

Pass-through 模式部署 Supernova

(界面方式和命令行方式)

网测科技

2023.03.01

目录

KVM 介绍	4
1. 安装 KVM 前准备	4
1.1 安装 CentOS 系统时注意:	4
1.2 禁用它如下:.....	5
1.3 验证 cpu 是否支持 KVM, 结果有 vmx (INTEL) 或 svm (AMD) 说明支持	5
1.4 在 BIOS 中开机虚拟化功能	5
1.5 关闭 XELinux.....	6
2. 安装 KVM	6
2.1 安装 KVM 包	6
2.2 安装 KVM 核心包——虚拟操作系统模拟器加速模块.....	7
2.3 重启宿主机, 加载 KVM 相关模块.....	7
2.4 查看 KVM 模块是否被正确加载	7
2.5 开启 KVM 服务, 并设置开机启动.....	7
2.6 查看操作结果, 出现 Active: active (running) 字样则说明运行情况良好.....	7
2.7 如果执行“开启 KVM 服务”报错请更新系统.....	8
3. 需要启用 PCI Pass-through, 在打开的文件中添加 intel_iommu=on 参数.....	8
3.1 更新 GRUB 后重启使之生效	8
4. 对 PCI 网卡分离.....	9
4.1 查看宿主机上的 PCI 设备并找到要操作的网卡.....	9
4.2 找到该设备的 PCI 编号.....	9
4.3 分离这两个 PCI 网卡.....	10
5.virt-manager 图形界面部署虚拟机.....	10
5.1 添加存储池.....	10
5.2 创建虚拟机.....	14
5.3 再增加一块 Data 盘 (Supernova 默认要用到两块硬盘)	16
5.4 把从主机上分离出来的 PCI 网卡添加到虚拟机上来	19
5.5 调整 CPU 和内存大小.....	21
6.启动虚拟机 Supernova	22
6.1 运行	22
6.2 成功运行后测试仪 dhcp 自动获取 IP 地址	24
6.3 登陆 Supernova 测试能否访问和运行用例	25
7 使用命令行方式部署虚拟机.....	26
7.1.创建 pass-through 文件夹	26
7.2 进入该目录上传镜像.....	26
7.3.创建 pass-through 的池并启动	26
7.5.上传虚拟机的 xml 文件.....	27
7.6 启动虚拟机.....	28
8.附加: VNC 配置方法	30
8.1 安装软件包.....	30
8.2 关闭防火墙.....	30

8.3 复制配置文件.....	30
8.4 编辑复制出来的配置文件.....	30
8.5 重新加载配置文件.....	30
8.6 设置 VNC 密码.....	30
8.7 开启 VNC 并设置成开机启动.....	31
8.8 启动.....	31
8.9 如果启动成功端口是监听状态(VNC 端口号默认 5900+1).....	31
8.10 客户端连接.....	31

KVM 介绍

KVM 是 Kernel-based Virtual Machine 的简称，是一个开源的系统虚拟化模块，自 Linux 2.6.20 之后集成在 Linux 的各个主要发行版本中。它使用 Linux 自身的调度器进行管理。KVM 目前已成为学术界的主流 VMM 之一。

KVM 的虚拟化需要硬件支持(如 Intel VT 技术或 AMD V 技术)。是基于硬件的完全虚拟化。

所谓 Pass-through 技术是指可以将 PCI/PCIe 设备绕过虚拟机平台直接分配给虚拟机使用
下文是将 Supernova 部署到 Pass-through 模式 KVM 的部署方法
宿主机：就是实体机

1. 安装 KVM 前准备

1.1 安装 CentOS 系统时注意：



1.2 禁用它如下:

命令: chkconfig NetworkManager off

命令: chkconfig network on

命令: service NetworkManager stop

1.3 验证 cpu 是否支持 KVM, 结果有 vmx (INTEL) 或 svm (AMD) 说明支持

命令: cat /proc/cpuinfo | egrep 'vmx|svm'

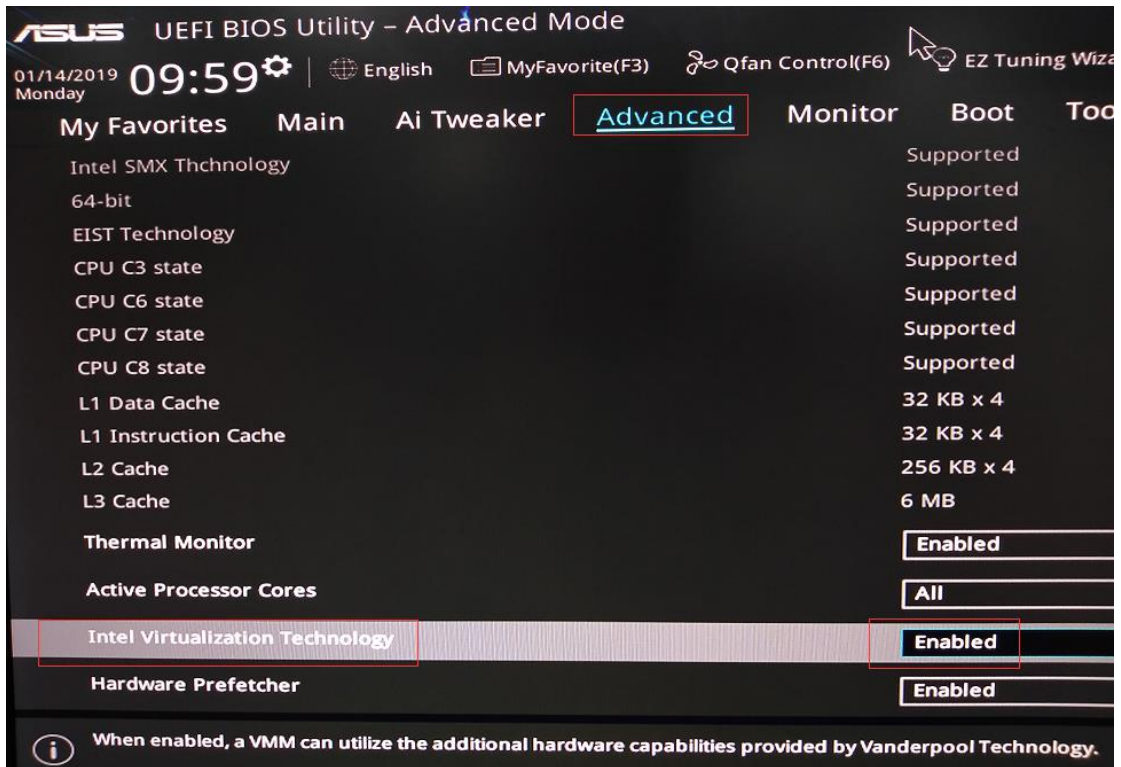
效果:

```
[root@localhost ~]# egrep '(vmx|svm)' /proc/cpuinfo
flags       : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx
pdpe1gb rdtscp lm constant_tsc art arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc aperfmperf eagerfpu pni pclmulqdq dtes64 monitor ds_cp
1 vmx smx est tm2 sse3 fma cx16 xtpr pdcm pcid sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx f16c rdrand lahf_lm abm 3dnowpref
etch epb intel_pt tpr_shadow vnmi flexpriority ept vpid fsgsbase tsc_adjust bmi1 hle avx2 smep bmi2 erms invpcid rtm mpx rdseed adx smap clflushopt
xsaveopt xsavec xgetbv1 dtherm ida arat pln pts hwp hwp_notify hwp_act_window hwp_epp
flags       : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx
pdpe1gb rdtscp lm constant_tsc art arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc aperfmperf eagerfpu pni pclmulqdq dtes64 monitor ds_cp
1 vmx smx est tm2 sse3 fma cx16 xtpr pdcm pcid sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx f16c rdrand lahf_lm abm 3dnowpref
etch epb intel_pt tpr_shadow vnmi flexpriority ept vpid fsgsbase tsc_adjust bmi1 hle avx2 smep bmi2 erms invpcid rtm mpx rdseed adx smap clflushopt
xsaveopt xsavec xgetbv1 dtherm ida arat pln pts hwp hwp_notify hwp_act_window hwp_epp
flags       : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx
pdpe1gb rdtscp lm constant_tsc art arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc aperfmperf eagerfpu pni pclmulqdq dtes64 monitor ds_cp
1 vmx smx est tm2 sse3 fma cx16 xtpr pdcm pcid sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx f16c rdrand lahf_lm abm 3dnowpref
etch epb intel_pt tpr_shadow vnmi flexpriority ept vpid fsgsbase tsc_adjust bmi1 hle avx2 smep bmi2 erms invpcid rtm mpx rdseed adx smap clflushopt
xsaveopt xsavec xgetbv1 dtherm ida arat pln pts hwp hwp_notify hwp_act_window hwp_epp
[root@localhost ~]#
```

1.4 在 BIOS 中开机虚拟化功能

方法: 开启按 delete 键进入 BIOS 中--用键盘方向键选中“Advanced”菜单--选中“CPU Configuration”—找到“Intel Virtualization Technology”开启

效果:



1.5 关闭 XELinux

命令: vi /etc/sysconfig/selinux

效果:

```
# 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=disabled
# SELINUXTYPE= can take one of three two values:
#   targeted - Targeted processes are protected,
#   minimum - Modification of targeted policy. Only selected processes are protected.
#   mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

2. 安装 KVM

2.1 安装 KVM 包

命令: yum -y install kvm

2.2 安装 KVM 核心包——虚拟操作系统模拟器加速模块

命令：yum -y install qemu-kvm qemu-kvm-tools

命令：yum -y install libvirt python-virtinst libvirt-python virt-manager libguestfs-tools
bridge-utils virt-install

说明：

libvirt：必须要装的核心工具

python-virtinst：包含 python 模块和工具（virt-install，virt-clone 和 virt-image）

virt-manager：虚拟机图形管理工具（宿主机有桌面环境时可以考虑安装，命令操作或者远程控制则不需要）

bridge-utils：虚拟机与外界通信的命令管理工具

virt-install：虚拟机安装工具

2.3 重启宿主机，加载 KVM 相关模块

命令：reboot

2.4 查看 KVM 模块是否被正确加载

命令：lsmod | grep kvm

出现以下信息则表示正确加载。

```
[root@localhost ~]# lsmod | grep kvm
kvm_intel          183621 0
kvm                586948 1 kvm_intel
irqbypass         13503 1 kvm
[root@localhost ~]#
```

2.5 开启 KVM 服务，并设置开机启动

命令：systemctl start libvirtd.service（开启）（如果报错请看 2.7）

命令：systemctl enable libvirtd.service（开机启动）

2.6 查看操作结果，出现 Active: active (running) 字样则说明运行情况良好

命令：systemctl status libvirtd（启动状态）

命令：systemctl is-enabled libvirtd（是否开机自动启动）

效果：

```
[root@localhost ~]# systemctl status libvirtd
● libvirtd.service - Virtualization daemon
  Loaded: loaded (/usr/lib/systemd/system/libvirtd.service; enabled; vendor preset: enabled)
  Active: active (running) since 2019-01-14 18:04:47 CST; 18min ago
    Docs: man:libvirtd(8)
          https://libvirt.org
  Main PID: 6050 (libvirtd)
    Tasks: 19 (limit: 32768)
  CGroup: /system.slice/libvirtd.service
          └─6050 /usr/sbin/libvirtd
             └─8500 /usr/sbin/dnsmasq --conf-file=/var/lib/libvirt/dnsmasq/default.conf --leasefile-ro --dhcp-script
                └─8501 /usr/sbin/dnsmasq --conf-file=/var/lib/libvirt/dnsmasq/default.conf --leasefile-ro --dhcp-script

1月 14 18:04:47 localhost.localdomain systemd[1]: Started Virtualization daemon.
1月 14 18:04:52 localhost.localdomain dnsmasq[8500]: started, version 2.76 cachesize 150
1月 14 18:04:52 localhost.localdomain dnsmasq[8500]: compile time options: IPV6 GNU-getopt DBus no-i18n IDN DHCP
1月 14 18:04:52 localhost.localdomain dnsmasq-dhcp[8500]: DHCP, IP range 192.168.122.2 -- 192.168.122.254, lease
1月 14 18:04:52 localhost.localdomain dnsmasq-dhcp[8500]: DHCP, sockets bound exclusively to interface virbr0
1月 14 18:04:52 localhost.localdomain dnsmasq[8500]: reading /etc/resolv.conf
1月 14 18:04:52 localhost.localdomain dnsmasq[8500]: using nameserver 180.76.76.76#53
1月 14 18:04:52 localhost.localdomain dnsmasq[8500]: read /etc/hosts - 2 addresses
1月 14 18:04:52 localhost.localdomain dnsmasq[8500]: read /var/lib/libvirt/dnsmasq/default.addnhosts - 0 address
1月 14 18:04:52 localhost.localdomain dnsmasq-dhcp[8500]: read /var/lib/libvirt/dnsmasq/default.hostsfile
Hint: Some lines were ellipsized, use -l to show in full.
```

```
[root@localhost ~]# systemctl is-enabled libvirtd
enabled
[root@localhost ~]#
```

2.7 如果执行“开启 KVM 服务”报错请更新系统

```
[root@localhost ~]# systemctl start libvirtd.service
Job for libvirtd.service failed because the control process exited with error code. See "systemctl status libvirtd.service" and "journalctl -xe" for details.
[root@localhost ~]#
```

命令: yum -y update

3. 需要启用 PCI Pass-through, 在打开的文件中添加

intel_iommu=on 参数

命令: vi /etc/default/grub

效果:

```
GRUB_TIMEOUT=5
GRUB_DISTRIBUTOR="$(sed 's, release .*$,g' /etc/system-release)"
GRUB_DEFAULT=saved
GRUB_DISABLE_SUBMENU=true
GRUB_TERMINAL_OUTPUT="console"
GRUB_CMDLINE_LINUX="crashkernel=auto rd.lvm.lv=centos/root rd.lvm.lv=centos/swap rhgb quiet intel_iommu=on"
GRUB_DISABLE_RECOVERY="true"
```

3.1 更新 GRUB 后重启使之生效

命令: grub2-mkconfig > /boot/grub2/grub.cfg

重启 reboot

4. 对 PCI 网卡分离

4.1 查看宿主机上的 PCI 设备并找到要操作的网卡

命令：lspci

效果：

```
[root@localhost ~]# lspci
00:00.0 Host bridge: Intel Corporation 8th Gen Core Processor Host Bridge/DRAM Registers (rev 07)
00:01.0 PCI bridge: Intel Corporation Xeon E3-1200 v5/E3-1500 v5/6th Gen Core Processor PCIe Controller (x16) (rev 07)
00:02.0 VGA compatible controller: Intel Corporation UHD Graphics 630 (Desktop)
00:08.0 System peripheral: Intel Corporation Xeon E3-1200 v5/v6 / E3-1500 v5 / 6th/7th Gen Core Processor Gaussian Mixture Model
00:14.0 USB controller: Intel Corporation 200 Series/Z370 Chipset Family USB 3.0 xHCI controller
00:16.0 Communication controller: Intel Corporation 200 Series PCH CSME HECI #1
00:17.0 SATA controller: Intel Corporation 200 Series PCH SATA controller [AHCI mode]
00:1b.0 PCI bridge: Intel Corporation 200 Series PCH PCI Express Root Port #17 (rev f0)
00:1b.2 PCI bridge: Intel Corporation 200 Series PCH PCI Express Root Port #19 (rev f0)
00:1b.4 PCI bridge: Intel Corporation 200 Series PCH PCI Express Root Port #21 (rev f0)
00:1c.0 PCI bridge: Intel Corporation 200 Series PCH PCI Express Root Port #1 (rev f0)
00:1c.1 PCI bridge: Intel Corporation 200 Series PCH PCI Express Root Port #2 (rev f0)
00:1c.2 PCI bridge: Intel Corporation 200 Series PCH PCI Express Root Port #3 (rev f0)
00:1c.4 PCI bridge: Intel Corporation 200 Series PCH PCI Express Root Port #5 (rev f0)
00:1c.5 PCI bridge: Intel Corporation 200 Series PCH PCI Express Root Port #6 (rev f0)
00:1c.6 PCI bridge: Intel Corporation 200 Series PCH PCI Express Root Port #7 (rev f0)
00:1c.7 PCI bridge: Intel Corporation 200 Series PCH PCI Express Root Port #8 (rev f0)
00:1d.0 PCI bridge: Intel Corporation 200 Series PCH PCI Express Root Port #9 (rev f0)
00:1f.0 ISA bridge: Intel Corporation Z370 Chipset LPC/eSPI controller
00:1f.2 Memory controller: Intel Corporation 200 Series/Z370 Chipset Family Power Management Controller
00:1f.3 Audio device: Intel Corporation 200 Series PCH HD Audio
00:1f.4 SMBus: Intel Corporation 200 Series/Z370 Chipset Family SMBus Controller
00:1f.6 Ethernet controller: Intel Corporation Ethernet Connection (2) I219-V
01:00.0 Ethernet controller: Intel Corporation 82599ES 10-Gigabit SFI/SFP+ Network Connection (rev 01)
01:00.1 Ethernet controller: Intel Corporation 82599ES 10-Gigabit SFI/SFP+ Network Connection (rev 01)
03:00.0 USB controller: ASMedia Technology Inc. ASM2142 USB 3.1 Host Controller
```

```
01:00.0 Ethernet controller: Intel Corporation 82599ES 10-Gigabit SFI/SFP+ Network Connection (rev 01)
01:00.1 Ethernet controller: Intel Corporation 82599ES 10-Gigabit SFI/SFP+ Network Connection (rev 01)
```

注意：上图 01:00.0 中 01 是 PCI bus number；00 是 PCI device number；0 是 Function number

4.2 找到该设备的 PCI 编号

命令：virsh nodedev-list

效果：

```
[root@localhost ~]# virsh nodedev-list
block_sda_wdc_wd10EZEX_08WN4A0_WD_WCC6Y6TCZVZL
computer
drm_card0
drm_renderD128
net_enp0s31f6_e0_d5_5e_b2_53_e1
net_lo_00_00_00_00_00_00
net_virbr0_nic_52_54_00_1e_1a_09
net_vnet0_fe_54_00_c1_cf_0d
pci_0000_00_00_0
pci_0000_00_01_0
pci_0000_00_02_0
pci_0000_00_08_0
pci_0000_00_14_0
pci_0000_00_16_0
pci_0000_00_17_0
pci_0000_00_1b_0
pci_0000_00_1b_2
pci_0000_00_1b_4
pci_0000_00_1c_0
pci_0000_00_1c_1
pci_0000_00_1c_2
pci_0000_00_1c_4
pci_0000_00_1c_5
pci_0000_00_1c_6
pci_0000_00_1c_7
pci_0000_00_1d_0
pci_0000_00_1f_0
pci_0000_00_1f_2
pci_0000_00_1f_3
pci_0000_00_1f_4
pci_0000_00_1f_6
pci_0000_01_00_0
pci_0000_01_00_1
pci_0000_03_00_0
scsi_0_0_0_0
scsi_generic_sg0
scsi_host0
scsi_host1
```

注意：在 virsh 中 “.” 和 “_” 变为 “_” 编号都是一样的支持显示格式不一样

4.3 分离这两个 PCI 网卡

命令: `virsh nodedev-detach pci_0000_01_00_0` `virsh nodedev-detach pci_0000_01_00_1`

效果:

```
[root@localhost ~]# virsh nodedev-detach pci_0000_01_00_0
已分离设备 pci_0000_01_00_0

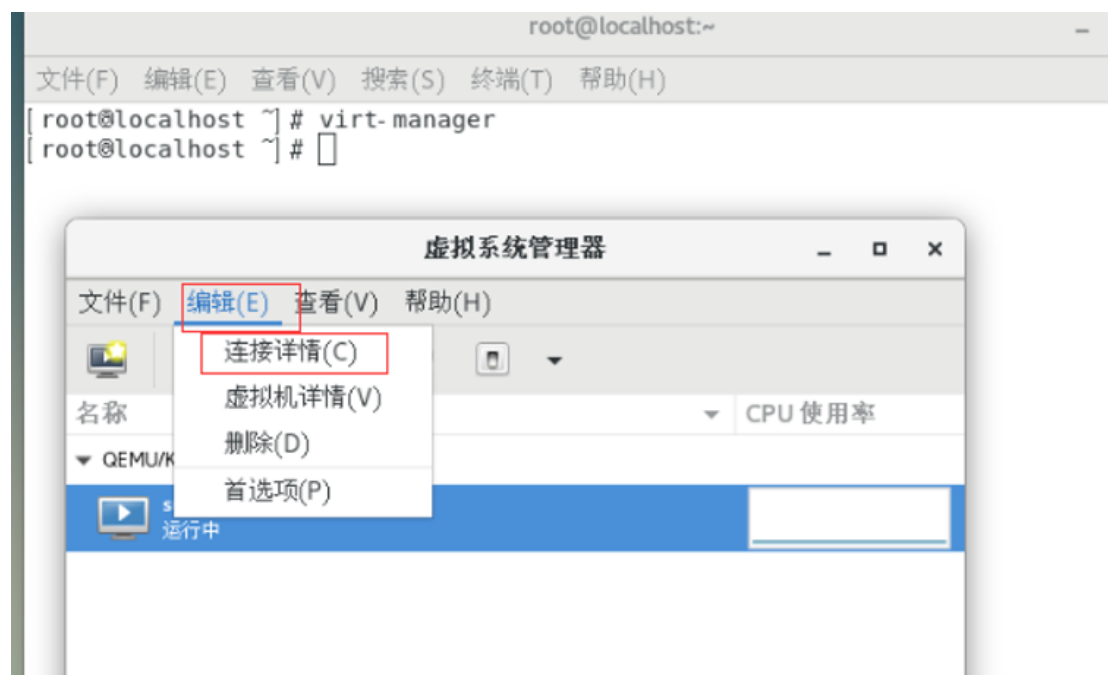
[root@localhost ~]# virsh nodedev-detach pci_0000_01_00_1
已分离设备 pci_0000_01_00_1
```

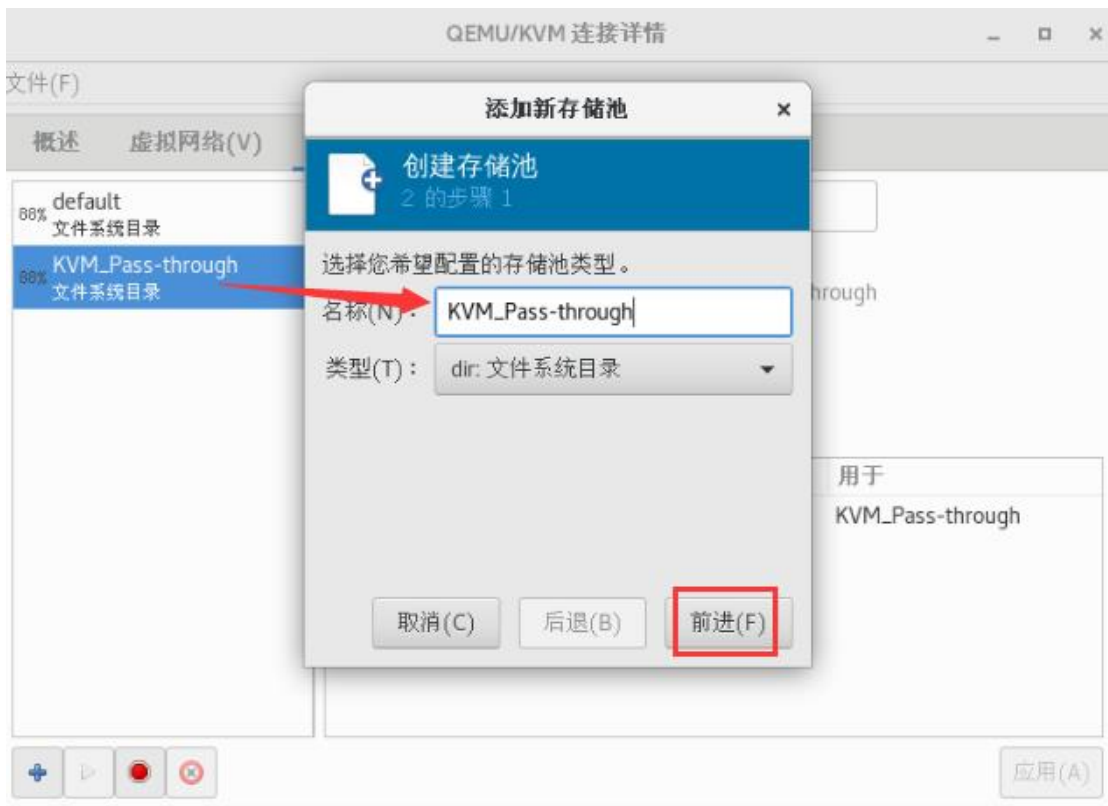
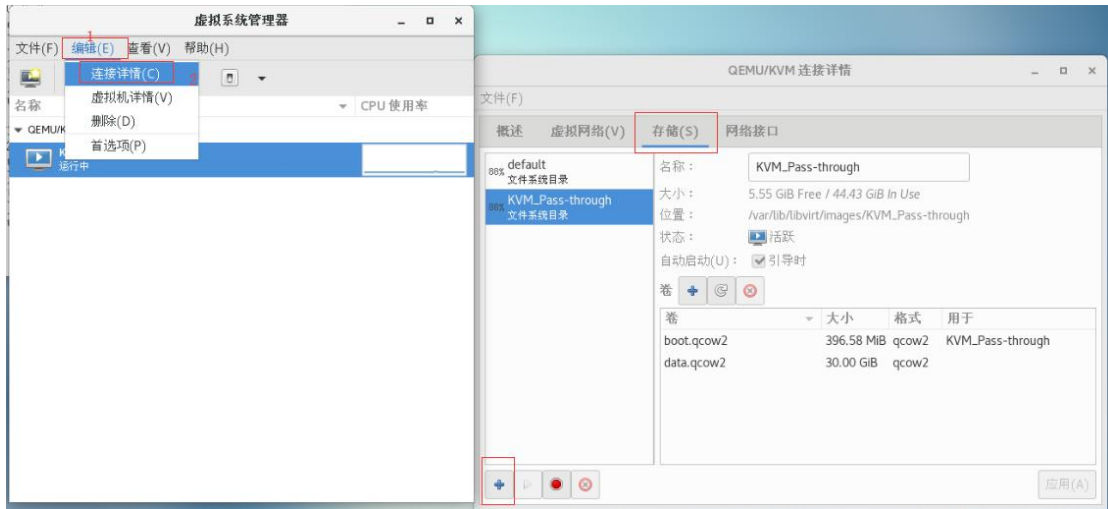
重启: `reboot`

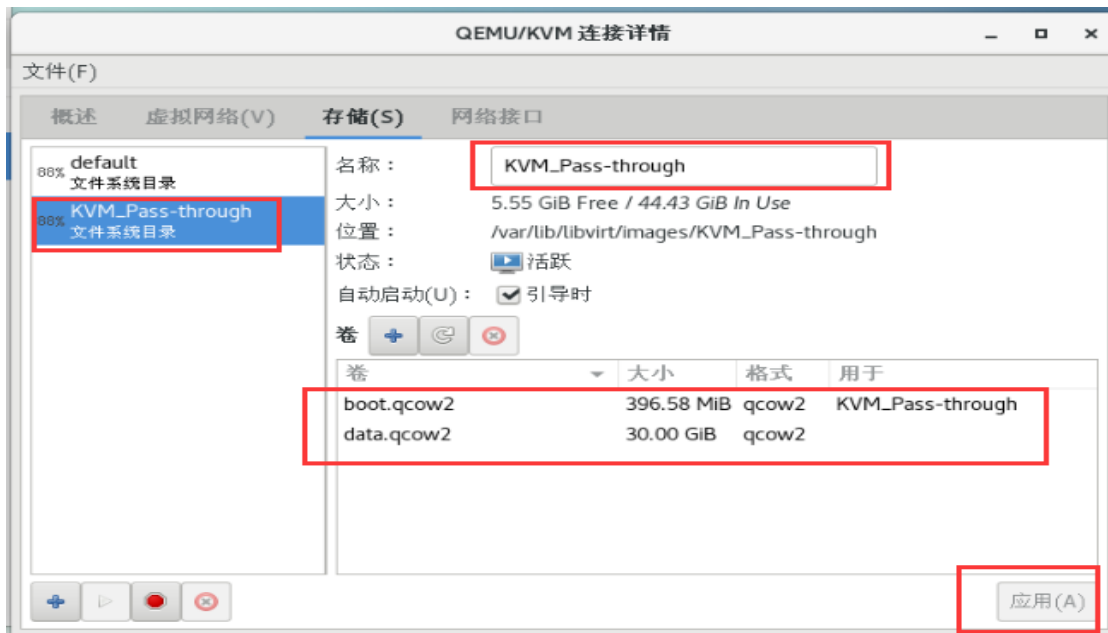
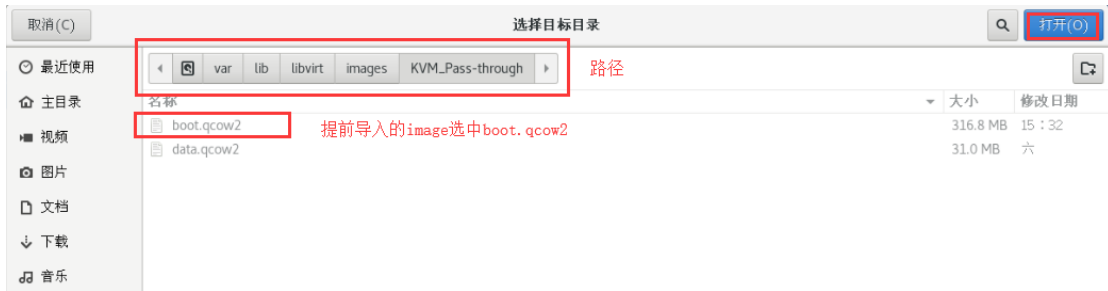
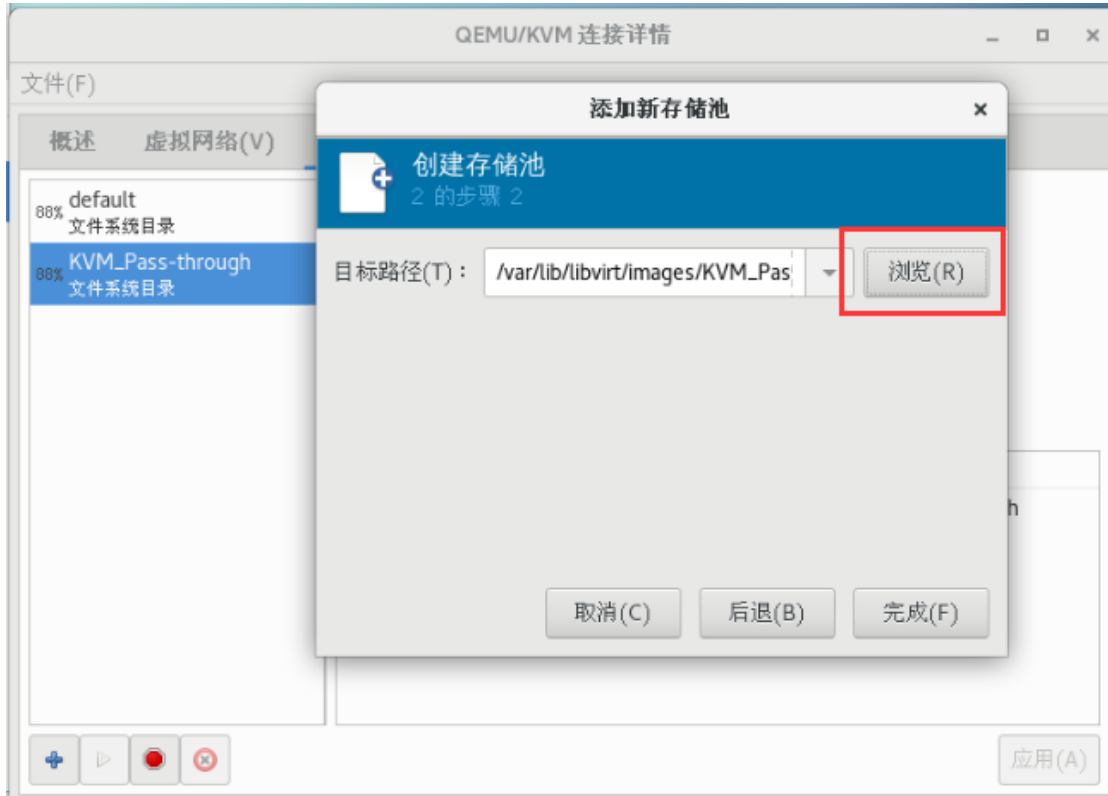
注: `virsh nodedev-detach pci_0000_网卡` 分离网卡
`virsh nodedev-reattach pci_0000_网卡` 取消分离

5. virt-manager 图形界面部署虚拟机

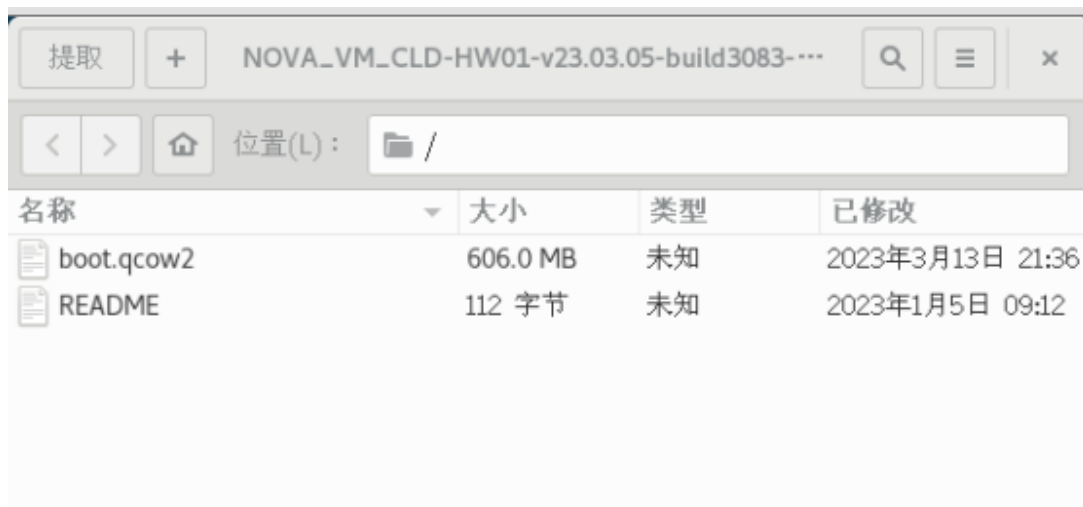
5.1 添加存储池







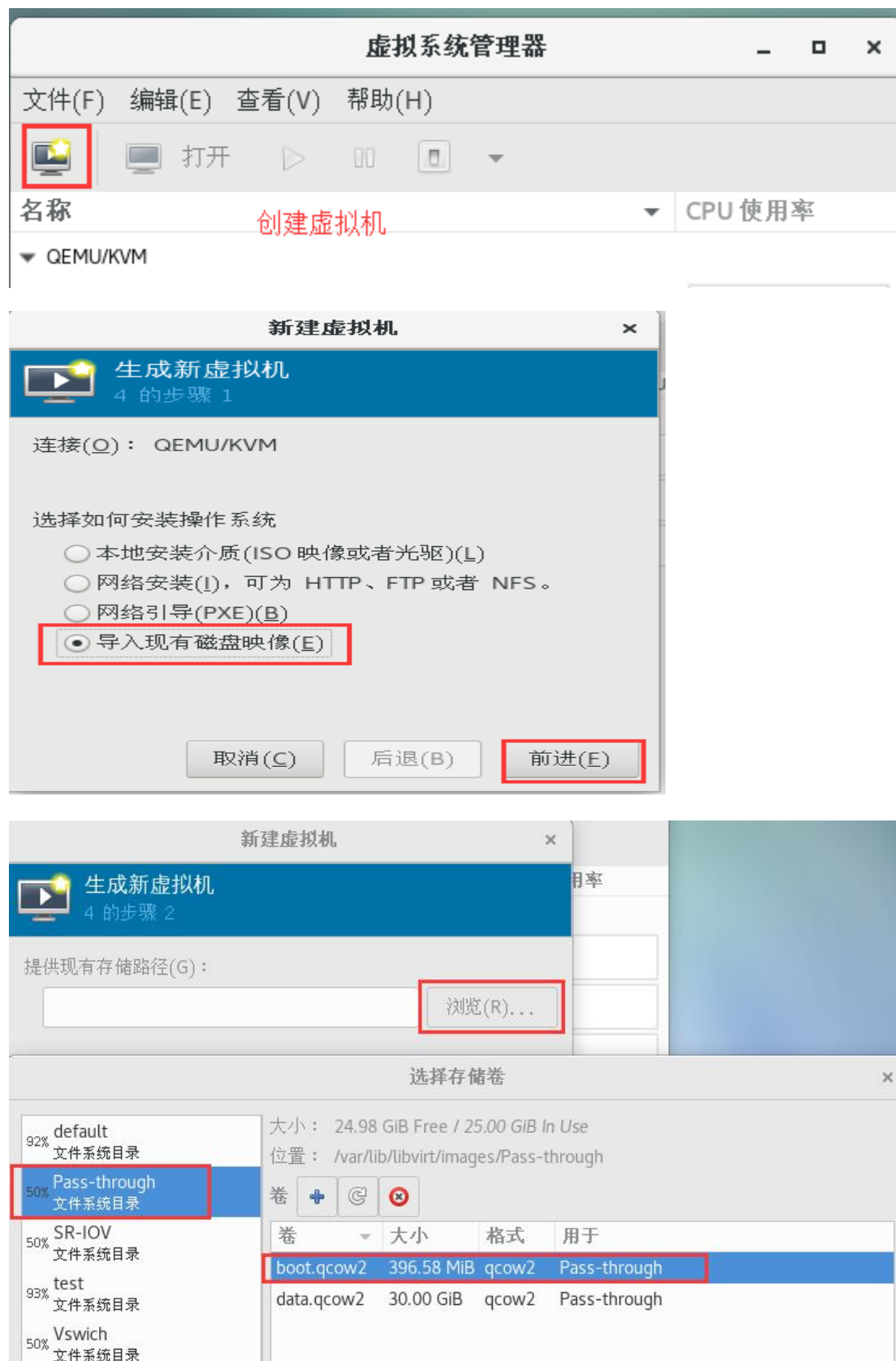
注意：从 23.03 版本开始需要手动创建数据盘
用 23.03 以后版本解压出的文件为 README 和系统盘 boot.qcow2

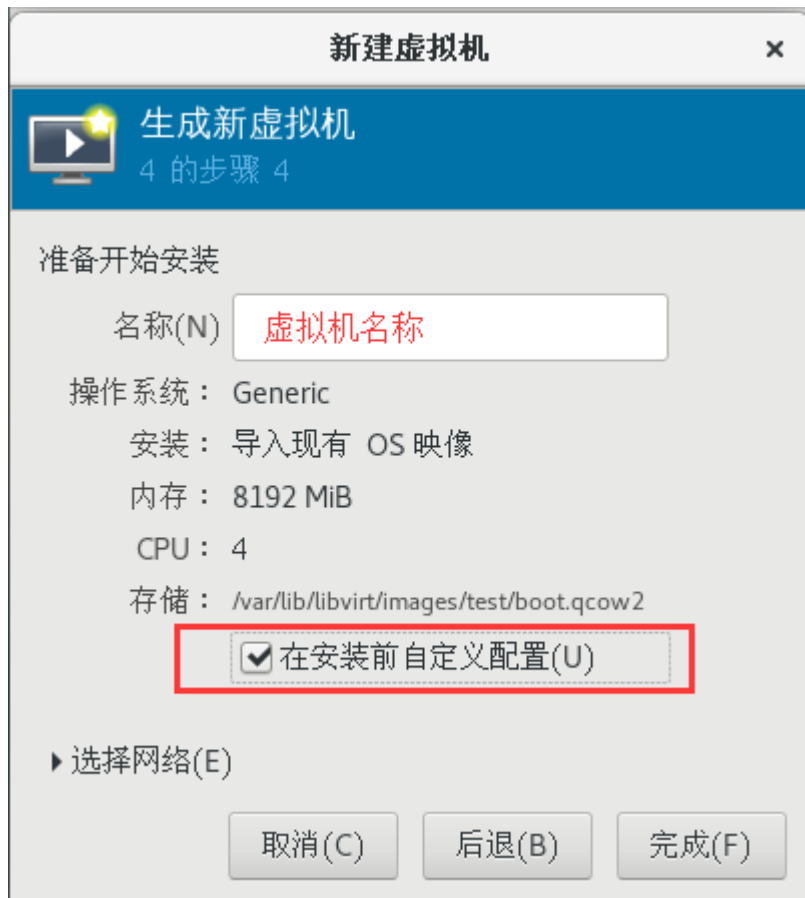
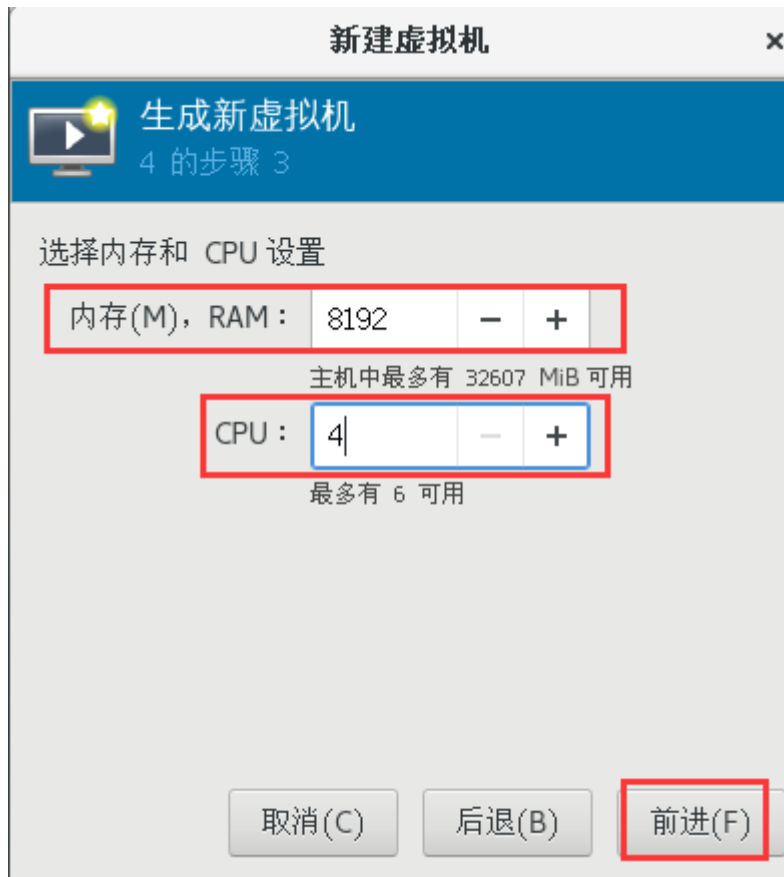


打开终端执行创建数据盘命令



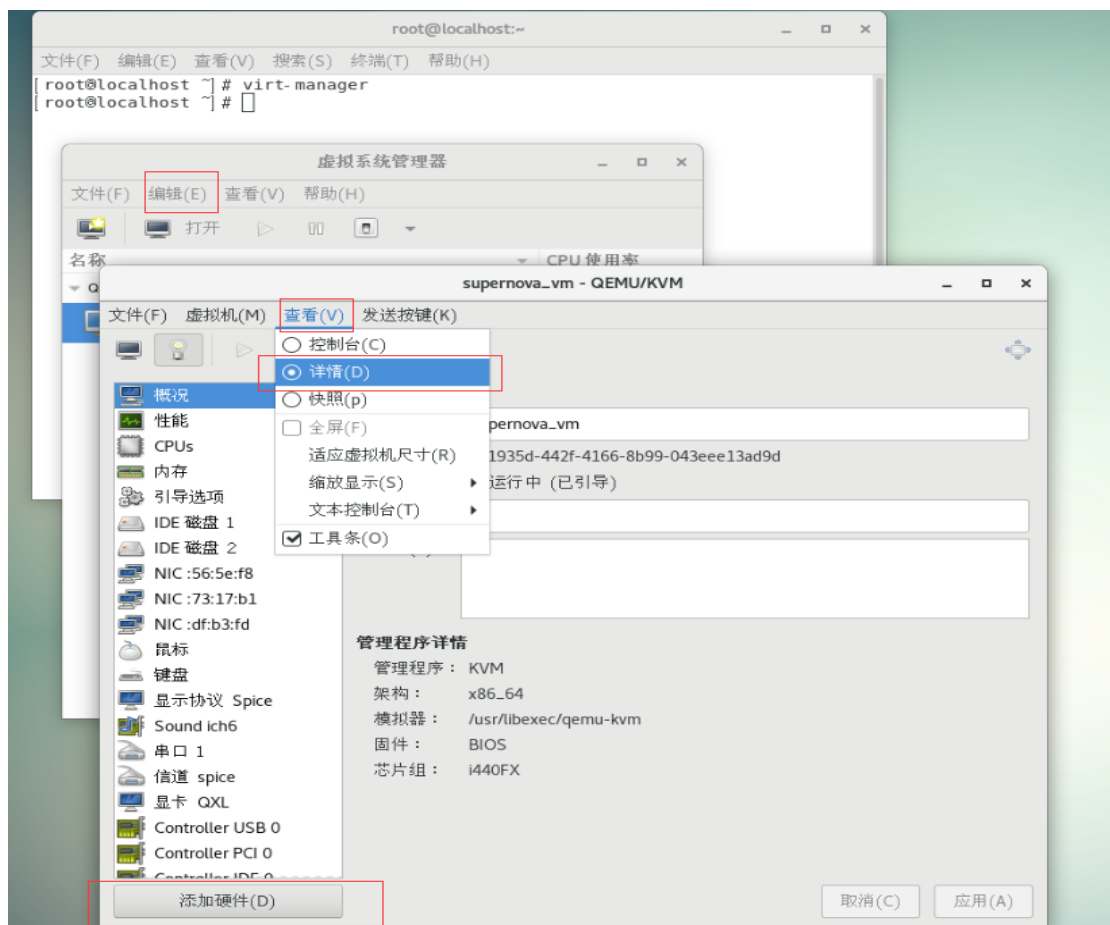
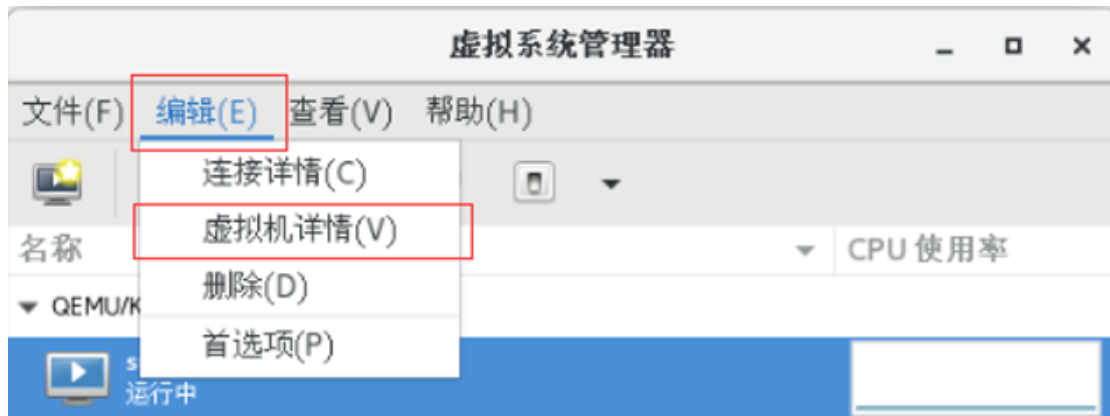
5.2 创建虚拟机

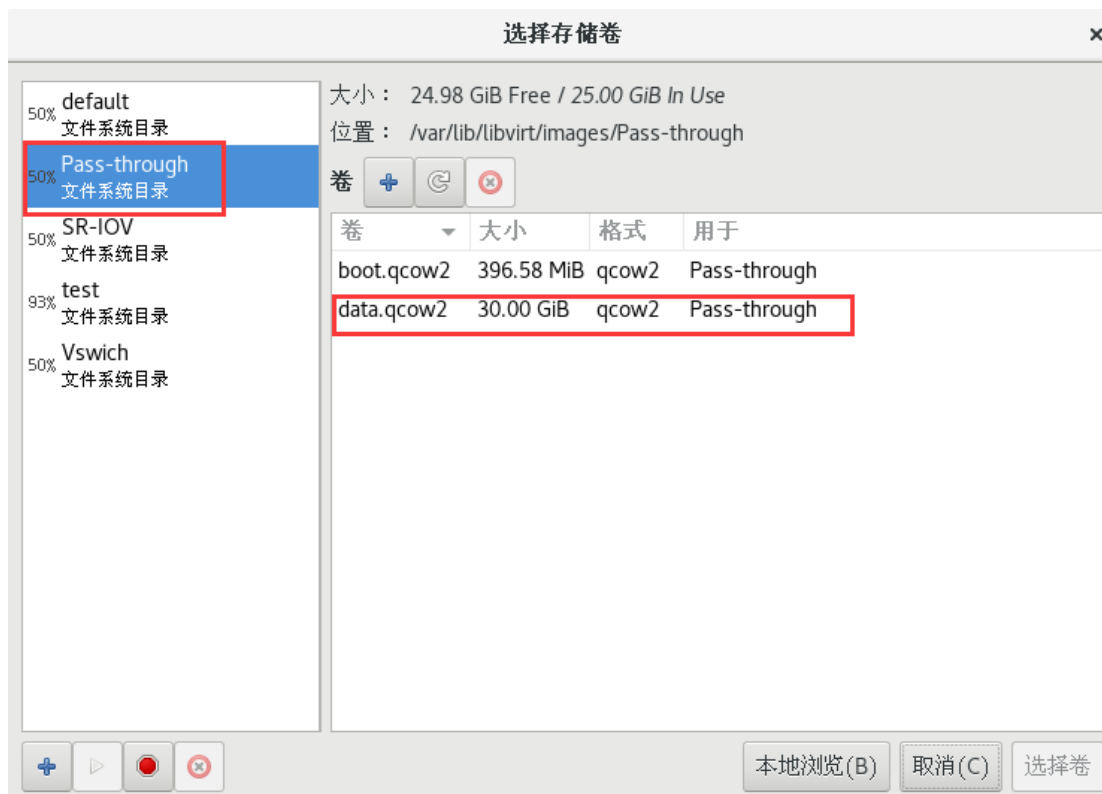
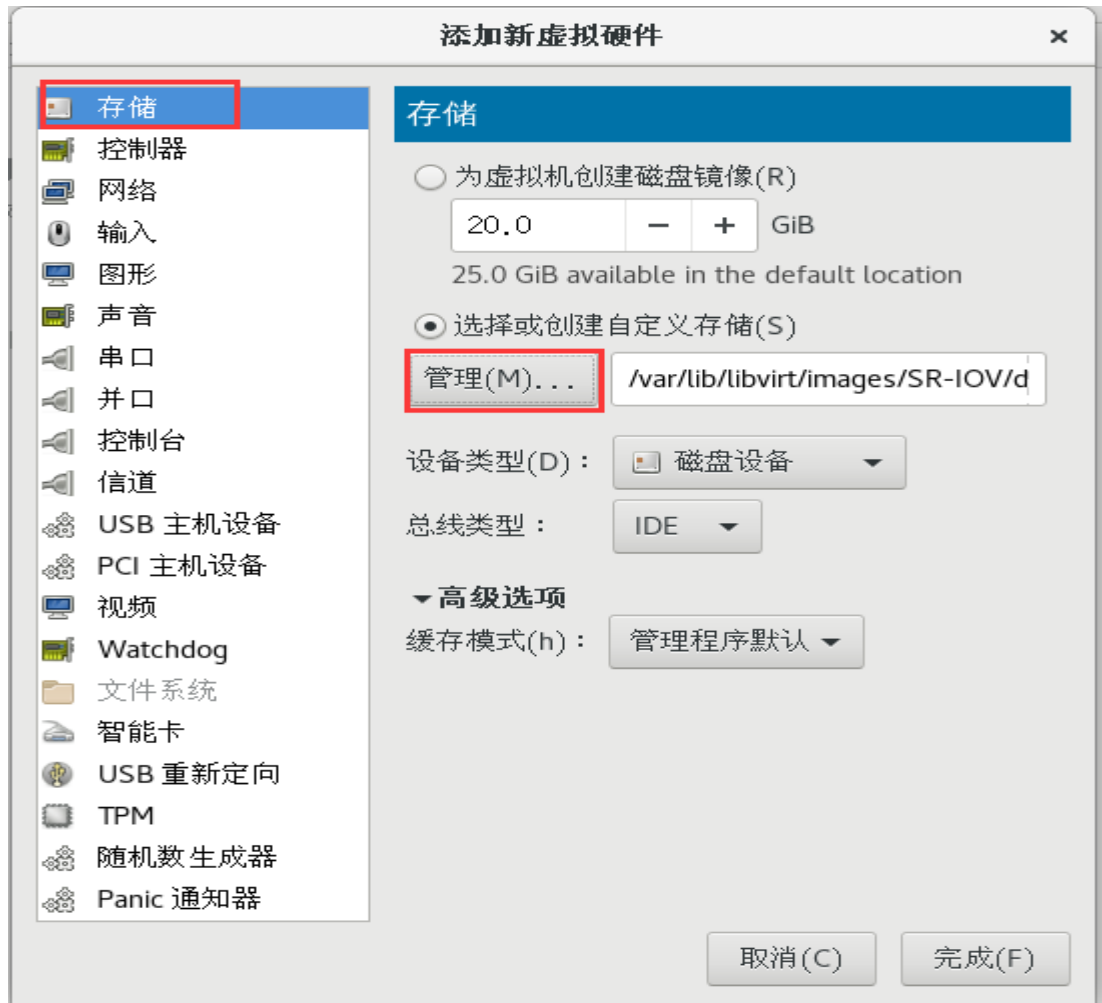




在创建虚拟机的过程中第一块 Boot 盘就有了

5.3 再增加一块 Data 盘（Supernova 默认要用到两块硬盘）





虚拟磁盘

源路径： /var/lib/libvirt/images/SR-IOV/boot.qcow2
设备类型： IDE 磁盘 1
存储大小： 396.58 MiB
只读(E)：
可共享(B)：

高级选项(o)

磁盘总线(u)： IDE
序列号(L)：

存储格式(t)： qcow2

性能选项(P)

缓存模式(h)： writethrough
IO 模式： 管理程序默认

添加硬件(D)

虚拟磁盘

源路径： /var/lib/libvirt/images/SR-IOV/data.qcow2
设备类型： IDE 磁盘 2
存储大小： 30.00 GiB
只读(E)：
可共享(B)：

高级选项(o)

磁盘总线(u)： IDE
序列号(L)：

存储格式(t)： qcow2

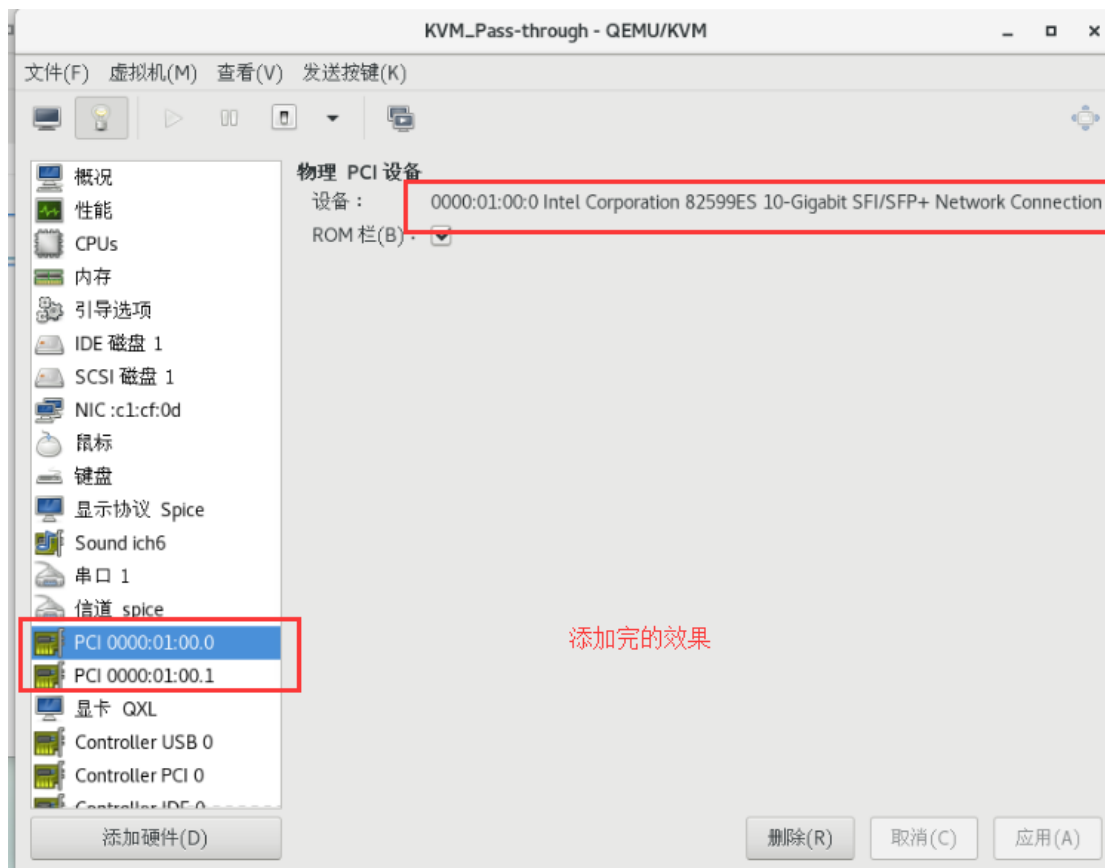
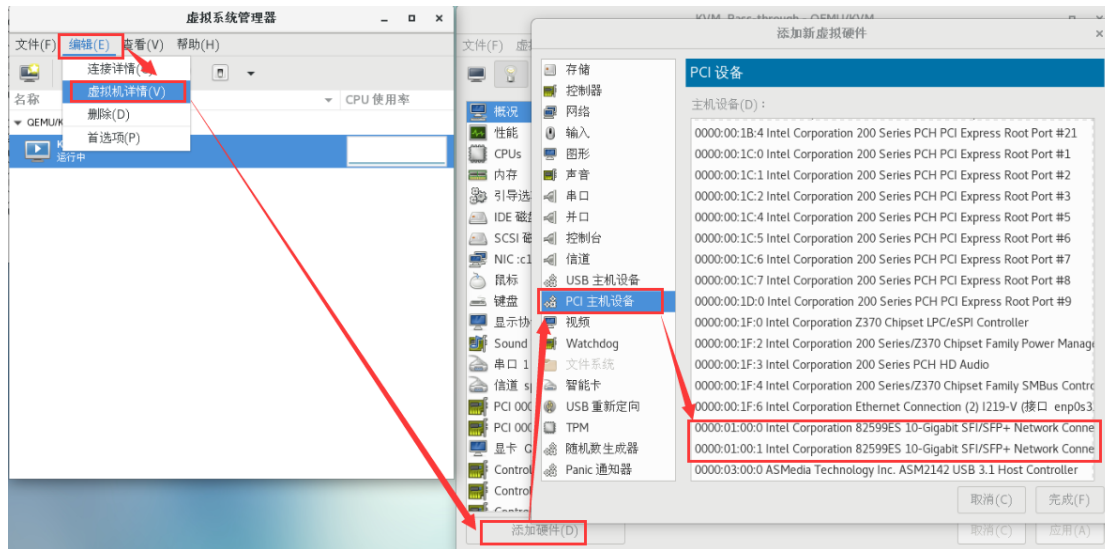
性能选项(P)

缓存模式(h)： writethrough
IO 模式： 管理程序默认

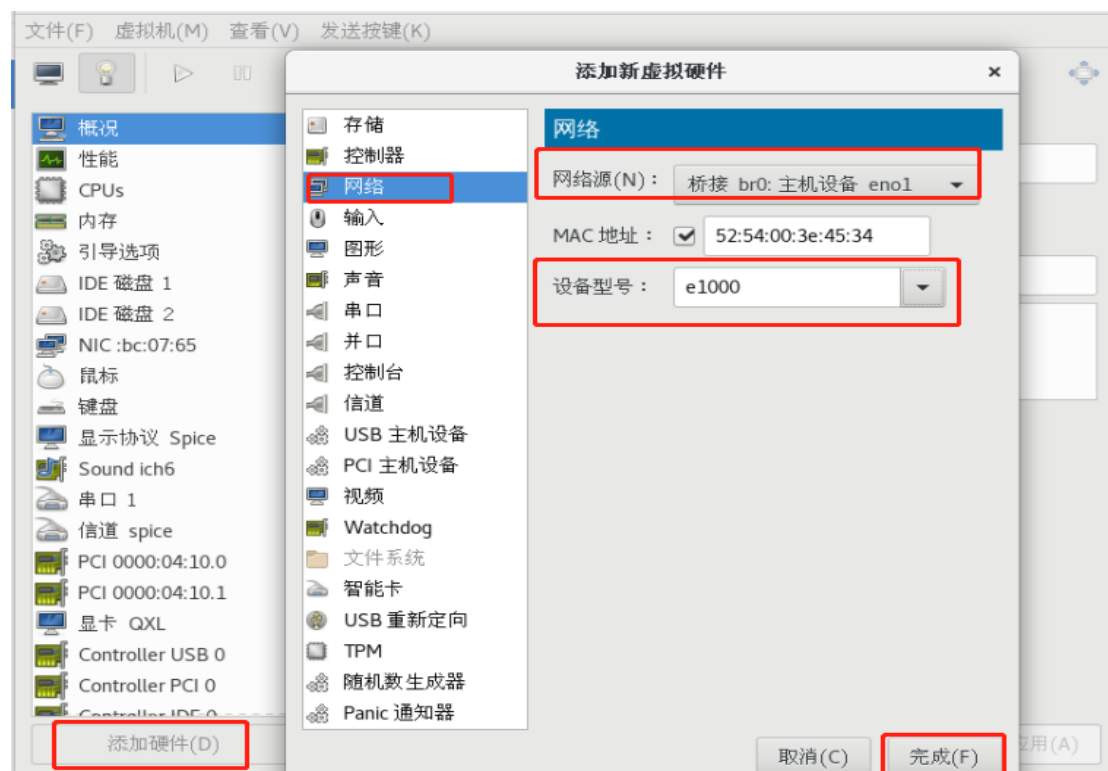
添加硬件(D)

5.4 把从宿主主机上分离出来的 PCI 网卡添加到虚拟机上来

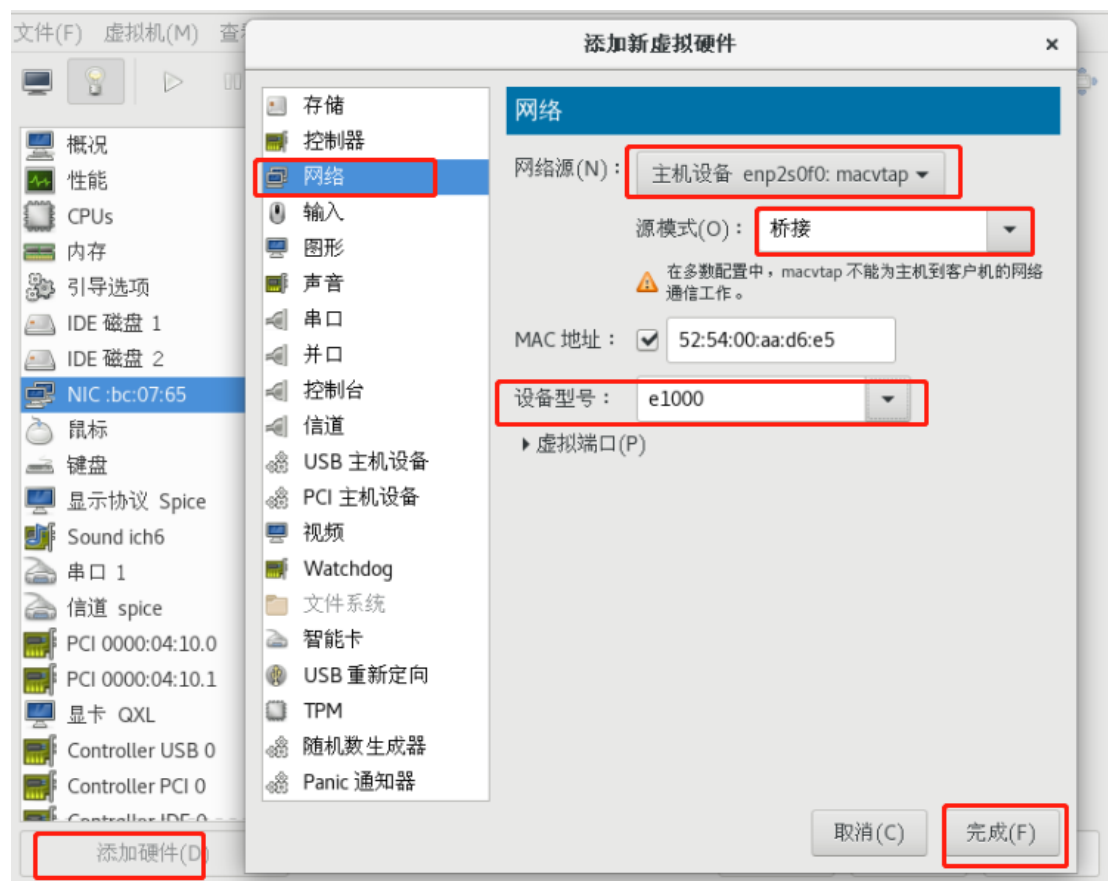
把从宿主主机上分离出来的 PCI 网卡添加到虚拟机上，作为测试口 port1 和测试口 port2



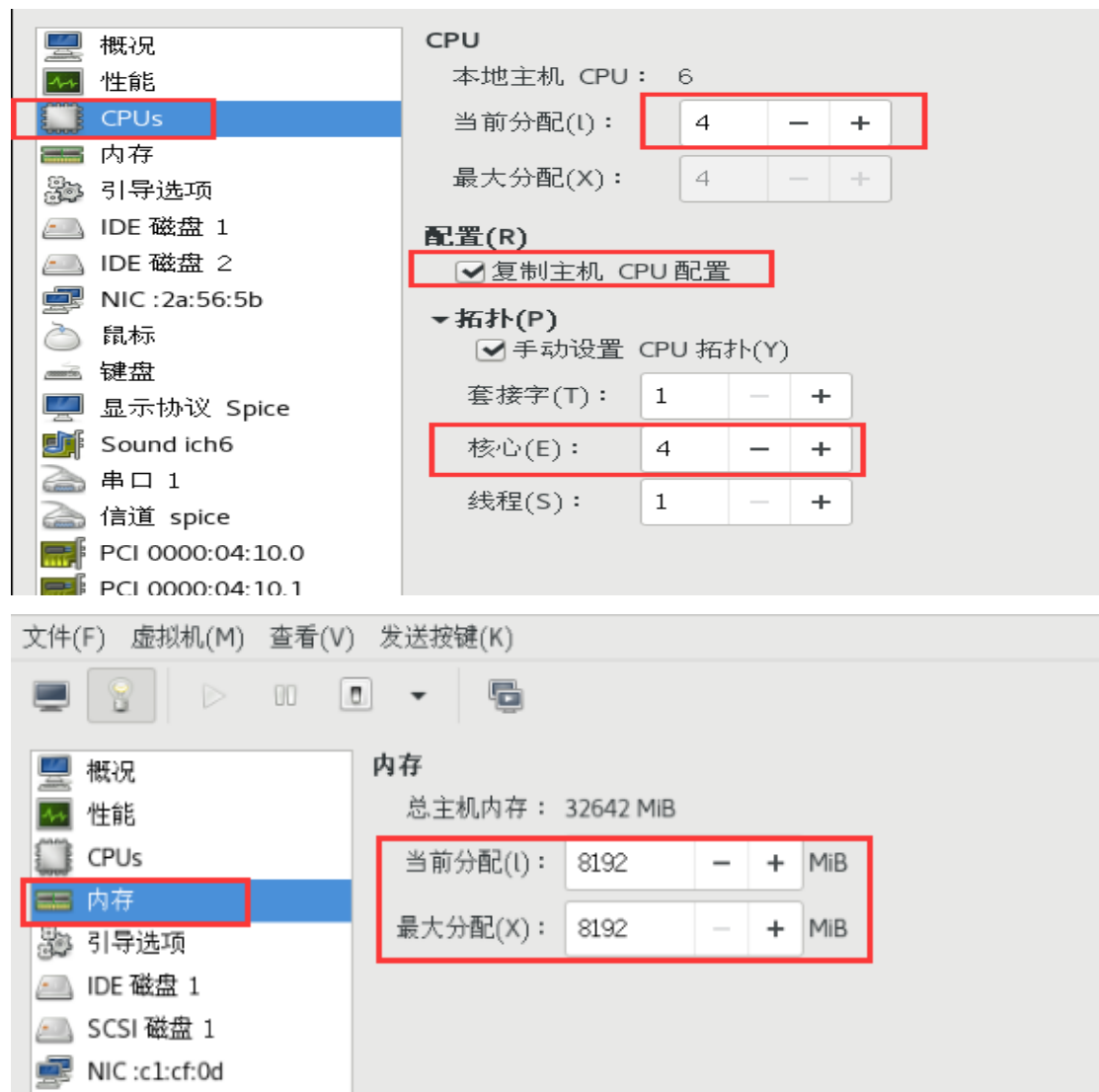
添加管理口 mgmt1 网卡, 点击添加硬件-网络, 网络源选择 br0, 设备型号选择 e1000, 作为 mgmt1 管理口, 设置后点击完成以完成网卡添加。



添加管理口 mgmt2 网卡, 点击添加硬件-网络, 网络源选择主机其他可用网卡, 如 enp2s0f0, 设备型号选择 e1000, 作为 mgmt2 管理口, 设置后点击完成以完成网卡添加。

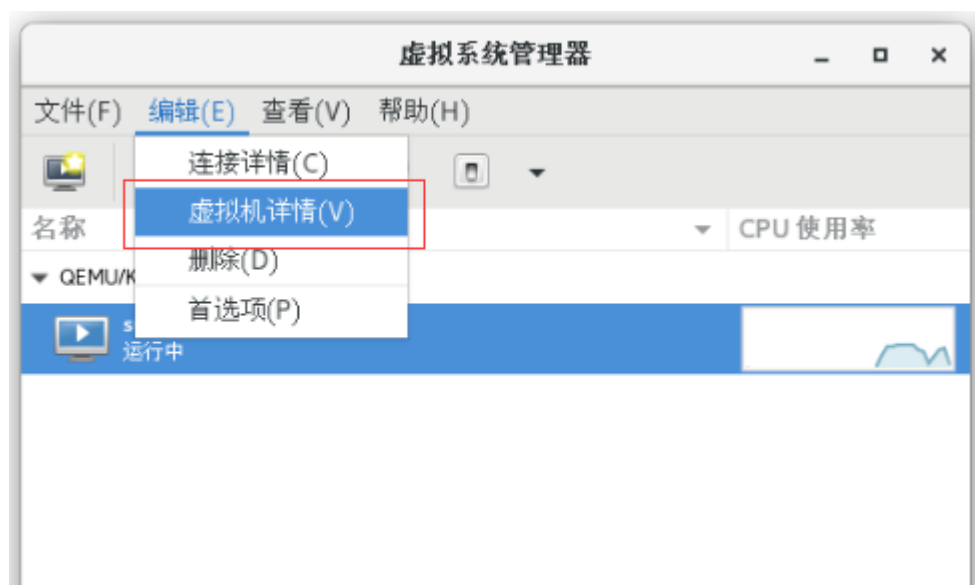
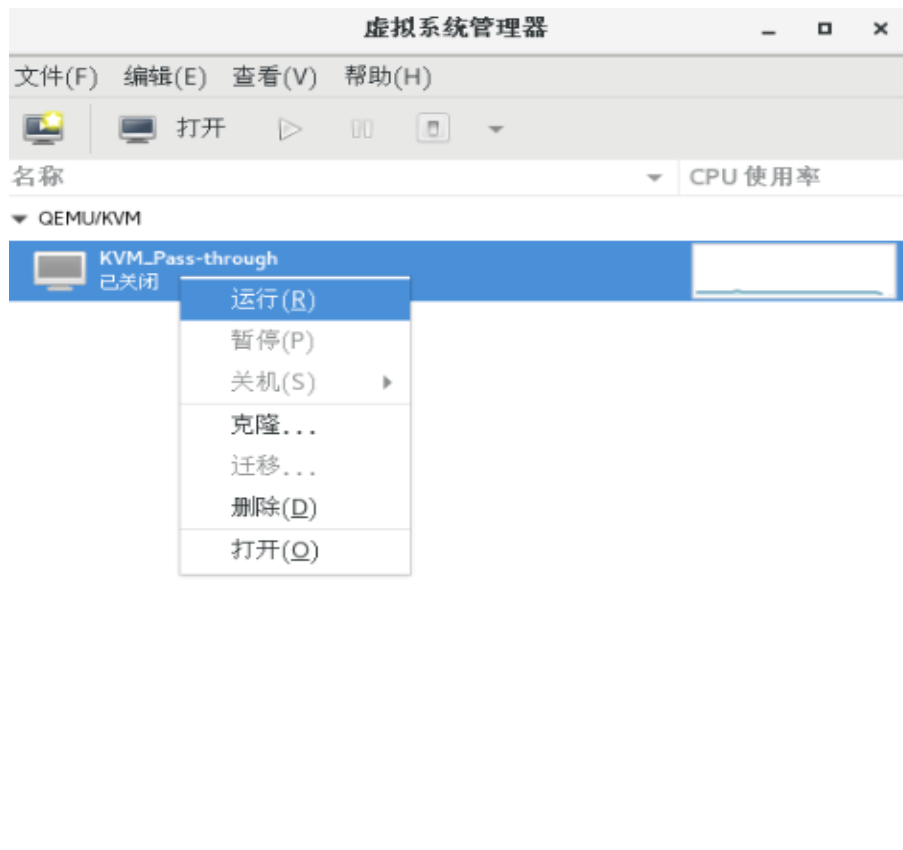


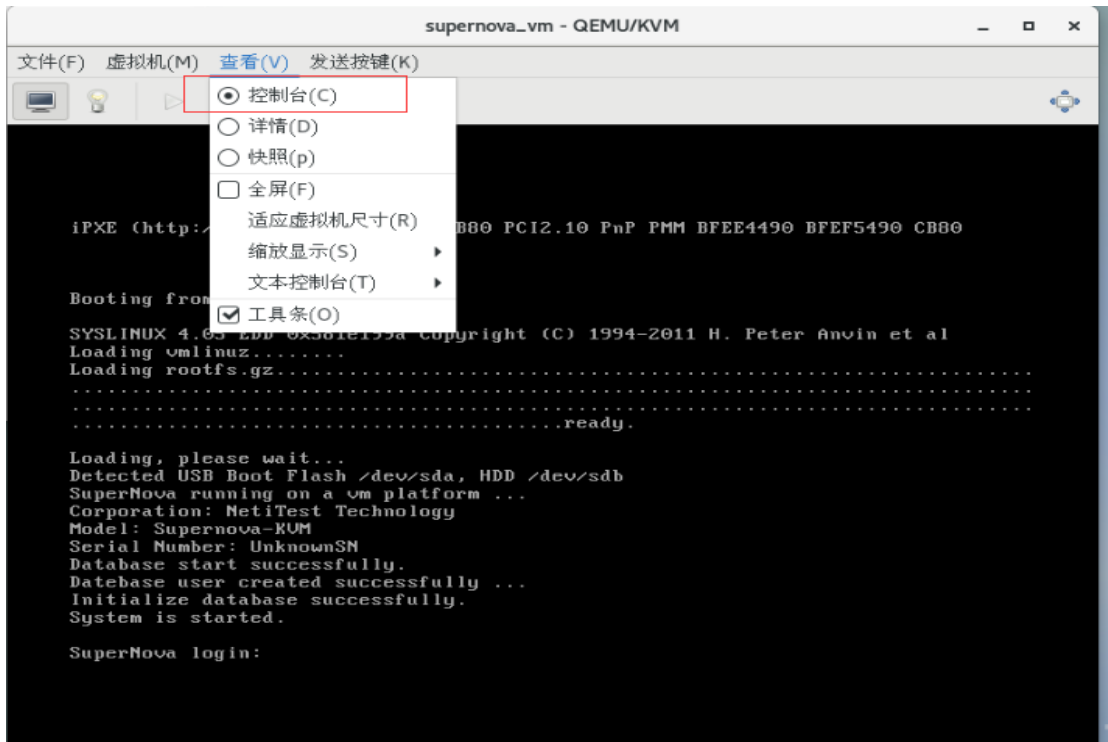
6.5 调整 CPU 和内存大小



6. 启动虚拟机 Supernova

6.1 运行





注：如果在启动过程中报 port1/port2 错误，进入到页面后发现刚报出错误的 port 没有显示出来那么请在 cd /etc/sysconfig/network-scripts 中加入相对应的接口。
先查看我 4 个 10G 口的网卡名称：

```

[root@localhost network-scripts]# ifconfig -a
br0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.16.95 netmask 255.255.255.0 broadcast 192.168.16.255
    inet6 fe80::8ad7:f6ff:fec4:216f prefixlen 64 scopeid 0x20<link>
    ether 88:d7:f6:c4:21:6f txqueuelen 1000 (Ethernet)
    RX packets 2707 bytes 189894 (185.4 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 532 bytes 91021 (88.8 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

enp0s31f6: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet6 fe80::8ad7:f6ff:fec4:216f prefixlen 64 scopeid 0x20<link>
    ether 88:d7:f6:c4:21:6f txqueuelen 1000 (Ethernet)
    RX packets 3452 bytes 462497 (451.6 KiB)
    RX errors 0 dropped 10 overruns 0 frame 0
    TX packets 547 bytes 95489 (93.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    1 device interrupt 16 memory 0xdf400000-df420000

enp1s0f0: flags=4098<BROADCAST,MULTICAST> mtu 1500
    ether 68:91:d0:61:be:cc txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    2

enp1s0f1: flags=4098<BROADCAST,MULTICAST> mtu 1500
    ether 68:91:d0:61:be:cd txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    3

enp4s0f0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet6 fe80::216:31ff:fef2:4942 prefixlen 64 scopeid 0x20<link>
    ether 00:16:31:f2:49:42 txqueuelen 1000 (Ethernet)
    RX packets 1 bytes 78 (78.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    4

enp4s0f1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet6 fe80::216:31ff:fef2:4943 prefixlen 64 scopeid 0x20<link>
    ether 00:16:31:f2:49:43 txqueuelen 1000 (Ethernet)
    RX packets 1 bytes 78 (78.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
  
```

在 cd /etc/sysconfig/network-scripts 中要一一对应；如果没有对应需要手动添加：

```
[root@localhost ~]# cd /etc/sysconfig/network-scripts/
[root@localhost network-scripts]# ls
ifcfg-br0          ifcfg-enp4s0f1  ifdown-ib         ifdown-ppp      ifdown-tunnel    ifup-ib         ifup-plusb      ifup-Team      network-functions
ifcfg-enp0s31f6   ifcfg-lo        ifdown-ppp       ifdown-routes  ifup              ifup-ippv6     ifup-post       ifup-TeamPort  network-functions-ipv6
ifcfg-enp1s0f0    ifdown         ifdown-ipv6     ifdown-sit     ifup-aliases     ifup-ipv6     ifup-ppp        ifup-tunnel
ifcfg-enp1s0f1    ifdown-bnep    ifdown-isdn     ifdown-Team    ifup-bnep        ifup-isdn     ifup-routes     ifup-wireless
ifcfg-enp4s0f0    ifdown-eth     ifdown-post     ifdown-TeamPort ifup-eth         ifup-plip     ifup-sit        init.ipv6-global
[root@localhost network-scripts]#
```

这是我的四块 10G 网卡，我用到的是 ifcfg-enp4s0f0 和 ifcfg-enp4s0f1 如果这个表中没有这两个是需要添加的添加的内容为：

```
vi ifcfg-enp4s0f0

BOOTPROTO=none
IPV6INIT=yes
IPV6_AUTOCONF=yes
DEVICE=enp4s0f0
ONBOOT=yes
```

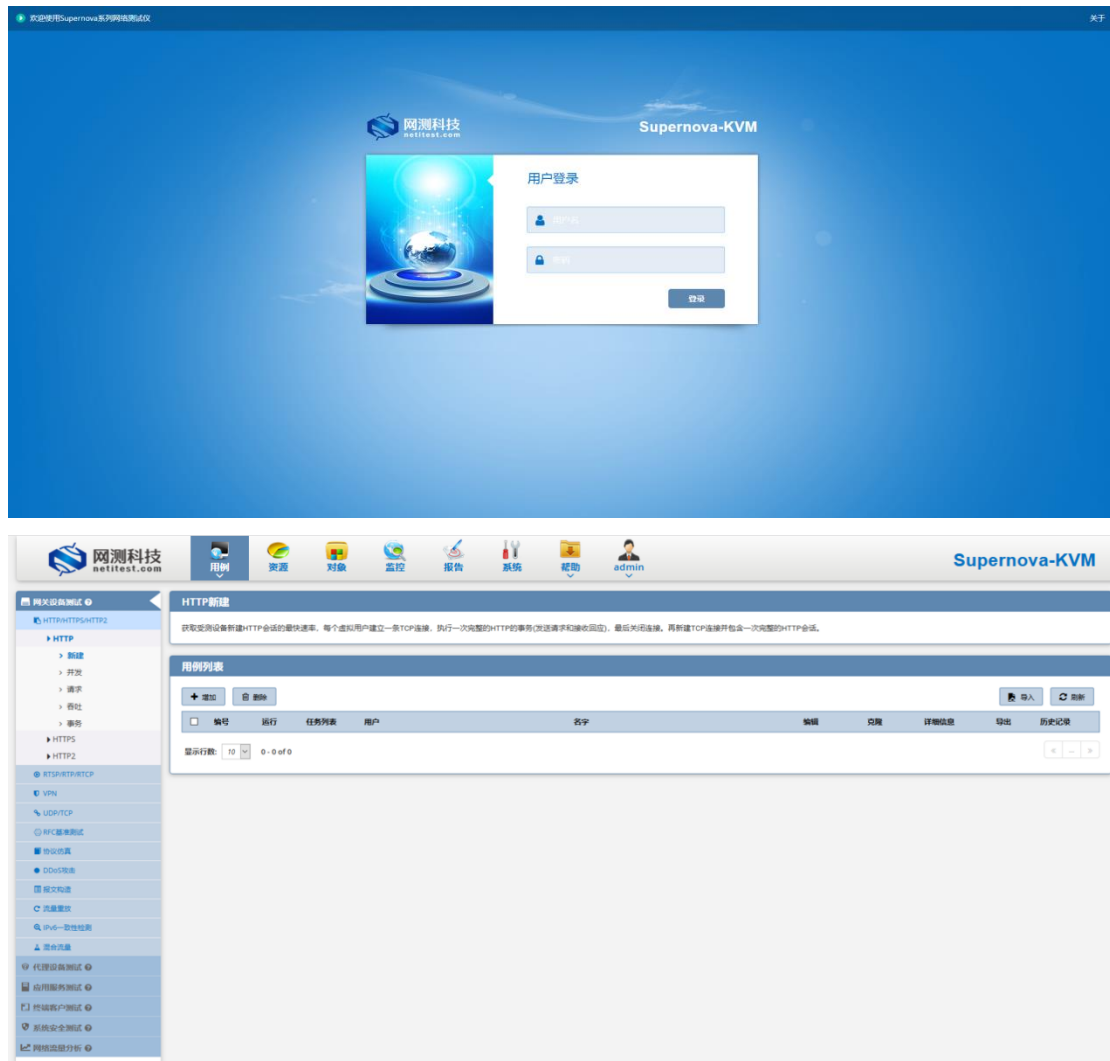
6.2 成功运行后测试仪 dhcp 自动获取 IP 地址

```
Entering listen mode: raw
Opening raw socket on ifindex 5
Got raw socket fd
Attached filter to raw socket fd
Created raw socket
Adapter index 5
MAC 52:54:00:ed:55:3b
Sending discover...
Waiting on select 3 seconds
Received a packet
Adapter index 5
MAC 52:54:00:ed:55:3b
Sending select for 192.168.18.45...
Waiting on select 3 seconds
Received a packet
Lease of 192.168.18.45 obtained, lease time 604800
Executing /usr/share/udhcpd/default.script bound
deleting routers
Entering listen mode: none
Version: 22.12.11 build2907
Build date: 20230225
kernel version: 5.10.112
System is started.

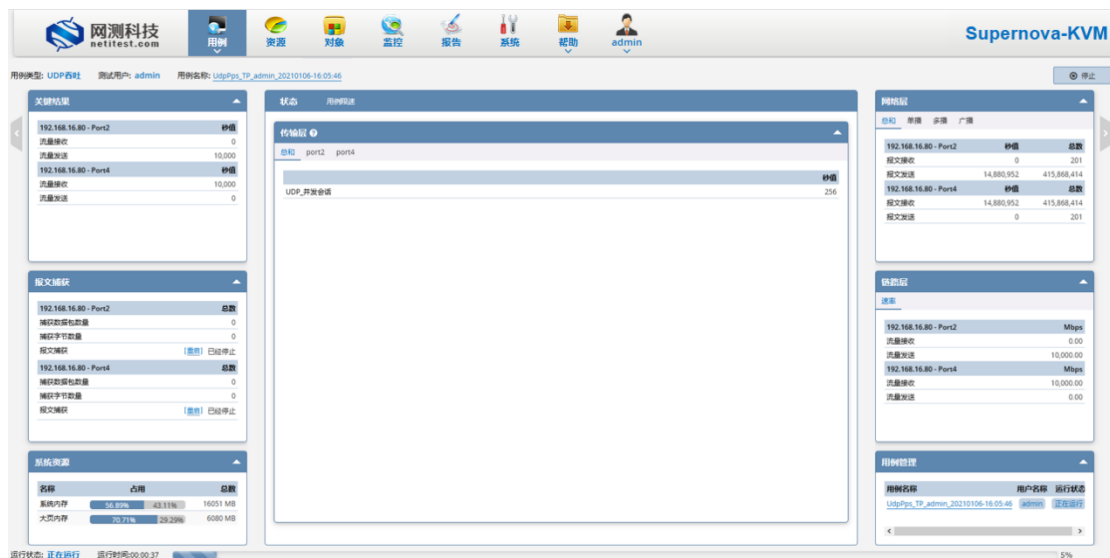
Supernova login:
```


6.3 登陆 Supernova 测试能否访问和运行用例

通过设置的管理 IP 访问 web 管理页面，初始登录账号密码：admin/admin



测试创建运行用例



7 使用命令行方式部署虚拟机

7.1.创建 pass-through 文件夹

```
[root@localhost home]# cd /home          进入到 home 目录下
[root@localhost home]# mkdir pass-through  创建 pass-through 文件夹
[root@localhost qemu]#
[root@localhost qemu]# cd /home
[root@localhost home]# ls
CLD-1 _novavm pass-through win02
```

7.2 进入该目录上传镜像

```
[root@localhost ~]# cd /home/pass-through  进入到/home/pass-through 文件夹下
[root@localhost pass-through]# rz          上传镜像文件
```

```
[root@localhost pass-through]# rz
rz waiting to receive.
Starting zmodem transfer. Press Ctrl+C to cancel.
Transferring NOVA_VM_CLD-HW01-v22.12.11-build2907-20230225.img.cloud.zip...
100% 604313 KB 26274 KB/sec 00:00:23 0 Errors
```

unzip 命令解压镜像

```
[root@localhost pass-through]# unzip NOVA_VM_CLD-HW01-v22.12.11-build2907-20230225.img.cloud.zip
```

```
[root@localhost pass-through]# ls
NOVA_VM_CLD-HW01-v22.12.11-build2907-20230225.img.cloud.zip
[root@localhost pass-through]# unzip NOVA_VM_CLD-HW01-v22.12.11-build2907-20230225.img.cloud.zip
Archive: NOVA_VM_CLD-HW01-v22.12.11-build2907-20230225.img.cloud.zip
  inflating: data.qcow2
  inflating: boot.qcow2
```

注意：从 23.03 版本开始需要手动创建数据盘

用 23.03 以后版本解压出的文件为 README 和系统盘 boot.qcow2

```
[root@localhost pass-through]# unzip NOVA_VM_CLD-HW01-v23.03.06-build3094-20230321-x86_64.img.cloud.zip
Archive: NOVA_VM_CLD-HW01-v23.03.06-build3094-20230321-x86_64.img.cloud.zip
  inflating: README
  inflating: boot.qcow2
[root@localhost pass-through]#
```

README 内容为创建数据盘命令

```
  inflating: boot.qcow2
[root@localhost pass-through]# cat README
Create 40G empty disk by following command:
# qemu-img create -f qcow2 -o preallocation=full ./data.qcow2 40G  执行创建数据盘命令
[root@localhost pass-through]# qemu-img create -f qcow2 -o preallocation=full ./data.qcow2 40G
Formatting './data.qcow2': fmt=qcow2 size=42949672960 encryption=off cluster_size=65536 preallocation='full' lazy_refcounts=off
[root@localhost pass-through]# ls
boot.qcow2 data.qcow2 NOVA_VM_CLD-HW01-v23.03.06-build3094-20230321-x86_64.img.cloud.zip README
```

7.3.创建 pass-through 的池并启动

```
[root@localhost pass-through]# virsh-----进入 virsh 工具
```

```
virsh # pool-define-as pass-through dir --target '/home/pass-through/' -----定义池 pass-through
```

```
virsh # pool-build pass-through-----构建池 pass-through
virsh # pool-start pass-through-----启动池 pass-through
virsh # pool-autostart pass-through-----池 pass-through 标记为自动启动
virsh # pool-list --all -----查看池运行状态
virsh # pool-refresh pass-through-----刷新池
```

```
virsh # pool-list --all
名称          状态          自动开始
-----
cld-1          活动          否
novavm        活动          是
pass-through   活动          是
win02         活动          是

virsh #
```

[root@localhost ~]# lspci |grep Ethernet-----查看网卡信息

```
[root@localhost pass-through]# lspci |grep Ethernet
01:00.0 Ethernet controller: Intel Corporation 82599ES 10-Gigabit SFI/SFP+ Network Connection (rev 01)
01:00.1 Ethernet controller: Intel Corporation 82599ES 10-Gigabit SFI/SFP+ Network Connection (rev 01)
04:00.0 Ethernet controller: Intel Corporation 82599ES 10-Gigabit SFI/SFP+ Network Connection (rev 01)
04:00.1 Ethernet controller: Intel Corporation 82599ES 10-Gigabit SFI/SFP+ Network Connection (rev 01)
07:00.0 Ethernet controller: Intel Corporation 82599ES 10-Gigabit SFI/SFP+ Network Connection (rev 01)
07:00.1 Ethernet controller: Intel Corporation 82599ES 10-Gigabit SFI/SFP+ Network Connection (rev 01)
08:00.0 Ethernet controller: Intel Corporation I210 Gigabit Network Connection (rev 03)
09:00.0 Ethernet controller: Intel Corporation I210 Gigabit Network Connection (rev 03)
[root@localhost pass-through]# █
```

8.4.3 分离网卡

```
virsh nodedev-detach pci_0000_07_00_0
virsh nodedev-detach pci_0000_07_00_1
```

```
[root@localhost pass-through]# virsh nodedev-detach pci_0000_07_00_0
已分离设备 pci_0000_07_00_0

[root@localhost pass-through]# virsh nodedev-detach pci_0000_07_00_1
已分离设备 pci_0000_07_00_1

[root@localhost pass-through]# █
```

8.5.上传虚拟机的 xml 文件



pass-through.xml

```
[root@localhost ~]# cd /etc/libvirt/qemu/
```

rz 上传 pass-through.xml

```
[root@localhost qemu]# rz
rz waiting to receive.
Starting zmodem transfer. Press Ctrl+C to cancel.
Transferring pass-through.xml...
 100%    4 KB    4 KB/sec    00:00:01    0 Errors
```

修改文件内容调整虚拟机配置

```
[root@localhost ~]# vim /etc/libvirt/qemu/pass-through.xml
```

```

<domain type='kvm'>
  <name>pass-through</name>
  <uuid>b/d2c5ae-7f40-437c-8a7b-0a624ed30662</uuid>
  <memory unit='KiB' size='12582912' currentMemory='12582912'>memory</memory>
  <currentMemory unit='KiB' size='12582912' currentMemory='12582912'>currentMemory</currentMemory>
  <vcpu placement='static' id='0'>4</vcpu>
  <os>
    <type arch='x86_64' machine='pc-i440fx-rhel7.0.0'>hvm</type>
    <boot dev='hd'>/>
  </os>
  </features>
  <cpu mode='host-model' check='partial'>
    <model fallback='allow'>/>
    <topology sockets='1' cores='4' threads='1'>/>
  </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' cache='writethrough'>/>
      <source file='/home/pass-through/boot.qcow2'>/>
      <target dev='hda' bus='ide'>/>
      <address type='drive' controller='0' bus='0' target='0' unit='0'>/>
    </disk>
    <disk type='file' device='disk'>
      <driver name='qemu' type='qcow2' cache='writethrough'>/>
      <source file='/home/pass-through/data.qcow2'>/>
      <target dev='hdb' bus='ide'>/>
      <address type='drive' controller='0' bus='0' target='0' unit='1'>/>
    </disk>
  </devices>
  <controller>
    <interface type='direct'>
      <mac address='52:54:00:7b:8f:c5'>/>
      <source dev='eno1' mode='bridge'>/>
      <model type='e1000'>/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x03' function='0x00'>/>
    </interface>
    <serial type='pty'>
      <target type='isa-serial' port='0'>
        <model name='isa-serial'>/>
      </target>
    </serial>
    <console type='pty'>
      <target type='serial' port='0'>/>
    </console>
    <channel type='spicevmc'>
      <target type='virtio' name='com.redhat.spice.0'>/>
      <address type='virtio-serial' controller='0' bus='0' port='1'>/>
    </channel>
    <input type='mouse' bus='ps2'>/>
    <input type='keyboard' bus='ps2'>/>
    <graphics type='spice' autoports='yes'>
      <listen type='address'>/>
      <image compression='off'>/>
    </graphics>
    <sound model='ich6'>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x04' function='0x00'>/>
    </sound>
    <video>
      <model type='qxl' ram='65536' vram='65536' vgamem='16384' heads='1' primary='yes'>/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x00'>/>
    </video>
    <hostdev mode='subsystem' type='pci' managed='yes'>
      <source>
        <address domain='0x0000' bus='0x07' slot='0x00' function='0x00'>/>
      </source>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x07' function='0x00'>/>
    </hostdev>
    <hostdev mode='subsystem' type='pci' managed='yes'>
      <source>
        <address domain='0x0000' bus='0x07' slot='0x00' function='0x01'>/>
      </source>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x08' function='0x00'>/>
    </hostdev>
    <redirdev bus='usb' type='spicevmc'>
      <address type='usb' bus='0' port='1'>/>
    </redirdev>
    <redirdev bus='usb' type='spicevmc'>
      <address type='usb' bus='0' port='2'>/>
    </redirdev>
    <memballoon model='virtio'>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x09' function='0x00'>/>
    </memballoon>
  </devices>

```

虚拟机名称和xml文件名称一致

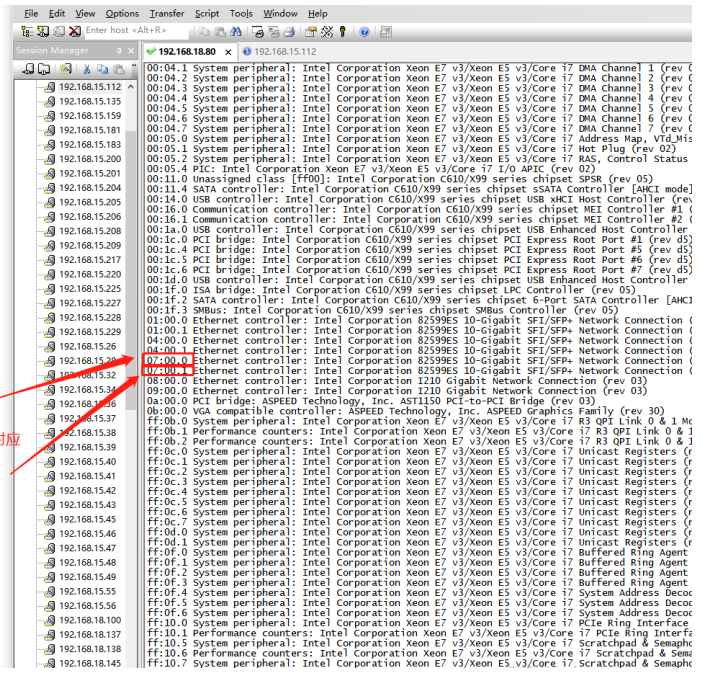
修改内存大小

核数修改

系统盘和数据盘名称和路径要和之前解压的路径正确

管理口名称一致

网卡名称对应



7.6 启动虚拟机

```

[root@localhost qemu]# virsh define pass-through.xml ----导入虚拟机配置
[root@localhost qemu]# virsh start pass-through-----启动虚拟机
[root@localhost ~]# virsh list -----查看虚拟机运行状态
[root@localhost qemu]# virsh console pass-through ----连接到虚拟机，登陆用 admin/admin.

```

```
Database is running now.
Database user created successfully.
Initialize database successfully.
Http Web service is started.
Https Web service is started.
Adapter index 3
MAC 52:54:00:7b:8f:c5
udhcpd (v1.22.1) started
Executing /usr/share/udhcpd/default.script deconfig
Entering listen mode: raw
Opening raw socket on ifindex 3
Got raw socket fd
Attached filter to raw socket fd
Created raw socket
Adapter index 3
MAC 52:54:00:7b:8f:c5
Sending discover...
Waiting on select 3 seconds 测试仪dhcp自动获取IP地
Received a packet 址
Adapter index 3
MAC 52:54:00:7b:8f:c5
Sending select for 192.168.18.49..
Waiting on select 3 seconds
Received a packet
Lease of 192.168.18.49 obtained, lease time 604800
Executing /usr/share/udhcpd/default.script bound
deleting routers
Entering listen mode: none
Version: 22.12.11 build2907
Build date: 20230225
kernel version: 5.10.112
System is started.

Supernova login: █
```

7.7 通过 web 登录测试仪



8. 附加：VNC 配置方法

8.1 安装软件包

命令：yum install -y tigervnc-server

8.2 关闭防火墙

命令：systemctl stop firewalld
systemctl disable firewalld

8.3 复制配置文件

命令：cp /lib/systemd/system/vncserver@.service
/etc/systemd/system/vncserver@:1.service

8.4 编辑复制出来的配置文件

命令：vi /etc/systemd/system/vncserver@:1.service

```
# `man vncviewer` manual page.

[Unit]
Description=Remote desktop service (VNC)
After=syslog.target network.target

[Service]
Type=simple 改为simple
# Clean any existing files in /tmp/.X11-unix environment
ExecStartPre=/bin/sh -c '/usr/bin/vncserver -kill %i > /dev/null 2>&1 || :'
ExecStart=/usr/sbin/runuser -l <root> -c "/usr/bin/vncserver %i"
PIDFile=/home/<root>/.vnc/%H%i.pid
ExecStop=/bin/sh -c '/usr/bin/vncserver -kill %i > /dev/null 2>&1 || :'

[Install]
WantedBy=multi-user.target
"/etc/systemd/system/vncserver@:1.service" 48L, 1827C
```

8.5 重新加载配置文件

命令：systemctl daemon-reload

8.6 设置 VNC 密码

命令：vncpasswd root

8.7 开启 VNC 并设置成开机启动

命令：`systemctl start vncserver@:1.service`
`systemctl enable vncserver@:1.service`

8.8 启动

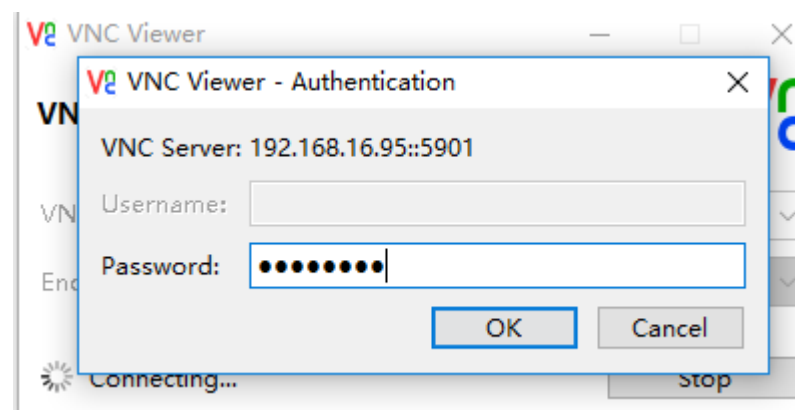
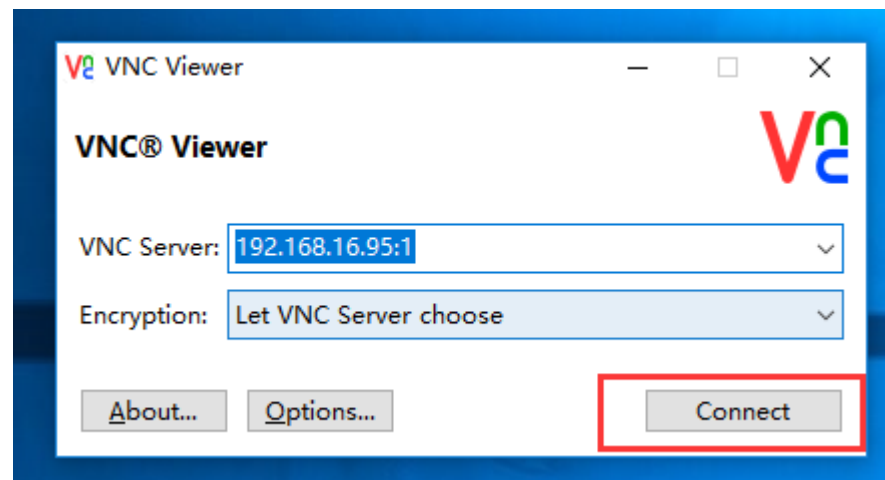
命令：`vncserver`

8.9 如果启动成功端口是监听状态(VNC 端口号默认 5900+1)

命令：`netstat -an |grep 5901`

```
[root@localhost ~]# netstat -an |grep 5901
tcp        0      0 0.0.0.0:5901        0.0.0.0:*        LISTEN
tcp6       0      0 :::5901           :::*              LISTEN
[root@localhost ~]#
```

8.10 客户端连接



测试连接成功。