目錄

[**一、KVM+WwbVirtMgr部署 1**](#_Toc2091979)

[**二、ESXi到KVM之v2v 38**](#_Toc2091987)

[**三、Other 46**](#_Toc2091988)

**一、KVM+****WebVirtMgr部署**

参考 URL1 : [https://www.cnblogs.com/nulige/p/9236191.html](%20https:/www.cnblogs.com/nulige/p/9236191.html)

URL2: <http://www.cnblogs.com/kevingrace/p/5737724.html>

一、基础配置

[root@kvm ~]# systemctl stop firewalld

[root@kvm ~]# systemctl disable firewalld

[root@kvm ~]# vim /etc/selinux/config

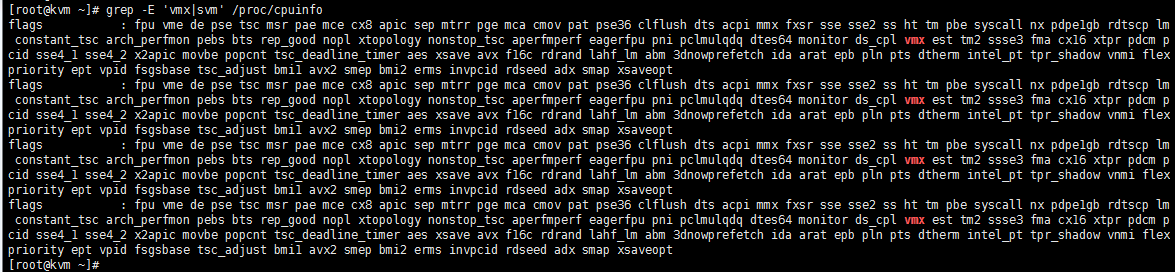
SELINUX=disabled

[root@kvm ~]# systemctl stop NetworkManager

[root@kvm ~]# systemctl disable NetworkManage

[root@kvm ~]# grep -E 'vmx|svm' /proc/cpuinfo

如果有过滤出vmx或svm关键字就代表支持虚拟化，vmx是Intel的CPU，svm是AMD的CPU



安装epel源

[root@kvm ~]# yum -y install wget

[root@kvm ~]# cd /etc/yum.repos.d/

[root@kvm yum.repos.d]# mv CentOS-Base.repo CentOS-Base.repo.bak

[root@kvm yum.repos.d]# wget -O /etc/yum.repos.d/CentOS-Base.repo http://mirrors.aliyun.com/repo/Centos-7.repo

[root@kvm yum.repos.d]# yum -y install epel-release

[root@kvm yum.repos.d]# yum clean all

[root@kvm yum.repos.d]# yum makecache

[root@kvm yum.repos.d]# yum install -y net-tools lrzsz

二、安装KVM

[root@kvm ~]# yum install -y qemu-kvm libvirt libvirt-python libguestfs-tools virt-install virt-manager python-virtinst libvirt-client virt-viewer

[root@kvm ~]# cd /etc/sysconfig/network-scripts

配置桥接网络

[root@kvm network-scripts]# vim ifcfg-enp1s0 #编辑真实网卡

TYPE=Ethernet

BOOTPROTO=static

DEFROUTE=yes

PEERDNS=yes

PEERROUTES=yes

NM\_CONTROLLED=no # 让enp1s0不受networkmanager的控制

IPV4\_FAILURE\_FATAL=no

NAME=enp1s0

DEVICE=enp1s0

ONBOOT=yes

BRIDGE=br0

IPADDR=10.148.54.30

NETMASK=255.255.252.0

GATEWAY=10.148.52.1

[root@kvm network-scripts]# cp ifcfg-enp1s0 ifcfg-br0

TYPE=Bridge

BOOTPROTO=static

DEFROUTE=yes

PEERDNS=yes

PEERROUTES=yes

NM\_CONTROLLED=no

IPV4\_FAILURE\_FATAL=no

NAME=enp1s0

DEVICE=br0

ONBOOT=yes

IPADDR=10.148.54.30

NETMASK=255.255.252.0

GATEWAY=10.148.52.1

[root@kvm network-scripts]# systemctl restart network

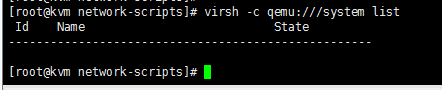
[root@kvm network-scripts]# systemctl start libvirtd #启动libvirt

[root@kvm network-scripts]# systemctl enable libvirtd

[root@kvm network-scripts]# systemctl status libvirtd

测试

[root@kvm network-scripts]# virsh -c qemu:/system list

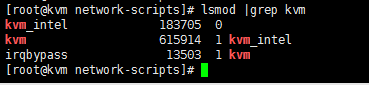


[root@kvm network-scripts]# virsh --version

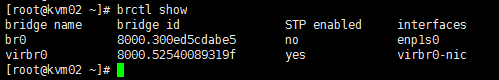
[root@kvm network-scripts]# virt-install --version

[root@kvm network-scripts]# ln -s /usr/libexec/qemu-kvm /usr/bin/

[root@kvm network-scripts]# lsmod |grep kvm #查看加载到内核



[root@kvm ~]# brctl show #查看网桥

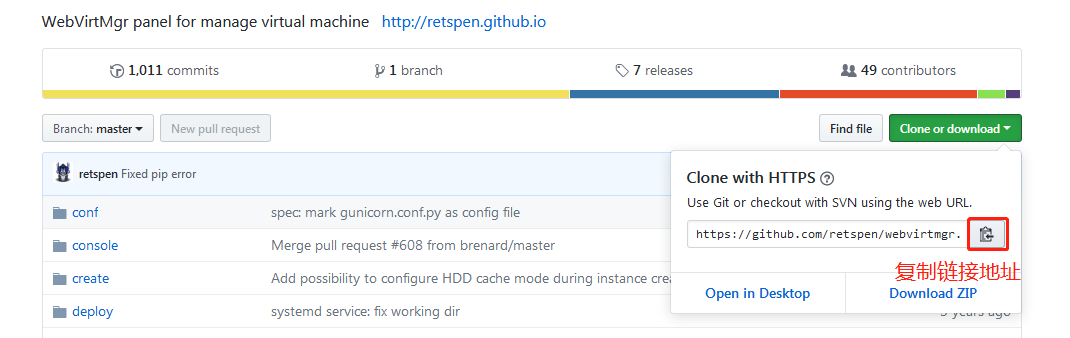


三、部署webvirtmgr

[root@kvm network-scripts]# yum -y install git python-pip libvirt-python libxml2-python python-websockify supervisor nginx python-devel

从git-hub中下载相关的webvirtmgr代码

URL : <https://github.com/retspen/webvirtmgr>



[root@kvm ~]# cd /usr/local/src

[root@kvm src]# git clone https://github.com/retspen/webvirtmgr.git

[root@kvm src]# cd webvirtmgr/

[root@kvm webvirtmgr]# pip install -r requirements.txt

pip install --trusted-host pypi.org --trusted-host files.pythonhosted.org  --upgrade  pip

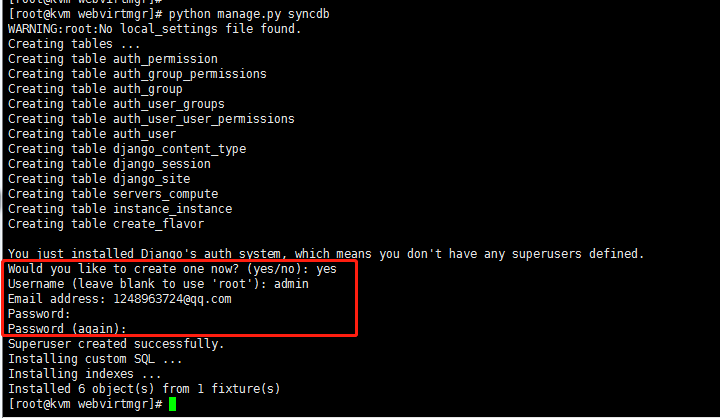
pip install --trusted-host pypi.org --trusted-host files.pythonhosted.org –r requirements.txt

安装数据库（备注：自带不需要安装，导入模块检查一下。）

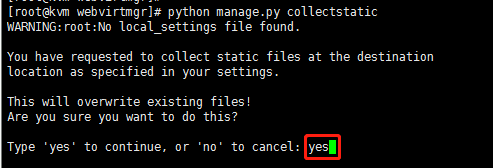
[root@kvm webvirtmgr]# python

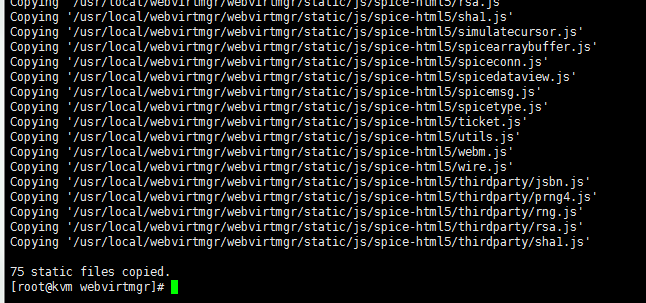
>>> import sqlite3

[root@kvm webvirtmgr]# python manage.py syncdb #创建webvirtmgr登录用户



[root@kvm webvirtmgr]# python manage.py collectstatic #生成配置文件





拷贝web到 相关目录

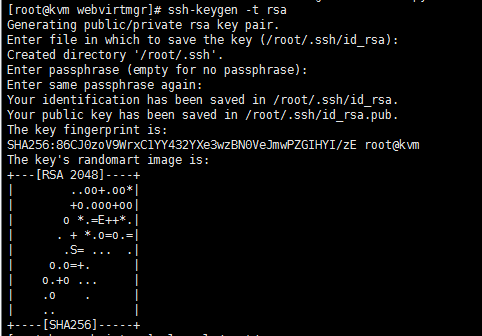
[root@kvm webvirtmgr]# mkdir -p /var/www

[root@kvm webvirtmgr]# cp -R /usr/local/src/webvirtmgr /var/www/

设置ssh

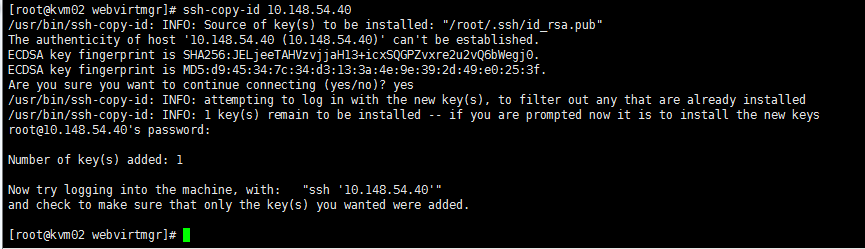
[root@kvm webvirtmgr]# ssh-keygen -t rsa #使用rsa算法进行加密

按照提示完成三次回车



[root@kvm webvirtmgr]# ssh-copy-id 10.148.54.30

# 由于这里webvirtmgr和kvm服务部署在同一台机器，所以这里本地信任。如果kvm部署在其他机器，那么这个是它的ip



[root@kvm webvirtmgr]# ssh 10.148.54.30 -L localhost:8000:localhost:8000 -L localhost:6080:localhost:60

编辑nginx配置文件

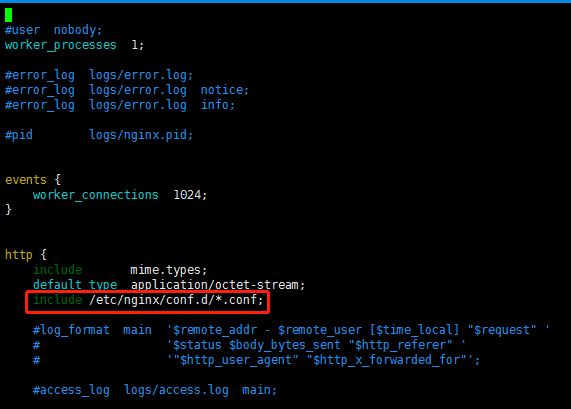
[root@kvm ~]# cd /etc/nginx/

[root@kvm nginx]# mv nginx.conf nginx.conf.bak

[root@kvm nginx]# cp nginx.conf.default nginx.conf

[root@kvm nginx]# vim nginx.conf

添加这行代码： include /etc/nginx/conf.d/\*.conf;



添加 /etc/nginx/conf.d/webvirtmgr.conf  配置文件

[root@kvm nginx]# vim /etc/nginx/conf.d/webvirtmgr.conf

server {

listen 80 default\_server;

server\_name $hostname;

#access\_log /var/log/nginx/webvirtmgr\_access\_log;

location /static/ {

root /var/www/webvirtmgr/webvirtmgr; # or /srv instead of /var

expires max;

}

location / {

proxy\_pass http://127.0.0.1:8000;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-for $proxy\_add\_x\_forwarded\_for;

proxy\_set\_header Host $host:$server\_port;

proxy\_set\_header X-Forwarded-Proto $scheme;

proxy\_connect\_timeout 600;

proxy\_read\_timeout 600;

proxy\_send\_timeout 600;

client\_max\_body\_size 1024M; # Set higher depending on your needs

}

}

[root@kvm nginx]# systemctl start nginx

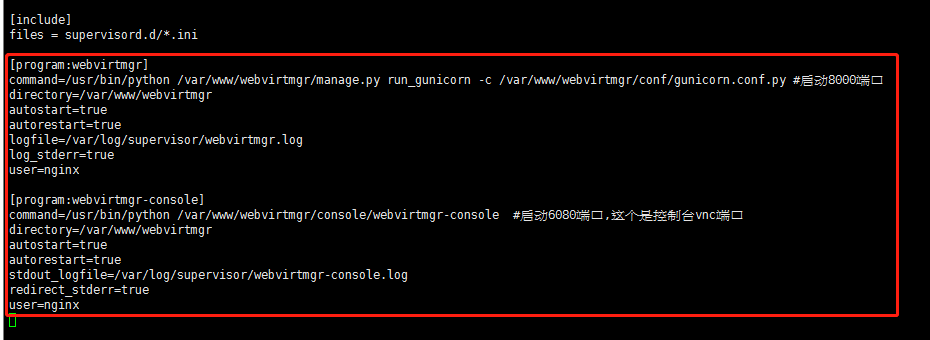
[root@kvm nginx]# systemctl enable nginx

[root@kvm nginx]# systemctl status nginx

[root@kvm ~]# chown -R nginx:nginx /var/www/webvirtmgr #授权

设置supervisor (如果iptables防火墙开启的话，就必须要开通80、8000、6080端口访问)

[root@kvm ~]# vim /etc/supervisord.conf #在文件末尾追加



[program:webvirtmgr]

command=/usr/bin/python /var/www/webvirtmgr/manage.py run\_gunicorn -c /var/www/webvirtmgr/conf/gunicorn.conf.py #启动8000端口

directory=/var/www/webvirtmgr

autostart=true

autorestart=true

logfile=/var/log/supervisor/webvirtmgr.log

log\_stderr=true

user=nginx

[program:webvirtmgr-console]

command=/usr/bin/python /var/www/webvirtmgr/console/webvirtmgr-console

directory=/var/www/webvirtmgr

autostart=true

autorestart=true

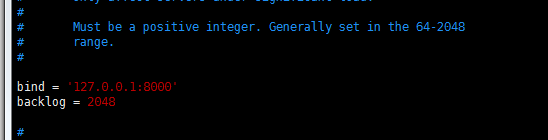
stdout\_logfile=/var/log/supervisor/webvirtmgr-console.log

redirect\_stderr=true

user=nginx

检查确保下面bind绑定的是本机的8000端口，这个在nginx配置中定义了，被代理的端口

[root@kvm ~]# vim /var/www/webvirtmgr/conf/gunicorn.conf.py

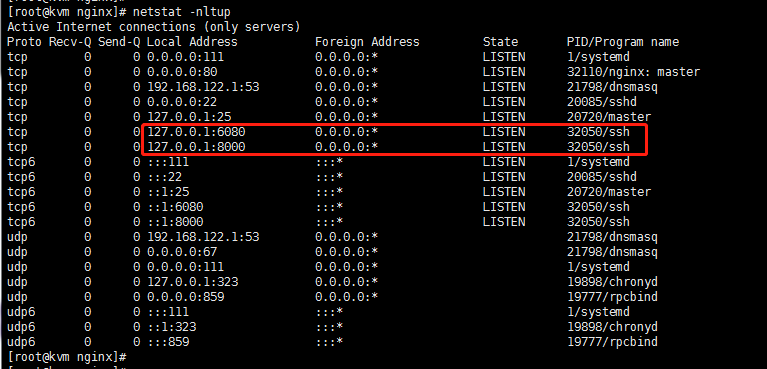


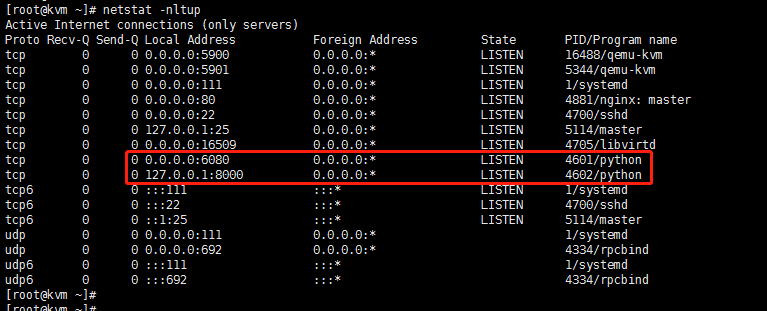
[root@kvm ~]# systemctl start supervisord

[root@kvm ~]# systemctl enable supervisord

[root@kvm ~]# systemctl status supervisord

[root@kvm ~]# netstat –nltup





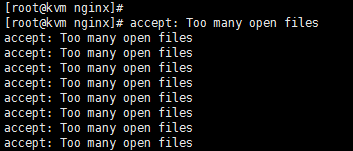
这两个端口进程服务刚开始是“ssh”，通过web访问后才变成为“python”

四、WEB设置

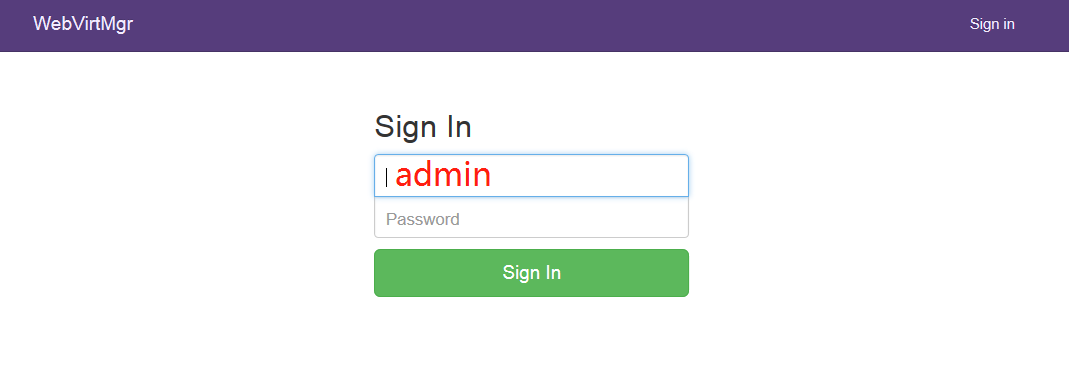
参考URL1 : <https://yq.aliyun.com/articles/46358>

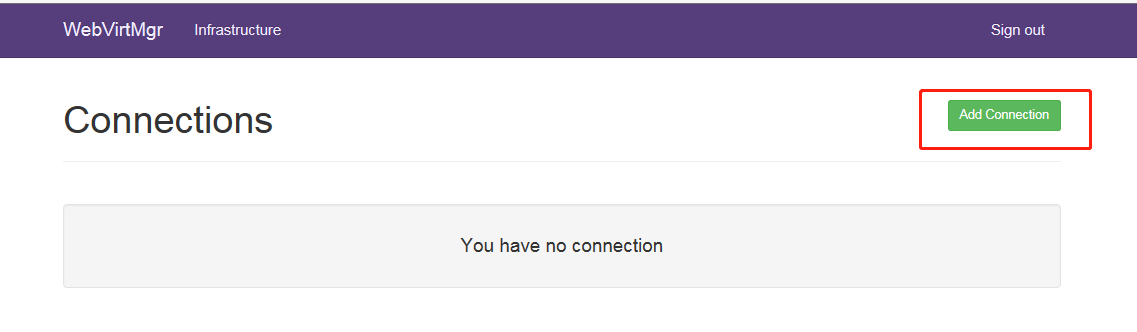
URL2 : <https://www.cnblogs.com/kevingrace/p/5739009.html>

Login: <http://10.148.54.30/login/>

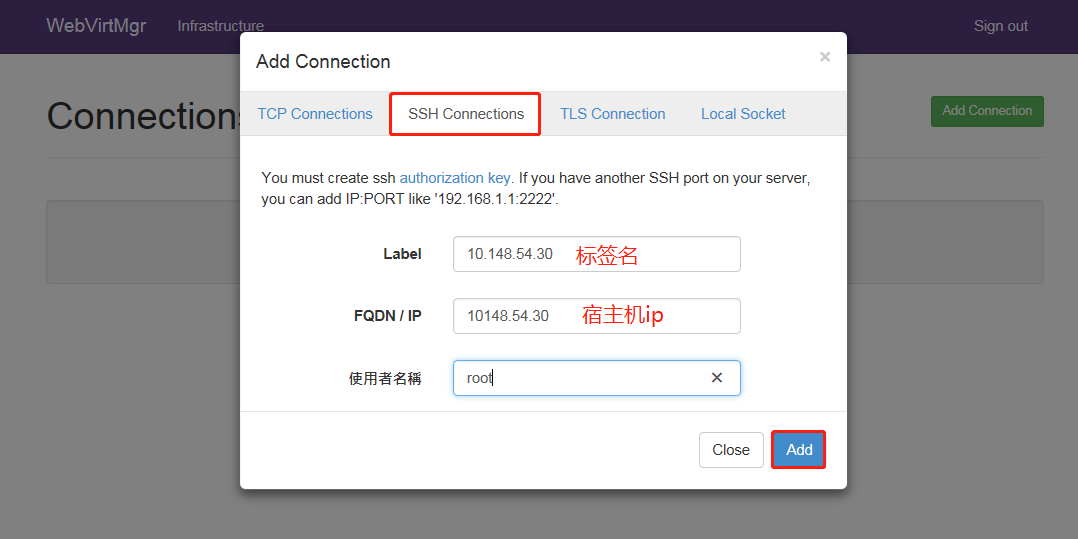


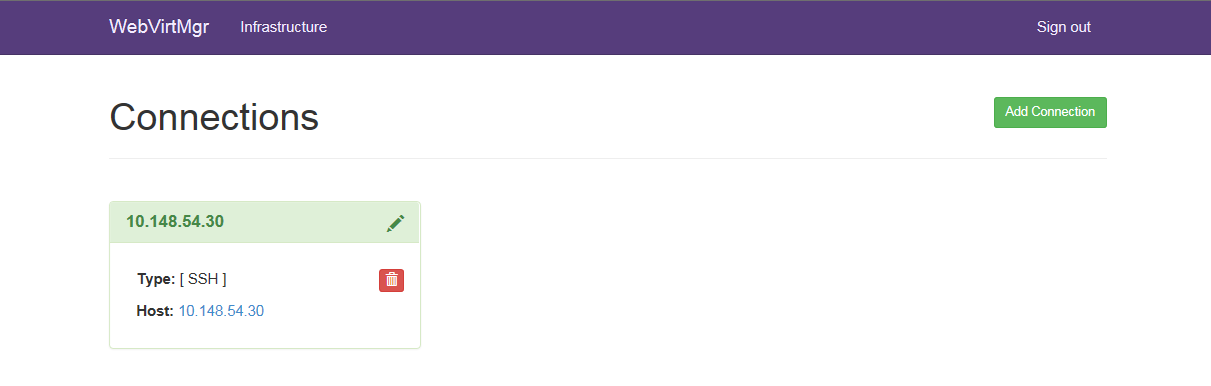
出现此问题关闭web ，Ctrl+D就可退出



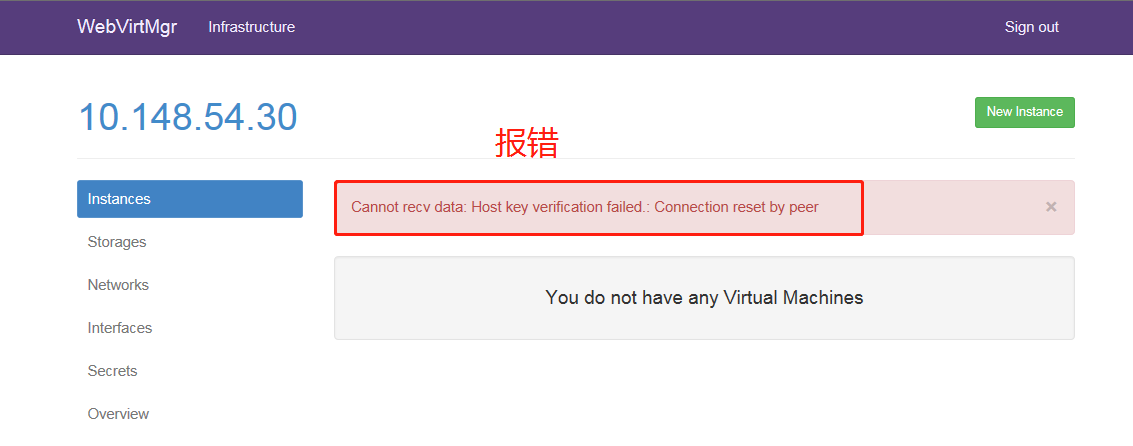


使用SSH连接：









解决措施：在webvirtmgr服务器（服务端）上（这里kvm和WebVirtMgr部署在同一台机器上）创建nginx用户家目录（默认nginx服务安装时是没有nginx家目录的），生成nginx的公私钥

[root@kvm ~]# cd /home/

[root@kvm home]# mkdir nginx

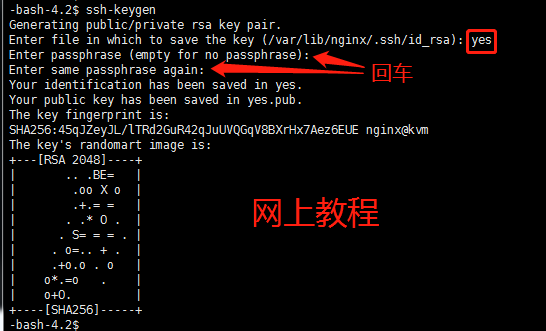
[root@kvm home]# chown nginx:nginx nginx/

[root@kvm home]# chmod 700 -R nginx/

[root@kvm home]# su - nginx -s /bin/bash

#网上教程是期间输入yes后直接回车，回车，结果最后使用ssh-copy-id时会报错“/var/lib/nginx/.pub': No such file or directory”

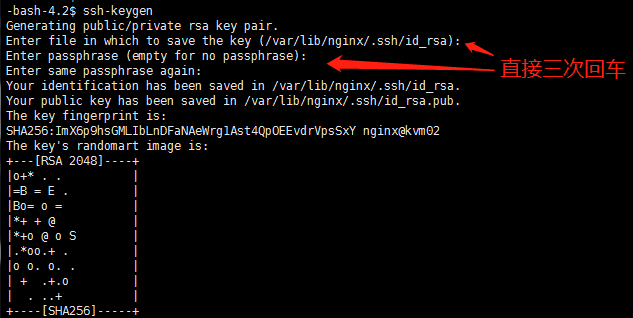
-bash-4.2$ ssh-keygen #生成证书



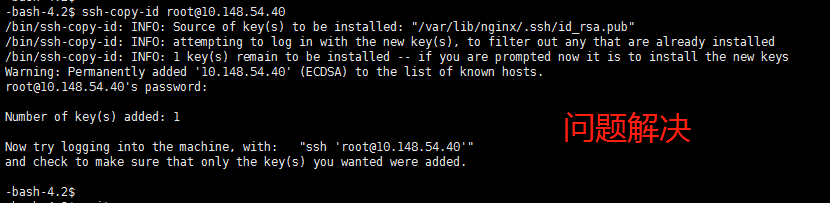




本人验证结果都不要输入“yes”，才能解决ssh-copy-id 时不报错“/var/lib/nginx/.pub': No such file or directory”







-bash-4.2$ touch ~/.ssh/config && echo -e "StrictHostKeyChecking=no\nUserKnownHostsFile=/dev/null" >> ~/.ssh/config

-bash-4.2$ chmod 0600 ~/.ssh/config

-bash-4.2$ ssh-copy-id root@10.148.54.30 #本机ip

#配置 libvirt ssh授权

[root@kvm ~]# vim /etc/polkit-1/localauthority/50-local.d/50-libvirt-remote-access.pkla

[Remote libvirt SSH access]

Identity=unix-user:root #注意这里采用的是root用户

Action=org.libvirt.unix.manage

ResultAny=yes

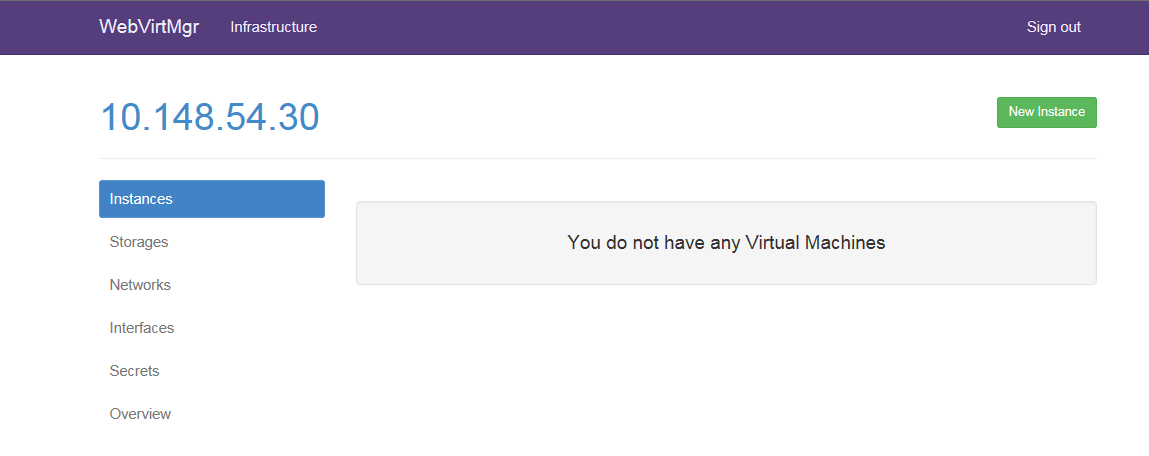
ResultInactive=yes

ResultActive=yes

[root@kvm ~]# chown -R root:root /etc/polkit-1/localauthority/50-local.d/50-libvirt-remote-access.pkla

[root@kvm ~]# systemctl restart nginx

[root@kvm ~]# systemctl restart libvirtd

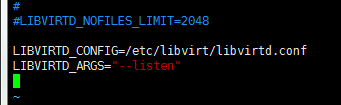


使用TCP连接（被连接端如同配置）：

[root@kvm ~]# vim /etc/sysconfig/libvirtd

LIBVIRTD\_CONFIG=/etc/libvirt/libvirtd.conf

LIBVIRTD\_ARGS="--listen"



[root@kvm ~]# vim /etc/libvirt/libvirtd.conf

listen\_tls = 0  
listen\_tcp = 1  
tcp\_port = "16509"  
listen\_addr = "0.0.0.0"  
auth\_tcp = "none"

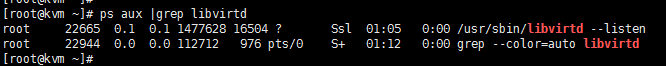
[root@kvm ~]# systemctl restart libvirtd #restart

如果重启libvirtd没生效那么使用命令行，如果生效了，执行下面命令就会报错

[root@kvm ~]# libvirtd --daemon --listen --config /etc/libvirt/libvirtd.conf



[root@kvm ~]# ps aux |grep libvirtd

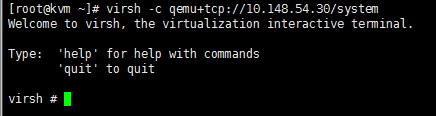


[root@kvm ~]# netstat -nltup |grep 16509 #查看端口

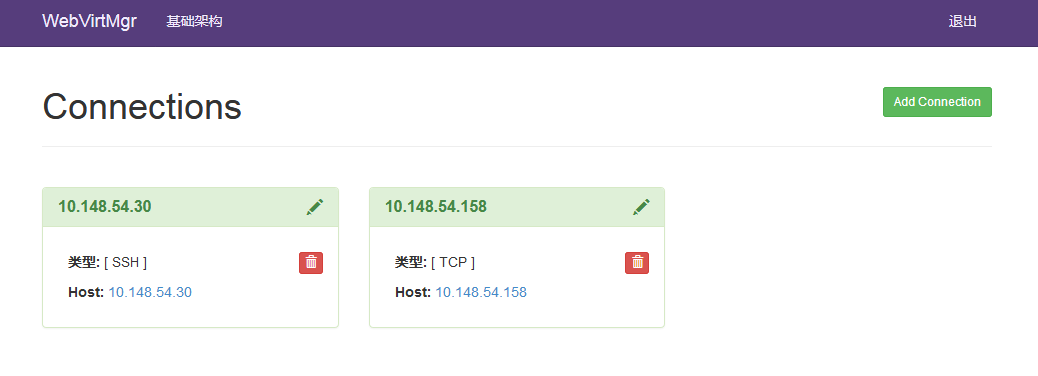


在source host连接target host远程libvirtd查看信息，或者连接本地主机

[root@kvm ~]# virsh -c qemu+tcp://10.148.54.30/system





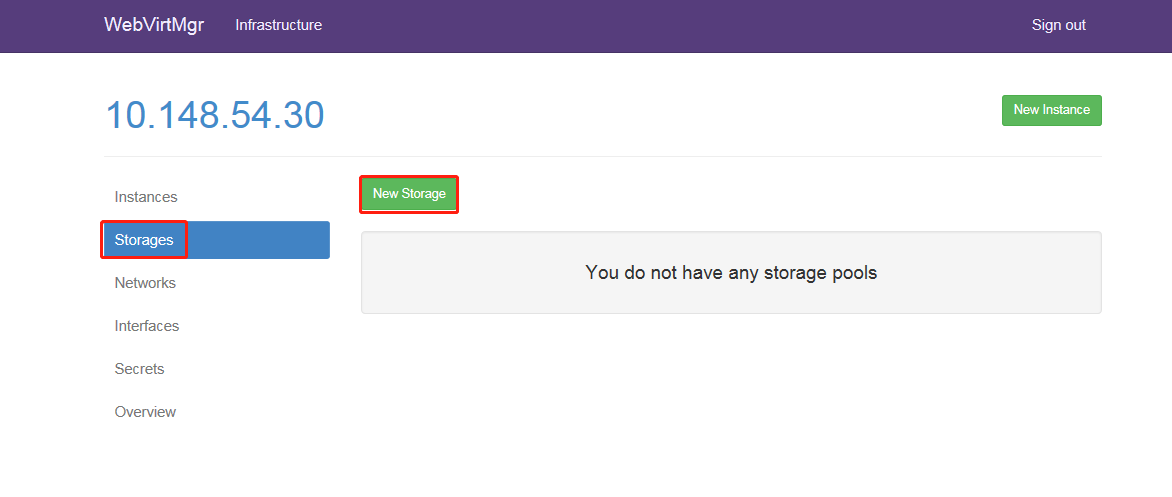


添加镜像

[root@kvm ~]# cd /var/lib/libvirt/images/ #上传ISO到此目录

点击“Storages”按钮，创建存储池（即创建磁盘镜像存放的位置）

1)创建镜像文件存放路径







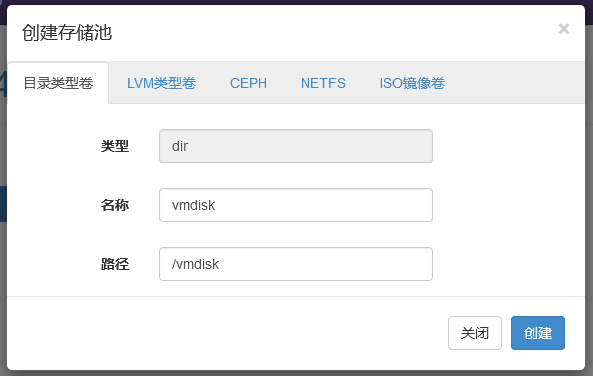


2)创建系统磁盘存放路径

注意：如果要创建新的存储池目录，首先要在宿主机上创建一个目录，然后在“path”设置中添加该目录

例：mkdir –p /vmdisk



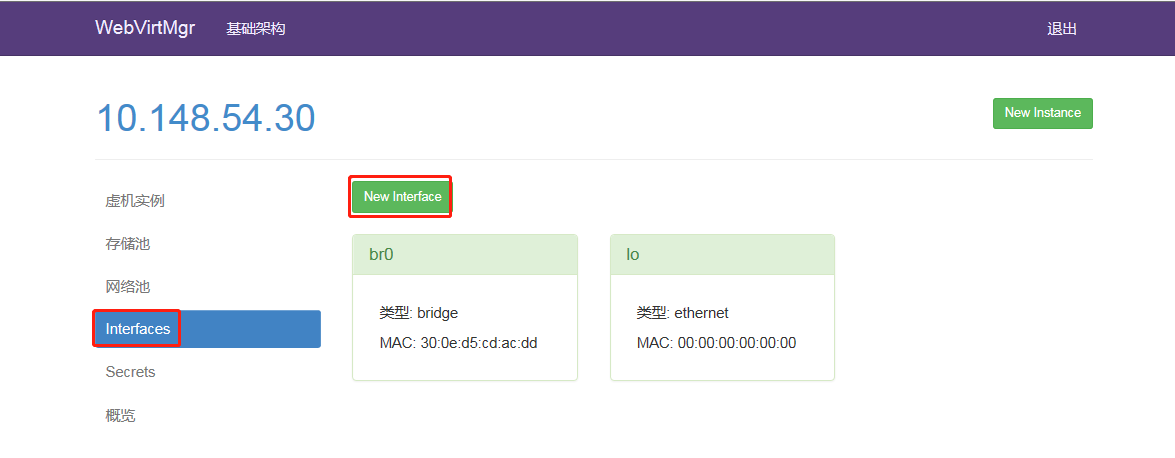






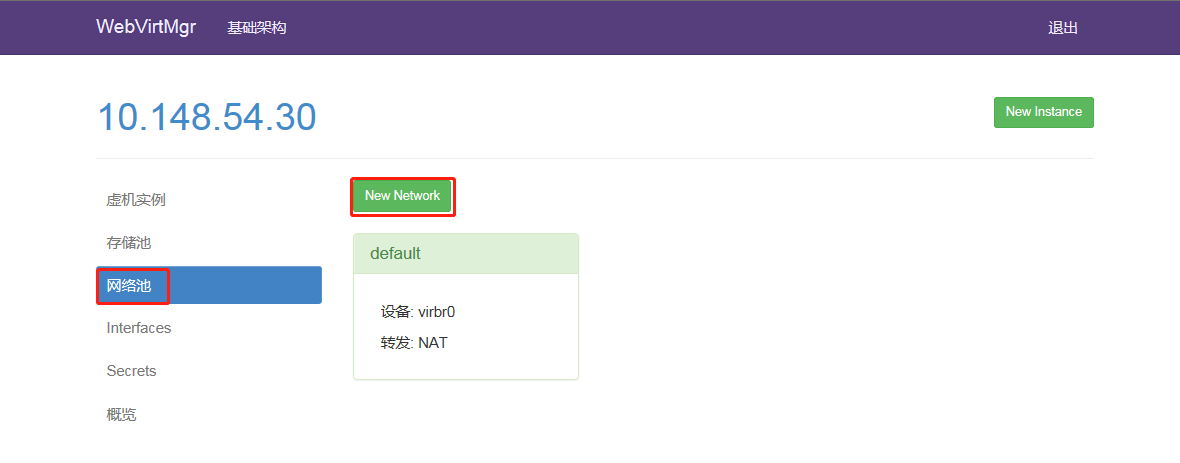


宿主机网卡的桥接模式设置

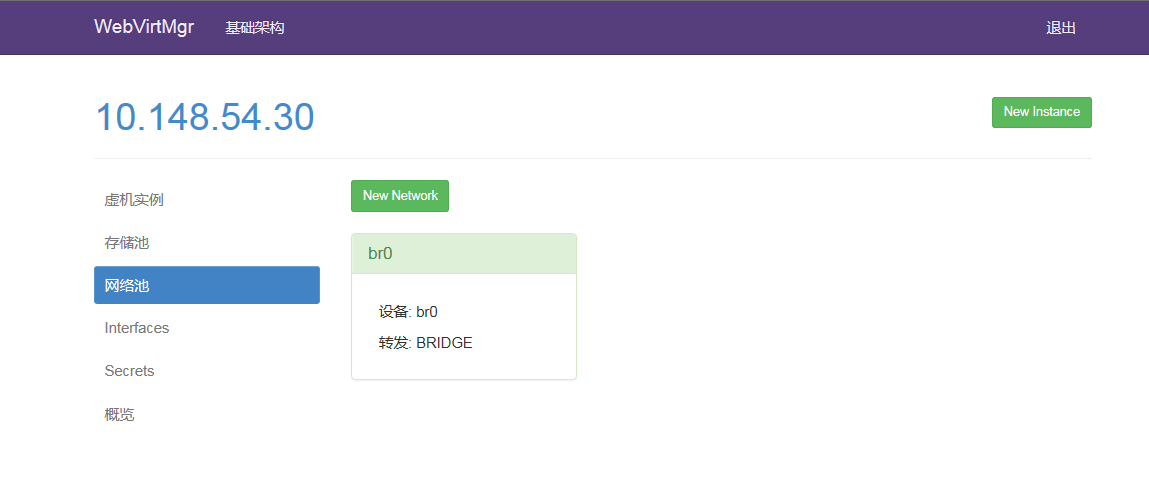


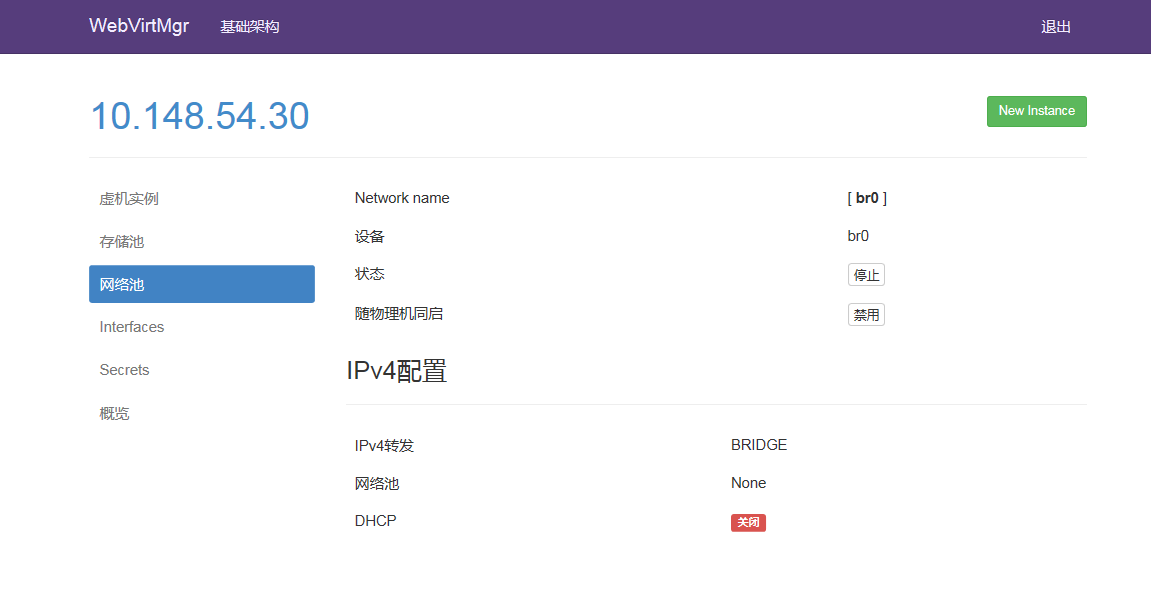


创建网络池

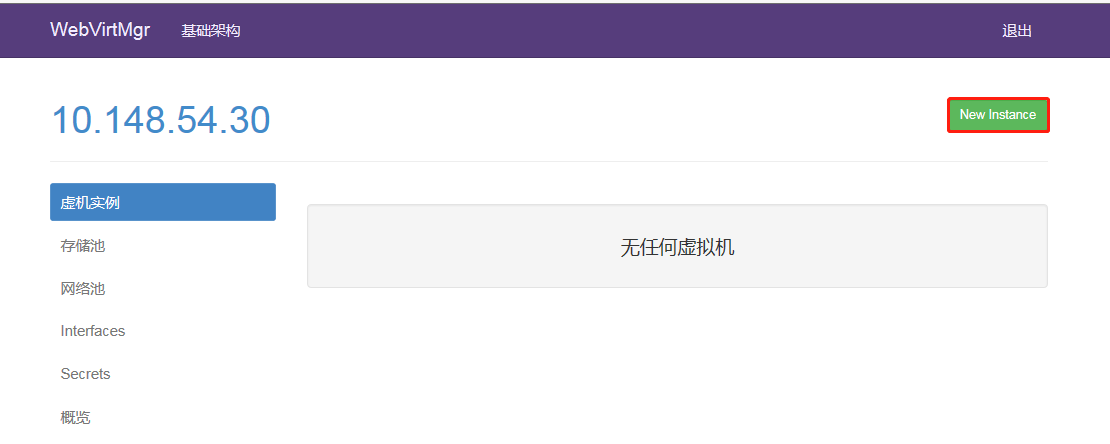








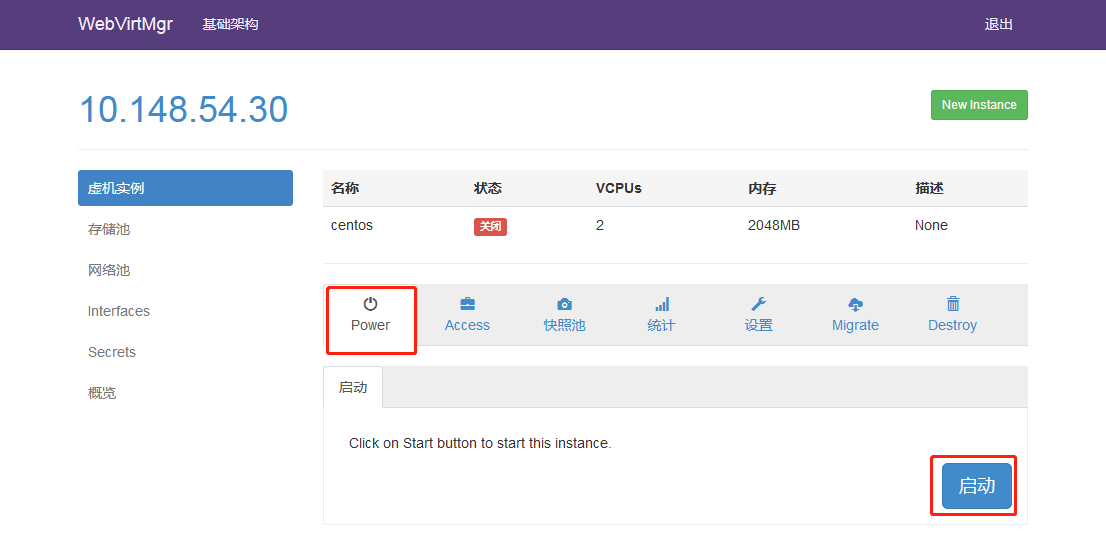
创建VM ( linux )

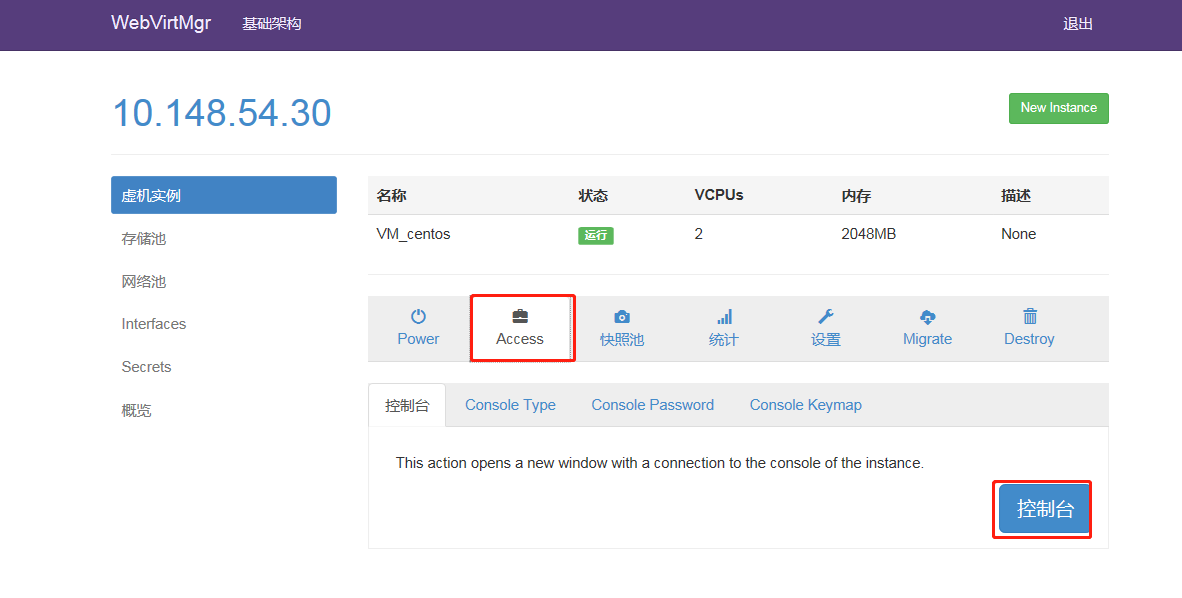






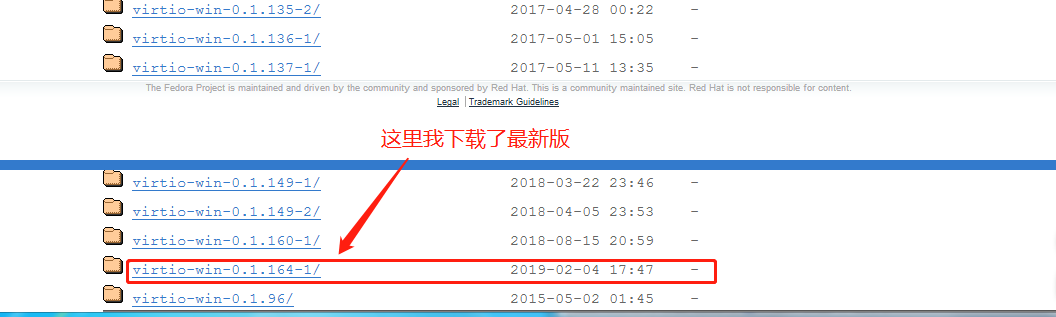


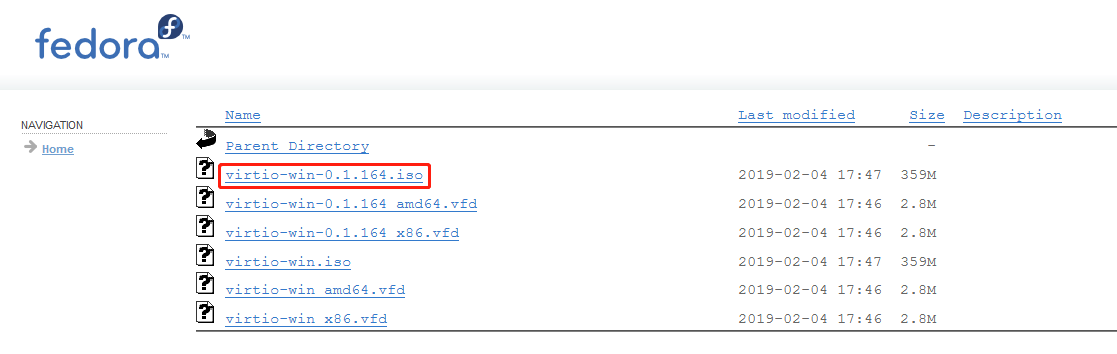


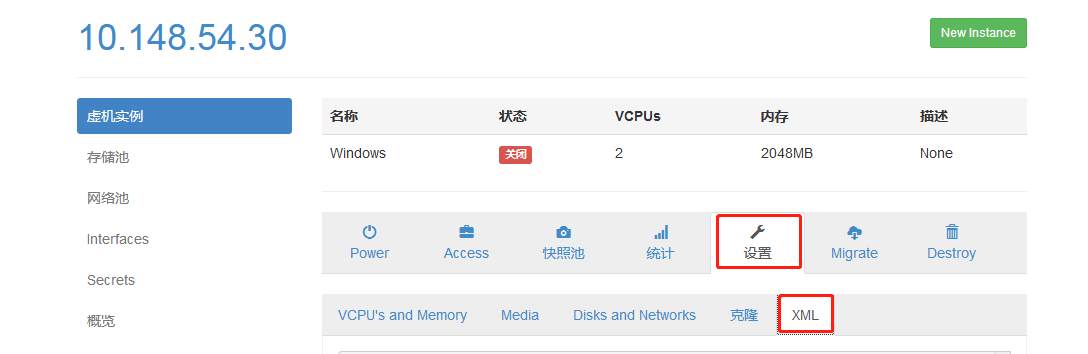


创建VM（windows）

驱动下载URL : <https://fedorapeople.org/groups/virt/virtio-win/direct-downloads/archive-virtio/>







编辑虚拟机配置文件，删掉原有的空的 cdrom，添加两个 cdrom 分别挂载 windows 系统 iso 镜像和 virtio-win 驱动 iso 镜像



<disk type='file' device='cdrom'>

<driver name='qemu' type='raw'/>

<source file='/ISO/WindServer 2012R2\_CN.iso'/>

<backingStore/>

<target dev='hda' bus='ide'/>

<readonly/>

<alias name='ide0-0-0'/>

<address type='drive' controller='0' bus='0' target='0' unit='0'/>

</disk>

<disk type='file' device='cdrom'>

<driver name='qemu' type='raw'/>

<source file='/ISO/virtio-win-0.1.164.iso'/>

<backingStore/>

<target dev='hdb' bus='ide'/>

<readonly/>

<alias name='ide0-0-1'/>

<address type='drive' controller='0' bus='0' target='0' unit='1'/>

</disk>

直接复制

<disk type='file' device='cdrom'>

<driver name='qemu' type='raw'/>

<source file='/var/lib/libvirt/images/virtio-win-0.1.164.iso'/>

<backingStore/>

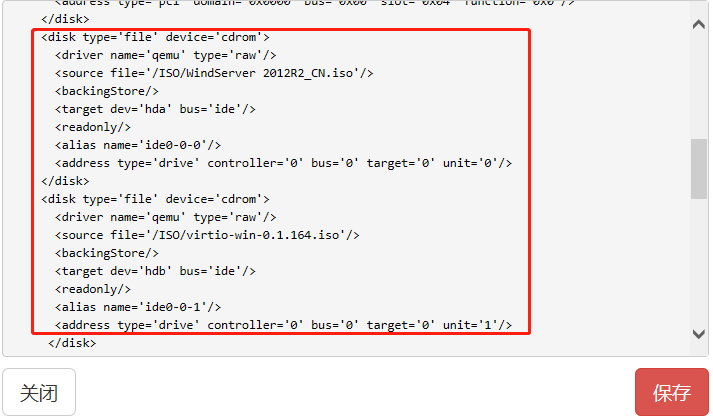
<target dev='hdb' bus='ide'/>

<readonly/>

<alias name='ide0-0-1'/>

<address type='drive' controller='0' bus='0' target='0' unit='1'/>

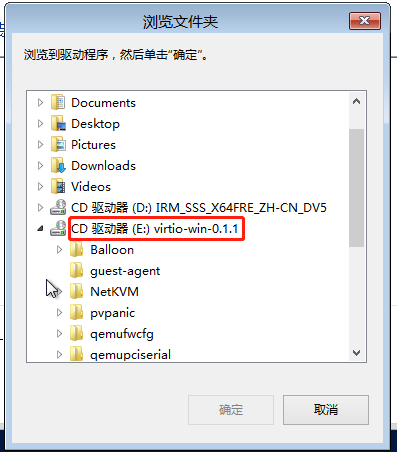
</disk>

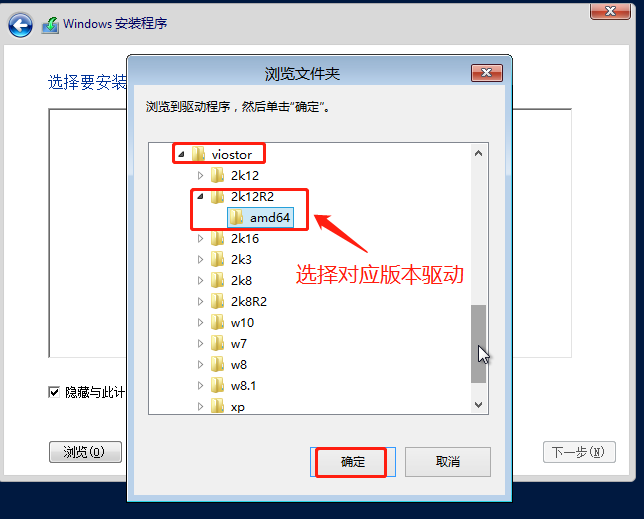


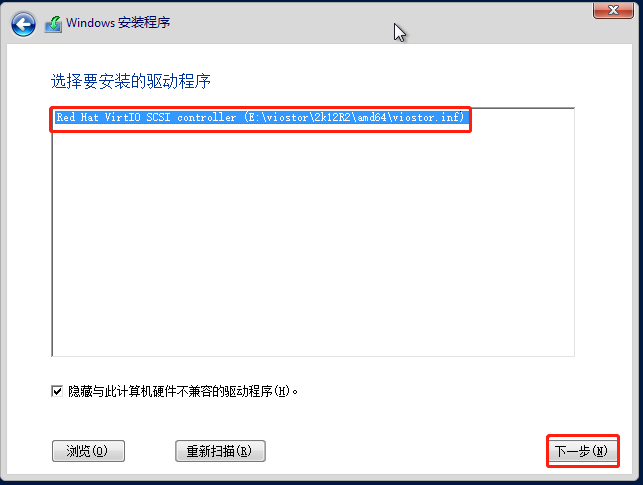












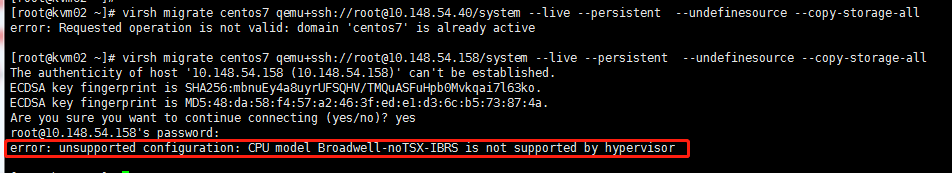


VM**基于本地存储的在线迁移**

**参考URL1 ：**<https://boke.wsfnk.com/archives/213.html>

**URL2 ：**<https://www.jb51.net/LINUXjishu/349638.html>

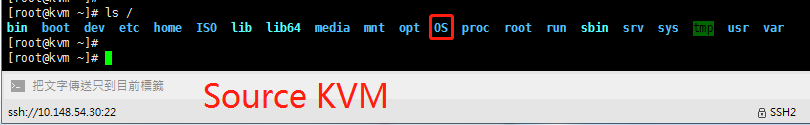
1）硬件要求： 必须是两个相同的CPU才可以

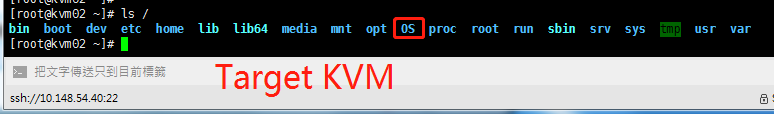


2)软件要求： source kvm和target kvm所用的存储池“系统磁盘目录名称”、和“目录路径”必须一致，断开CD-ROM媒体连接，双方都要ssh认证 ，

[root@localhost ~]# su - nginx -s /bin/bash

“-bash-4.2$ ssh-copy-id root@10.148.54.30” #Copy target KVM id





为了防止日后不同KVM VM迁移，设置libvirtd uuid

[root@kvm ~]# sed -i "/#host\_uuid/ahost\_uuid = \"`uuidgen`\"" /etc/libvirt/libvirtd.conf

[root@kvm ~]# systemctl restart nginx

[root@kvm ~]# systemctl restart libvirtd

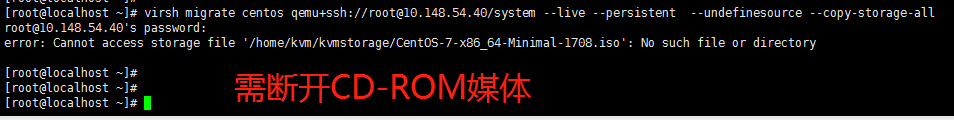
待测试：需要有实时存储迁移（块迁移）支持，标准的RHEL/Centos7的qemu-kvm不支持，oVirt或RHEV支持, ev表示企业级, （安装它会替换原本的qemu-kvm组件为qemu-kvm-ev）

[root@kvm ~]# yum –y install centos-release-qemu-ev qemu-kvm-ev

命令行迁移

[root@kvm ~]# virsh migrate centos7 qemu+ssh://root@10.148.54.40/system --live --persistent --undefinesource --copy-storage-all #在线迁移

No.1 Question

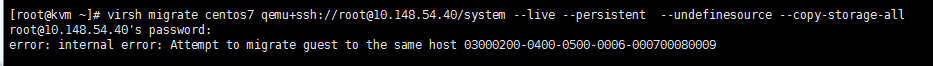


No.2 Question目的地的主机名解析为本地主机，但迁移需要FQDN

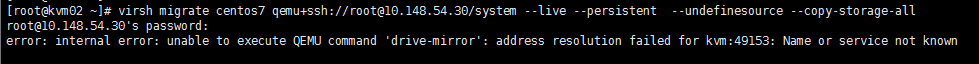


[root@localhost ~]# hostname 10.148.54.40 #登录目标主机临时修改主机名为本机IP

No.3 Question尝试将客户端迁移到相同的主机（修改libvirtd配置文件）



No.4 Question地址解析失败

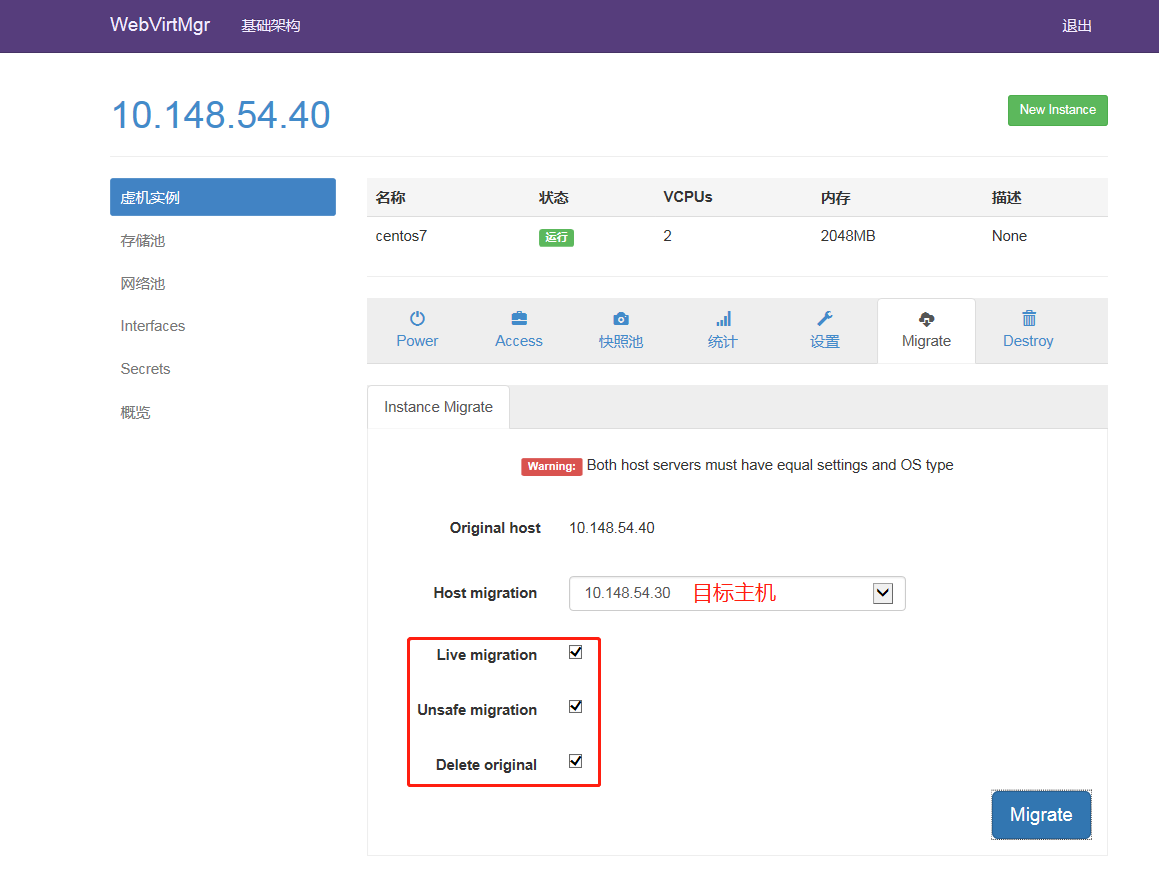


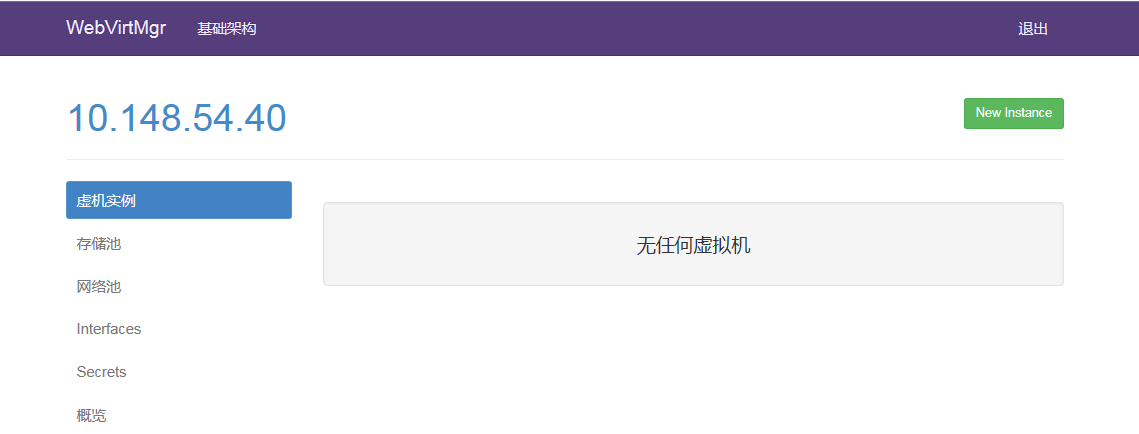
[root@kvm02 ~]# sed -i "/#host\_uuid/ahost\_uuid = \"`uuidgen`\"" /etc/libvirt/libvirtd.conf

[root@kvm ~]# systemctl restart nginx

[root@kvm ~]# systemctl restart libvirtd

图形化迁移





二、**ESXi到KVM之v2v**

## 一、VMwareESXi虚拟平台下linux系统迁移到KVM虚拟平台

## 1、从ESXi导出OVF模板（先关机才能导出）

## 

## 

## 

## 2、创建虚拟机存放目录

## [root@kvm ~]# yum -y install virt-\*

## [root@kvm /]# virt-v2v -i ova /esxi/proxy60.ova -o local -os /OS -of qcow2 -b br0

## 参数说明: -i ova 读取VMware的ova 文件

## -o local 输出到本地的磁盘镜像和一个配置文件

## -os 指定到KVM中的哪个存储池

## -of 迁移转出的虚拟机磁盘格式

## -b 指定KVM中的虚拟网桥

## 

## 在“/OS”目录下会生成文件

## 

## 3、修改网卡配置

## 

## <interface type='bridge'>

## <source bridge='br0'/>

## <model type='virtio'/>

## <address type='pci' domain='0x0000' bus='0x00' slot='0x03' function='0x0'/>

## </interface>

## 4、启动VM

## [root@kvm /]# mv OS/proxy60.xml /etc/libvirt/qemu #移动配置文件到指定路径

## [root@kvm /]# virsh define /etc/libvirt/qemu/proxy60.xml #导入VM

## 

## 查看虚拟机列表

## 

## [root@kvm /]# virsh start proxy60 #启动VM

## 

## root@kvm /]# virsh autostart proxy60 #开机自启

## 5、配置VM及检查可用性

## 1）查看网卡名，修改网卡名和网卡配置文件

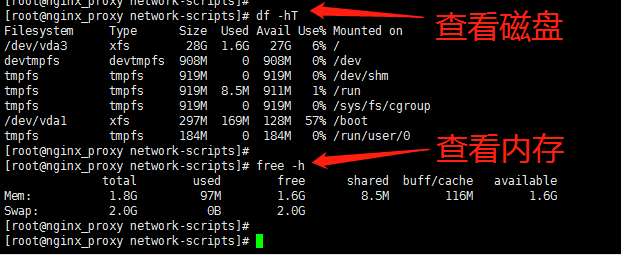
## 2）备份此网卡“ifcfg-Wired\_connection\_1”，否则network.service启动失败，

## 因为Centos7系统启动时不是读取现有网卡文件，而是读取ifcfg-Wired\_connection\_1这个文件

## 3）重启网卡和网卡服务

4）检查网络、查看对应服务、查看磁盘和内存，经测试一切OK





## 二、VMwareESXi虚拟平台下windows系统迁移到KVM虚拟平台

## 1、创建虚拟机存放目录

## [root@kvm /]# virt-v2v -i ova /esxi/winFTP.ova -o local -os /OS -of qcow2 -b br0

## 

## 2、修改网卡配置

## 

## <interface type='bridge'>

## <!-- bridge mapped from "VM Network" to "br0" -->

## <source bridge='br0'/>

## <model type='rtl8139'/>

## <address type='pci' domain='0x0000' bus='0x00' slot='0x03' function='0x0'/>

## </interface>

## # 'rtl8139'为自动生成的网卡型号, 不要修改为virtio, 否则系统无网卡驱动

## 3、启动VM

## [root@kvm /]# mv OS/winFTP.xml /etc/libvirt/qemu #移动配置文件到指定路径

## [root@kvm /]# virsh define /etc/libvirt/qemu/winFTP.xml #导入VM

## [root@kvm /]# virsh start winFTP.xml

## 4、配置VM及检查可用性

## 1）网络类型会变成DHCP，需修改回之前的类型和配置IP

## 2）检查网络、驱动、应用服务、磁盘和内存，经测试一切OK

## 

## 

## 

**三、Other**

虚拟机xml配置文件路径

[root@kvm /]# ls /etc/libvirt/qemu

autostart centos.xml networks Wind2012.xml

虚拟机运行pid路径

[root@kvm /]# ls /run/libvirt/qemu/

Wind2012.pid Wind2012.xml

虚拟机启动时对应VNC端口

[root@kvm /]# virsh dumpxml wind2012 |grep vnc

<graphics type='vnc' port='5901' autoport='yes' listen='0.0.0.0'>

Qemu-kvm创建虚拟磁盘

[root@kvm ~]# qemu-img create -f qcow2 /OS/centos.img 20G

学习链接1、<https://blog.51cto.com/13659301/2157751>

2、<http://www.cnblogs.com/clsn/p/8366251.html>

问题:1

Kvm创建实例时遇到500内部错误,同时images也出现500错误?

问题成因:由于上传的镜像文件在centos上显示为乱码,故导致此错误 ,

解决方案:更改镜像文件名称即可.改成纯英文+数字.

问题2:删除镜像实例后(未删除硬盘及xml文件)如何恢复

进入/etc/libvirt/qemu下寻找对应xml文件

使用virsh create xxx.xml即可

3.kvm扩容问题

Windows:先将vm关机,然后运行,开机扩容会报错.

qemu-img resize /vmdisk/LUA.img +200G

4.关于开机扩容问题,找不到扩容的磁盘还在测试-2020/07/28

先关机 再启动.先关机再启动.

要关机后,再启动, ,cold reboot, not warm reboot,所以不能直接重启.

问题成因: cold reboot 可以将硬盘信息更改写入到bios里面.

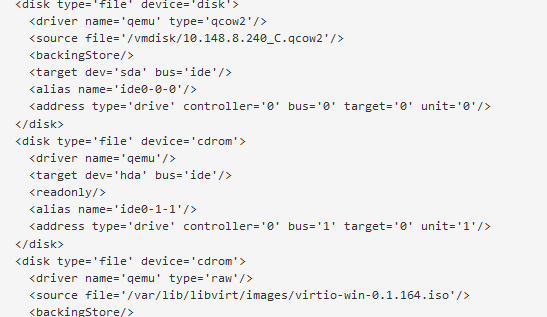
5. **Error message misleads users when 2 or more IDE controllers are configured，only one single ide 。。。。。。**

关于kvm挂载 多个 迁移后强转格式的硬盘问题, 建立instance时 选择virtIO(对应着SCSI模式,不选virtIO对应IDE模式)模式,然后安装驱动

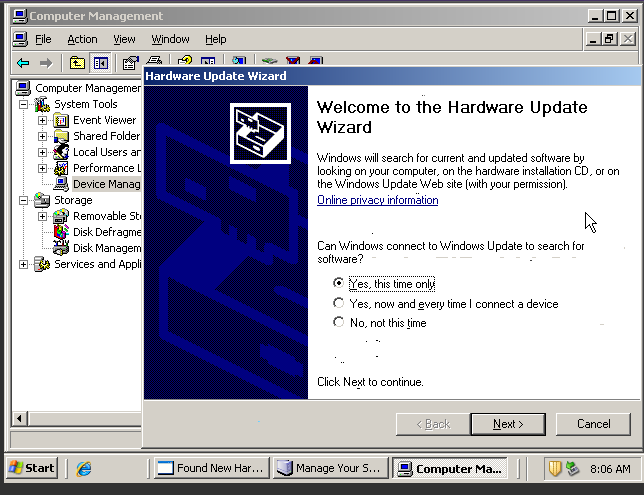
6.关于windows2003 主机迁移后选择virtIo模式下的蓝屏问题解决,

系统盘还是IDE,但是挂载的却是SCSI,

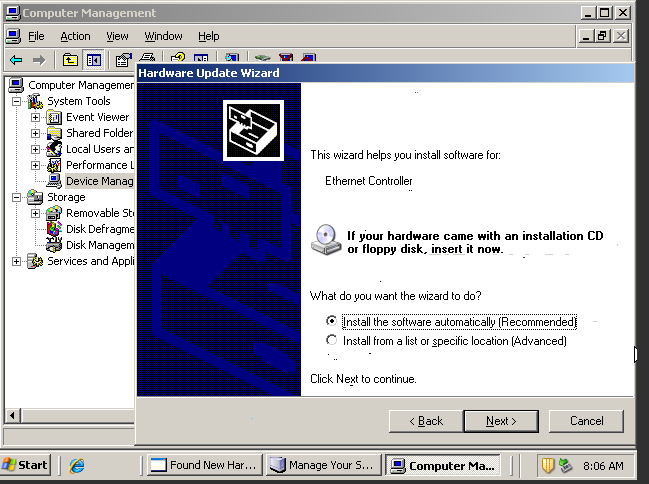
修改xml文件,如图所示,将c盘文件的设备驱动改为dev=sda,并bus=ide,使得c盘可以正常启动，后续的多个硬盘可以通过virtIO模式挂载在主机上。



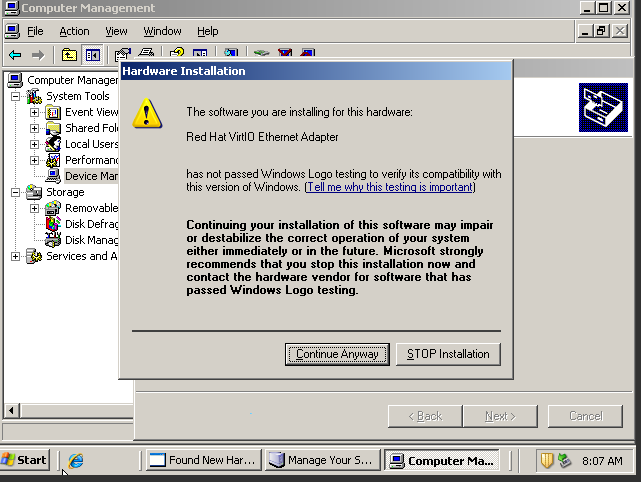
然后开机进入系统，因为之前挂在了virtio的驱动，所以在设备管理界面更新，scsi设备驱动，PCI设备驱动，网卡驱动



选择第一项，只更新一次。



选择自动搜索。因为已经挂上virt-164.iso了，让他自己搜索。



按照提示直接安装就好。，安装驱动后可以看到硬盘，但是对应盘符可能会有所改变（推测是起点不是vda，而是vdb的缘故,应该是硬盘驱动更新后盘符消失.）。

后续如何修改c盘为virtIO模式，重装系统可以。

具体配置如下：

<domain type='kvm' id='23'>

<name>10.148.8.240-t</name>

<uuid>500a6f83-da1f-4c22-ed6d-4ea5eda85639</uuid>

<description>None</description>

<memory unit='KiB'>4194304</memory>

<currentMemory unit='KiB'>4194304</currentMemory>

<vcpu placement='static'>4</vcpu>

<resource>

<partition>/machine</partition>

</resource>

<os>

<type arch='x86\_64' machine='pc-i440fx-rhel7.0.0'>hvm</type>

<boot dev='hd'/>

<boot dev='cdrom'/>

<bootmenu enable='yes'/>

</os>

<features>

<acpi/>

<apic/>

<pae/>

</features>

<cpu mode='custom' match='exact' check='full'>

<model fallback='forbid'>Skylake-Server-IBRS</model>

<vendor>Intel</vendor>

<feature policy='disable' name='ds'/>

<feature policy='disable' name='acpi'/>

<feature policy='require' name='ss'/>

<feature policy='disable' name='ht'/>

<feature policy='disable' name='tm'/>

<feature policy='disable' name='pbe'/>

<feature policy='disable' name='dtes64'/>

<feature policy='disable' name='monitor'/>

<feature policy='disable' name='ds\_cpl'/>

<feature policy='disable' name='vmx'/>

<feature policy='disable' name='smx'/>

<feature policy='disable' name='est'/>

<feature policy='disable' name='tm2'/>

<feature policy='disable' name='xtpr'/>

<feature policy='disable' name='pdcm'/>

<feature policy='disable' name='dca'/>

<feature policy='disable' name='osxsave'/>

<feature policy='disable' name='tsc\_adjust'/>

<feature policy='require' name='clflushopt'/>

<feature policy='disable' name='intel-pt'/>

<feature policy='require' name='pku'/>

<feature policy='disable' name='ospke'/>

<feature policy='require' name='avx512vnni'/>

<feature policy='disable' name='md-clear'/>

<feature policy='require' name='stibp'/>

<feature policy='require' name='ssbd'/>

<feature policy='require' name='hypervisor'/>

<feature policy='disable' name='arat'/>

</cpu>

<clock offset='utc'/>

<on\_poweroff>destroy</on\_poweroff>

<on\_reboot>restart</on\_reboot>

<on\_crash>restart</on\_crash>

<devices>

<emulator>/usr/libexec/qemu-kvm</emulator>

<disk type='file' device='disk'>

<driver name='qemu' type='qcow2'/>

<source file='/vmdisk/10.148.8.240\_C.qcow2'/>

<backingStore/>

<target dev='sda' bus='ide'/>

<alias name='ide0-0-0'/>

<address type='drive' controller='0' bus='0' target='0' unit='0'/>

</disk>

<disk type='file' device='cdrom'>

<driver name='qemu'/>

<target dev='hda' bus='ide'/>

<readonly/>

<alias name='ide0-1-1'/>

<address type='drive' controller='0' bus='1' target='0' unit='1'/>

</disk>

<disk type='file' device='cdrom'>

<driver name='qemu' type='raw'/>

<source file='/var/lib/libvirt/images/virtio-win-0.1.164.iso'/>

<backingStore/>

<target dev='hdb' bus='ide'/>

<readonly/>

<alias name='ide0-0-1'/>

<address type='drive' controller='0' bus='0' target='0' unit='1'/>

</disk>

<disk type='file' device='disk'>

<driver name='qemu' type='qcow2'/>

<source file='/vmdisk/10.148.8.240\_D.qcow2'/>

<backingStore/>

<target dev='vdb' bus='virtio'/>

<alias name='virtio-disk1'/>

<address type='pci' domain='0x0000' bus='0x00' slot='0x05' function='0x0'/>

</disk>

<disk type='file' device='disk'>

<driver name='qemu' type='qcow2'/>

<source file='/vmdisk/10.148.8.240\_E.qcow2'/>

<backingStore/>

<target dev='vdc' bus='virtio'/>

<alias name='virtio-disk2'/>

<address type='pci' domain='0x0000' bus='0x00' slot='0x06' function='0x0'/>

</disk>

<disk type='file' device='disk'>

<driver name='qemu' type='qcow2'/>

<source file='/vmdisk/10.148.8.240\_F.qcow2'/>

<backingStore/>

<target dev='vdd' bus='virtio'/>

<alias name='virtio-disk3'/>

<address type='pci' domain='0x0000' bus='0x00' slot='0x07' function='0x0'/>

</disk>

<disk type='file' device='disk'>

<driver name='qemu' type='qcow2'/>

<source file='/vmdisk/10.148.8.240\_G.qcow2'/>

<backingStore/>

<target dev='vde' bus='virtio'/>

<alias name='virtio-disk4'/>

<address type='pci' domain='0x0000' bus='0x00' slot='0x08' function='0x0'/>

</disk>

<controller type='usb' index='0' model='piix3-uhci'>

<alias name='usb'/>

<address type='pci' domain='0x0000' bus='0x00' slot='0x01' function='0x2'/>

</controller>

<controller type='pci' index='0' model='pci-root'>

<alias name='pci.0'/>

</controller>

<controller type='ide' index='0'>

<alias name='ide'/>

<address type='pci' domain='0x0000' bus='0x00' slot='0x01' function='0x1'/>

</controller>

<interface type='bridge'>

<mac address='52:54:00:52:73:dc'/>

<source network='br1' bridge='br1'/>

<target dev='vnet6'/>

<model type='virtio'/>

<alias name='net0'/>

<address type='pci' domain='0x0000' bus='0x00' slot='0x03' function='0x0'/>

</interface>

<serial type='pty'>

<source path='/dev/pts/12'/>

<target type='isa-serial' port='0'>

<model name='isa-serial'/>

</target>

<alias name='serial0'/>

</serial>

<console type='pty' tty='/dev/pts/12'>

<source path='/dev/pts/12'/>

<target type='serial' port='0'/>

<alias name='serial0'/>

</console>

<input type='mouse' bus='ps2'>

<alias name='input0'/>

</input>

<input type='tablet' bus='usb'>

<alias name='input1'/>

<address type='usb' bus='0' port='1'/>

</input>

<input type='keyboard' bus='ps2'>

<alias name='input2'/>

</input>

<graphics type='vnc' port='5906' autoport='yes' listen='0.0.0.0'>

<listen type='address' address='0.0.0.0'/>

</graphics>

<video>

<model type='cirrus' vram='16384' heads='1' primary='yes'/>

<alias name='video0'/>

<address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x0'/>

</video>

<memballoon model='virtio'>

<alias name='balloon0'/>

<address type='pci' domain='0x0000' bus='0x00' slot='0x09' function='0x0'/>

</memballoon>

</devices>

<seclabel type='dynamic' model='dac' relabel='yes'>

<label>+107:+107</label>

<imagelabel>+107:+107</imagelabel>

</seclabel>

</domain>

##虚拟机克隆猴网络出现问题

,配置无错误,无论怎样都不通,找一下富鸿网,看是不是管控了.

###关于kvm脚本,

1. Pid-vm-monitor.sh :

因为kvm宿主机上无法看到对应vm名称的PID，所以写了这个脚本，使得宿主机可以看到特定名称的vm目前占用的cpu，内存使用情况，后续还可以添加iostat 配合pid查看。目前只是top命令。

1. change\_vm\_name.sh：

因为kvm虚拟机修改名字需要手动修改xml文件以及硬盘名称，所以写了这个脚本配合直接修改后关闭虚拟机，重新定义xml文件，后期优化可以增加扩容选项。

1. nfs\_client\_install.sh：

安装nfs\_client,因为k8s后端存储使用nfs，所以需要批量安装一下。

1. create\_vm.sh：

使用xml模板新建一台指定名称的虚拟机，但是后续装机需要手动到网页点击，后续优化可以使用webvirtmgr克隆选项，批量建立虚拟机。只需要修改克隆后的名称，mac 还有硬盘名称 ，再进入克隆后的虚拟机修改ip地址即可

1. ansible\_scp.sh

拷贝repo文件到ansible 的node虚拟机，docker.repo,k8s.repo,epel.repo,主要是这三个。

1. ipvs.modules：

建立k8s集群需要使用到ipvs模块时 ，加载ipvs模块及算法，查看是否成功，

# lsmod|grep ip\_vs

1. linux\_init.sh

linux初始化脚本，包含selinux和防火墙关闭，swap设备是否关闭，安装必要的vim编辑器和mlocate，lrzsz常用工具等。配置DNS解析，

参考网页：<https://www.cnblogs.com/kevingrace/p/5897424.html>