redis/src/  $ ./redis-cli -h 192.168.190.9 -p 6379  192.168.190.9:6379> ping ---测试redis是否可以用  PONG |



## 6.10修改项目的配置文件

项目的配置文件在kvm-cloud/cloud-management/src/main/resources目录下的application.yaml文件（也就是管理端的配置文件，只需要修改管理端配置文件即可）

|  |
| --- |
| $ cd /opt/kvm-cloud/cloud-management/src/main/resources  $ cp application.yaml application.yaml.bak  修改的内容如下，用红色字体标出  $ vim application.yaml  app:  #系统组件网卡默认驱动，可选值为virtio e1000 rtl8139,变更该值需要删除原系统组件重建，切为最新版本的系统组件模版文件  systemComponentNetworkDriver: virtio  systemComponentCpu: 1  systemComponentMemory: 524288  systemComponentCpuSpeed: 500  #通过模版创建是否启用backing file,为false时通过clone磁盘实现,如果磁盘空间充足，可修改为false，该值为false时通过模版创建系统耗时会变长  enableVolumeBack: false  manager-uri: http://192.168.190.9:8080/  over-cpu: 4.0  over-memory: 1.5  work:  thread:  size: 16  boss:  thread:  size: 4  oauth2:  auth-uri: https://192.168.190.9:8083/oauth2/authorize  client-id: NLMAv72e  client-secret: fd74a2cb8b62d4011ff5945b0204d07f68366848  enable: false  id-path:  - userId  redirect-uri: http://192.168.190.9:8081/#/Login  title: Oauth2 Login  token-uri: https://192.168.190.9:8083/oauth2/token  user-uri: https://192.168.190.9:8083/oauth2/user/info  server:  port: 8080  servlet:  context-path: /  spring:  datasource:  driver-class-name: com.mysql.cj.jdbc.Driver  druid:  filters: stat,wall  initial-size: 5  max-active: 20  max-open-prepared-statements: 50  max-pool-prepared-statement-per-connection-size: 20  max-wait: 60000  min-evictable-idle-time-millis: 300000  min-idle: 5  pool-prepared-statements: true  test-on-borrow: false  test-on-return: false  test-while-idle: true  time-between-eviction-runs-millis: 60000  validation-query: SELECT 1  password: HotDB@2013  type: com.alibaba.druid.pool.DruidDataSource  url: jdbc:mysql://127.0.0.1:3306/cj\_kvm\_cloud?characterEncoding=utf8&characterSetResults=utf8&system\_time\_zone=Asia/Shanghai&autoReconnect=true&failOverReadOnly=false&useSSL=false&useUnicode=true  username: root  servlet:  multipart:  enabled: true  file-size-threshold: 0  max-file-size: 100GB  max-request-size: 100GB  redis:  redisson:  config: |  singleServerConfig:  idleConnectionTimeout: 10000  connectTimeout: 10000  timeout: 3000  retryAttempts: 3  retryInterval: 1500  address: "redis://192.168.190.9:6379"  password:  database: 0  subscriptionsPerConnection: 5  subscriptionConnectionMinimumIdleSize: 1  subscriptionConnectionPoolSize: 50  connectionMinimumIdleSize: 32  connectionPoolSize: 64  database: 2  dnsMonitoringInterval: 5000  threads: 16  nettyThreads: 32  codec: !<org.redisson.codec.JsonJacksonCodec> {}  transportMode: "NIO" |

## 6.11启动管理端和客户端

（1）手动启动

管理端：

|  |
| --- |
| $ cd /opt/kvm-cloud/cloud-management/target/  $ nohup java -jar cloud-management-1.0-SNAPSHOT.jar &  $ tailf nohup.out  查看输出日志，无报错则启动正常，可访问[http://192.168.190.9:8080](http://192.168.200.9:8080)  默认账号密码是：admin/123456 |

客户端：

|  |
| --- |
| $ cd /opt/kvm-cloud/cloud-agent/target  $ nohup java -jar cloud-agent-1.0-SNAPSHOT.jar &  $ tailf nohup.out  查看输出日志，无报错则启动正常 |

## 6.12配置nginx文件下载

项目在使用中，需要通过nginx下载本地的系统模板和iso镜像文件，因此需要进行下载

系统模板和iso镜像需要先下载下来，地址如下：

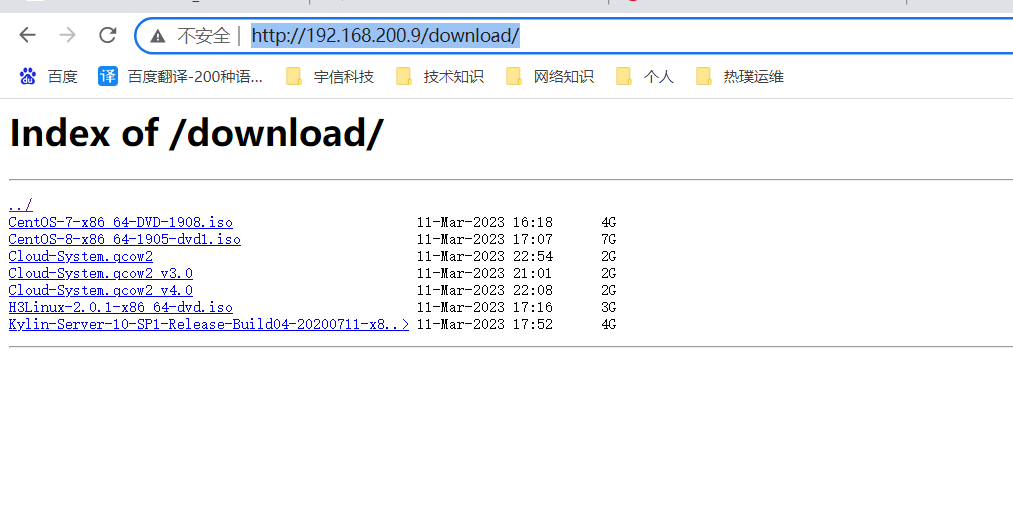
|  |
| --- |
| 系统模板：  [http://192.168.190.9/download/Cloud-System.qcow2](http://192.168.200.9/download/Cloud-System.qcow2)  iso镜像：  http://192.168.190.9/download/CentOS-7-x86\_64-DVD-1908.iso  iso镜像可以根据自己需求从官网下载，然后上传到主机 |

Nginx配置文件下载

|  |
| --- |
| $ yum –y install nginx  $ mkdir -p /data1/download  $ cd /etc/nginx/conf.d/  $ cp default.conf default.conf.bak  在server模块添加如下内容：  $ vim default.conf  location /download {  root /data1;  #add\_header Content-Disposition: "attachment";  #add\_header Content-Type application/octet-stream;  autoindex on;  autoindex\_exact\_size off;  autoindex\_localtime on;  }    $ nginx –t ##回显如下则正常  nginx: the configuration file /etc/nginx/nginx.conf syntax is ok  nginx: configuration file /etc/nginx/nginx.conf test is successful  $ systemctl enable nginx  $ systemctl start nginx |

Nginx启动完成之后，需要吧事先下载好的基础模板和iso镜像，上传至服务器的/data1/download目录下。。

浏览器访问[http://192.168.190.9/download/](http://192.168.200.9/download/)



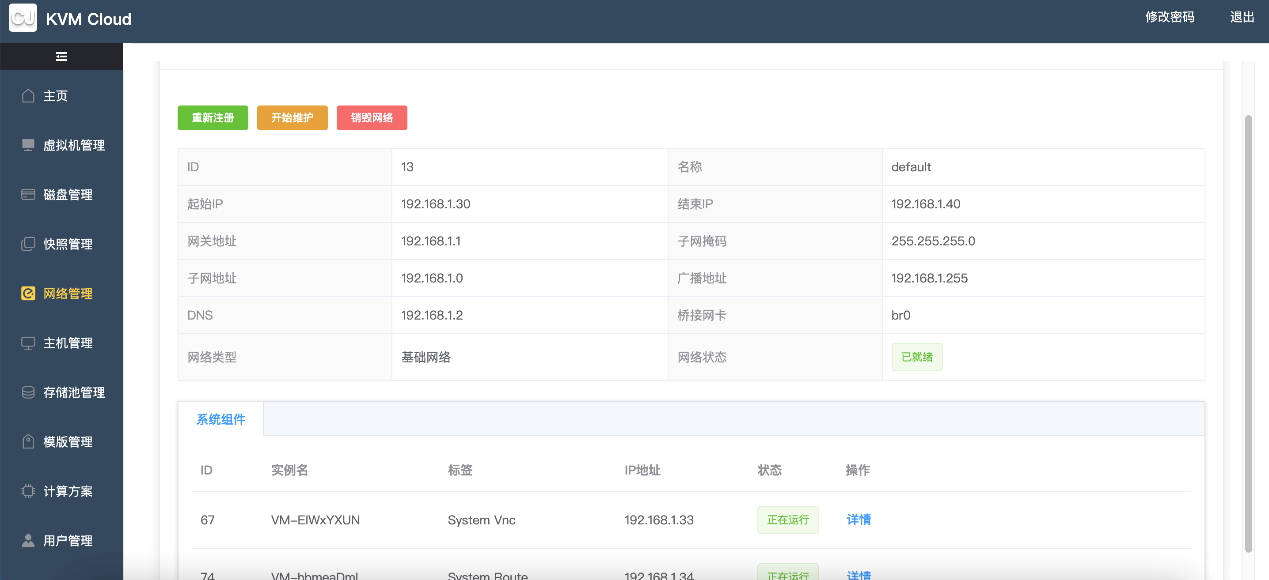
# 7.kvm-cloud平台基础操作

以上部署完成之后，通过浏览器访问<http://主机IP:8080>

默认账号密码为：admin/123456

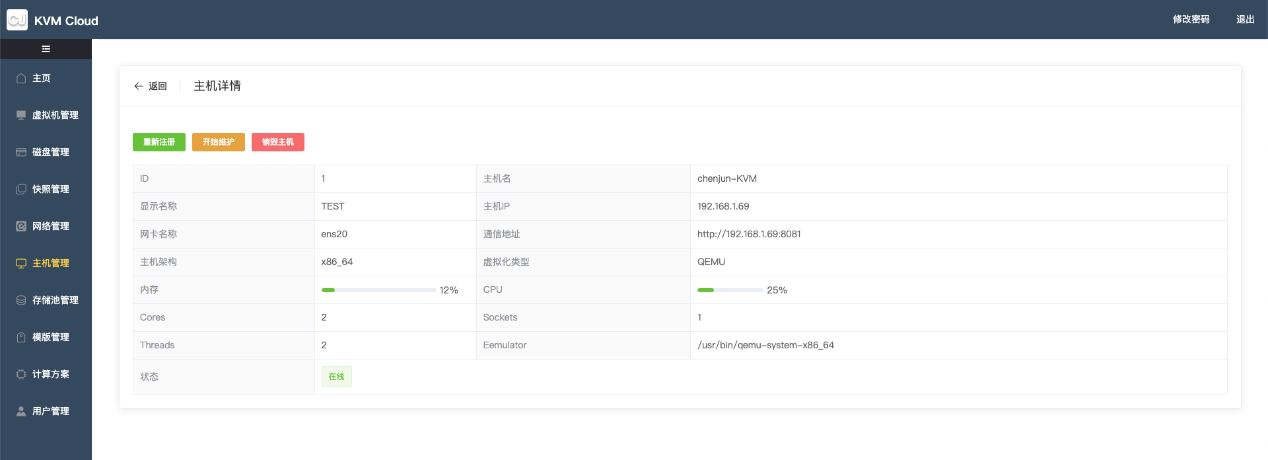
## 7.1 创建基础网络

采用桥接网络配置，IP地址段与主机主机段需保持一致，可通过起始IP与结束IP和主机网络进行分离，防止IP冲突.

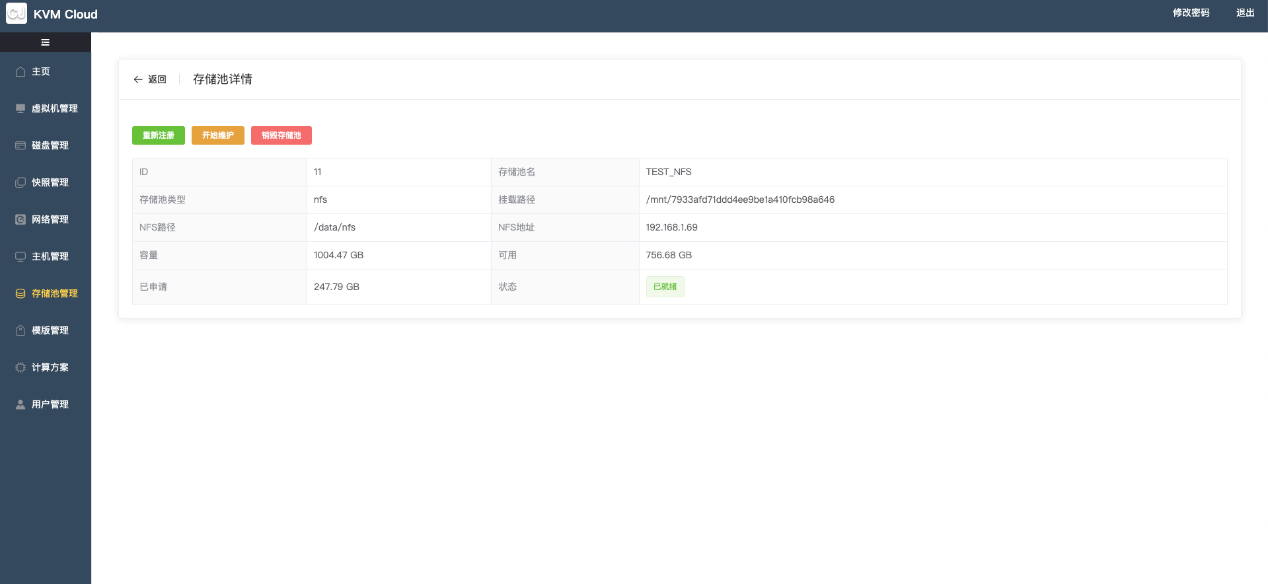


## 7.2 创建主机

注意创建主机时若主机错误，则需要查看在客户端agent启动是否异常。

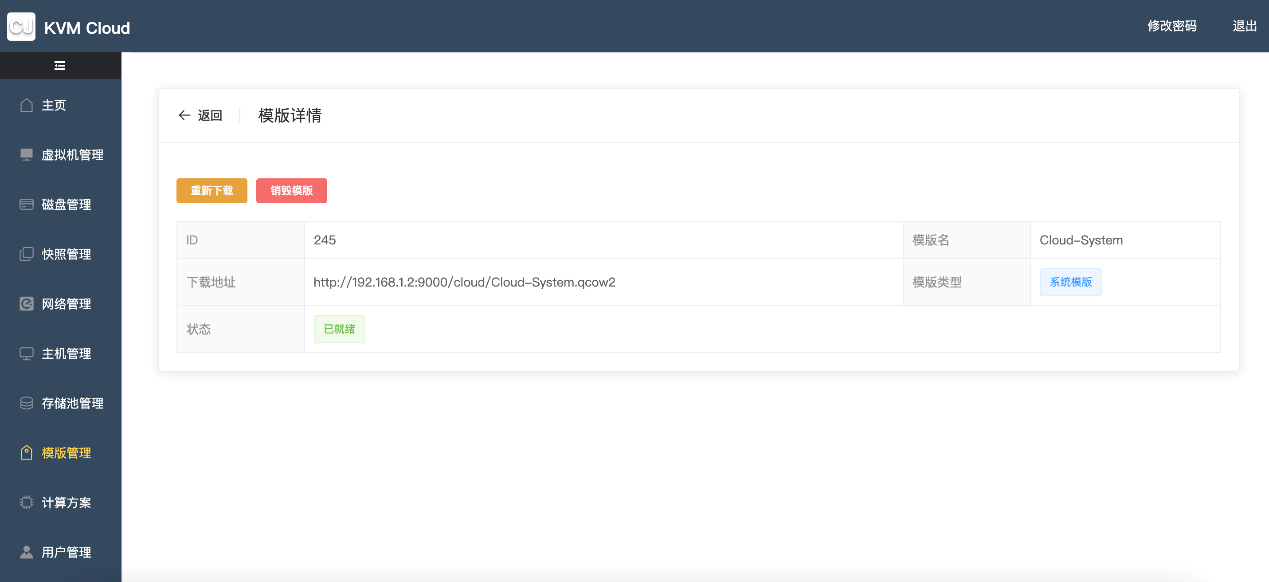


## 7.3 创建存储池(只支持nfs)



## 7.4 上传基础模板

注意创建时，模板类型要选择系统模板

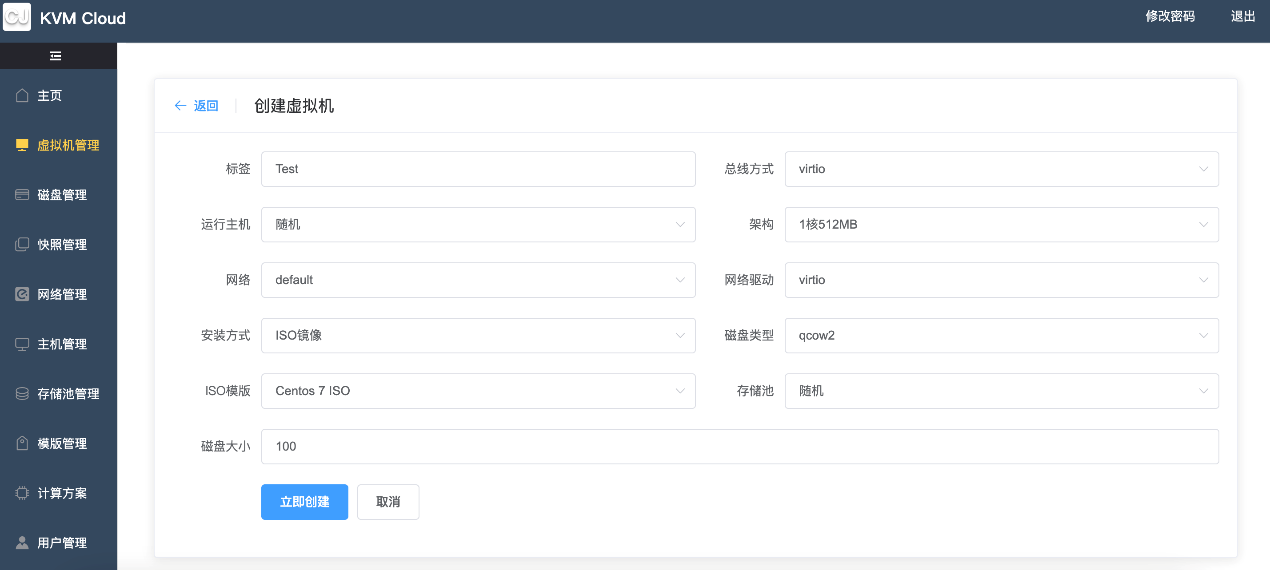


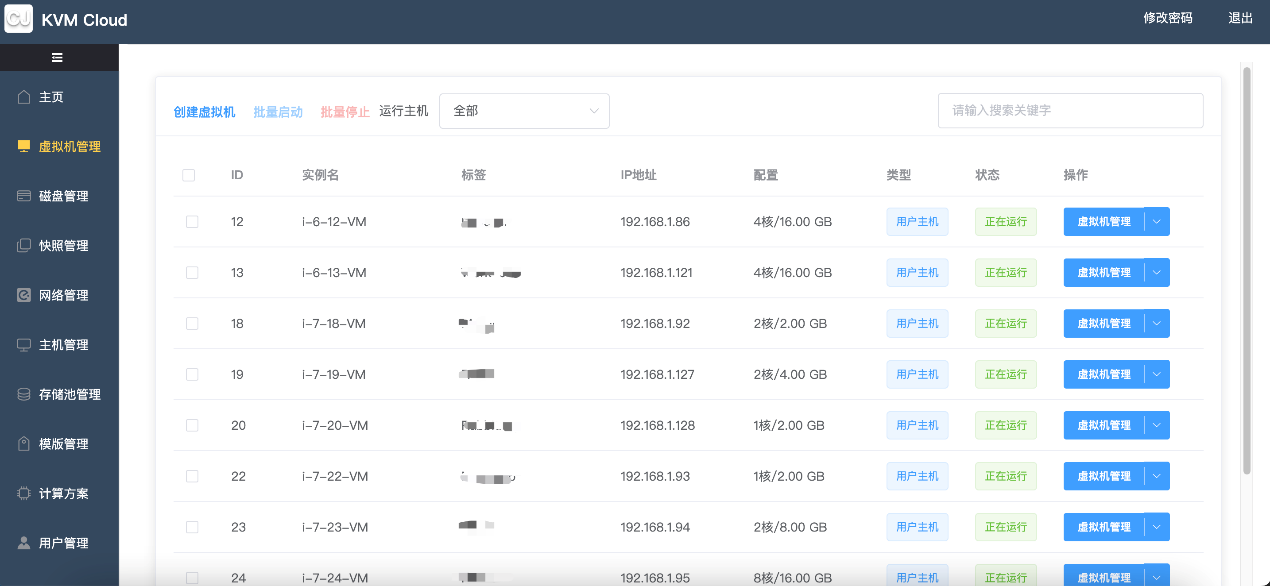
上传iso镜像使用相同的方法，只需要将模板类型修改为iso

系统模板上传完成后，会根据之前的基础网络创建系统虚拟机用于route转发和vnc链接（等待时间较长）。可点击【网络管理】-【网络详情】，在下方会有系统虚拟机的详情。

## 7.5 创建虚拟机

当两个系统虚拟机创建完成之后，就可以进行虚拟机的创建。





# 8.相关问题

## 8.1 找不到配置文件问题导致数据库连接问题

管理端的yaml文件 和 客户端的properties 文件内容分别为management和agent项目下的application.yaml和application.properties的文件，运行时自行修改名称及相关配置

## 8.2 关于备份与恢复

对数据库和存储池进行完整备份；

数据无价，建议对虚拟机中的数据进行备份

## 8.3 关于网络隔离

目前不支持自动创建隔离网络，可以通过创建两个桥接网络的方式进行隔离,大致步骤如下:

1)、通过eth0创建桥接网络br0

2)、通过br0创建vlan网络(例如br0.2)

3)、通过创建的vlan再次创建桥接网络(例如br2)

4)、在管理界面初始化网络br2

5)、等待该网络的console和route虚拟机初始化完成

6)、创建一台br0的网络的跳板虚拟机

7)、为该跳板机附加br2对应的网络

8)、外部访问vlan所对应的虚拟机，可通过该跳板机进行访问

## 8.4 windows系统附加磁盘不识别问题

windows没有virto的驱动，请安装virtio-win-0.1.185.iso驱动

## 8.5 服务器掉电重启后处理

1、服务器掉电重启后，请在页面手动关闭所有自己创建的虚拟机，然后重新启动，系统虚拟机有自动检测重启功能，无需处理

2、掉电可能引起虚拟磁盘损坏，如无法启动，可通过qemu-img check检查并进行相应修复

## 8.6 虚拟机虚拟化嵌套

|  |
| --- |
| 1、验证KVM 宿主机是否启用了嵌套虚拟化：  基于 Intel 的处理器运行以下命令：  $ cat /sys/module/kvm\_intel/parameters/nested  基于 AMD 的处理器运行以下命令：  $ cat /sys/module/kvm\_amd/parameters/nested  上述命令输出N /0表示嵌套虚拟化是禁用的。如果我们得到的输出是Y/1 则表示在您的宿主机已启用嵌套虚拟化  2、如果需要启用嵌套虚拟化，使用以下内容创建一个文件名为/etc/modprobe.d/kvm-nested.conf 的文件：  $ touch /etc/modprobe.d/kvm-nested.conf  $ cat >> /etc/modprobe.d/kvm-nested.conf << EOF  options kvm-intel nested=1  options kvm-intel enable\_shadow\_vmcs=1  options kvm-intel enable\_apicv=1  options kvm-intel ept=1  EOF  3、reboot 重启机器  $ reboot  4、现在验证嵌套虚拟化功能是否启用  $ cat /sys/module/kvm\_intel/parameters/nested |

# 9项目一键启停脚本

当前一键启停脚本在192.168.190.9机器的/opt/kvm-bin目录下，分别是start.sh和shutdown.sh

start.sh：

|  |
| --- |
| #!/bin/bash  #author:HotDB  #用途：用于启动kvm-cloud平台  source /etc/profile  local\_ip=`ifconfig |grep "192.168" |awk "NR==1" |awk '{print $2}'`  echo -e "------开始启动kvm-cloud管理端------ \n"  echo -e "------jar包在/opt/kvm-cloud/cloud-managent/target目录下------\n"  cd /opt/kvm-cloud/cloud-management/target && nohup java -jar cloud-management-1.0-SNAPSHOT.jar >management.log 2>&1 &  if [ $? == 0 ]  then  echo -e "------kvm-cloud管理端启动成功,可通过/opt/kvm-cloud/cloud-management/target/management.log查看日志------\n"  echo -e "请访问web页面：http://$local\_ip:8080------\n"  echo -e "------开始启动kvm-cloud-agent端------\n"  echo -e "------jar包在/opt/kvm-cloud/cloud-agent/target目录下------\n"  cd /opt/kvm-cloud/cloud-agent/target && nohup java -jar cloud-agent-1.0-SNAPSHOT.jar >agent.log 2>&1 &  if [ $? == 0 ]  then  echo -e "------kvm-cloud代理端启动成功,可通过/opt/kvm-cloud/cloud-agent/target/agent.log查看日志------\n"  else  echo -e "------kvm-cloud代理端启动失败,请查看/opt/kvm-cloud/cloud-agent/target/agent.log日志------\n"  fi  else  echo -e "------kvm-cloud管理端启动失败,请查看/opt/kvm-cloud/cloud-management/target/management.log日志------\n"  fi |

shutdown.sh:

|  |
| --- |
| #!/bin/bash  #author:HotDB  #用途：用于停止kvm-cloud平台  source /etc/profile  local\_ip=`ifconfig |grep "192.168" |awk "NR==1" |awk '{print $2}'`  management\_pid=`ps -ef |grep java |grep cloud-management |awk '{print $2}'`  agent\_pid=`ps -ef |grep java |grep cloud-agent |awk '{print $2}'`  echo -e "------开始停止kvm-cloud管理端------ \n"  #echo -e "------jar包在/opt/kvm-cloud/cloud-managent/target目录下------\n"  #cd /opt/kvm-cloud/cloud-management/target && nohup java -jar cloud-management-1.0-SNAPSHOT.jar >management.log 2>&1 &  echo -e "------当前management的PID为：$management\_pid"  kill -9 $management\_pid  if [ $? == 0 ]  then  echo -e "------kvm-cloud管理端停止成功------\n"  # echo -e "请访问web页面：http://$local\_ip:8080------\n"  echo -e "------开始停止kvm-cloud-agent端------\n"  # echo -e "------jar包在/opt/kvm-cloud/cloud-agent/target目录下------\n"  # cd /opt/kvm-cloud/cloud-agent/target && nohup java -jar cloud-agent-1.0-SNAPSHOT.jar >agent.log 2>&1 &  echo -e "------当前agent的PID为：$agent\_pid"  kill -9 $agent\_pid  if [ $? == 0 ]  then  echo -e "------kvm-cloud代理端停止成功,可通过/opt/kvm-cloud/cloud-agent/target/agent.log查看日志------\n"  else  echo -e "------kvm-cloud代理端停止失败,请查看/opt/kvm-cloud/cloud-agent/target/agent.log日志------\n"  fi  else  echo -e "------kvm-cloud管理端停止失败,请查看/opt/kvm-cloud/cloud-management/target/management.log日志------\n"  fi |