KVM_SR-IOV 上部署 Supernova

(界面方式和命令行方式)

网测科技

2023.03.01

目录

版 本	说　明	时 间
V1.0	初版拟制	2019/01/15
V1.1	增加“1.7：支持 CPU 虚拟化透传”	2019/09/10
V1.2	增加 8 使用命令行方式部署虚拟机	2022/09/09

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。
网卡是一对 Intel 82599-10G 网卡
VT-d 的性能非常好，但是它的物理设备只能分配给一个客户机使用。为了实现多个虚机共享一个物理设备，并且达到直接分配的目的，PCI-SIG 组织发布了 SR-IOV （Single Root I/O Virtualization and sharing） 规范，它定义了一个标准化的机制用以原生地支持实现多个客户机共享一个设备。不过，目前 SR-IOV （单根 I/O 虚拟化）最广泛地应用还是网卡上。

SR-IOV 使得一个单一的功能单元（比如，一个以太网端口）能看起来像多个独立的物理设备。一个带有 SR-IOV 功能的物理设备能被配置为多个功能单元。SR-IOV 使用两种功能（function）：

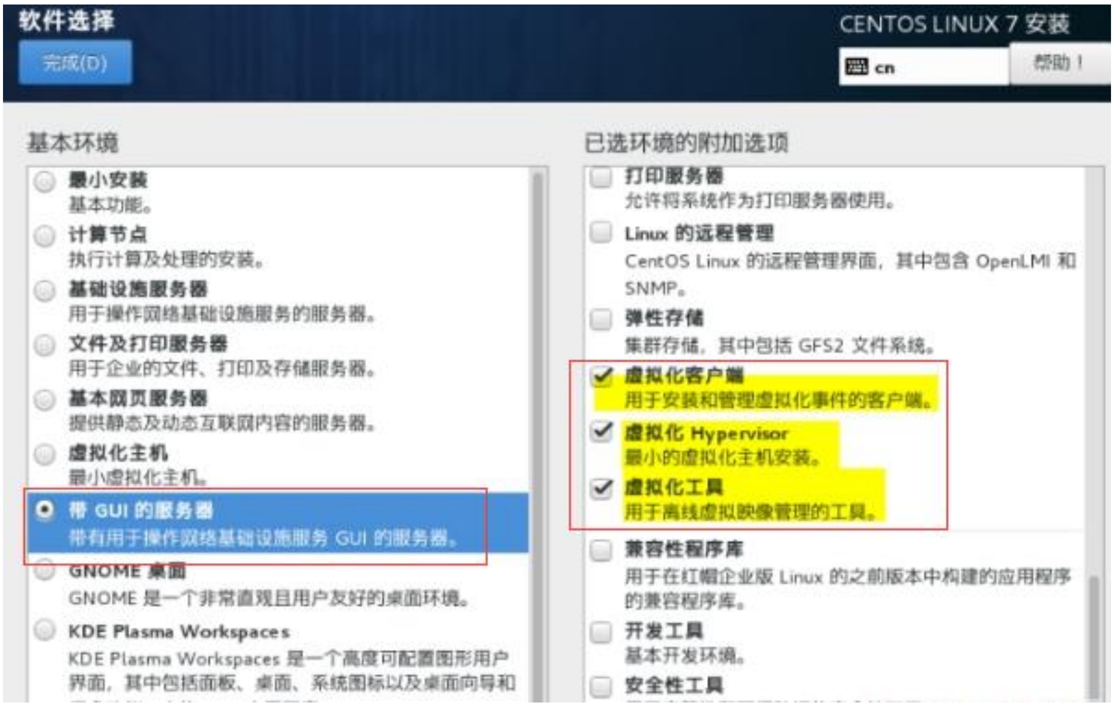
- 物理功能（Physical Functions，PF）：这是完整的带有 SR-IOV 能力的 PCIe 设备。PF 能像普通 PCI 设备那样被发现、管理和配置。
- 虚拟功能（Virtual Functions，VF）：简单的 PCIe 功能，它只能处理 I/O。每个 VF 都是从 PF 中分离出来的。每个物理硬件都有一个 VF 数目的限制。一个 PF，能被虚拟成多个 VF 用于分配给多个虚拟机。

从 SRIOV 的中文字面不难理解，它属于 VT-d 技术的一个分支，要实现 SRIOV 功能，前提条件就是你的网卡首先要支持 SRIOV，你的主板要支持 VT-d 技术（支持 VT-d 自然也就支持 SRIOV）
Hypervisor 能将一个或者多个 VF 分配给一个虚机。在某一时刻，一个 VF 只能被分配给一个虚机。一个虚机可以拥有多个 VF。在虚机的操作系统看来，一个 VF 网卡看起来和一个普通网卡没有区别。SR-IOV 驱动是在内核中实现的。
宿主机：就是实体机

主题内容

1. 安装 KVM 前准备

1.1 安装 CentOS 系统时注意：

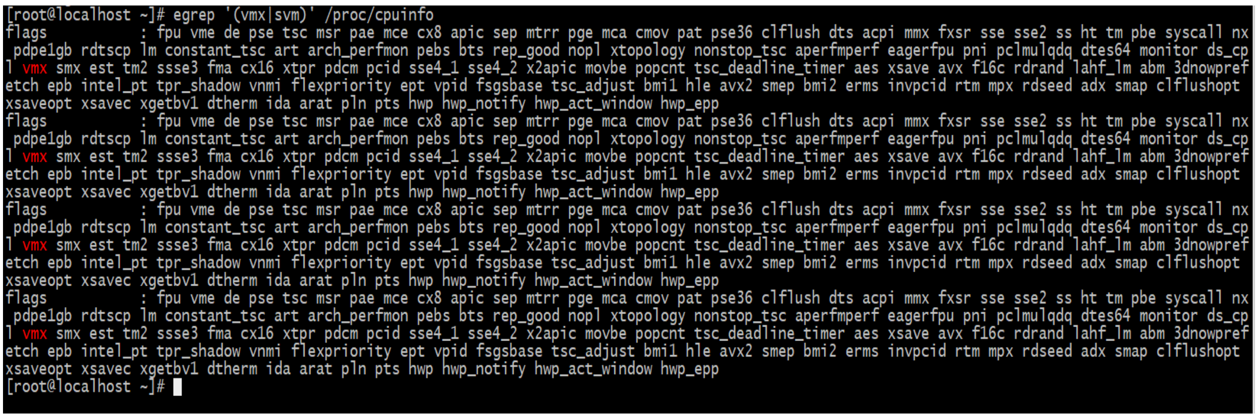


1.2 禁用它如下:

命令：chkconfig NetworkManager off
命令：chkconfig network on
命令：service NetworkManager stop

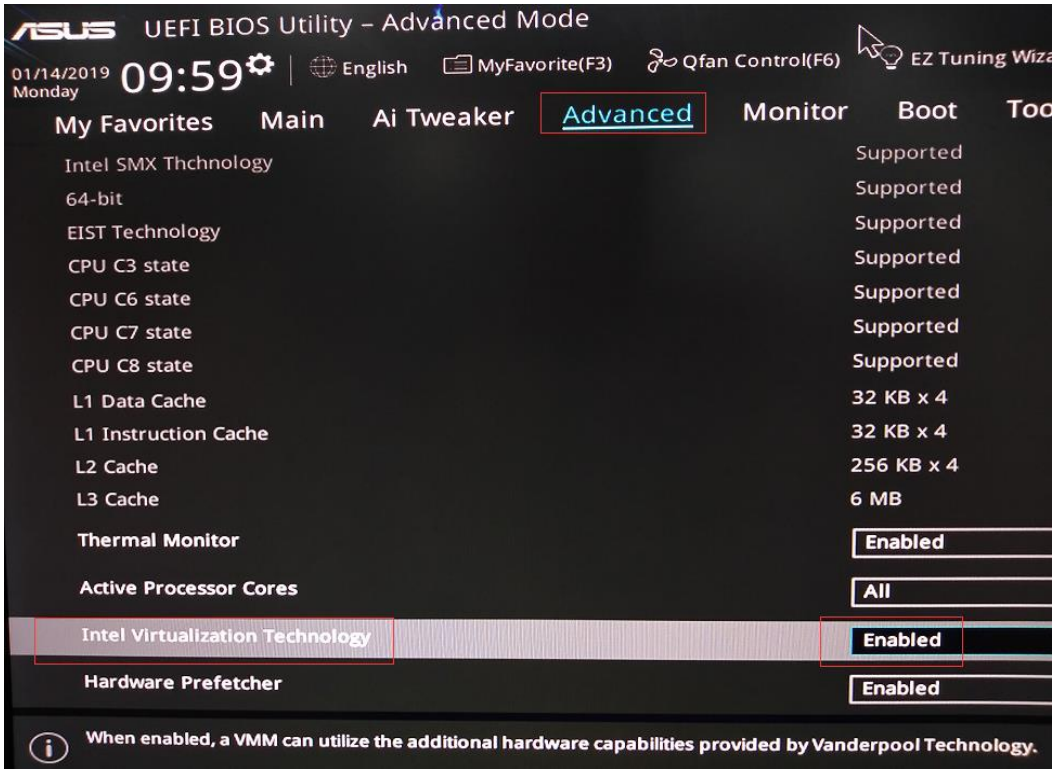
1.3 验证 cpu 是否支持 KVM，结果有 vmx（INTEL）或 svm（AMD）说明支持

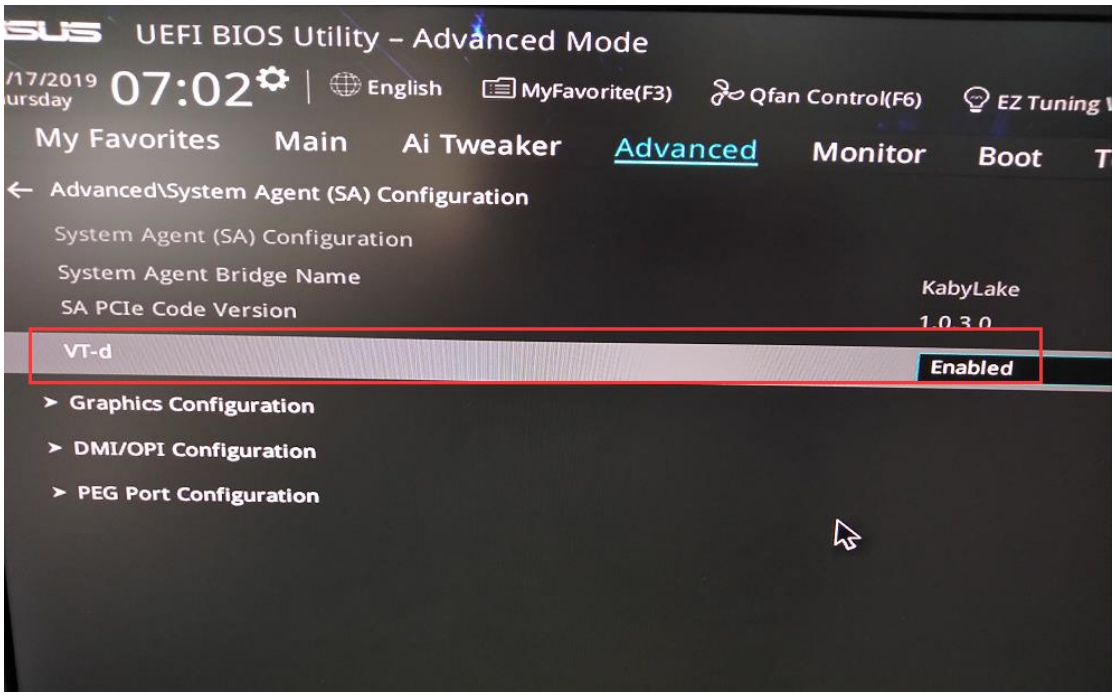
命令：cat /proc/cpuinfo | egrep 'vmx|svm'
效果：



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　找到“VT-D”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
```

1.6 查看物理网卡是否支持 SR-IOV(单根虚拟化)，说明支持 SR-IOV

命令：lspci -v |grep SR-IOV
效果：

```
[root@localhost ~]# lspci -v| grep SR-IOV
        Capabilities: [160] Single Root I/O Virtualization (SR-IOV)
        Capabilities: [160] Single Root I/O Virtualization (SR-IOV)
[root@localhost ~]#
```

1.7 支持 CPU 虚拟化透传

命令：echo 'options kvm_intel nested=1' >/etc/modprobe.d/kvm-nested.conf

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 libguests-tools bridge-utils virt-install
说明：
libvirt：必须要装的核心工具
pyhon-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 一 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-scri
           └─8501 /usr/sbin/dnsmasq --conf-file=/var/lib/libvirt/dnsmasq/default.conf --leasefile-ro --dhcp-scri

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. 需要在内核中启用，在打开的文件中添加 intel_iommu=on 参数和 pci=assign-busses 和 pci=realloc 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 pci=assign-busses pci=realloc"
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. 确认网卡设备的驱动类型

命令：ethtool -i enp1s0f0
效果：

```
[root@localhost ~]# ethtool -i enp1s0f0
driver: ixgbe
version: 5.1.0-k-rh7.6
firmware-version: 0x00011bab
expansion-rom-version:
bus-info: 0000:01:00.0
supports-statistics: yes
supports-test: yes
supports-eeprom-access: yes
supports-register-dump: yes
supports-priv-flags: yes
[root@localhost ~]#
```

4.1 可以查看某个物理网卡支持多少 VF （虚拟网卡）

命令：cat /sys/class/net/enp1s0f0/device/sriov_totalvfs
效果：

```
[root@localhost ~]# cat /sys/class/net/enp1s0f0/device/sriov_totalvfs
63
```

4.2 若需要重新加载驱动，先移除驱动，再加载

命令：modprobe -r ixgbe （正常安装，不需要执行）
效果：

```
[root@localhost ~]# ethtool -i enp1s0f0
Cannot get driver information: No such device
[root@localhost ~]#
```

已经找不到这个网卡了 因为已经成功将驱动移除
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 举例每个网卡创建一个 VF

命令：echo 1 > /sys/class/net/enp1s0f0/device/sriov_numvfs
echo 1 > /sys/class/net/enp1s0f1/device/sriov_numvfs

```
[root@localhost ~]# echo 1 > /sys/class/net/enp1s0f0/device/sriov_numvfs
[root@localhost ~]# echo 1 > /sys/class/net/enp1s0f1/device/sriov_numvfs
```

命令含义为将实体网卡 enp1s0f0 虚拟出一个 VF；echo 2 就是虚拟出 2 个 VF；

4.4 分离虚拟出来的网卡

```
[root@localhost ~]# lspci |grep Ethernet
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)
02:10.0 Ethernet controller: Intel Corporation 82599 Ethernet Controller Virtual Function (rev 01)
02:10.1 Ethernet controller: Intel Corporation 82599 Ethernet Controller Virtual Function (rev 01)
```

命令：virsh nodedev-detach pci_0000_02_10_0

virsh nodedev-detach pci_0000_02_10_1

效果：

```
[root@localhost ~]# virsh nodedev-detach pci_0000_02_10_0
已分离设备 pci_0000_02_10_0

[root@localhost ~]# virsh nodedev-detach pci_0000_02_10_1
已分离设备 pci_0000_02_10_1

[root@localhost ~]#
```

这里是可以选择添加的（直到 6 以前）：

命令：vi /etc/rc.d/rc.local

效果：

```
#!/bin/bash
# THIS FILE IS ADDED FOR COMPATIBILITY PURPOSES
#
# It is highly advisable to create own systemd services or udev rules
# to run scripts during boot instead of using this file.
#
# In contrast to previous versions due to parallel execution during boot
# this script will NOT be run after all other services.
#
# Please note that you must run 'chmod +x /etc/rc.d/rc.local' to ensure
# that this script will be executed during boot.

touch /var/lock/subsys/local
echo 1 > /sys/class/net/enp1s0f0/device/sriov_numvfs
echo 1 > /sys/class/net/enp1s0f1/device/sriov_numvfs
~
~
~
```

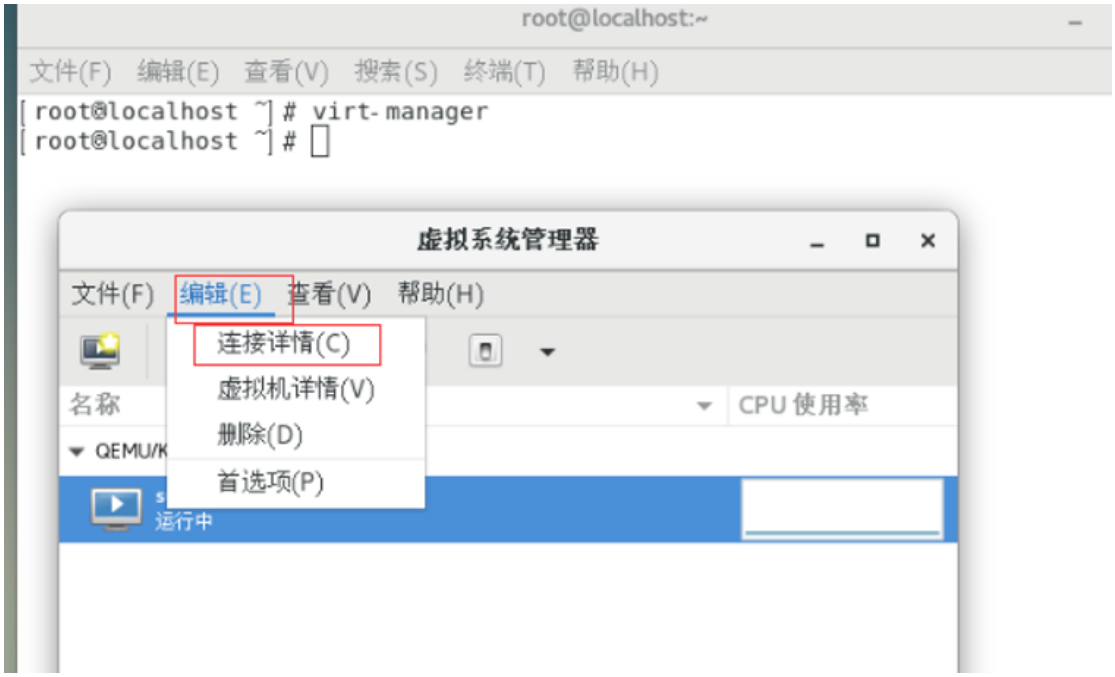
命令：vi /etc/rc.d/rc.local

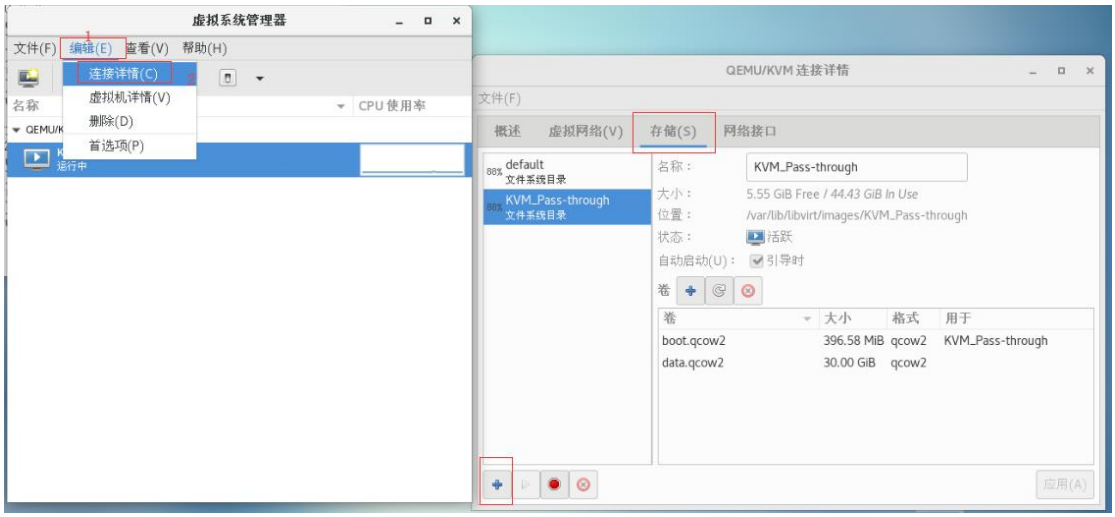
效果：

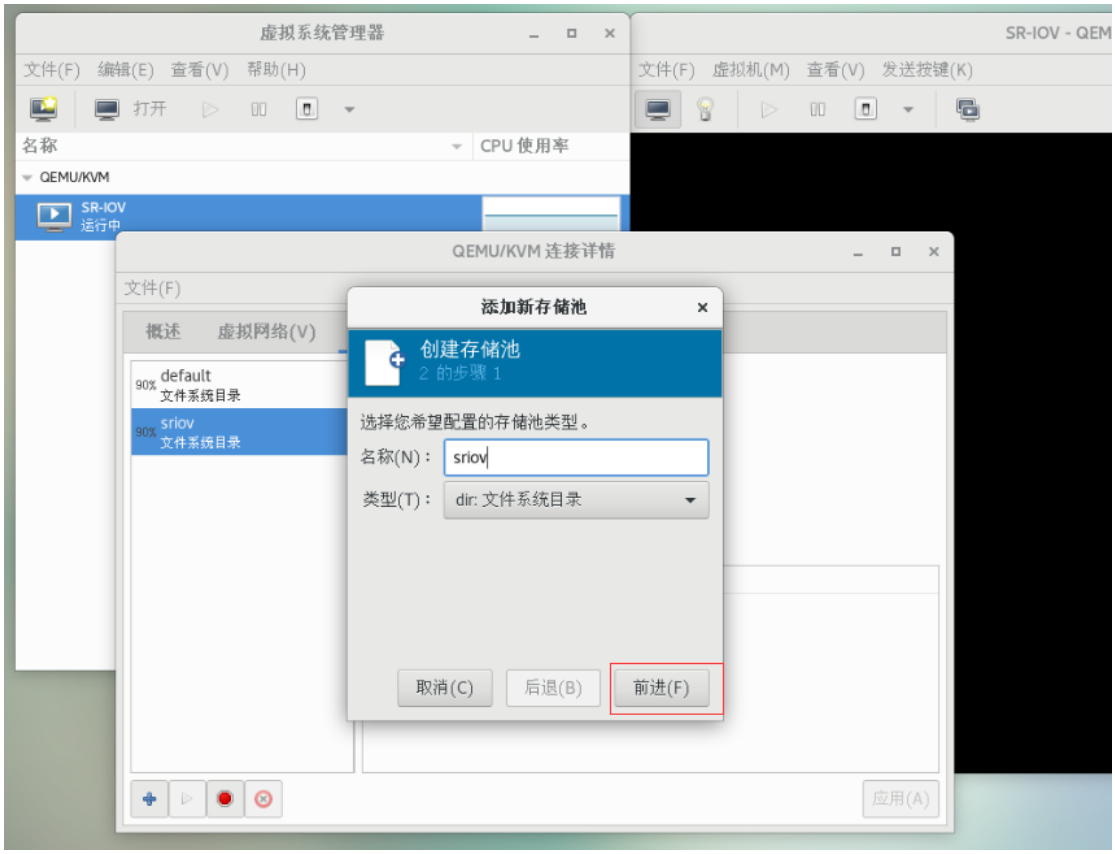
```
# It is highly advisable to create own systemd services or udev rules
# to run scripts during boot instead of using this file.
#
# In contrast to previous versions due to parallel execution during boot
# this script will NOT be run after all other services.
#
# Please note that you must run 'chmod +x /etc/rc.d/rc.local' to ensure
# that this script will be executed during boot.

touch /var/lock/subsys/local

echo 1 > /sys/class/net/enp1s0f0/device/sriov_numvfs
echo 1 > /sys/class/net/enp1s0f1/device/sriov_numvfs

ip link set enp1s0f0 vf 0 mac aa:bb:cc:dd:ee:00
ip link set enp1s0f1 vf 0 mac aa:bb:cc:dd:ee:01
~
~
~
```

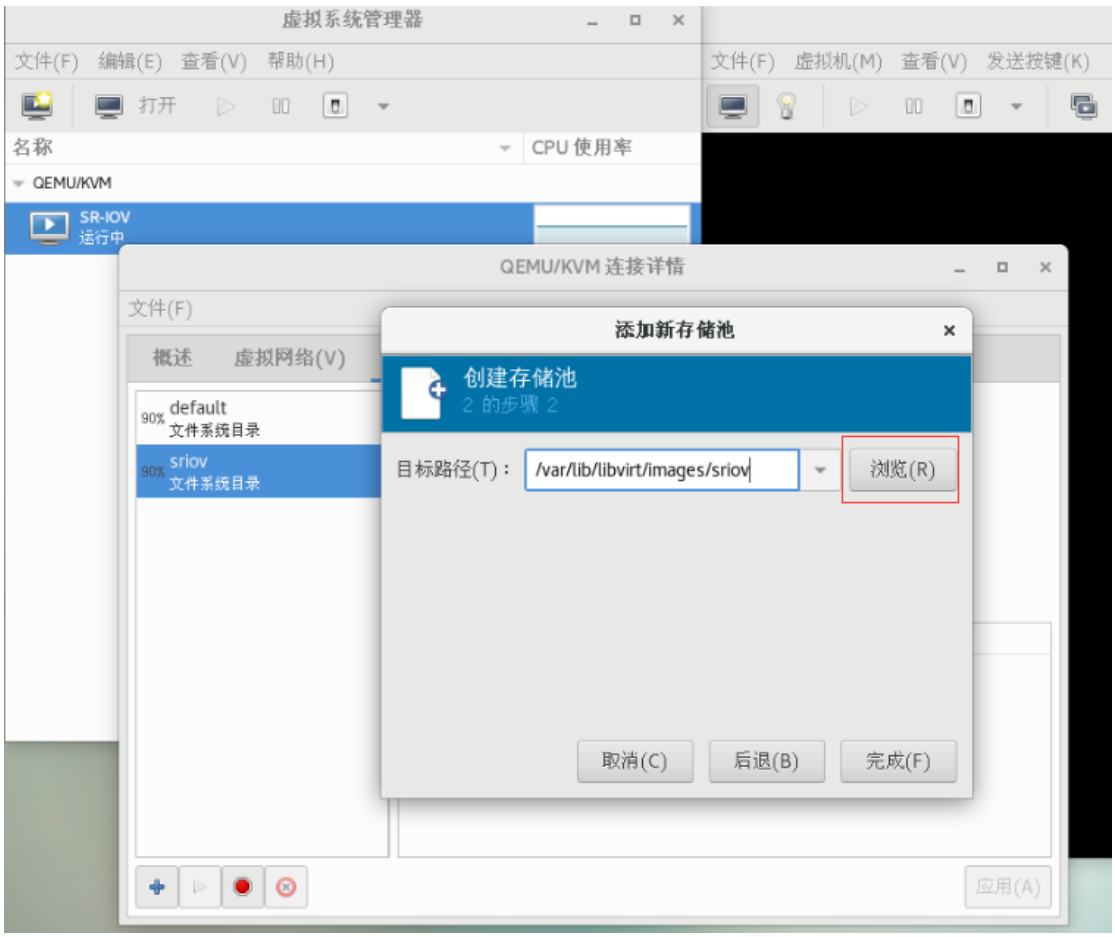
5. 界面方式部署 KVM

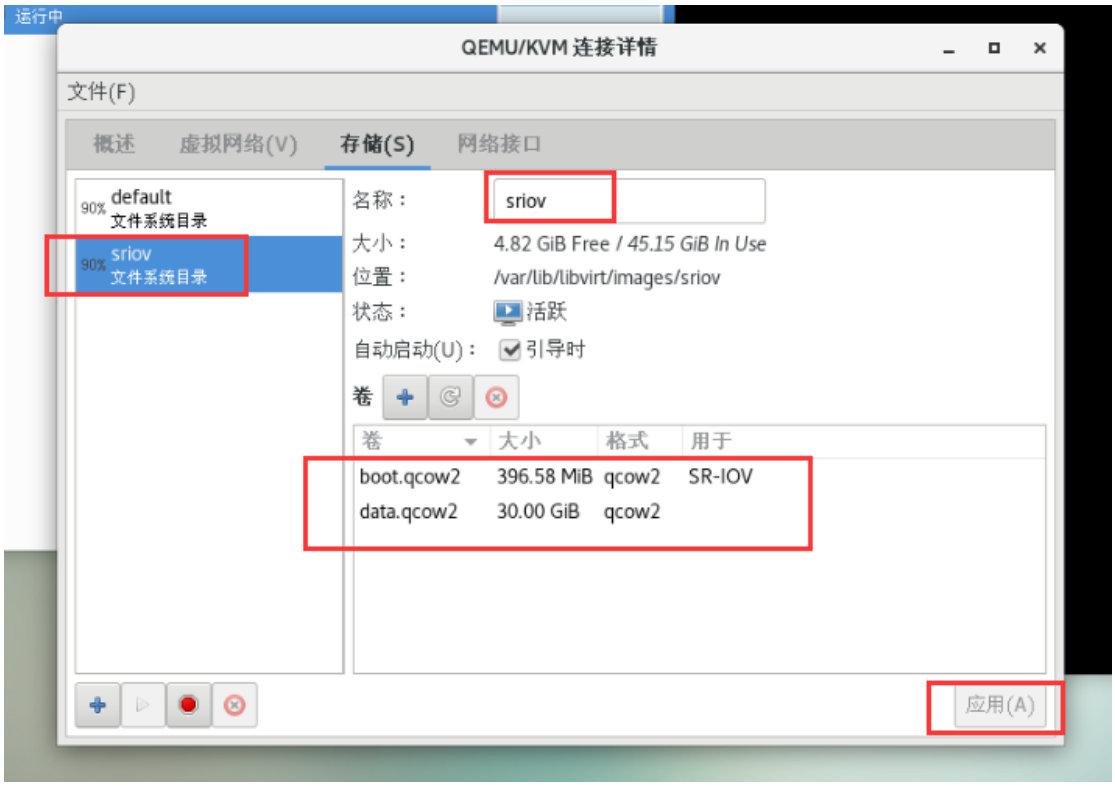
5.1 添加存储池






虚拟系统管理器
文件(F) 编辑(E) 查看(V) 帮助(H)
打开
名称
CPU 使用率
QEMU/KVM
SR-IOV
运行中
SR-IOV - QEM
文件(F) 虚拟机(M) 查看(V) 发送按键(K)
QEMU/KVM 连接详情
文件(F)
概述
虚拟网络(V)
90% default
文件系统目录
90% sriov
文件系统目录
添加新存储池
创建存储池
2 的步骤 1
选择您希望配置的存储池类型。
名称(N)：
sriov
类型(T)：
dir: 文件系统目录
取消(C)
后退(B)
前进(F)
应用(A)


虚拟系统管理器
文件(F) 编辑(E) 查看(V) 帮助(H)
打开
名称
CPU 使用率
QEMU/KVM
SR-IOV
运行中
文件(F) 虚拟机(M) 查看(V) 发送按键(K)
QEMU/KVM 连接详情
文件(F)
概述
虚拟网络(V)
90% default
文件系统目录
90% sriov
文件系统目录
添加新存储池
创建存储池
2 的步骤 2
目标路径(T)：
/var/lib/libvirt/images/sriov
浏览(R)
取消(C)
后退(B)
完成(F)
应用(A)



注意：从 23.03 版本开始需要手动创建数据盘

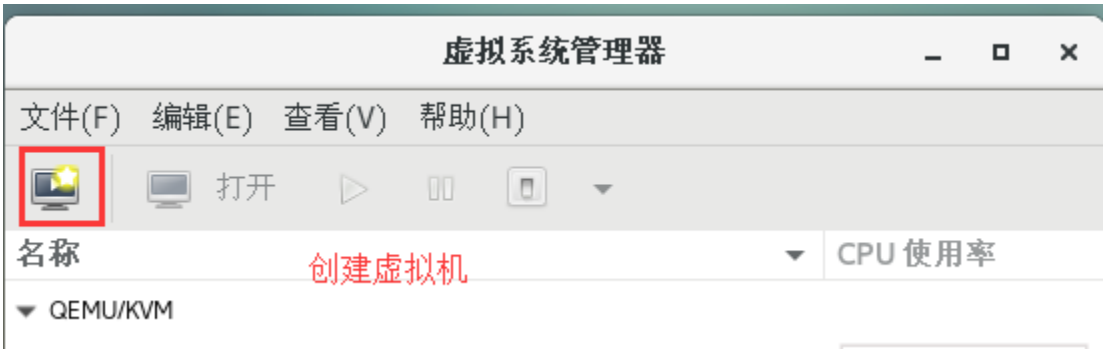
用 23.03 以后版本解压出的文件为 README 和系统盘 boot.qcow2

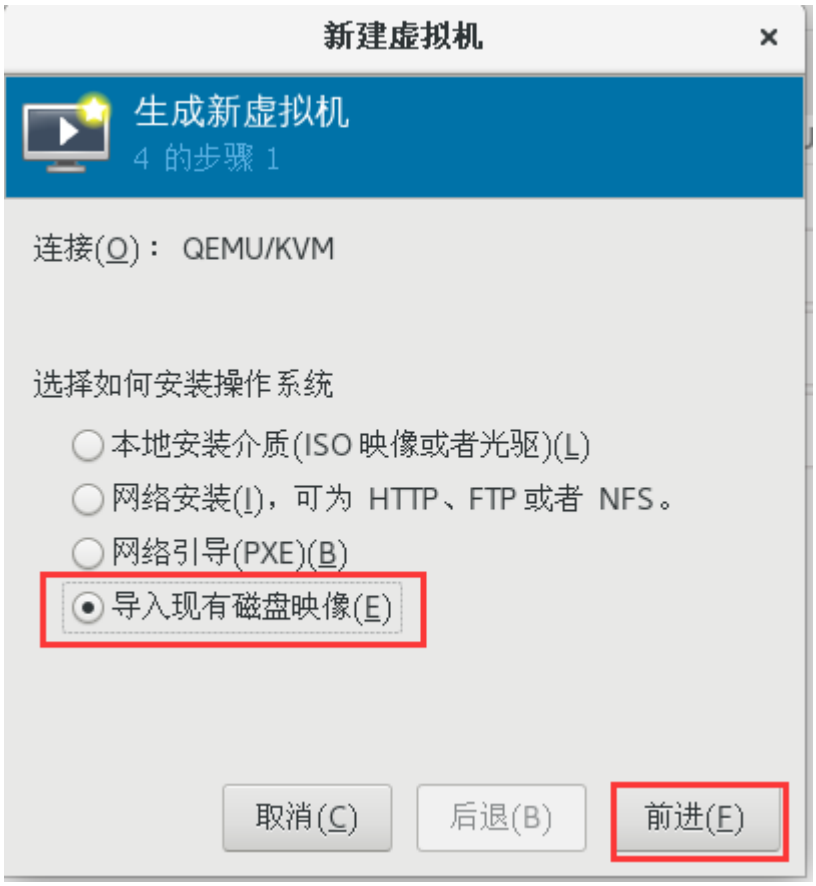
打开终端执行创建数据盘命令

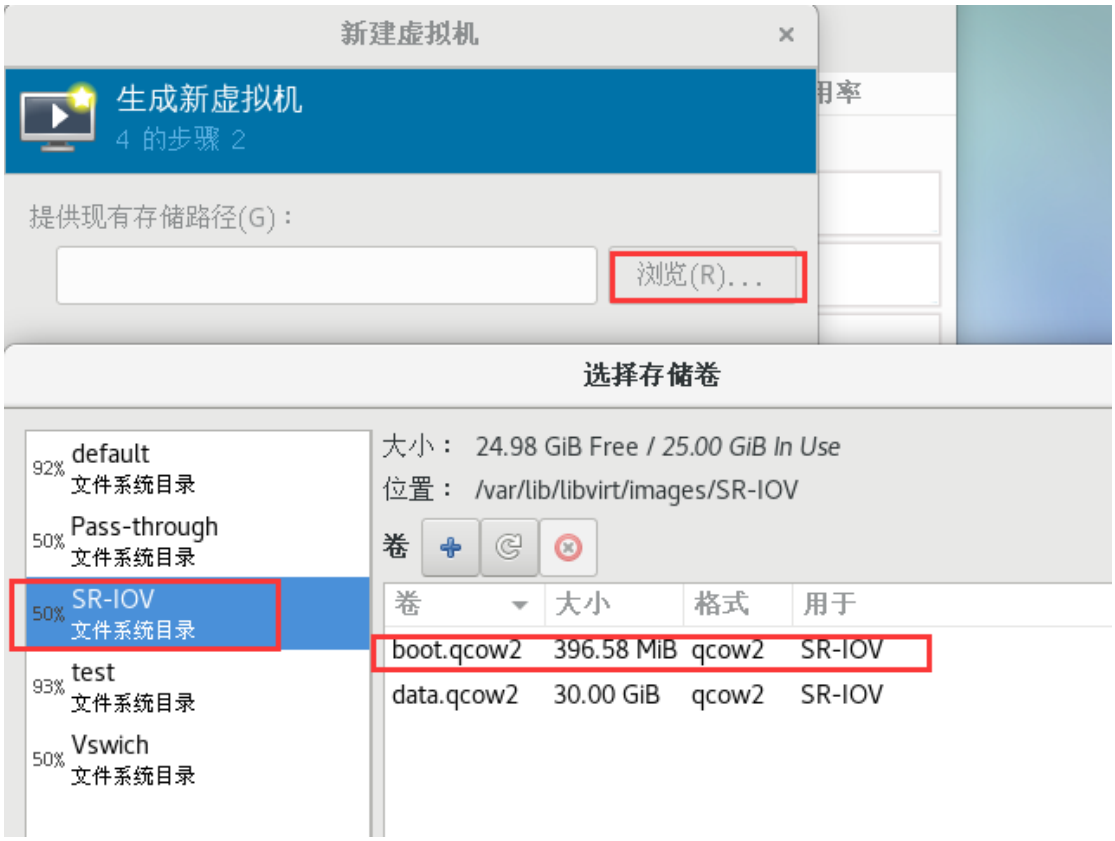

打开(O)
README
~/.cache/.fr-9r3zvk
保存(S)
Create 40G empty disk by following command:
qemu-img create -f qcow2 -o preallocation=full ./data.qcow2 40G

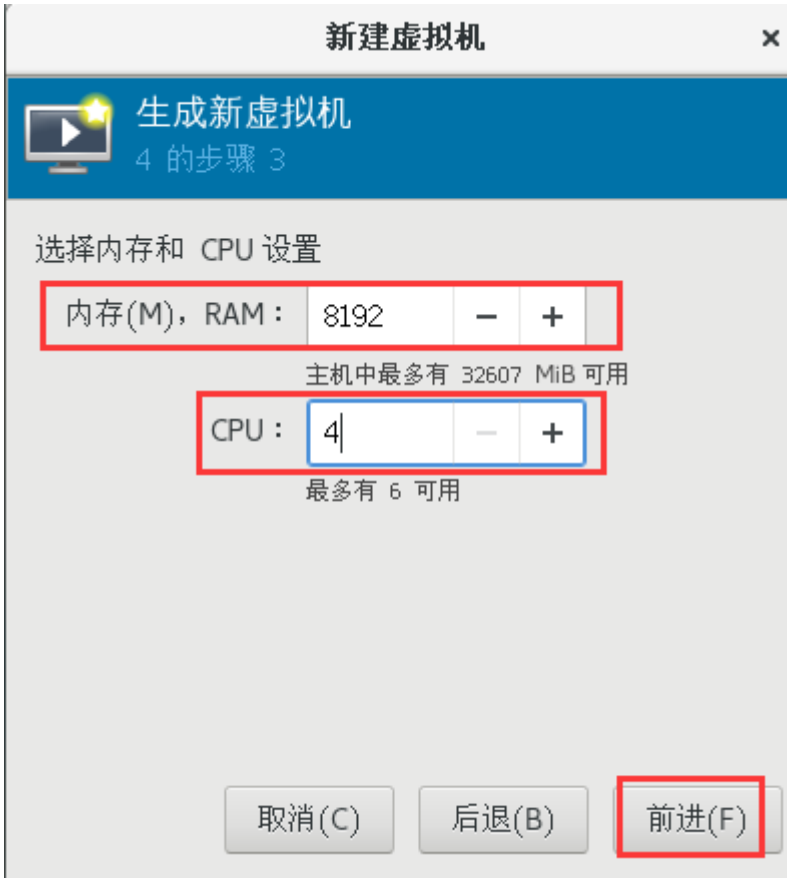

root@localhost:~
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
[root@localhost ~]# qemu-img create -f qcow2 -o preallocation=full ./data.qcow2
40G
Formatting './data.qcow2', fmt=qcow2 size=42949672960 encryption=off cluster_siz
e=65536 preallocation=full lazy_refcounts=off
[root@localhost ~]# ls
anaconda-ks.cfg data.qcow2 README 公共 视频 文档 音乐
boot.qcow2 initial-setup-ks.cfg root 模板 图片 下载 桌面
[root@localhost ~]#

5.2 创建虚拟机


虚拟系统管理器
文件(F) 编辑(E) 查看(V) 帮助(H)
打开
名称
创建虚拟机
CPU 使用率
QEMU/KVM

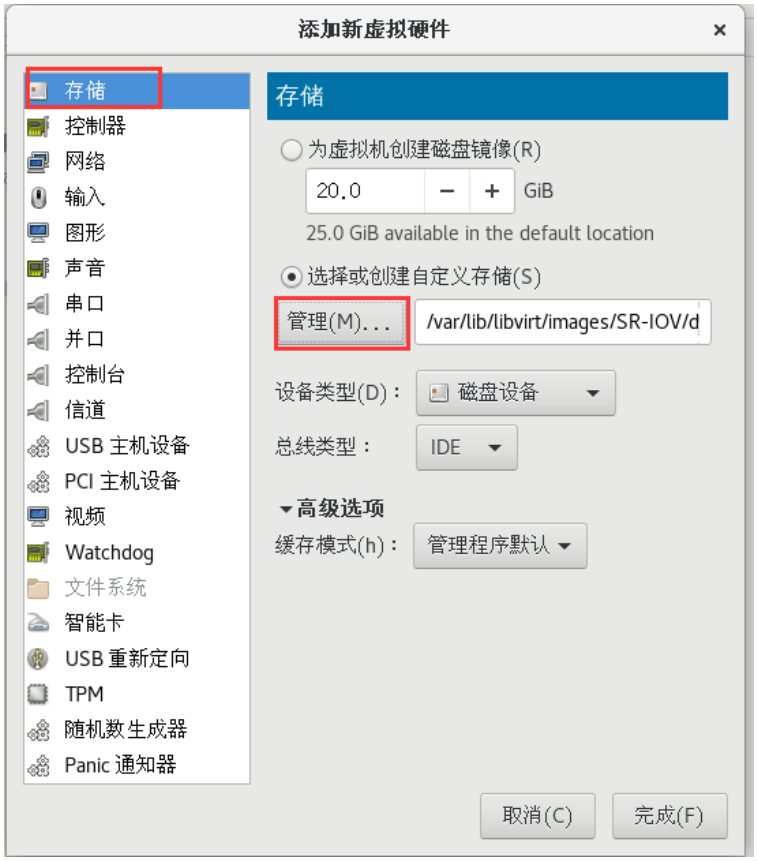

新建虚拟机
生成新虚拟机
4 的步骤 1
连接(O)： QEMU/KVM
选择如何安装操作系统
本地安装介质(ISO 映像或者光驱)(L)
网络安装(I)，可为 HTTP、FTP 或者 NFS。
网络引导(PXE)(B)
导入现有磁盘映像(E)
取消(C)
后退(B)
前进(F)

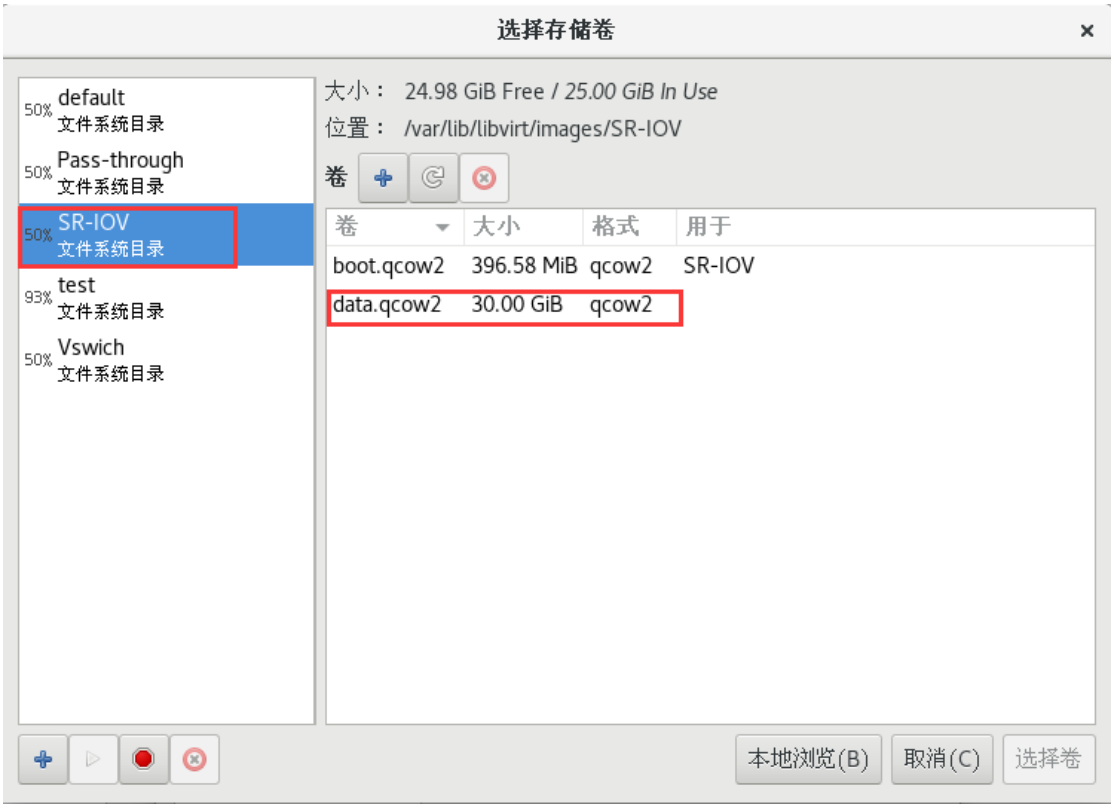

新建虚拟机
生成新虚拟机
4 的步骤 2
提供现有存储路径(G)：
浏览(R)...
选择存储卷
92% default
文件系统目录
50% Pass-through
文件系统目录
50% SR-IOV
文件系统目录
93% test
文件系统目录
50% Vswich
文件系统目录
大小： 24.98 GiB Free / 25.00 GiB In Use
位置： /var/lib/libvirt/images/SR-IOV
卷
卷 大小 格式 用于
boot.qcow2 396.58 MiB qcow2 SR-IOV
data.qcow2 30.00 GiB qcow2 SR-IOV

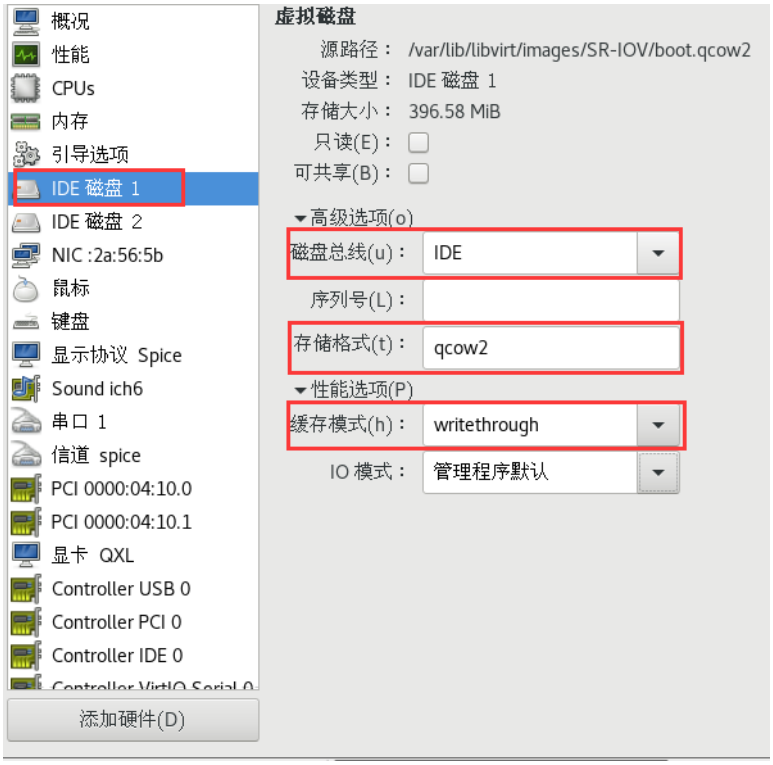


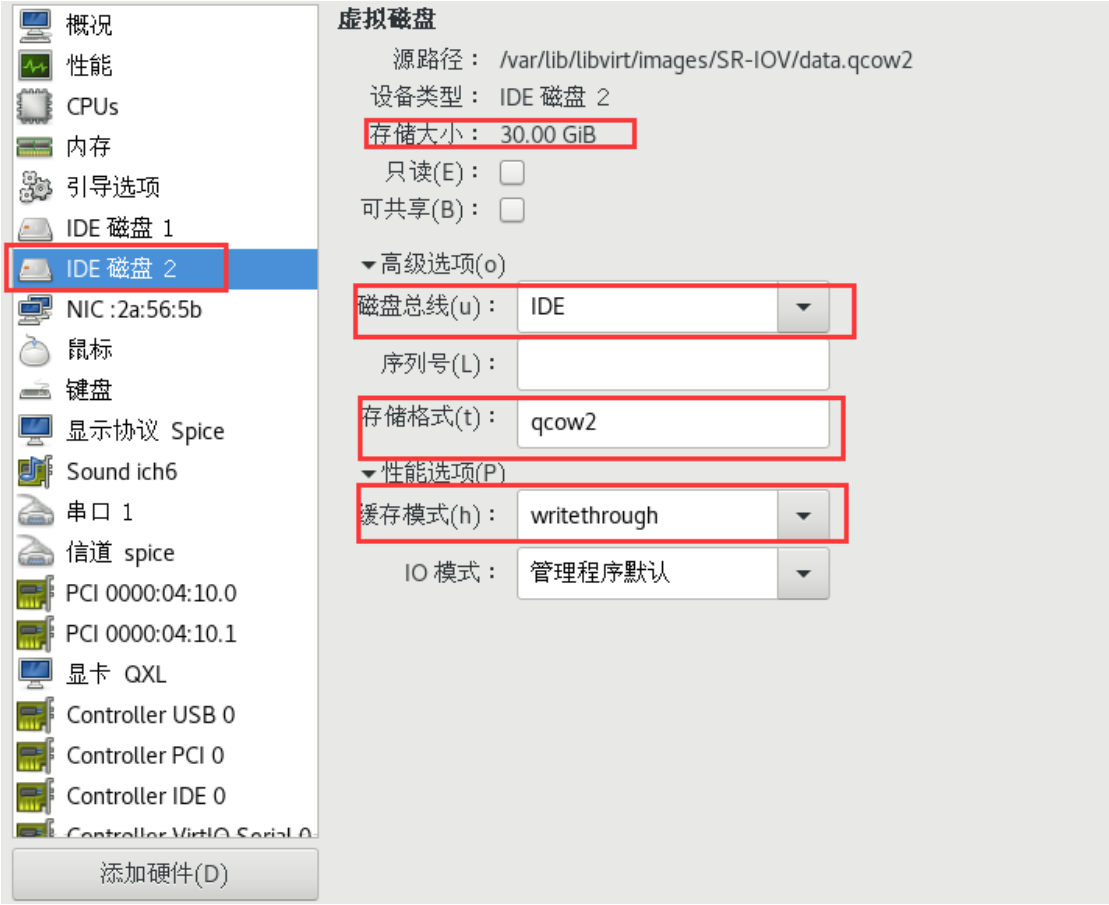
在创建虚拟机的过程中第一块 Boot 盘就有了

5.3 再增加一块 Data 盘（SuperNova 默认要用到两块硬盘）

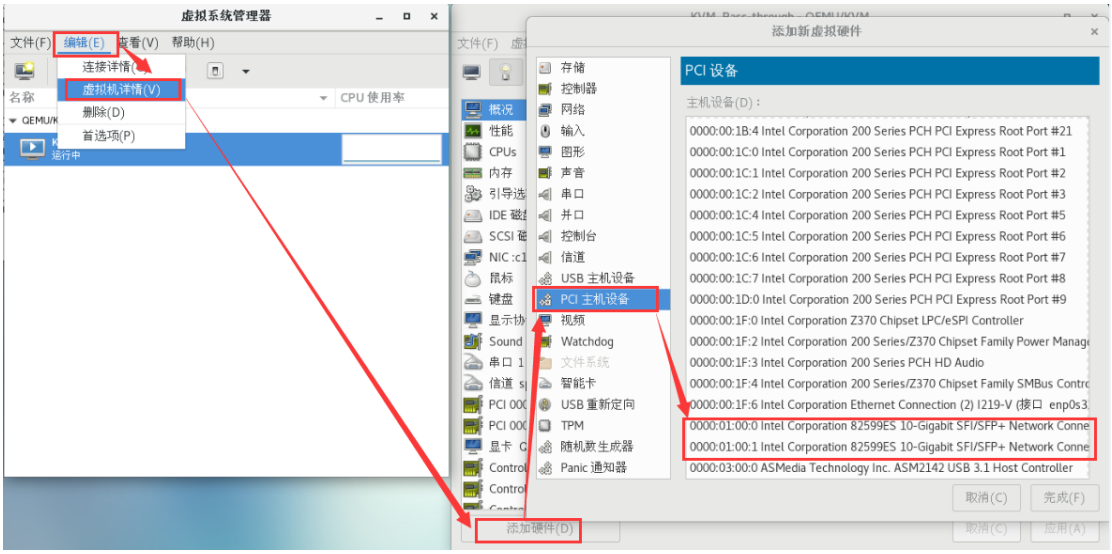


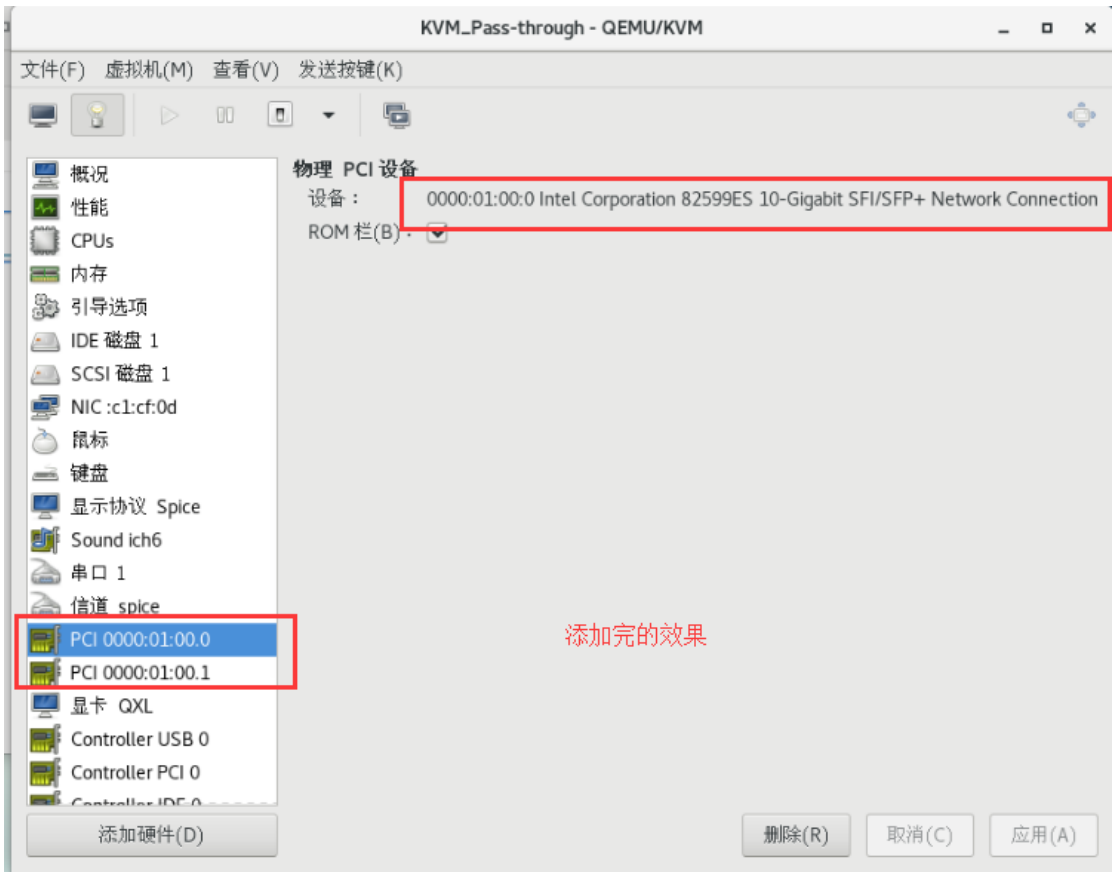

选择存储卷
default
文件系统目录
Pass-through
文件系统目录
SR-IOV
文件系统目录
test
文件系统目录
Vswich
文件系统目录
大小： 24.98 GiB Free / 25.00 GiB In Use
位置： /var/lib/libvirt/images/SR-IOV
卷
卷
大小
格式
用于
boot.qcow2
396.58 MiB
qcow2
SR-IOV
data.qcow2
30.00 GiB
qcow2
本地浏览(B)
取消(C)
选择卷

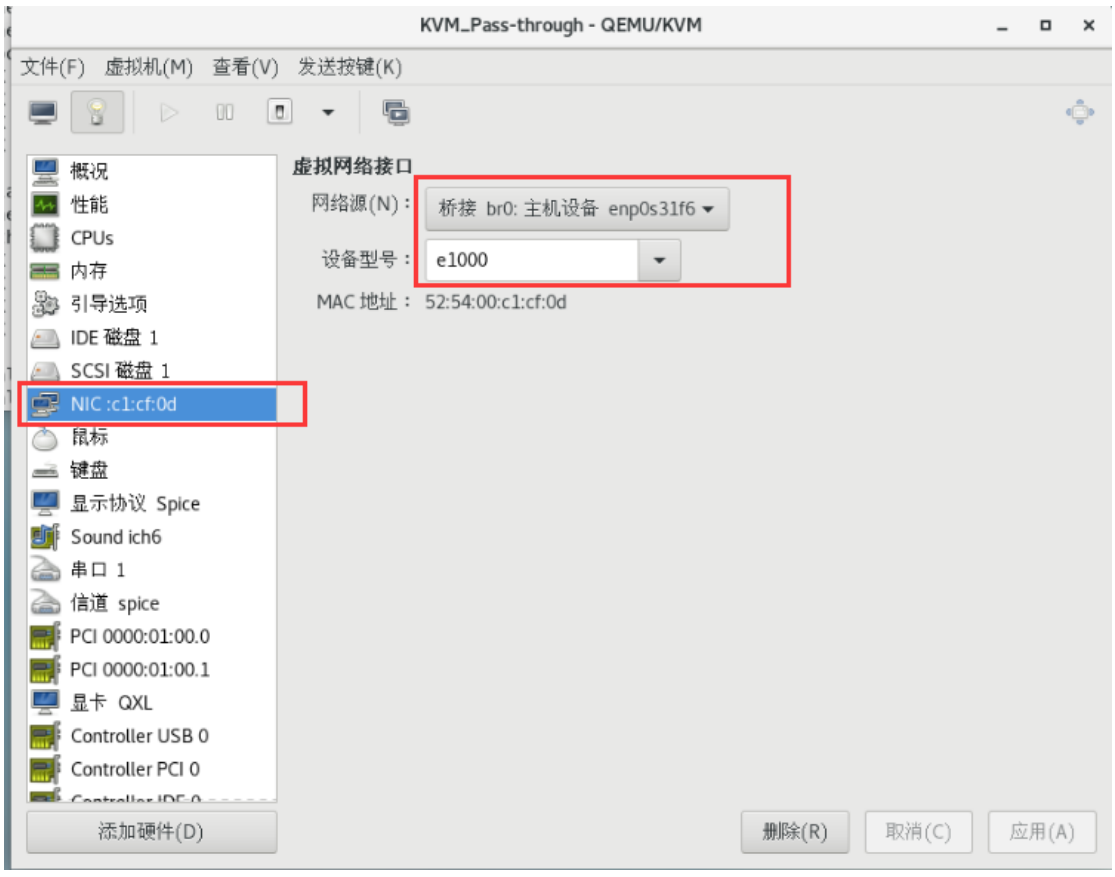

概况
性能
CPUs
内存
引导选项
IDE 磁盘 1
IDE 磁盘 2
NIC :2a:56:5b
鼠标
键盘
显示协议 Spice
Sound ich6
串口 1
信道 spice
PCI 0000:04:10.0
PCI 0000:04:10.1
显卡 QXL
Controller USB 0
Controller PCI 0
Controller IDE 0
添加硬件(D)
虚拟磁盘
源路径： /var/lib/libvirt/images/SR-IOV/boot.qcow2
设备类型： IDE 磁盘 1
存储大小： 396.58 MiB
只读(E)：
可共享(B)：
高级选项(o)
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



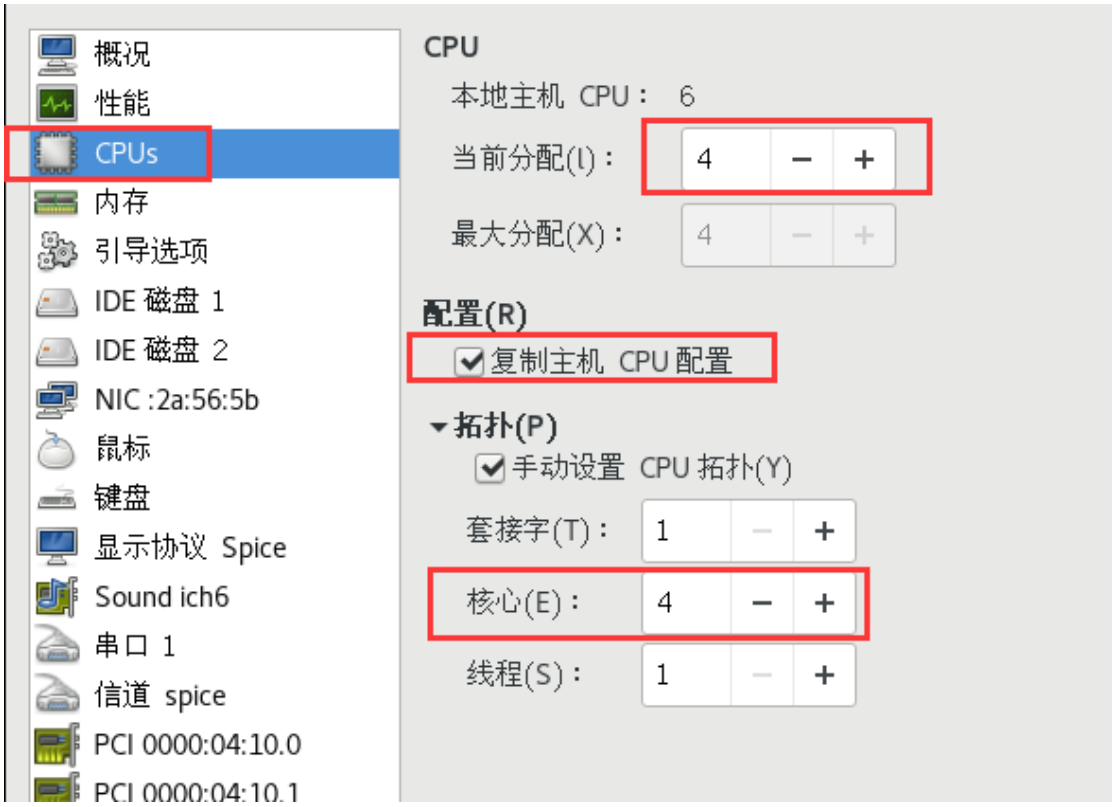
6.4 把从宿主机上分离出来的 PCI 网卡添加到虚拟机上来

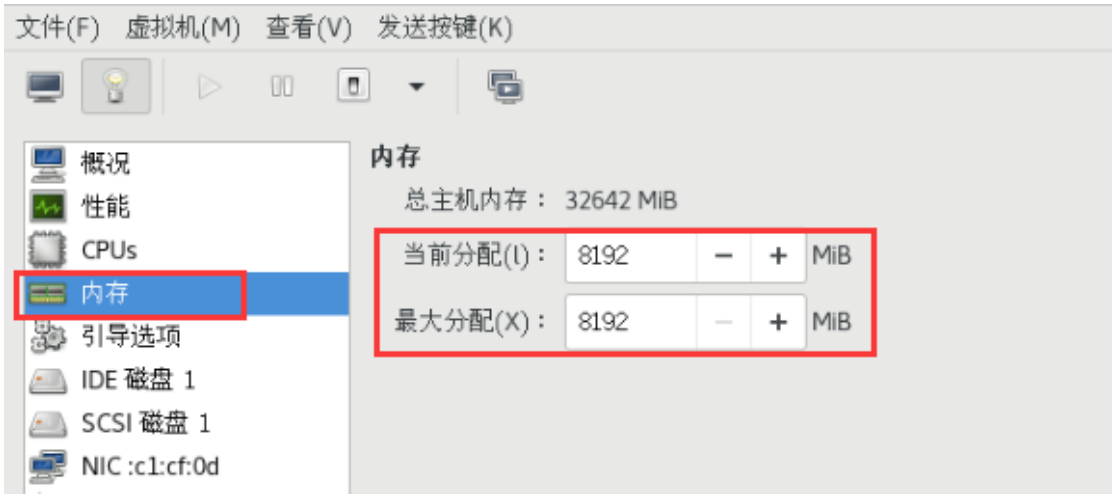



KVM_Pass-through - QEMU/KVM
文件(F) 虚拟机(M) 查看(V) 发送按键(K)
概况
性能
CPUs
内存
引导选项
IDE 磁盘 1
SCSI 磁盘 1
NIC :c1:cf:0d
鼠标
键盘
显示协议 Spice
Sound ich6
串口 1
信道 spice
PCI 0000:01:00.0
PCI 0000:01:00.1
显卡 QXL
Controller USB 0
Controller PCI 0
添加硬件(D)
物理 PCI 设备
设备： 0000:01:00:0 Intel Corporation 82599ES 10-Gigabit SFI/SFP+ Network Connection
ROM 栏(B)：
添加完的效果
删除(R)
取消(C)
应用(A)


KVM_Pass-through - QEMU/KVM
文件(F) 虚拟机(M) 查看(V) 发送按键(K)
概况
性能
CPUs
内存
引导选项
IDE 磁盘 1
SCSI 磁盘 1
NIC :c1:cf:0d
鼠标
键盘
显示协议 Spice
Sound ich6
串口 1
信道 spice
PCI 0000:01:00.0
PCI 0000:01:00.1
显卡 QXL
Controller USB 0
Controller PCI 0
添加硬件(D)
虚拟网络接口
网络源(N)： 桥接 br0: 主机设备 enp0s31f6
设备型号： e1000
MAC 地址： 52:54:00:c1:cf:0d
删除(R)
取消(C)
应用(A)

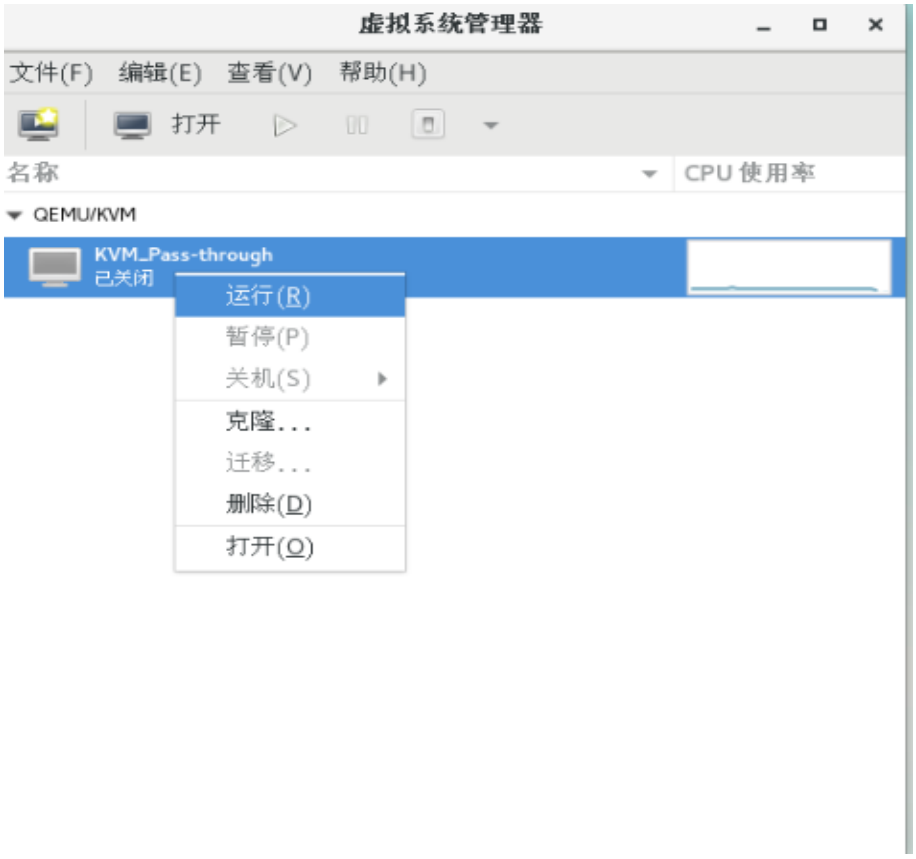
5.5 调整 CPU 和内存大小





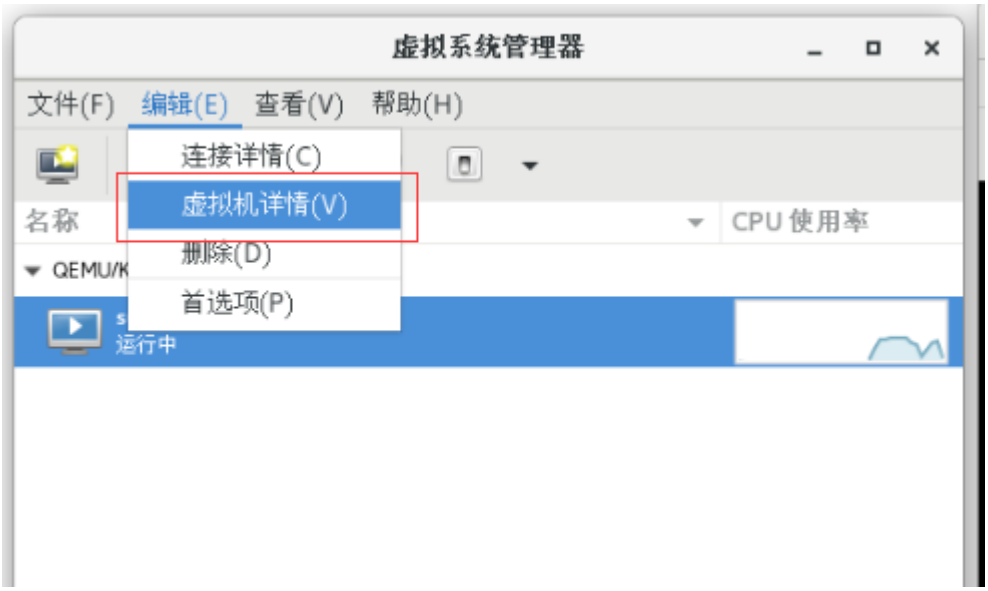
6. 运行 KVM 启动 Supernova

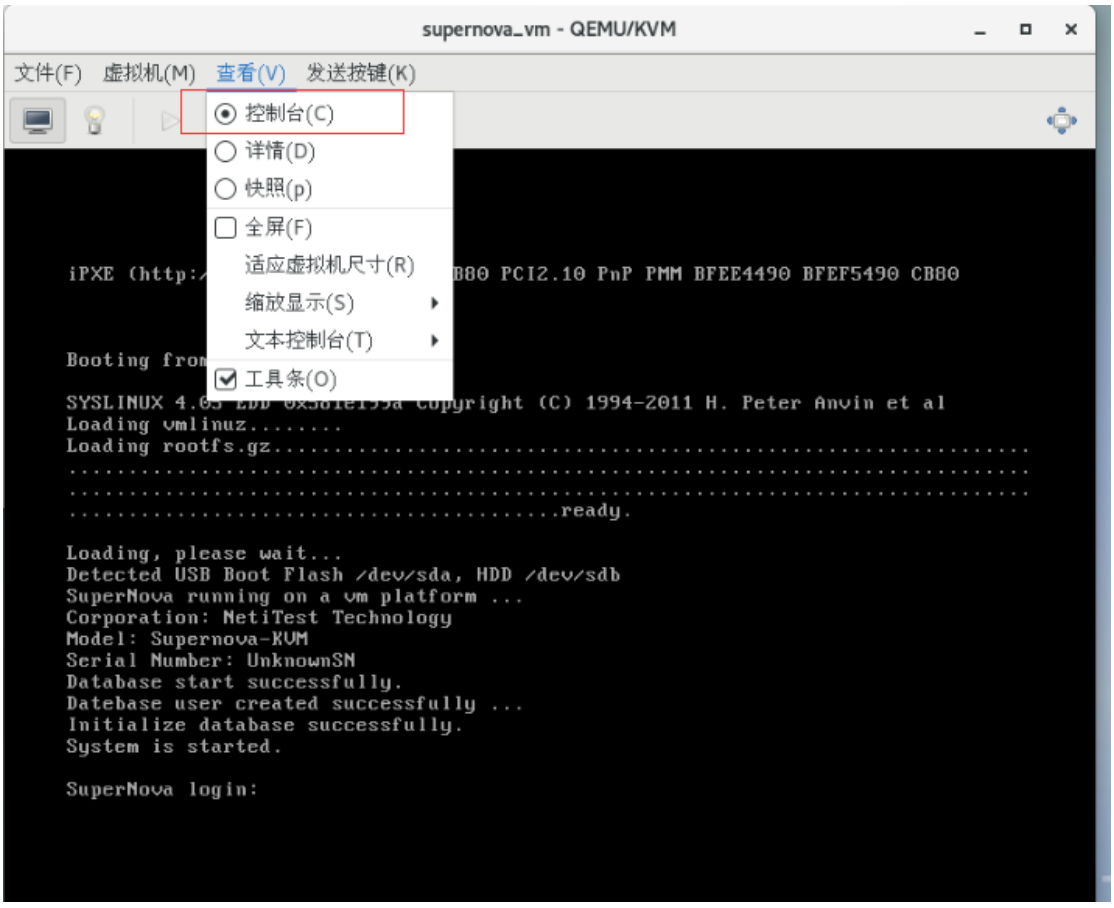
6.1 运行 KVM



注：如果启动虚拟机失败，报错为：pci…请更换宿主机网卡插槽位置；
请在 vi /etc/fstab 下增加 devpts /dev/pts devpts gid=5,mode=620 0 0
如下图：

```
[root@localhost centos_chroot]# cat /etc/fstab

#
# /etc/fstab
# Created by anaconda on Tue Jan  8 15:57:37 2019
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
/dev/mapper/centos-root /                       xfs     defaults        0 0
UUID=52cd09d9-03f7-405f-85f0-ed0eec22ee8a /boot                   xfs     defaults        0 0
/dev/mapper/centos-home /home                   xfs     defaults        0 0
/dev/mapper/centos-swap swap                    swap    defaults        0 0
devpts    /dev/pts   devpts    gid=5,mode=620   0   0
[root@localhost centos_chroot]#
```





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：

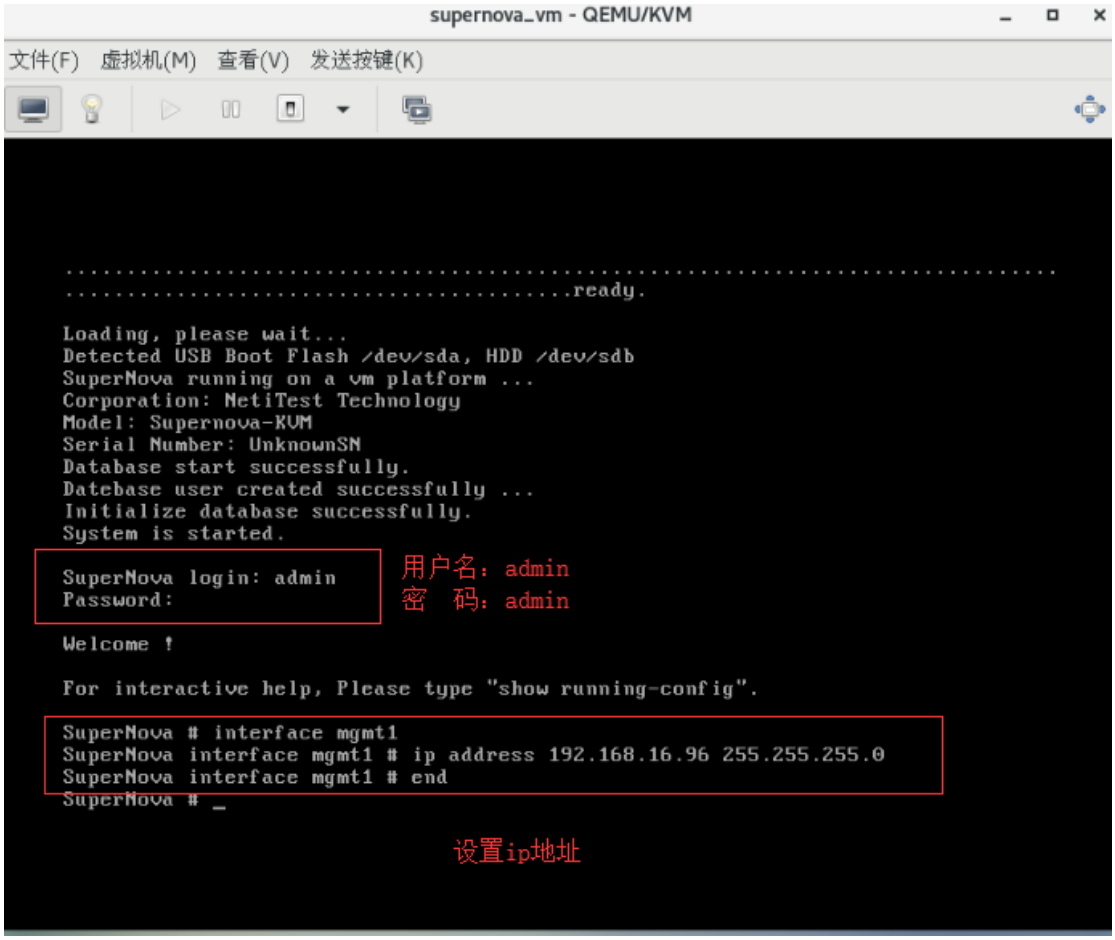
```
[root@localhost ~]# cd /etc/sysconfig/network-scripts/
[root@localhost network-scripts]# ls
ifcfg-br0           ifcfg-enp4s0f1  ifdown-ib    ifdown-ppp      ifdown-tunnel  ifup-ib    ifup-plusb  ifup-Team        network-functions
ifcfg-enp0s31f6     ifcfg-lo        ifdown-ippp  ifdown-routes   ifup           ifup-ippp  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：

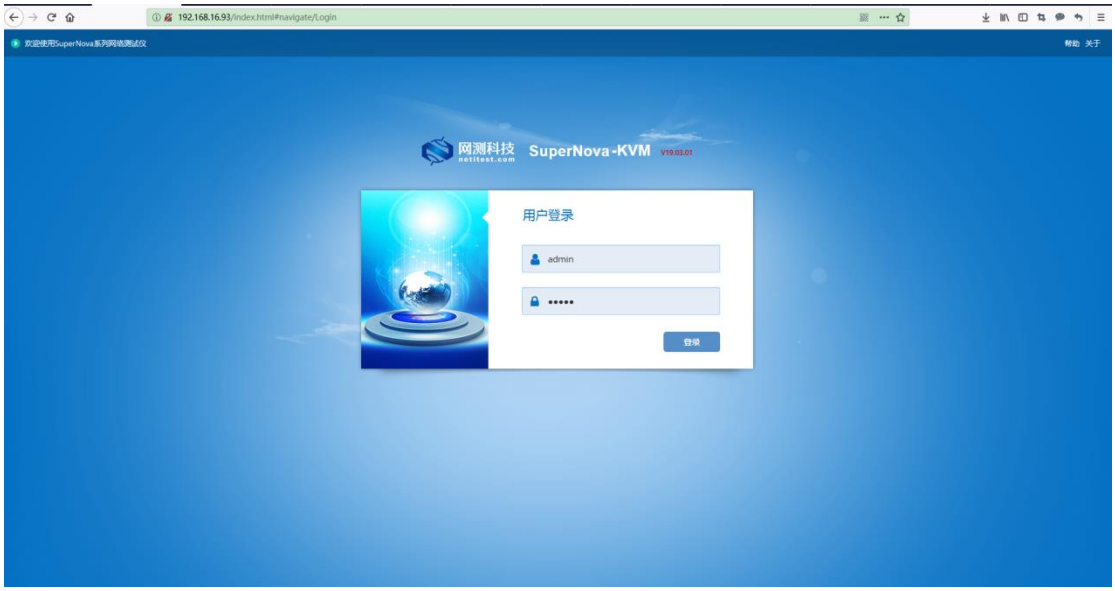
```
vi ifcfg-enp4s0f0
```

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

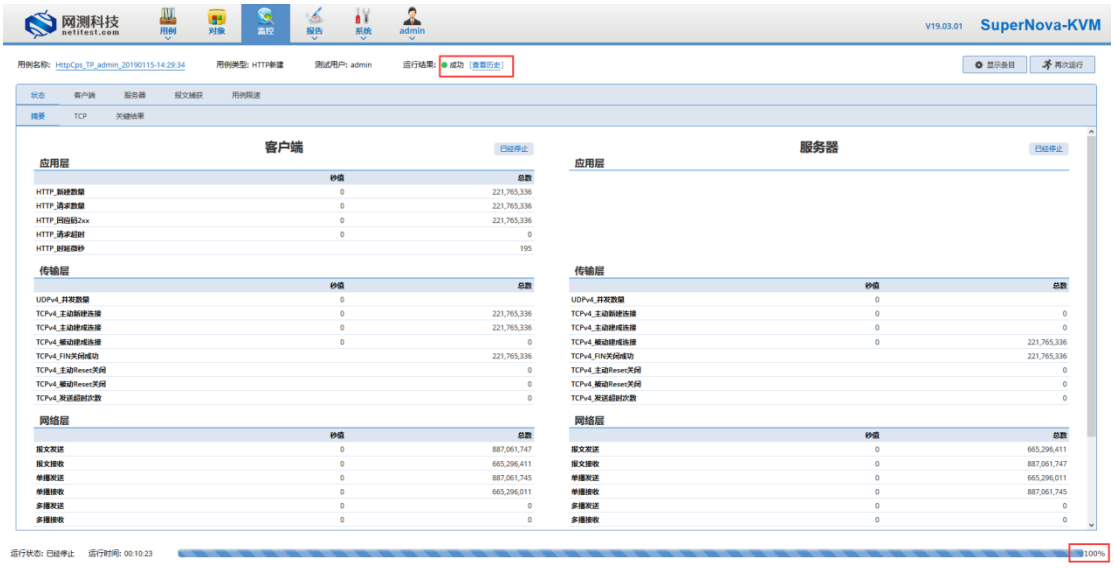
6.2 成功运行后设置 ip 地址



6.3 登陆界面



6.4 配置用例并成功运行



7 使用命令行方式部署虚拟机

7.1.创建 sriov 文件夹

[root@localhost home]# cd /home　　进入到 home 目录下
[root@localhost home]# mkdir sriov　　创建 sriov 文件夹

```
[root@localhost ~]#  cd /home
[root@localhost home]# mkdir sriov
[root@localhost home]# ls
admin  CLD  sriov
[root@localhost home]#
```

8.2 进入该目录上传镜像

NOVA_VM_CLD-HW01-v22.12.11-build2907-20230225.img.cloud.zip

[root@localhost ~]# cd /home/sriov　　进入到/home/sriov 文件夹下

[root@localhost sriov]# rz　　上传镜像文件

```
[root@localhost sriov]# 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    10242 KB/sec    00:00:59       0 Errors
```

[root@localhost sriov]# unzip NOVA_VM_CLD-HW01-v22.12.11-build2907-20230225.img.cloud.zip　　unzip 命令解压镜像

```
[root@localhost sriov]# 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
[root@localhost sriov]#
```

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 sriov]# 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 sriov]#
```

README 内容为创建数据盘命令

```
[root@localhost sriov]# 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 sriov]# 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 sriov]#
[root@localhost sriov]# ls
boot.qcow2  data.qcow2  NOVA_VM_CLD-HW01-v23.03.06-build3094-20230321-X86_64.img.cloud.zip  README
[root@localhost sriov]#
```

8.3.创建 sriov 的池并启动

[root@localhost sriov]# virsh-----进入 virsh 工具

virsh # pool-define-as sriov dir --target '/home/sriov/' -----定义池 sriov

virsh # pool-build sriov-----构建池 sriov

virsh # pool-start sriov------启动池 sriov

virsh #pool-autostart sriov------池 sriov 标记为自动启动

virsh # pool-list --all ------查看池运行状态

virsh #pool-refresh sriov------刷新池

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

virsh # pool-build sriov
构建池 sriov

virsh # pool-start sriov
池 sriov 已启动

virsh # pool-autostart sriov
池 sriov 标记为自动启动

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
 sriov                活动     是
 win02                活动     是

virsh # pool-refresh sriov
池 sriov 被刷新
```

8.4.分离网卡

8.4.1 ifconfig 查看可用网卡名称

```
enp101s0f0: flags=6211<UP,BROADCAST,RUNNING,SLAVE,MULTICAST>  mtu 1500
        ether 0c:42:a1:49:94:44  txqueuelen 1000  (Ethernet)
        RX packets 579  bytes 71796 (70.1 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 590  bytes 72824 (71.1 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp101s0f1: flags=6211<UP,BROADCAST,RUNNING,SLAVE,MULTICAST>  mtu 1500
        ether 0c:42:a1:49:94:44  txqueuelen 1000  (Ethernet)
        RX packets 579  bytes 71796 (70.1 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 581  bytes 72044 (70.3 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

8.4.2 命令含义为将实体网卡 enp101s0f0 虚拟出一个 VF；echo 2 就是虚拟出 2 个 VF;

echo 1 > /sys/class/net/enp101s0f0/device/sriov_numvfs

echo 1 > /sys/class/net/enp101s0f0/device/sriov_numvfs

[root@localhost ~]# lspci |grep Ethernet------查看虚拟出来的网卡

```
[root@localhost ~]# lspci |grep Ethernet
19:00.0 Ethernet controller: Intel Corporation Ethernet Connection X722 for 10GBASE-T (rev 09)
19:00.1 Ethernet controller: Intel Corporation Ethernet Connection X722 for 10GBASE-T (rev 09)
65:00.0 Ethernet controller: Mellanox Technologies MT27800 Family [ConnectX-5]
65:00.1 Ethernet controller: Mellanox Technologies MT27800 Family [ConnectX-5]
65:00.2 Ethernet controller: Mellanox Technologies MT27800 Family [ConnectX-5 Virtual Function]
65:01.2 Ethernet controller: Mellanox Technologies MT27800 Family [ConnectX-5 Virtual Function]
b3:00.0 Ethernet controller: Mellanox Technologies MT27800 Family [ConnectX-5]
b3:00.1 Ethernet controller: Mellanox Technologies MT27800 Family [ConnectX-5]
```

8.4.3 分离虚拟网卡

```
virsh nodedev-detach pci_0000_65_00_2
virsh nodedev-detach pci_0000_65_01_2
```

```
[root@localhost ~]# virsh nodedev-detach pci_0000_65_00_2
已分离设备 pci_0000_65_00_2

[root@localhost ~]# virsh nodedev-detach pci_0000_65_01_2
已分离设备 pci_0000_65_01_2
```

注网卡分离和取消分离的命令为

virsh nodedev-detach pci_0000_网卡　　分离网卡

virsh nodedev-reattach pci_0000_网卡　　取消分离

8.5.上传虚拟机的 xml 文件

sr-iov.xml

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

rz 上传 sr-iov.xml

```
[root@localhost qemu]# rz
rz waiting to receive.
Starting zmodem transfer.  Press Ctrl+C to cancel.
Transferring sr-iov.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/sr-iov.xml
```

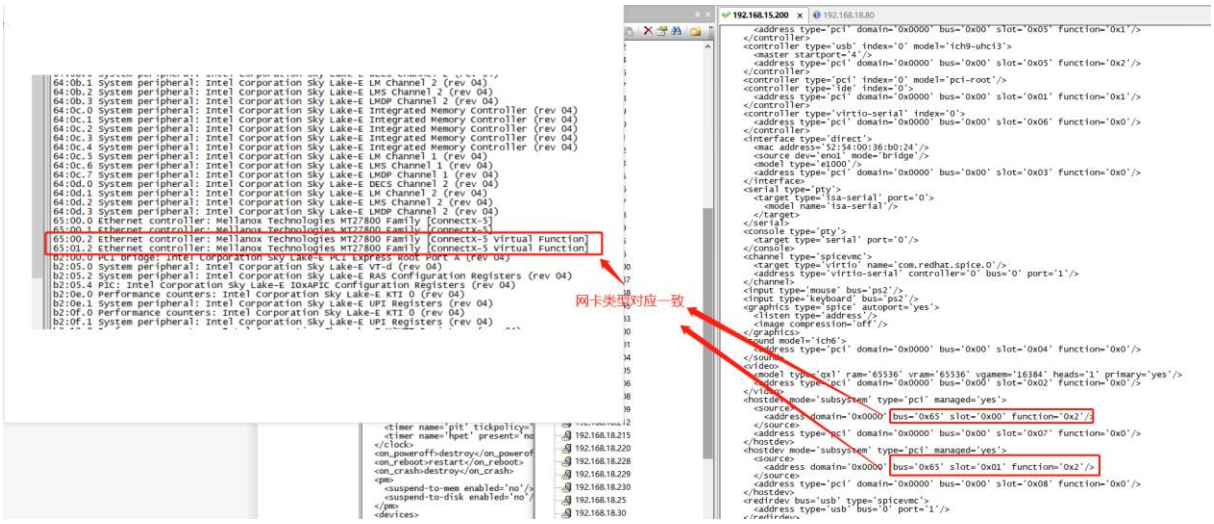
```
<domain type='kvm'>
  <name>sr-iov</name>
  <uuid>a9f99618-0589-4180-9007-8c4341f34f84</uuid>
  <memory unit='KiB'>8388608</memory>
  <currentMemory unit='KiB'>8388608</currentMemory>
  <vcpu placement='static'>4</vcpu>
  <os>
    <type arch='x86_64' machine='pc-i440fx-rhel7.0.0'>hvm</type>
    <boot dev='hd'/>
  </os>
  <features>
    <acpi/>
    <apic/>
  </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/sriov/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/sriov/data.qcow2'/>
      <target dev='hdb' bus='ide'/>
      <address type='drive' controller='0' bus='0' target='0' unit='1'/>
    </disk>
```

名称和xml文件名称一致

内存大小

核数

系统盘和数据盘路径和名称要与之前解压出来的一致



8.6 启动虚拟机

[root@localhost qemu]# virsh define sr-iov.xml ----导入虚拟机配置

[root@localhost qemu]# virsh start sr-iov-----启动虚拟机

[root@localhost ~]# virsh list -----查看虚拟机运行状态

[root@localhost qemu]# virsh console sr-iov----连接到虚拟机，登陆用 admin/admin.

```
[root@localhost ~]# virsh define novavm.xml
定义域 novavm（从 novavm.xml）

[root@localhost ~]# virsh start novavm
域 novavm 已开始

[root@localhost ~]# virsh list
 Id    名称                         状态
----------------------------------------------------
 6     novavm                         running

[root@localhost ~]# virsh console novavm
连接到域 novavm
换码符为 ^]

kvm: no hardware support

Loading, please wait...
mount: no /proc/mounts
Detected USB Boot Flash /dev/sda, Log Disk /dev/sdb
Supernova is formatting log disk ...
Error: /dev/sdb: unrecognised disk label
sh: 32.2: bad number
Make msdos lable on /dev/sdb ... Done
Delete all partition on /dev/sdb ... Done
Create Swap partition on /dev/sdb1 ... Done
Create log partition on disk /dev/sdb2  ... Done
Format log disk /dev/sdb2 ... Done
Log disk format successfully.
mount -t hugetlbfs nodev /mnt/huge ... Done
mount --make-shared /mnt/huge/ ... Done
mount -t hugetlbfs hugetlbfs /dev/hugepages ... Done
Generate SSL DH parameters ... Done
Netitest running on a vm platform ...
Corporation: Netitest Technology
Model: Supernova-KVM
Serial Number: UnknownSN
Database start successfully.
Database root user is not exists, creating users!
Datebase user created successfully.
Initialize database successfully.
Http Web service is started.
Https Web service is started.
Version: 22.09.08 build2604
Build date: 20220908
kernel version: 5.10.112
System is started.

Supernova login:
```

启动完成

8.7 登陆虚拟机后配置虚拟机 ip 和网关

Supernova # interface mgmt1
Supernova interface mgmt1 # ip address 192.168.18.205 255.255.255.0
Supernova interface mgmt1 # end
Supernova # unset ip route 0.0.0.0/0 192.168.1.1 mgmt1
Supernova # ip route 0.0.0.0/0 192.168.18.1 mgmt1 default

```
Supernova login: admin
Password:

Welcome !

For interactive help, Please type "show running-config".

Supernova # show running-config
show running-config                 show running-config.
show system status                  System status.
show interface firmware             Interface firmware version.
show hardware info                  Hardware information.
show system interface               Show network interfaces and configurations.
show network route                  Show default route.
show system setting                 Show system setting.
show system memsize                 Show memory total size.
show timezone help                  Display support time zone options.
interface mgmt1                     Config mgmt1
interface mgmt2                     Config mgmt2
ip route                            Config route
unset ip route                      Delete route
hostname                            Config hostname
ping                                PING command.
date <mm/dd/yyyy hh:mm:ss>          Set date.
date <mm/dd/yyyy hh:mm:ss Zone>     Set date include time zone.(Use: show timezone help)
reboot                              Reboot SuperNova.
shutdown                            Shutdown SuperNova.
factoryreset                        Factoryreset SuperNova.
formatlogdisk                       Format storage.
exit                                Exit the CLI.

network config usage:
interface mgmt1
ip address 192.168.1.99 255.255.255.0
end
interface mgmt2
ip address 10.1.1.99 255.255.255.0
end

route config usage:
ip route 0.0.0.0/0 192.168.1.1 mgmt1 default
ip route 10.1.0.0/16 10.1.1.1 mgmt2

unset ip route x.x.x.0/24 x.x.x.1 mgmt1

hostname config usage:
hostname <string>

web server port usage:
http port 80
https port 443

Supernova # interface mgmt1
Supernova interface mgmt1 # ip address 192.168.18.205 255.255.255.0
Supernova interface mgmt1 # end
Supernova # unset ip route 0.0.0.0/0 192.168.1.1 mgmt1
Supernova # ip route 0.0.0.0/0 192.168.18.1 mgmt1 default
Supernova #
```

配置ip和网关后，可以在浏览器直接登陆测试仪了

8.8 通过 web 登录测试仪

9. 附加：VNC 配置方法

9.1 安装软件包

命令:
yum check-update

yum install -y tigervnc-server

yum groupinstall "X Window System"

yum groupinstall "fonts"

yum install gnome-classic-session gnome-terminal nautilus-open-terminal control-center liberation-mono-fonts

unlink /etc/systemd/system/default.target

ln -sf /lib/systemd/system/graphical.target /etc/systemd/system/default.target

9.2 关闭防火墙

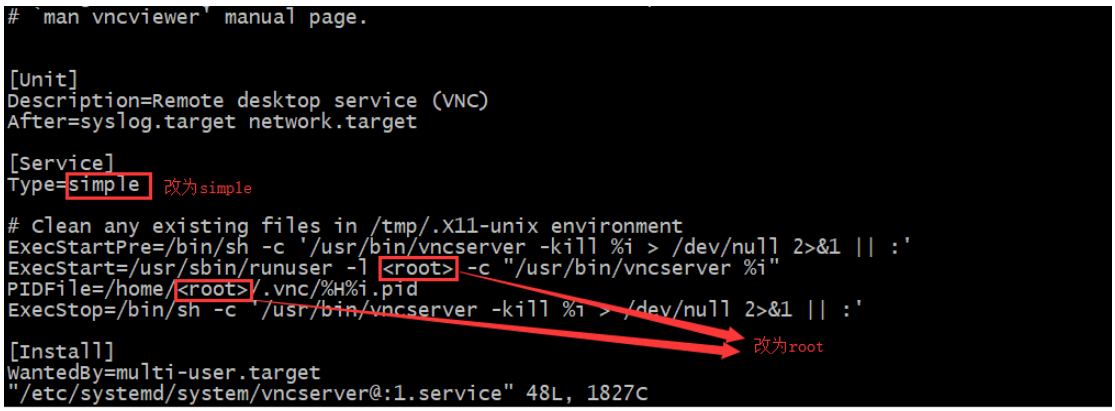
命令：systemctl stop firewalld
systemctl disable firewalld
systemctl status firewalld ----查看防火墙状态

9.3 复制配置文件

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

9.4 编辑复制出来的配置文件

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



9.5 重新加载配置文件

命令：systemctl daemon-reload

9.6 设置 VNC 密码

命令：vncpasswd root

9.7 开启 VNC 并设置成开机启动

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

9.8 启动

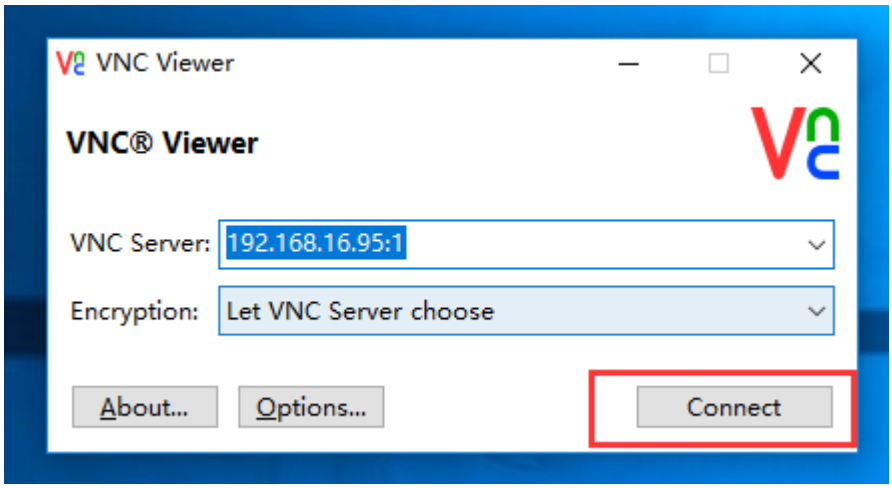
命令：vncserver

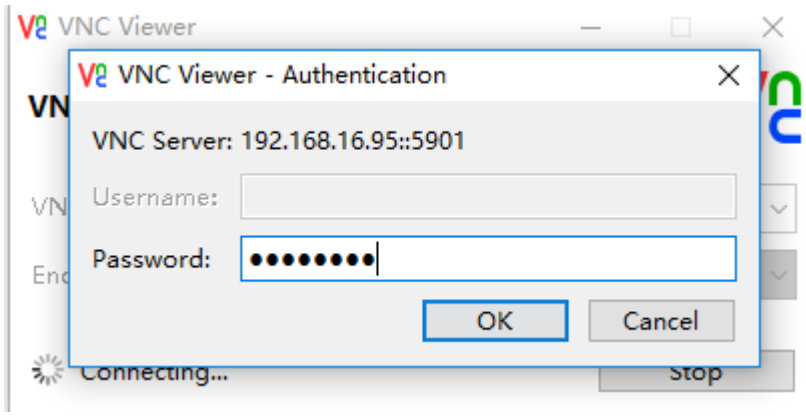
9.9 如果启动成功端口是监听状态(VNC 端口号默认 5900+1)

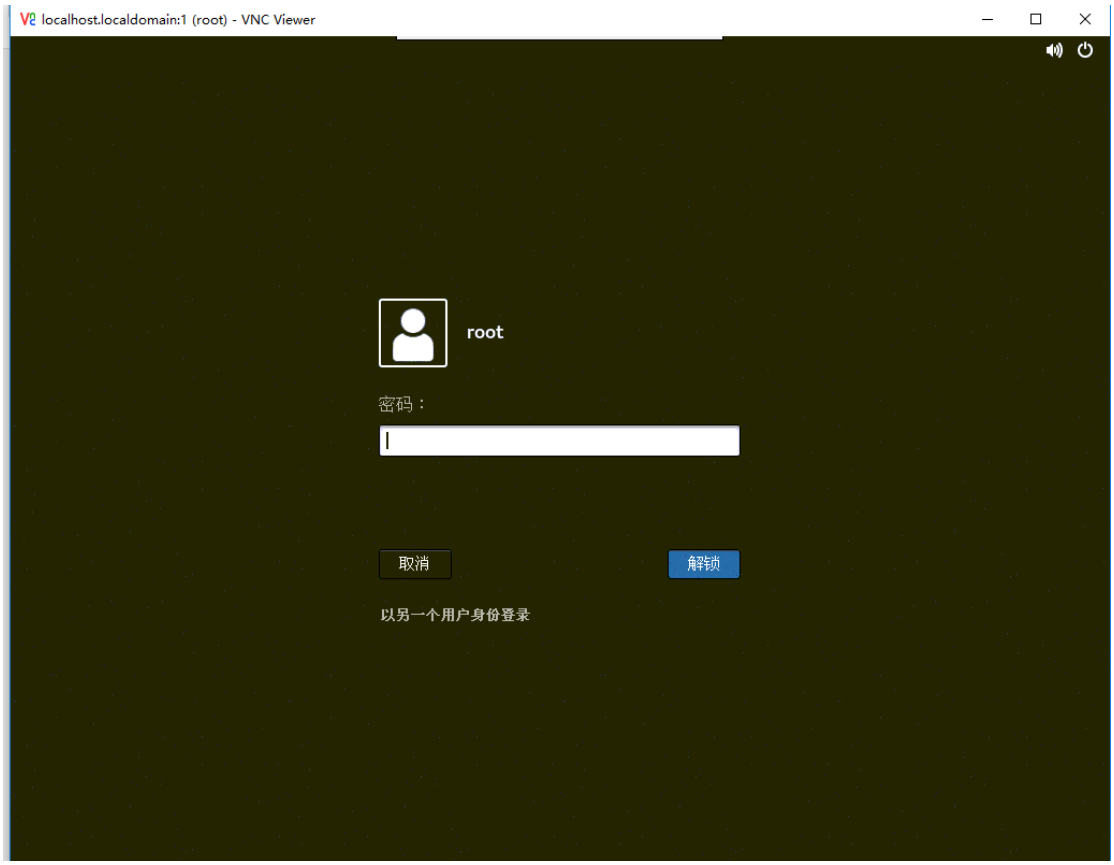
命令：netstat -an |grep 5901

```
[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 ~]#
```

9.10 vnc 客户端连接







查看运行状态

```
# systemctl status vncserver@:.service
# systemctl is-enabled vncserver@.service
```

vncserver 启动失败的一种情况

```
#vncserver

Warning: localhost.localdomain:1 is taken because of /tmp/.X11-
unix/X1
Remove this file if there is no X server localhost.localdomain:1

Warning: localhost.localdomain:2 is taken because of /tmp/.X11-
unix/X2
Remove this file if there is no X server localhost.localdomain:2

New ‘localhost.localdomain:3 (root)‘ desktop is
localhost.localdomain:3

Starting applications specified in /root/.vnc/xstartup
Log file is /root/.vnc/localhost.localdomain:3.log
```

解决办法：

删除 / tmp/.X11-unix/的内容

重启 vnvserver

```
#  systemctl restart vncserver@:2.service
```

Linux 删除文件夹和文件的命令

-r 就是向下递归，不管有多少级目录，一并删除
-f 就是直接强行删除，不作任何提示的意思

删除文件夹实例：
rm -rf /var/log/httpd/access
将会删除/var/log/httpd/access 目录以及其下所有文件、文件夹

删除文件使用实例：
rm -f /var/log/httpd/access.log
将会强制删除/var/log/httpd/access.log 这个文件