KVM 基本操作

一、KVM 软件安装

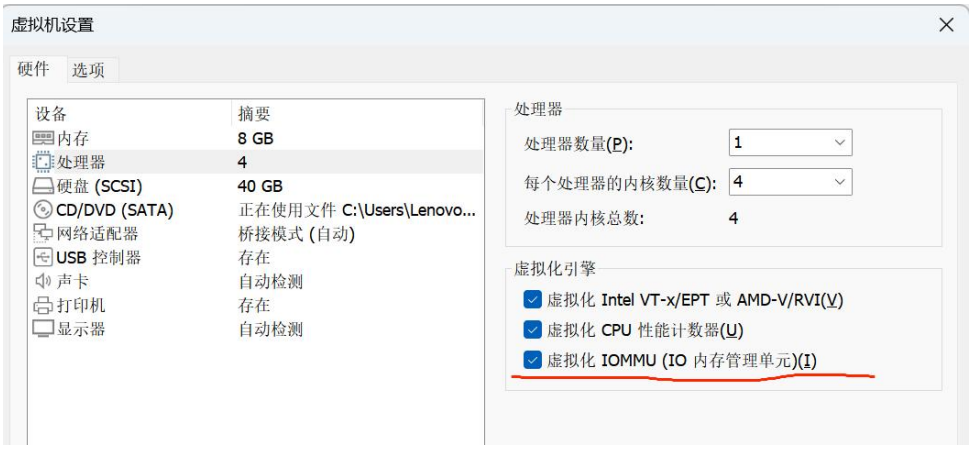
1.环境准备

（1）环境要求：

VMwareworkstation17.0

Centos7.0　64 位，选择带图形界面。

（2）Centos 安装要求，在虚拟机 CPU 设置中勾选以下选项，让 CPU 支持虚拟化。



Centos 内存最好不低于 8GB。

硬盘不低于 20GB。

2.查看 CPU 是否支持 VT 技术

cat /proc/cpuinfo | grep -E 'vmx|svm'

3. 清理环境，卸载 KVM

yum remove `rpm -qa | egrep 'qemu|virt|KVM'` -y

rm -rf /var/lib/libvirt /etc/libvirt/

4、安装软件

yum install *qemu* *virt* librbd1-devel -y

安装内容包括：qemu-KVM 主包、API 接口libvirt、图形管理程序 virt-manager。

5.启动服务

systemctl start libvirtd

6. 查看模块是否加载

lsmod | grep kvm

#查看是否有 KVM_intel 和 KVM 这两个模块

二、GuestOS 安装

1、安装和管理 GuestOS 的方式

（1）图形方式 virt-manager：

操作简单，直观，是一种常用的方式。

（2）完全文本模式

配置复杂，初学者不建议。

（3）命令行模式

常用的一种方式，通常是通过“虚拟机模板+配置文件”的方式实现 Guest OS 的添加。

（4）Cockpit 方式

Web 页面方式。

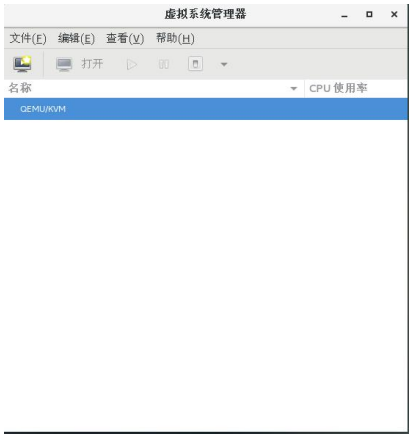
2、图形界面方式安装 GuestOS

（1）安装前准备

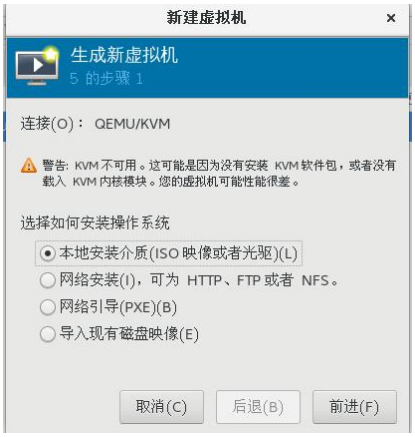
下载 GuestOS 需要安装的操作系统版本，并将其上传到宿主机。本教程 Guest OS 将安装 Centos6.5 的无界面版本。

（2）创建虚拟机

[harry@localhost ~]$ virt-manager



点击“创建虚拟机”图标或者在文件菜单中选择“新建虚拟机”。选择“本地介质安装”，点击“前进”，在第二步中选择“使用 ISO 映像”，在此之前应当将 GuestOS 操作系统的安装镜像文件上传到本地机中。第三步设置 CPU 和内存，此例中可以选内存 4GB，CPU 1 颗。第四步为虚拟机设置磁盘的容量，此例中选 20GB。第五步是设置的虚拟机参数，选择网卡连接方式默认是 NAT 模式。



设置完以上步骤后，点击“完成”，创建虚拟机并启动 GuestOS 操作系统的安装。安装过程略。

3、命令行方式安创建虚拟机（复制虚拟机）

虚拟机的组成部分

（1）虚拟机配置文件

```
root@noviciate ~]# ls /etc/libvirt/qemu
VM1.xml    networks
```

（2）储存虚拟机的介质

```
[root@noviciate ~]# ls /var/lib/libvirt/images
VM1.qcow2
```

（3）根据配置文件创建虚拟机

需要有磁盘镜像文件：

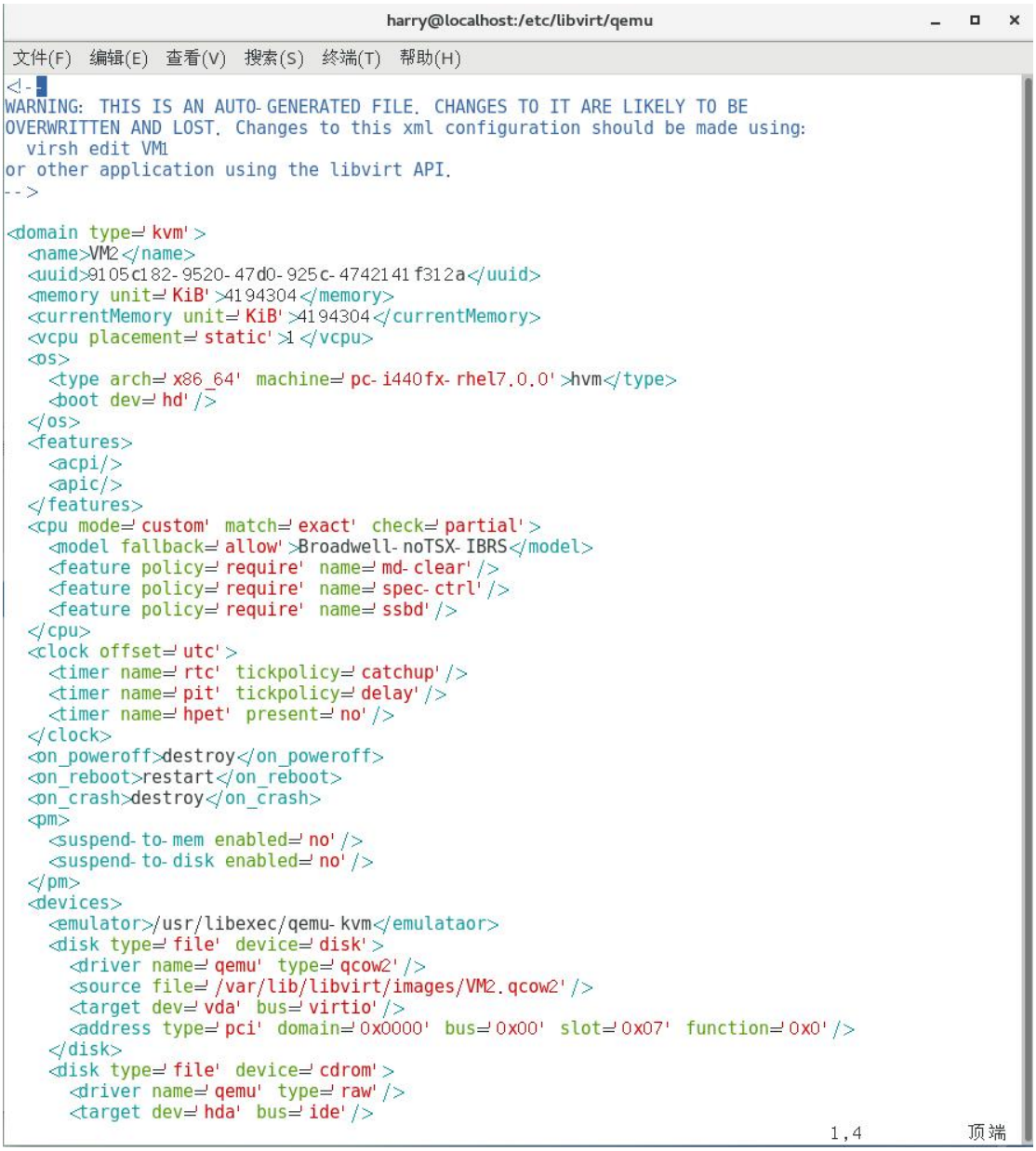
```
[root@noviciate ~]#cd    /var/lib/libvirt/images
[root@noviciate ~]# cp VM1.qcow2 VM2.qcow2
```

需要有配置文件

```
[root@noviciate ~]#cd /etc/libvirt/qemu
[root@noviciate ~]# cp VM1.xml.xml    VM2.xml
```

（4）配置文件需要修改必要的内容

```
[root@noviciate ~]# vim    VM2.xml
```



修改内容包括：虚拟机的名字、虚拟机的 uuid、虚拟机的磁盘文件、虚拟机的 Mac 地址。

（5）创建虚拟机:

[root@localhost qemu]# virsh define /etc/libvirt/qemu/VM2.xml

（6）重启一下 libvirt

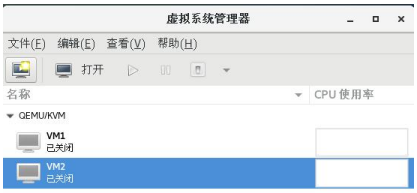
[root@localhost qemu]# systemctl restart libvirtd

（7）开启宿主机路由功能

[root@noviciate ~]vim /etc/sysctl.conf

[root@noviciate ~]sysctl -p

（8）可以启动虚拟机，测试效果。



注意：如果克隆的虚拟机网卡不能启动，请按下述方法执行：

udev 将 mac 与网卡名称的对应关系保存在 # vi /etc/udev/rules.d/70-persistent-net.rules 中，可以看到文件内容如下

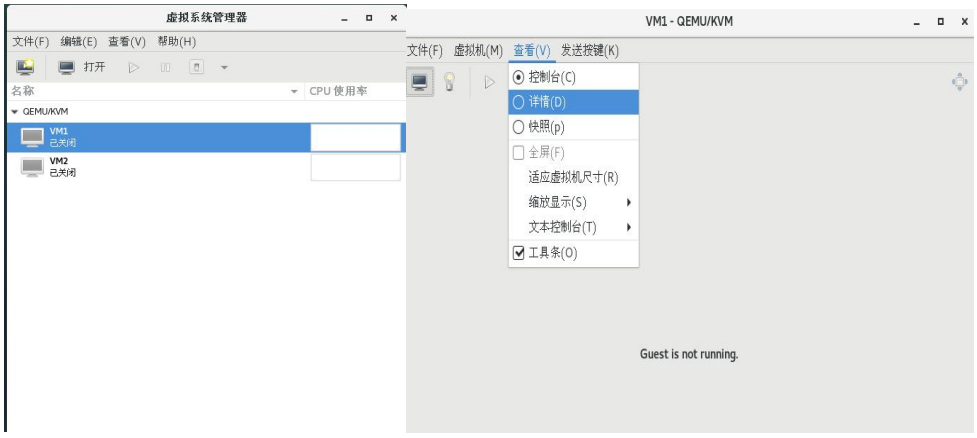
# PCI device 0x8086:0x100f (e1000)
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}=="00:0c:29:7b:60:38", ATTR{type}=="1", KERNEL=="eth*", NAME="eth0"

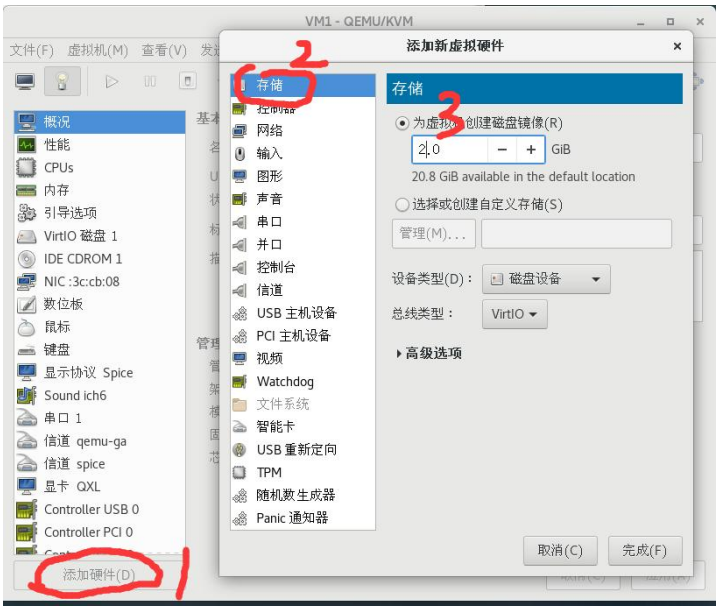
# PCI device 0x8086:0x100f (e1000)
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}=="00:0c:29:29:b9:c5", ATTR{type}=="1", KERNEL=="eth*", NAME="eth1"
处理步骤
步骤 1：将克隆出的虚拟机中只要删除与 NAME="eth0" 相关的行，并把下行的 eth1 的改为 eth0 ；并记录一下 ATTR{address} 的值；

步骤 2：记录此克隆机 MAC 地址，然后编辑 # vi /etc/sysconfig/network-scripts/ifcfg-eth0 将 HWADDR 及 IPADDR 修改一下；如果克隆机和被克隆机该文件中的 uuid 相同，请修改一下。
步骤 3：重启网络服务或者重启系统。

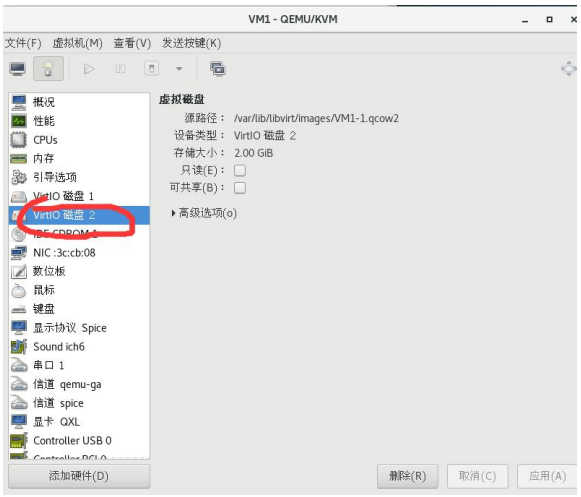
三、GuestOS 的配置修改

1、图形界面模式
虚拟机管理器中双击虚拟机，打开虚拟机窗口。选择“查看”、“详情”，





按图中顺序操作，点击“完成”，添加完毕。



添加完毕后，可以启动虚拟机，登录后用 lsblk 命令查看是否安装成功。

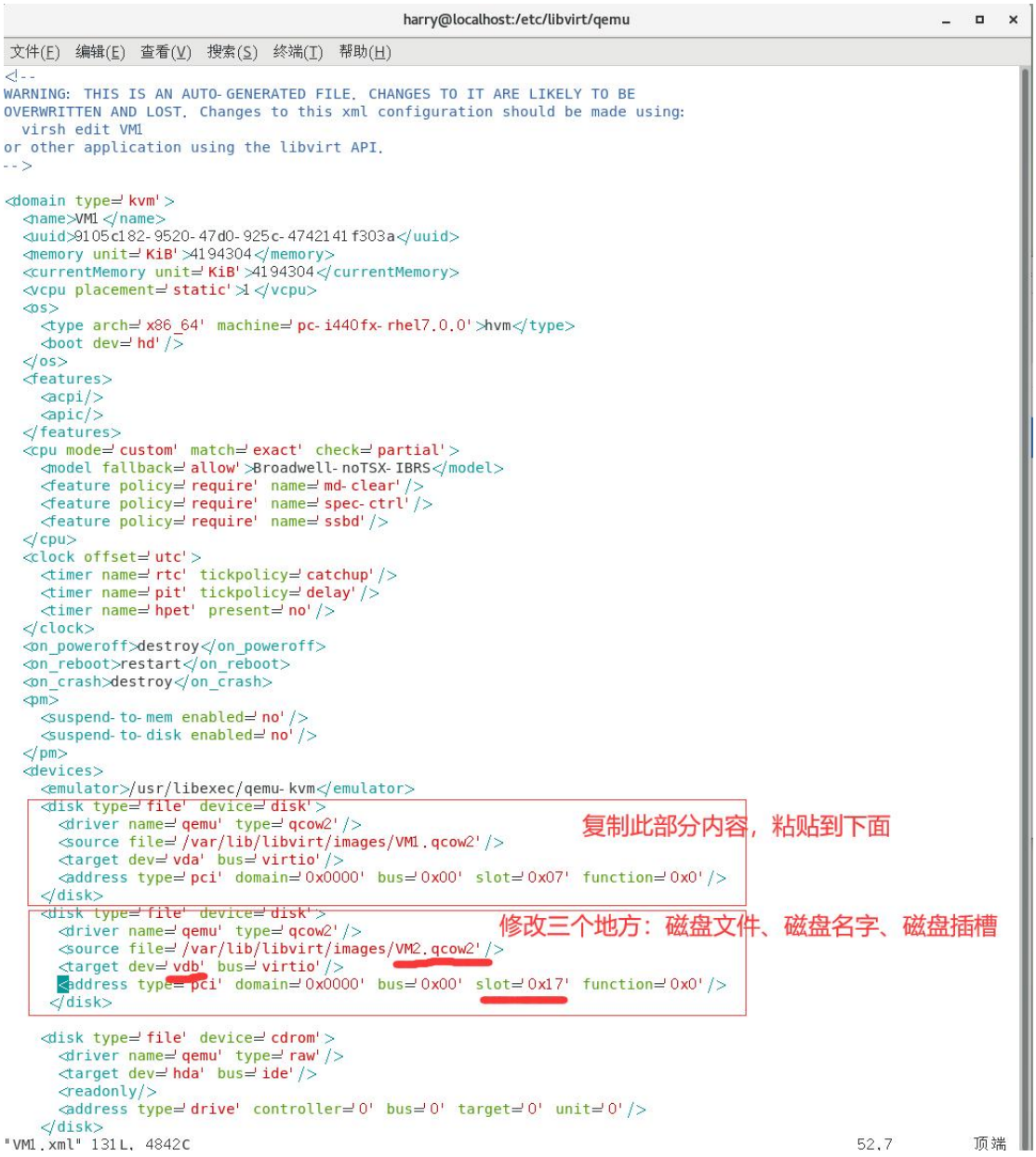
2、通过配置文件更改虚拟配置（以添加磁盘为例）

（1）关闭要修改的虚拟机

（2）打开虚拟机配置文件

[root@localhost images]# cd /etc/libvirt/qemu/

[root@localhost qemu]# vim VM1.xml



（3）创建空白磁盘

[root@localhost qemu]# qemu-img create -f qcow2 /var/lib/libvirt/images/VM2.qcow2 2G

Formatting '/var/lib/libvirt/images/VM2.qcow2', fmt=qcow2 size=2147483648 encryption=off cluster_size=65536 lazy_refcounts=off

（4）重新定义虚拟机

[root@localhost qemu]# virsh define /etc/libvirt/qemu/VM1.xml

定义域 VM1（从 /etc/libvirt/qemu/VM1.xml）

（5）重新启动 libvirtd 服务

[root@localhost qemu]# systemctl restart libvirtd

（6）测试

打开虚拟机用 lsblk 命令查看虚拟机磁盘。

四、存储管理

存储池： KVM 必须要配置一个目录当作他存储磁盘镜像(存储卷)的目录，我们称这个目录为存储池。默认存储池 /var/lib/libvirt/images/文件夹。

1.创建基于文件夹的存储池（目录）
[root@localhost ~]# mkdir -p /data/vmfs
2.定义存储池与其目录
[root@localhost ~]# virsh pool-define-as vmdisk --type dir --target /data/vmfs
3.创建已定义的存储池
(1) 创建已定义的存储池
[root@localhost ~]# virsh pool-build vmdisk
(2) 查看已定义的存储池，存储池不激活无法使用。
[root@localhost ~]#virsh pool-list --all
4.激活并自动启动已定义的存储池
[root@localhost ~]# virsh pool-start vmdisk
[root@localhost ~]# virsh pool-autostart vmdisk
这里vmdisk 存储池就已经创建好了，可以直接在这个存储池中创建虚拟磁盘文件了。
5.在存储池中创建虚拟机存储卷
[root@localhost ~]# virsh vol-create-as vmdisk oeltest03.qcow2 20G --format qcow2

创建完卷后可以将卷挂载到虚拟机上，操作同前面介绍。
注 1:KVM 存储池主要是体现一种管理方式，可以通过挂载存储目录，lvm 逻辑卷的方式创建存储池，
虚拟机存储卷创建完成后，剩下的操作与无存储卷的方式无任何区别了。
注 2:KVM 存储池也要用于虚拟机迁移任务。
6.存储池相关管理命令
(1)在存储池中删除虚拟机存储卷
[root@localhost ~]# virsh vol-delete --pool vmdisk oeltest03.qcow2
(2)取消激活存储池
[root@localhost ~]# virsh pool-destroy vmdisk
(3)删除存储池定义的目录/data/vmfs
[root@localhost ~]# virsh pool-delete vmdisk
(4)取消定义存储池
[root@localhost ~]# virsh pool-undefine vmdisk

五、KVM 管理
1.KVM 基本管理
查看 启动 关闭 重启 重置 查看
查看虚拟机:
[root@localhost ~]# virsh list
[root@localhost ~]# virsh list --all
```
 Id    Name    State
----------------------------------------------------
 2     vm1     running
```

查看 KVM 虚拟机配置文件(X)：
[root@localhost ~]# virsh dumpxml name

将 node4 虚拟机的配置文件保存至node6.xml(X):

[root@localhost ~]# virsh dumpxml node4 > /etc/libvirt/qemu/node6.xml

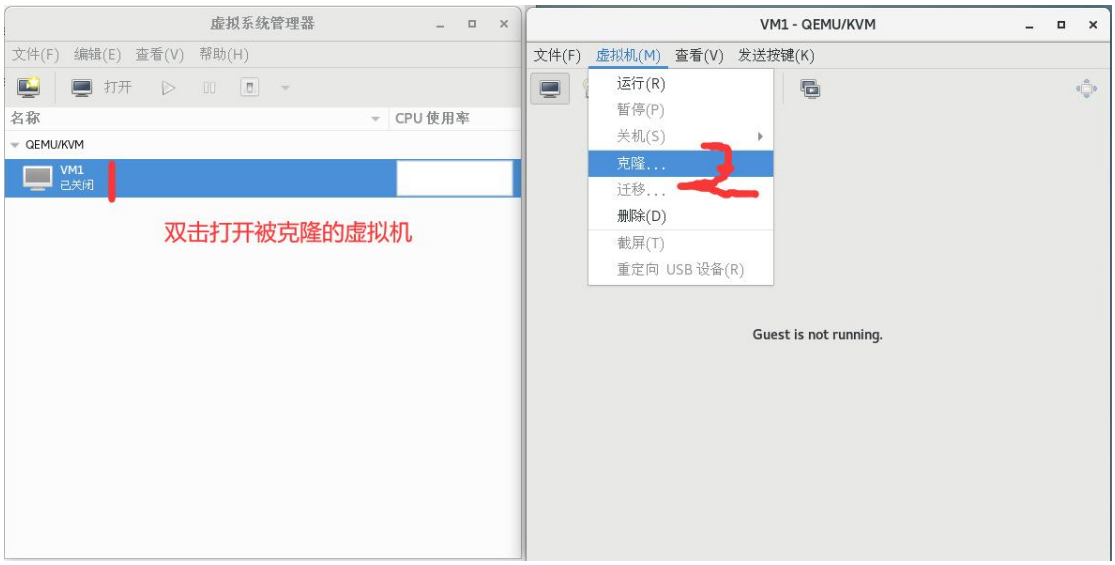
修改 node6 的配置文件(X)：
[root@localhost ~]# virsh edit node6

如果直接用vim 编辑器修改配置文件的话，需要重启 libvirtd 服务
启动:
[root@localhost ~]# virsh start vm1
Domain vm1 started

暂停虚拟机：
#virsh suspend vm_name
恢复虚拟机：
#virsh resume vm_name
关闭：
方法 1：
[root@localhost ~]# virsh shutdown vm1
Domain vm1 is being shutdown
方法 2(X)：
[root@localhost ~]# virsh destroy vm1
Domain vm1 destroyed
重启：
[root@localhost ~]# virsh reboot vm1
Domain vm1 is being reboote
重置:
[root@localhost ~]# virsh reset vm1
Domain vm1 was reset
删除虚拟机
[root@localhost ~]# virsh undefine vm2
Domain vm2 has been undefined
注意:虚拟机在开启的情况下 undefine 是无法删除的，但是如果再 destroy 会直接被删除掉
虚拟机开机自动启动:
[root@localhost ~]# virsh autostart vm1
域 vm1 标记为自动开始
[root@localhost ~]# ls /etc/libvirt/qemu/autostart/ //此目录默认不存在，在有开机启动的虚拟机时自动创建
vm1.xml
域 vm1 取消标记为自动开始
[root@localhost ~]# virsh autostart --disable vm1

查看所有开机自启的 guest os:
[root@Noviciate ~]# ls /etc/libvirt/qemu/autostart/
[root@Noviciate ~]# virsh list --all --autostart
2、克隆虚拟机
方法一：图形界面法

在 virt-manager 中双击打开被克隆的虚拟机，要保证虚拟机是处于关机状态，在虚拟机管理器中按下图操作。



虚拟机克隆后网卡的 Mac 地址、Ip 地址、主机的名字、uuid 等参数是与原主机是一样的，执行克隆操作后虚拟的配置文件上述参数自动被修改。

方法二：采用命令行方式克隆

```
[root@localhost ~]# virt-clone -o vm1 --auto-clone
 WARNING  设置图形设备端口为自动端口，以避免相互冲突。
 正在分配  'vm1-clone.qcow2' | 6.0 GB 00:00:05
 成功克隆  'vm1-clone'。
```

-o 参数指定被克隆的虚拟机名字。

[root@localhost ~]# virt-clone -o vm1 -n vm2 --auto-clone
WARNING 设置图形设备端口为自动端口，以避免相互冲突。
正在分配 'vm2.qcow2' | 6.0 GB 00:00:06
成功克隆 'vm2'。
-n 参数指定克隆机的名字。
[root@localhost ~]# virt-clone -o vm1 -n vm2 -f /var/lib/libvirt/images/vm2.img
正在克隆
vm1.img | 8.0 GB 01:03
Clone 'vm2' created successfully
-f 参数指定为克隆机生成一个磁盘镜像文件。

3、增量镜像
引入增量镜像的目的：
通过一个基础镜像（node.img），里面把各个虚拟机都需要的环境都搭建好，然后基于这个 镜像建立起一个个增量镜像，每个增量镜像对应一个虚拟机，虚拟机对镜像中所有的改变都记 录在增量镜像里面，基础镜像始终保持不变。
优点：
节省磁盘空间，快速复制虚拟机。
实验环境：
基本镜像文件：VM1.qcow2 虚拟机 ID：VM1
增量镜像文件：VM3.qcow2 虚拟机 ID：VM2
要求
以基本镜像文件 VM1.qcow2 为基础，创建一个镜像文件 VM3.qcow2，以此创建一个虚拟机 VM3，虚拟机 VM2 的改变将存储于 VM3.qcow2 中
步骤：
（1）创建增量镜像文件
[root@localhost ~]# cd /var/lib/libvirt/images/
[root@localhost images]# ls
VM1.qcow2 VM2.qcow2
[root@localhost images]# qemu-img create -b VM1.qcow2 -f qcow2 VM3.qcow2
Formatting 'VM3.qcow2', fmt=qcow2 size=9663676416 backing_file='VM1.qcow2' encryption=off
cluster_size=65536 lazy_refcounts=off
[root@localhost images]# qemu-img info VM3.qcow2
image: VM3.qcow2
file format: qcow2
virtual size: 9.0G (9663676416 bytes)
disk size: 196K
cluster_size: 65536
backing file: VM1.qcow2
Format specific information:
compat: 1.1
lazy refcounts: false
（2）创建虚拟机 VM3 的 XML 配置文件
[root@localhost images]# cp /etc/libvirt/qemu/VM1.xml /etc/libvirt/qemu/VM3.xml

按下面的提示进行修改

```
<!--
WARNING: THIS IS AN AUTO-GENERATED FILE. CHANGES TO IT ARE LIKELY TO BE
OVERWRITTEN AND LOST. Changes to this xml configuration should be made using:
  virsh edit VM1
or other application using the libvirt API.
-->
<domain type='kvm'>
  <name>VM3</name>    #虚拟机名，须修改，否则与基本虚拟机冲突
  <uuid>2b9bc095-892c-46e1-8add-afdbde51d49e</uuid>  #uuid，须修改，否则与基本虚拟机冲突,只需要修改其中一位即可。
  <memory unit='KiB'>4194304</memory>
  <currentMemory unit='KiB'>4194304</currentMemory>
  <vcpu placement='static'>1</vcpu>
  <os>
    <type arch='x86_64' machine='pc-i440fx-rhel7.0.0'>hvm</type>
    <boot dev='hd'/>
  </os>
  <features>
    <acpi/>
    <apic/>
  </features>
  <cpu mode='custom' match='exact' check='partial'>
    <model fallback='allow'>Broadwell-noTSX-IBRS</model>
    <feature policy='require' name='md-clear'/>
    <feature policy='require' name='spec-ctrl'/>
    <feature policy='require' name='ssbd'/>
  </cpu>
  <clock offset='utc'>
    <timer name='rtc' tickpolicy='catchup'/>
    <timer name='pit' tickpolicy='delay'/>
    <timer name='hpet' present='no'/>
  </clock>
  <on_poweroff>destroy</on_poweroff>
  <on_reboot>restart</on_reboot>
  <on_crash>destroy</on_crash>
  <pm>
    <suspend-to-mem enabled='no'/>
    <suspend-to-disk enabled='no'/>
  </pm>
  <devices>
    <emulator>/usr/libexec/qemu-kvm</emulator>
    <disk type='file' device='disk'>
      <driver name='qemu' type='qcow2'/>
      <source file='/var/lib/libvirt/images/VM3.qcow2'/>#将原指向/virhost/KVM_node/node.img 改为 VM3.qcow2
      <target dev='vda' bus='virtio'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x07' function='0x0'/>
    </disk>
    <disk type='file' device='disk'>
      <driver name='qemu' type='qcow2'/>
      <source file='/var/lib/libvirt/images/VM2.qcow2'/>
      <target dev='vdb' bus='virtio'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x17' function='0x0'/>
    </disk>
    <disk type='file' device='cdrom'>
      <driver name='qemu' type='raw'/>
      <target dev='hda' bus='ide'/>
      <readonly/>
      <address type='drive' controller='0' bus='0' target='0' unit='0'/>
    </disk>
    <controller type='usb' index='0' model='ich9-ehci1'>
    <interface type='network'>
      <mac address='52:54:00:17:ed:83'/>#修改网卡 MAC，防止冲突
      <source network='default'/>
      <model type='virtio'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x03' function='0x0'/>
    </interface>
```

(3) 根据 xml 配置定义虚拟机 VM3.xml

```
[root@localhost images]# virsh define      /etc/libvirt/qemu/VM3.xml
定义域 VM3（从 /etc/libvirt/qemu/VM3.xml）
[root@localhost images]# virsh start VM3
域 VM3 已开始
```

(4)测试

```
[root@localhost images]# virsh start VM3
域 VM3 已开始
[root@localhost images]# du -h VM1.qcow2
9.1G VM1.qcow2
[root@localhost images]# du -h VM3.qcow2
7.3M       VM3.qcow2
[root@localhost images]#
```

可以向虚拟机 VM3 上传一个文件，然后再次检查两个虚拟机磁盘空间的变化。

4、虚拟机创建快照

```
[root@localhost images]#    virsh snapshot-create-as VM1 VM1.snap1
已生成域快照 VM1.snap1
[root@localhost images]# ls
VM1.qcow2    VM2.qcow2    VM3.qcow2
[root@localhost images]# virsh snapshot-list VM1
 名称                 生成时间                 状态
------------------------------------------------------------
 VM1.snap1            2023-10-28 22:32:49 -0400 shutoff
```

说明：如果虚拟机的磁盘格式为 RAW 格式，请将该格式转换为 qcow2 格式然后再创建快照，否则容易报错。

恢复快照操作

关闭虚拟机，恢复到快照

```
[root@localhost images]# virsh shutdown VM1
[root@localhost images]# virsh snapshot-revert VM1    VM1.snap1
```

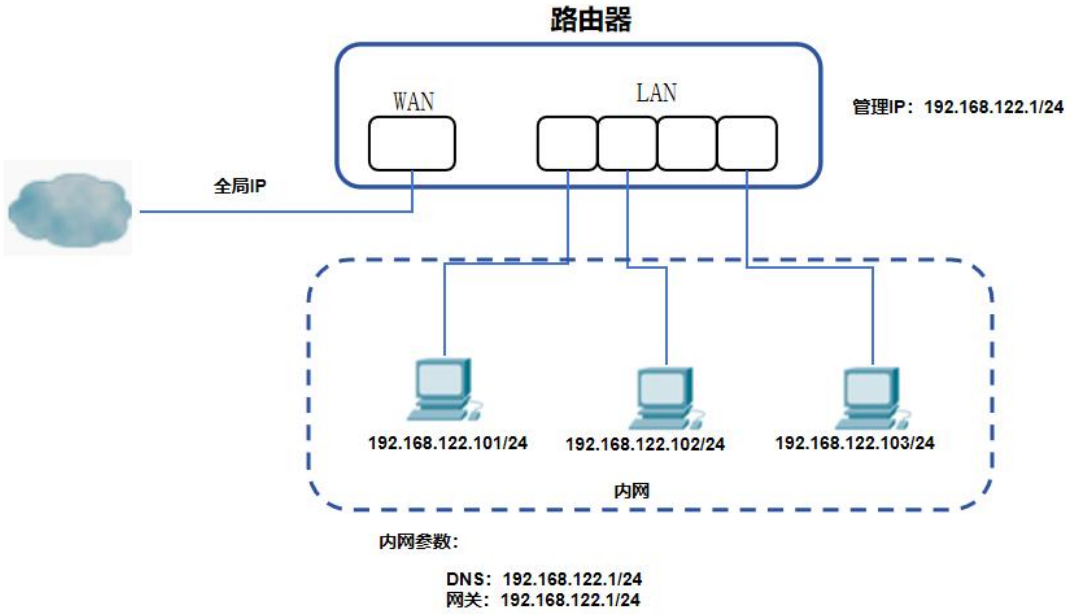
四、KVM 网络管理

1、KVM 网络类型

（1）NAT 模式

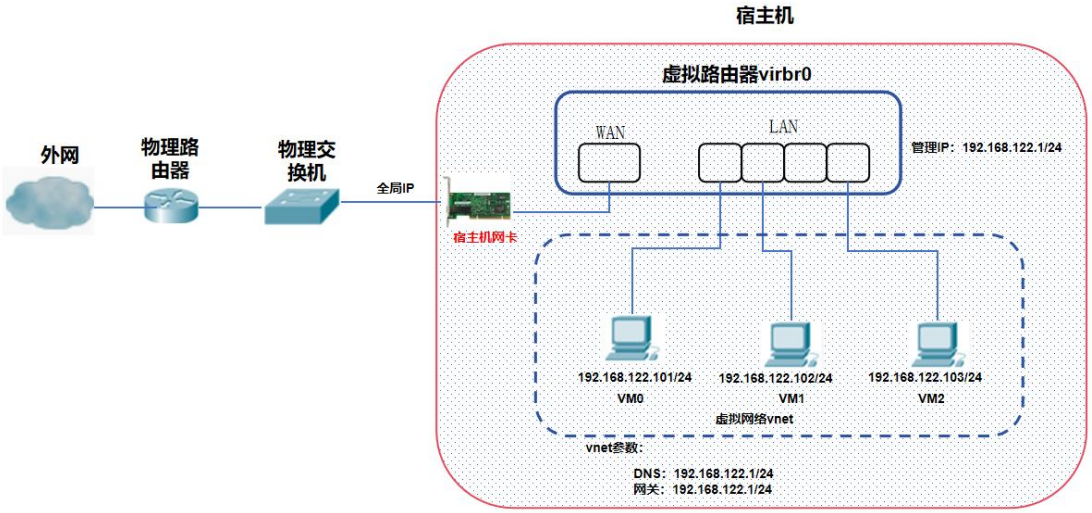
网络地址转换地址 NAT（Network Address Translation）主要用于实现位于内部网络的主机访问外部网络的功能。通过 NAT 技术可以将私网地址转化为公网地址，并且多个私网地址用户可以公用一个公网地址，这样既可以保证网络互通，又节约了公网地址。

下图是物理网络采用 NAT 模式的拓扑示意图。



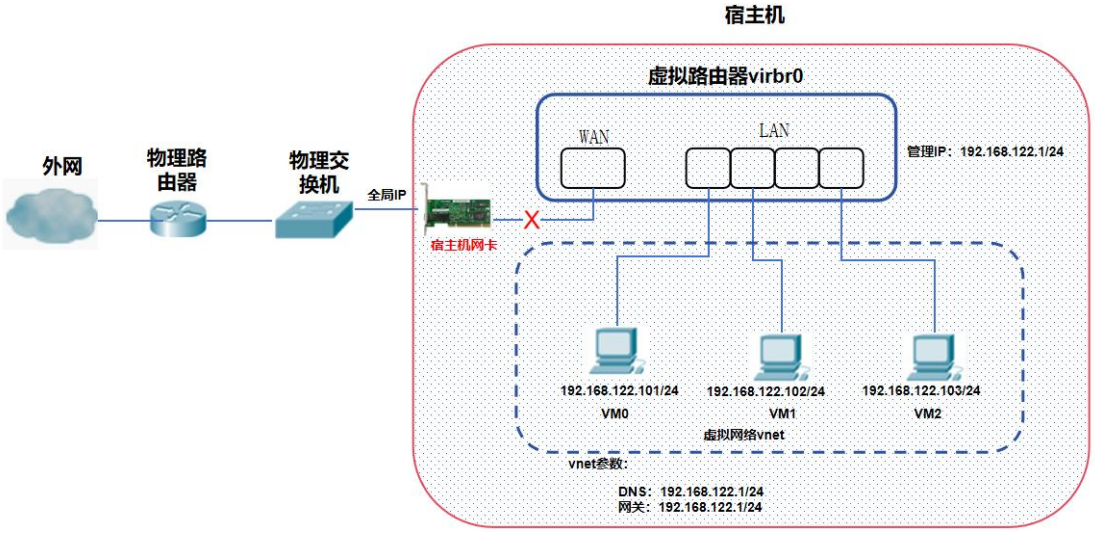
KVM 的 NAT 地址转换如下图所示。

NAT 模式:把定义在网桥之中的网络地址通过 NAT 转换，离开网桥(宿主机)，这个网桥内的网络必须指定 1 个宿主机的物理网卡作为 NAT 端口。该模式将源地址 ip 改为物理网卡 ip 发送给目标地址，目标地址 ip 回传给物理网卡，在将报文发送至虚拟主机。NAT 网络是 1 个虚拟网络。该模式下宿主机的网卡相当于物理网络中真实路由器的 WAN 接口，或者可以认为宿主机的网卡充当了虚拟路由器的 WAN 接口。



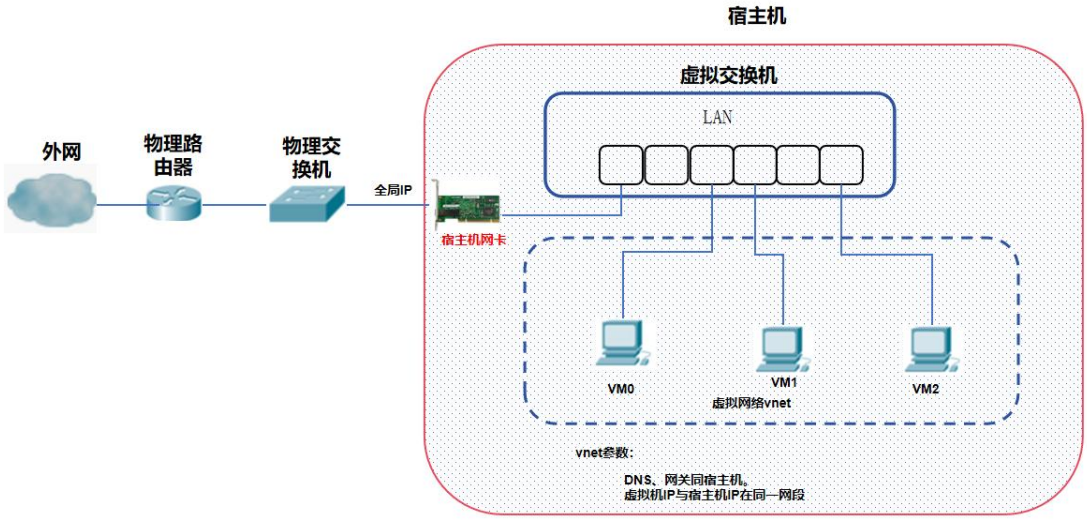
（2）隔离模式 isolated

隔离模式下不允许虚拟机与宿主机外部通信。其拓扑如下：



（3）桥接模式 bridge

桥接模式相当于虚拟机直接通过虚拟交换机连接外网，虚拟机的 IP 与宿主机 IP 在同一个网段，其拓扑结构如下：

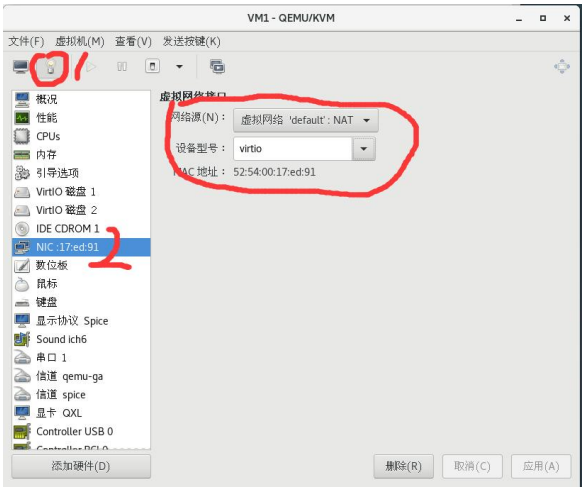


桥接模式的虚拟交换机也称作网桥。

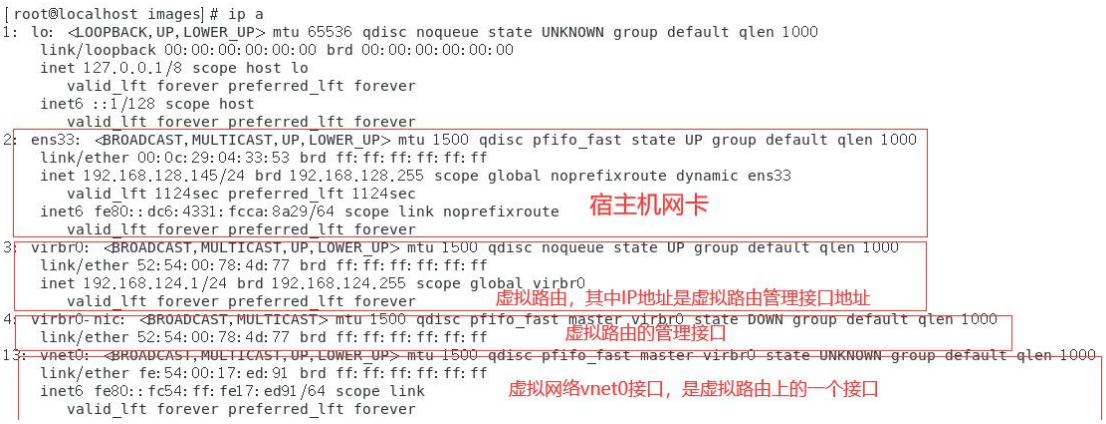
2、查看当前虚拟机的网络连接模式

图形界面方式查看：

在宿主机打开 virt-manager，双击虚拟机，打开虚拟机窗口，见下图。



用命令查看当前宿主机上的虚拟路由




```
[root@localhost images]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:04:33:53 brd ff:ff:ff:ff:ff:ff
    inet 192.168.128.145/24 brd 192.168.128.255 scope global noprefixroute dynamic ens33
       valid_lft 1124sec preferred_lft 1124sec
    inet6 fe80::dc6:4331:fcca:8a29/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: virbr0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether 52:54:00:78:4d:77 brd ff:ff:ff:ff:ff:ff
    inet 192.168.124.1/24 brd 192.168.124.255 scope global virbr0
       valid_lft forever preferred_lft forever
4: virbr0-nic: <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo_fast master virbr0 state DOWN group default qlen 1000
    link/ether 52:54:00:78:4d:77 brd ff:ff:ff:ff:ff:ff
13: vnet0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast master virbr0 state UNKNOWN group default qlen 1000
    link/ether fe:54:00:17:ed:91 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::fc54:ff:fe17:ed91/64 scope link
       valid_lft forever preferred_lft forever
```

root@localhost images]# brctl show

bridge name	bridge id	STP enabled	interfaces
virbr0	8000.525400784d77	yes	virbr0-nic
			vnet0

从上图可以看出当前虚拟路由上连接了两个接口，分别是管理接口和 vnet0 接口。

可以通过以下命令对虚拟路由删除和添加接口。

从交换机上把 vnet0网卡删除：brctl delif virbr0 vnet0

添加 vnet网卡到交换机上 :brctl addif virbr0 vnet0

3、采用配置文件的方式创建桥接模式网络

以下操作在在宿主机上操作。

步骤 1：查看本机 IP 地址和网关。

```
[root@localhost images]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:04:33:53 brd ff:ff:ff:ff:ff:ff
    inet 192.168.128.145/24 brd 192.168.128.255 scope global noprefixroute dynamic ens33
       valid_lft 1570sec preferred_lft 1570sec
    inet6 fe80::dc6:4331:fcca:8a29/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: virbr0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default qlen 1000
    link/ether 52:54:00:78:4d:77 brd ff:ff:ff:ff:ff:ff
    inet 192.168.124.1/24 brd 192.168.124.255 scope global virbr0
       valid_lft forever preferred_lft forever
4: virbr0-nic: <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo_fast master virbr0 state DOWN group default qlen 1000
    link/ether 52:54:00:78:4d:77 brd ff:ff:ff:ff:ff:ff
[root@localhost images]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.128.2   0.0.0.0         UG    100    0        0 ens33
192.168.124.0   0.0.0.0         255.255.255.0   U     0      0        0 virbr0
192.168.128.0   0.0.0.0         255.255.255.0   U     100    0        0 ens33
```

步骤 2：创建 ifcfg-br0 文件

[root@localhost images]# cd /etc/sysconfig/network-scripts/

[root@localhost network-scripts]# vim ifcfg-br0

```
TYPE=Bridge
NAME=br0
DEVICE=br0
ONBOOT="yes"
BOOTPROTO=static
IPADDR=192.168.128.145
GATEWAY=192.168.128.2
NETMASK=255.255.255.0
DNS2=114.114.114.114
DNS2=8.8.8.8
```

步骤 3：修改当前网卡的配置文件，将宿主机网卡绑定到网桥上。

备份当前网卡的配置文件，以备恢复配置。

[root@localhost network-scripts]# cp ifcfg-ens33 ifcfg-ens33bak

清空当前配置文件中的内容，并添加以下内容。

[root@localhost network-scripts]# vim ifcfg-ens33

```
文件(F)  编辑(E)  查看(V)  搜索(S)  终端(T)  帮助(H)
DEVICE="enp33"
ONBOOT="yes"
BRIDGE=br0
```

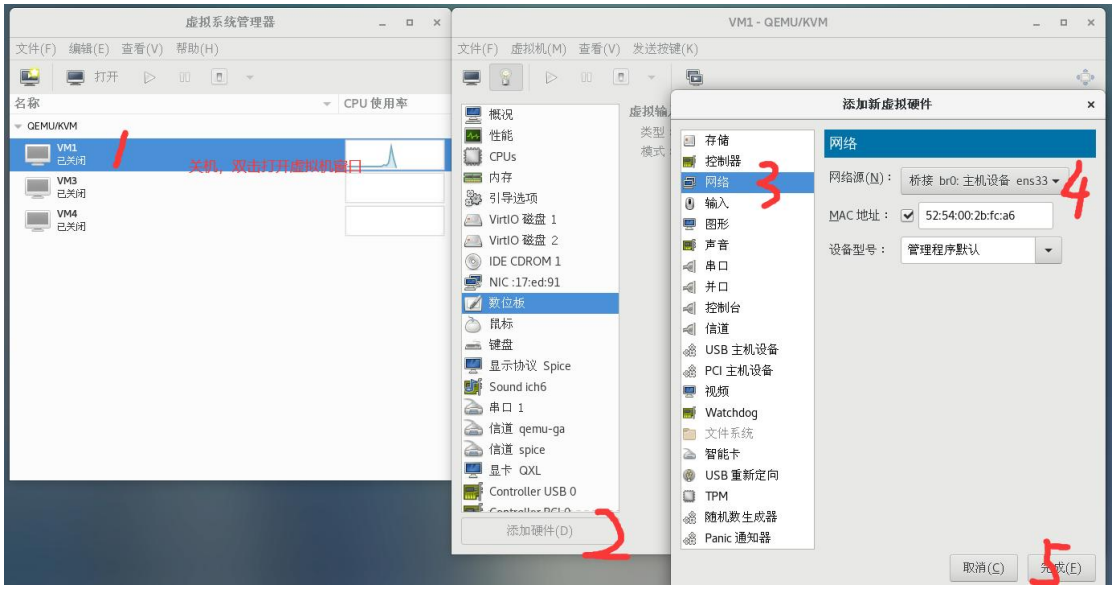
步骤 4：重启 libvirtd 服务

[root@localhost network-scripts]# systemctl restart libvirtd

步骤 5：重启 network 服务

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

步骤 5：给虚拟机添加网卡，并选择桥接到新添加的 br0。



步骤 6：启动虚拟机，查看新添加网卡的状态及 IP 地址，看看是否获得了与宿主机同网段的 IP 地址。

问题：添加新的网卡后虚拟机/etc/sysconfig/network-scripts 文件夹下可能没有网卡的配置文件，虚拟机网卡不一定能够启动，请自行百度解决该问题。

4、采用配置文件的方式创建 NAT 模式网络

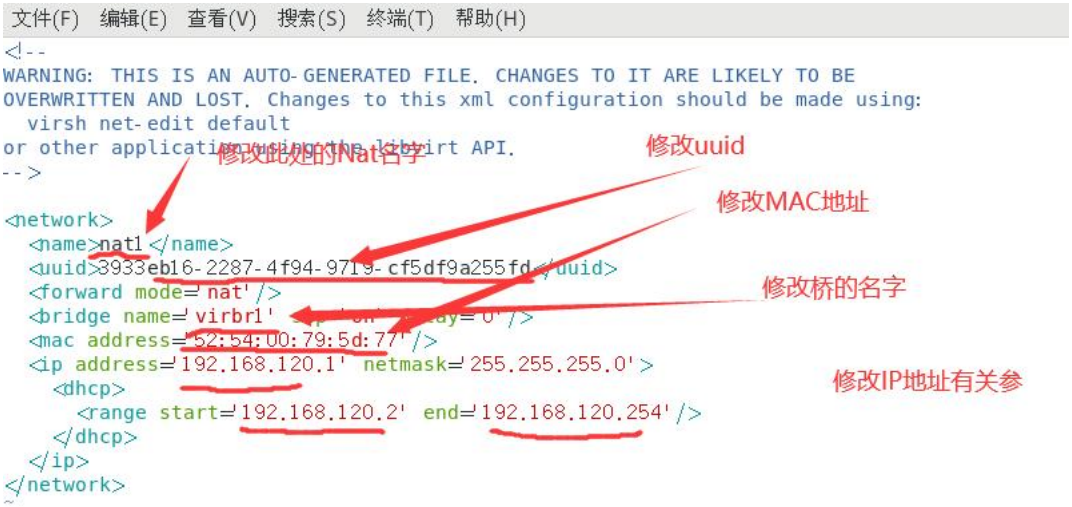
步骤 1：确保虚拟机关机，在宿主上将默认的 NAT 配置文件复制一份儿，并命名为新的文件名。

[root@localhost network-scripts]# cd /etc/libvirt/qemu/networks/

[root@localhost networks]# cp default.xml nat1.xml

步骤 2：修改 nat1.xml 文件

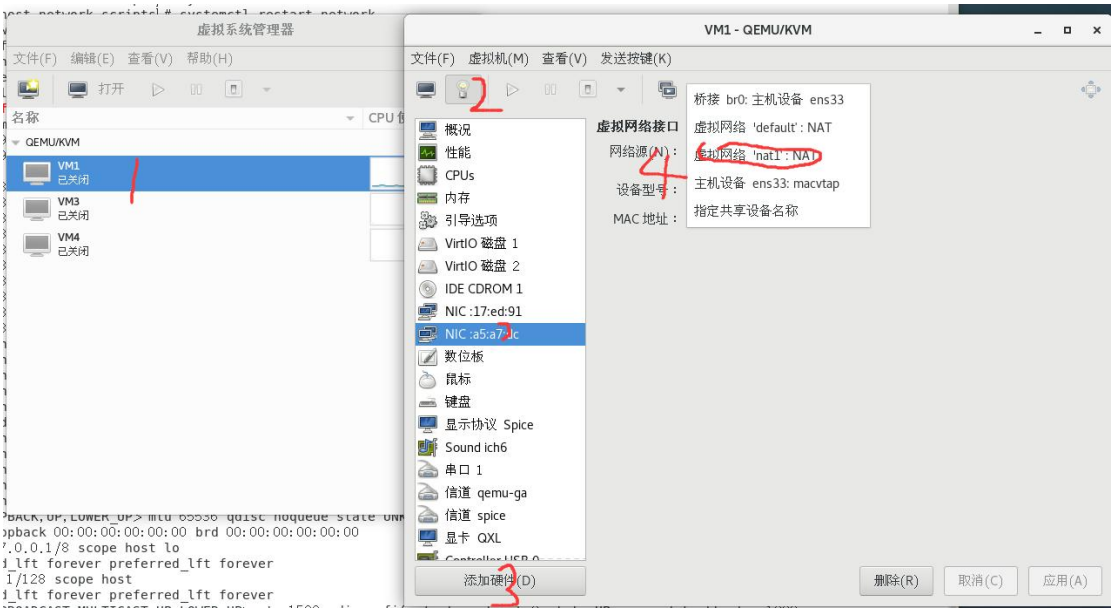
[root@localhost networks]# vim nat1.xml



步骤 3：重启 libvirtd 服务

[root@localhost networks]# systemctl restart libvirtd

步骤 4：进入 virt-mananger 虚拟机管理程序，双击要添加 nat 网络的虚拟机，按下面步骤操作。



步骤 5：启动虚拟机，测试虚拟机是否已经获得 nat1 网络的 IP 地址。

问题：添加新的网卡后虚拟机/etc/sysconfig/network-scripts 文件夹下可能没有网卡的配置文件，虚拟机网卡不一定能够启动，请自行百度解决该问题。