

# 一、物理机安装依赖包

以 Armbian/Debian/ubuntu 为例（其它操作系统请查询对应的命令）

首先验证物理机是否支持 kvm 虚拟化：

```
ls -l /dev/kvm
```

```
dmesg | grep kvm
```

```
root@gtking-pro:~# ls -l /dev/kvm
crw-rw----+ 1 root kvm 10, 232  7月 23 00:25 /dev/kvm
root@gtking-pro:~# dmesg|grep kvm
[    0.632204] kvm [1]: IPA Size Limit: 40 bits
[    0.632940] kvm [1]: vgic interrupt IRQ9
[    0.633098] kvm [1]: Hyp mode initialized successfully
root@gtking-pro:~#
```

如果结果如上图，那 kvm 支持就没问题，否则一般是物理机的内核没开启 kvm 支持（需要重新换个支持 kvm 的内核），或者是物理机根本就不支持 kvm！

按照 arm 的官方文档，cortex-a53 以上的 cpu 都是支持 kvm 的。

```
apt install gconf2 qemu-system qemu-system-arm qemu-utils  qemu-efi libvirt-daemon-system libvirt-clients bridge-utils virtinst virt-manager seabios vgabios  gir1.2-spiceclientgtk-3.0 xauth
```

x11 字库（可选）

```
apt install fonts-noto*
```

桌面环境(可选)

```
apt install tasksel
```

运行 tasksel，选择至少一个桌面环境即可

## 二、windows 客户机安装 ssh 客户端及 x11 server

### 2.1 Armbian(linux)端 (ssh 服务端+x11 客户端)

首先，确认远程服务器上的 SSH 服务端开启了 X11Forwarding 功能(默认开启):

```
# 编辑 /etc/ssh/sshd_config 文件

X11Forwarding yes

# 如果之前未开启，保存配置文件后重启 sshd

systemctl restart sshd
```

接下来，远程服务器上安装 `xauth` 包。**如果远程服务器安装时带桌面环境，`xauth` 包已经默认安装**，可以跳过这一步。无界面版的远程服务器需手动安装：

```
# CentOS

yum install -y xorg-x11-xauth

# Debian/Ubuntu

apt install -y xauth
```

## 2.2 Windows/Linux/MacOS 端 (ssh 客户端 + x11 服务端)

接着，在本地电脑上安装 X Server 程序。

运行带桌面环境的 Ubuntu、[Debian](#)、Fedora、[CentOS](#) 等 Linux 发行版的本地电脑，已经自带 X Server，可以略过这一步。

[Windows](#)、[MacOS](#) 系统需要自行下载 X Server 程序：MacOS 可到 <https://www.xquartz.org/> 下载 XQuartz 程序，Windows 可到 <https://sourceforge.net/projects/vcxsrv/> 下载 VcXsrv，或到 <https://sourceforge.net/projects/xming/> 下载 Xming，安装并运行 X Server 程序。

Ssh 客户端是 linux 时，开启 **X11 Forwarding** 选项，ssh 连接到远程服务器，运行 GUI 程序：

```
# -X 选项开启 X11 Forwarding
```

```
ssh -X user@host
```

```
# 运行远程 GUI 程序，界面将在本地电脑上显示出来
```

```
virt-manager
```

ssh 客户端是 windows 时，ssh 工具可以用 putty、xshell、securecr 等等，x11 server 可以用 xshell 自带的，或者 xming、vcxsrv、cygwin x11 等。

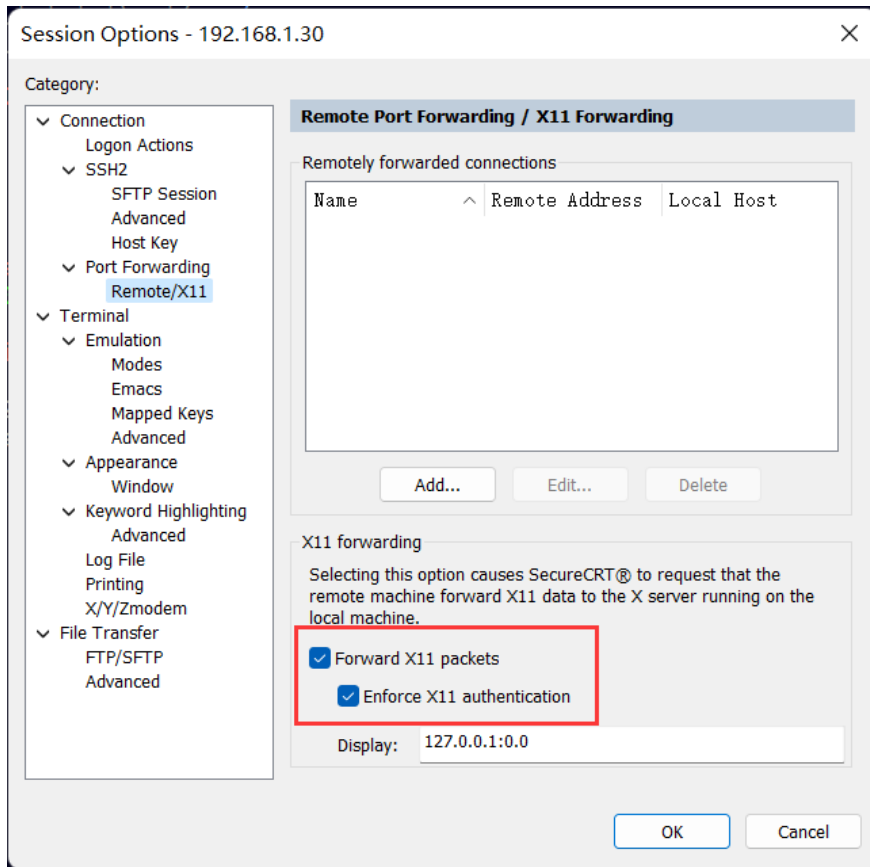
以 securecr+xming 为例，先双击 Xming 图标



启动之后，图标在右下角（需要防火墙入站规则允许 xming）



然后打开 Securecr session 配置，按图开启 forward x11 选项



接下来，ssh 连接到 linux 服务器（即 armbian），在#提示符下运行 `virt-manager`，并等待十多秒钟，就会看到虚拟机管理的图形界面了。



### 三、物理机网络配置

注意：如果物理机只有单网卡的话，要把网络改成桥接，以便与虚机共用网卡。

以 armbian/Debian/ubuntu 为例：（其它操作系统请自行查询网络配置方式）

文件名：/etc/network/interfaces.d/br0

```
# eth0 setup
```

```
allow-hotplug eth0
```

```
iface eth0 inet manual
```

```
    pre-up    ifconfig $IFACE up
```

```
    pre-down ifconfig $IFACE down
```

```
# Bridge setup
```

```
auto br0
```

```
iface br0 inet static
```

```
    bridge_ports eth0
```

```
    bridge_stp off
```

```
    bridge_waitport 0
```

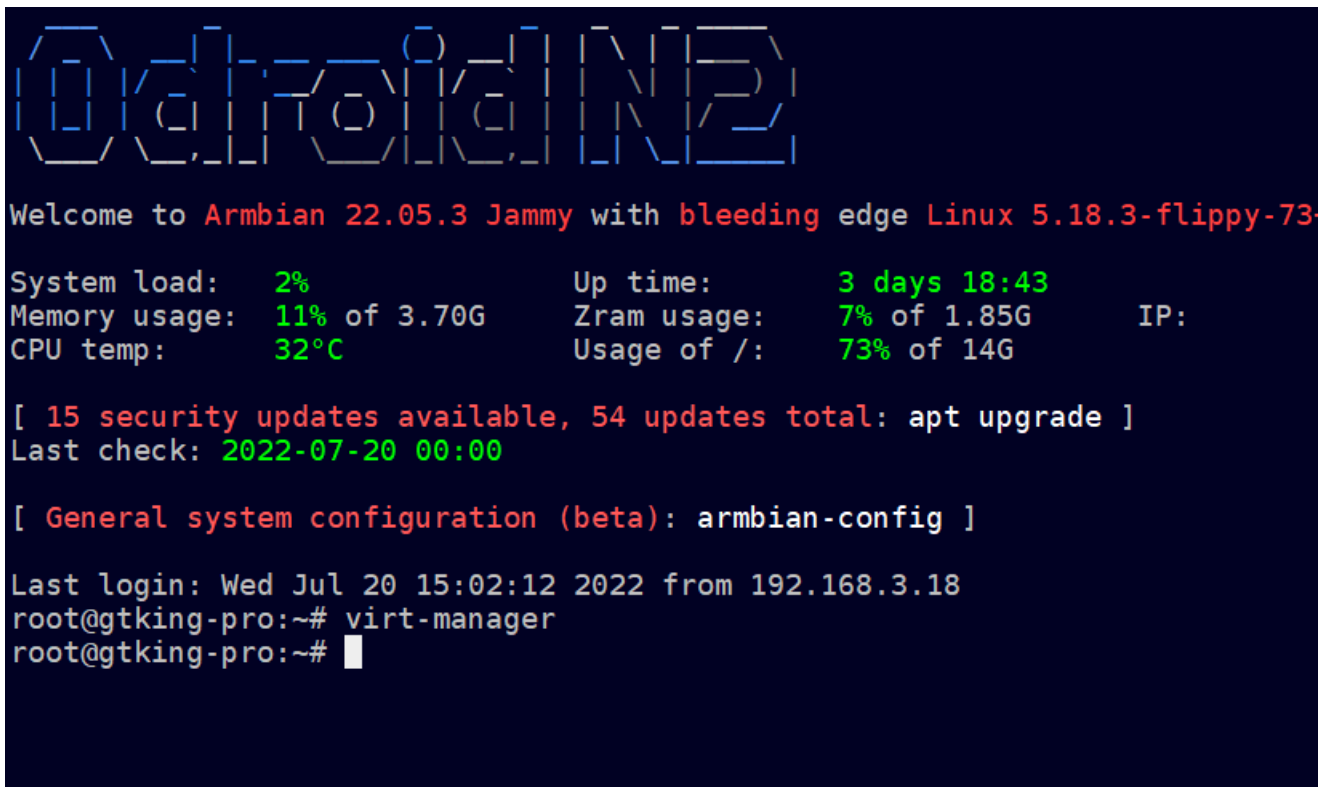
```
    bridge_fd 0
```

```
    address 192.168.3.22
```

```
broadcast 192.168.3.255
netmask 255.255.255.0
gateway 192.168.3.1
dns-nameservers 192.168.3.1
```

## 四、安装过程截图

qemu 的固件镜像后缀是 .qcow2，把镜像上传到物理机的 /var/lib/libvirt/images/目录下，名字可以任意改。  
运行 virt-manager （或桌面环境下点击“虚拟机管理”图标）

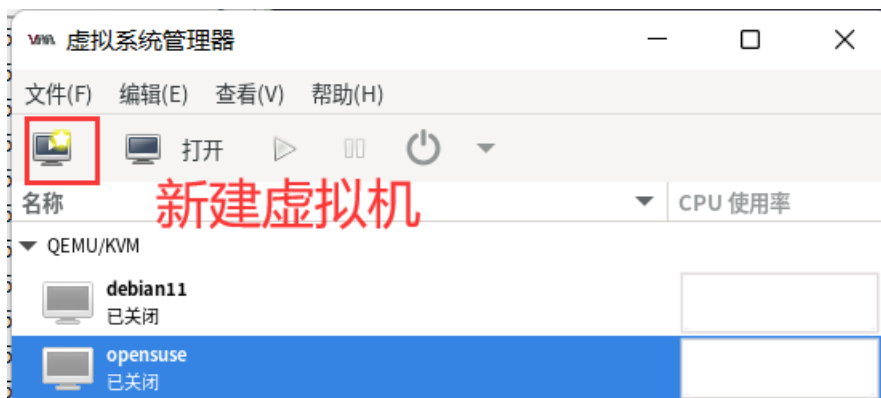


```
OdroidNE
Welcome to Armbian 22.05.3 Jammy with bleeding edge Linux 5.18.3-flippy-73
System load:  2%           Up time:      3 days 18:43
Memory usage: 11% of 3.70G  Zram usage:  7% of 1.85G   IP:
CPU temp:     32°C         Usage of /:   73% of 14G

[ 15 security updates available, 54 updates total: apt upgrade ]
Last check: 2022-07-20 00:00

[ General system configuration (beta): armbian-config ]

Last login: Wed Jul 20 15:02:12 2022 from 192.168.3.18
root@gtking-pro:~# virt-manager
root@gtking-pro:~#
```



新建虚拟机

×

 创建新虚拟系统

步骤 1, 共 4 步

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 (HTTP、HTTPS 或 FTP) (I)

☒ 导入现有磁盘映像(E)

☐ 手动安装(N)

▼ 架构选项

虚拟类型(V):

KVM

▼

架构(A):

aarch64

▼

机器类型(M):

virt

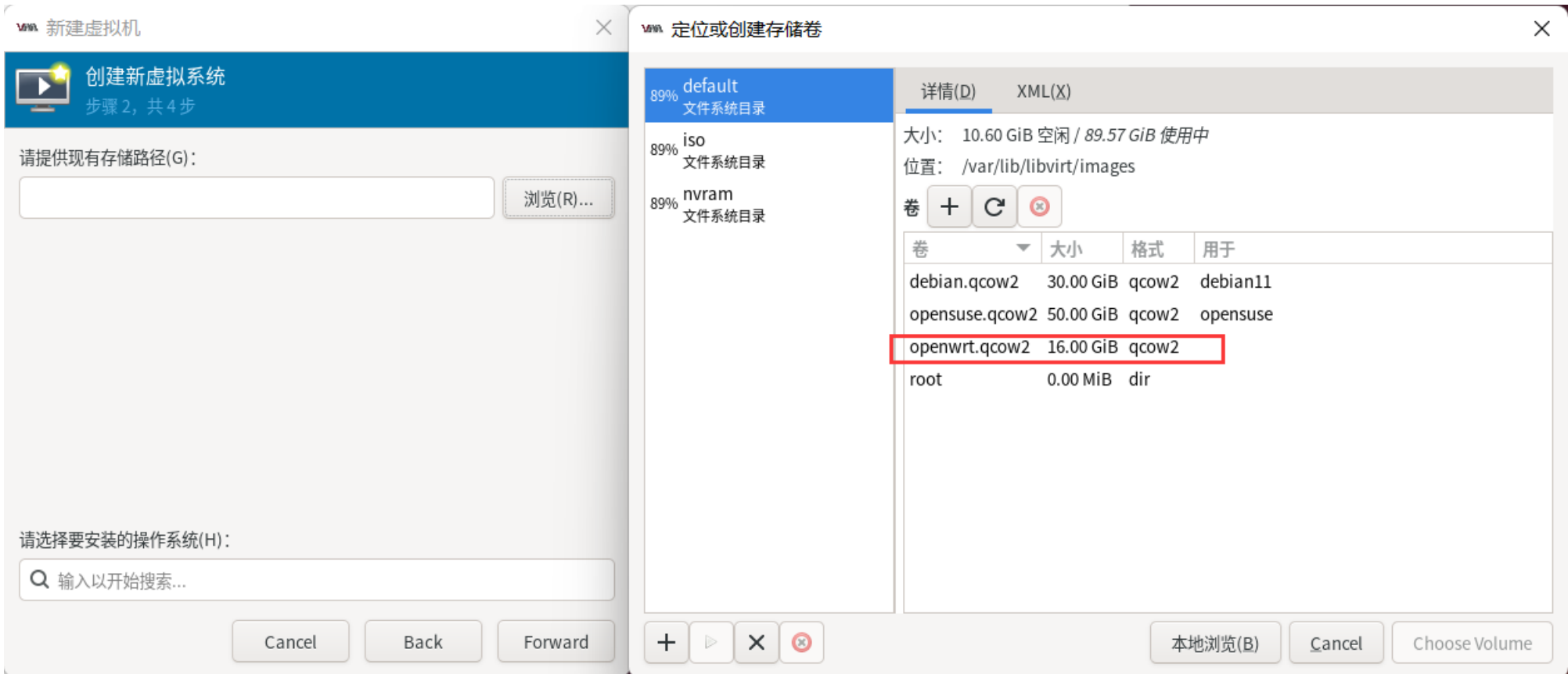
▼

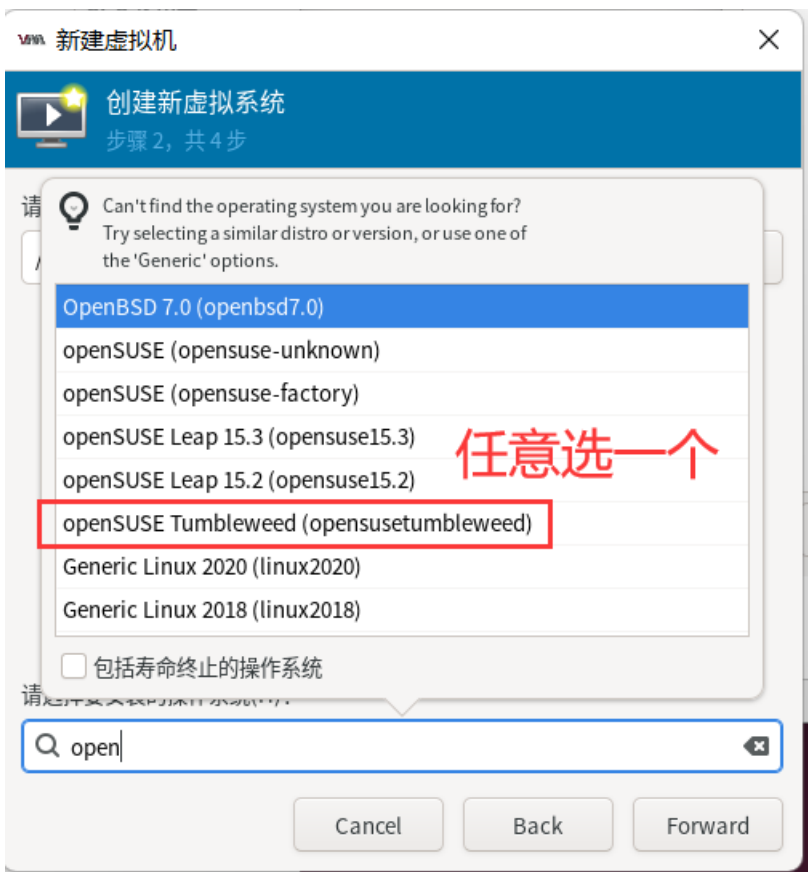
Cancel

Back

Forward







VMware

新建虚拟机

×



创建新虚拟系统

步骤 3, 共 4 步

请选择内存和 CPU 设置:

内存(M):

2048

—

+

主机中最多有 3784 MiB 可用

CPU 数(P):

2

—

+

最多可用 6

Cancel

Back

Forward

新建虚拟机

创建新虚拟系统  
步骤 4, 共 4 步

准备开始安装

名称(N):

openwrt

操作系统:

openSUSE Tumbleweed

安装:

导入现有操作系统映像

内存:

2048 MiB

CPU 数:

2

存储:

.../lib/libvirt/images/openwrt.qcow2

☒ 在安装前自定义配置(U)

选择网络(E)

桥接设备...

设备名称(V):

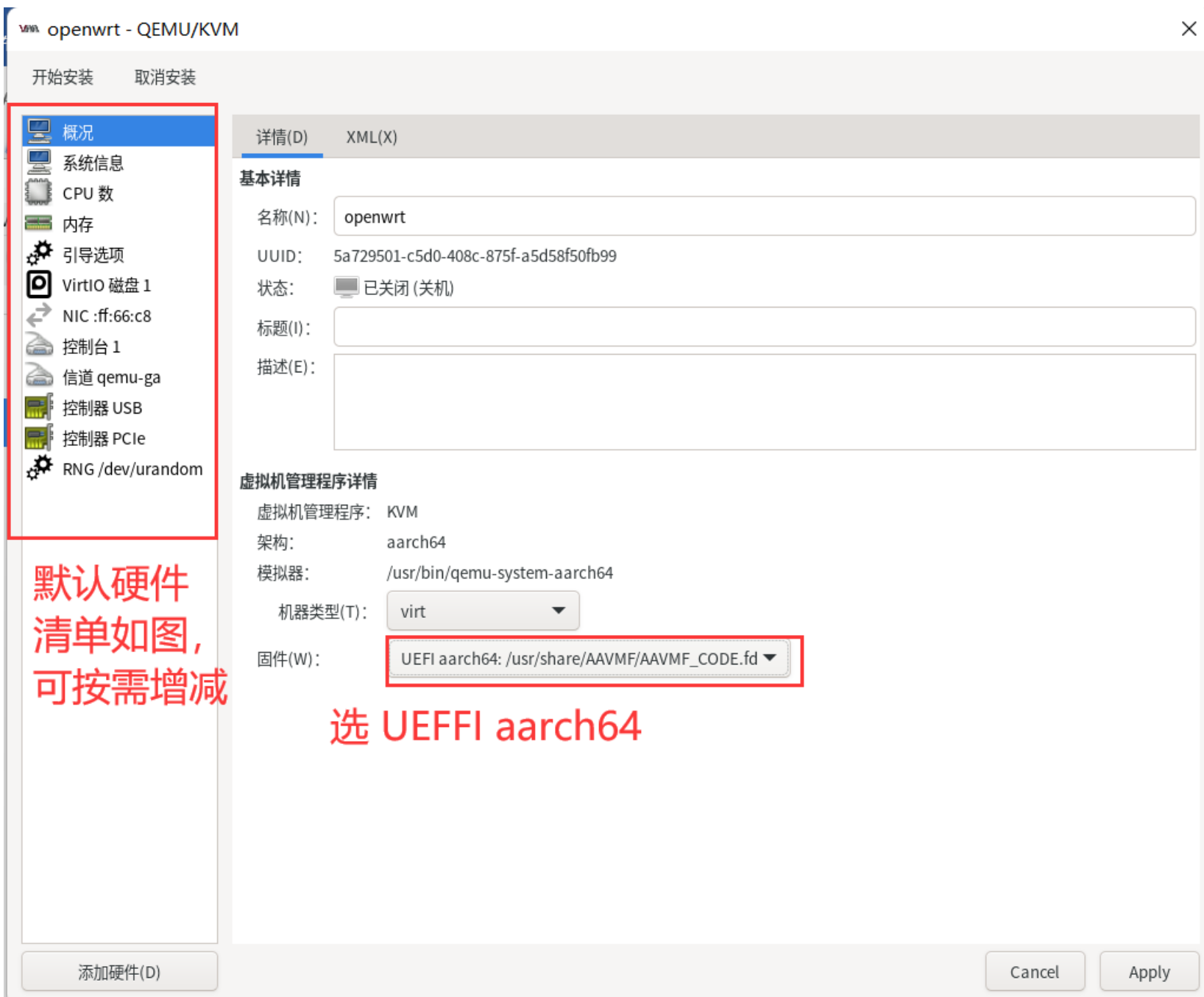
br0

共享网口选桥接，独立网口选macvtap

Cancel

Back

完成(F)



默认硬件  
清单如图,  
可按需增减

选 UEFI aarch64

开始安装

取消安装

- 概况
- 系统信息
- CPU 数
- 内存
- 引导选项
- VirtIO 磁盘 1
- NIC :ff:66:c8**
- 控制台 1
- 信道 qemu-ga
- 控制器 USB
- 控制器 PCIe
- RNG /dev/urandom

详情(D) XML(X)

## 虚拟网络接口

网络源(N): 桥接设备...

设备名称(V): br0

设备型号: virtio

MAC 地址: 52:54:00:ff:66:c8

IP 地址(P): 未知

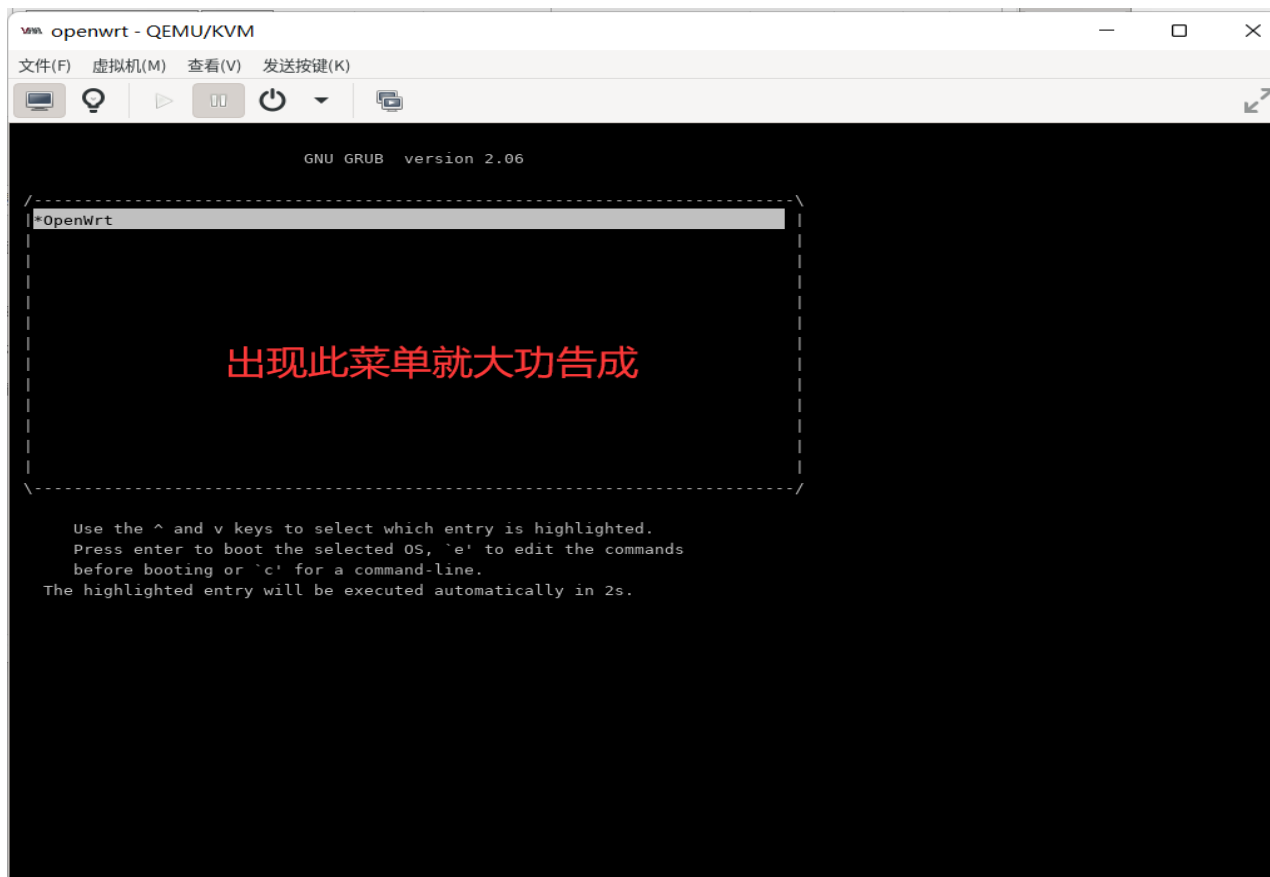
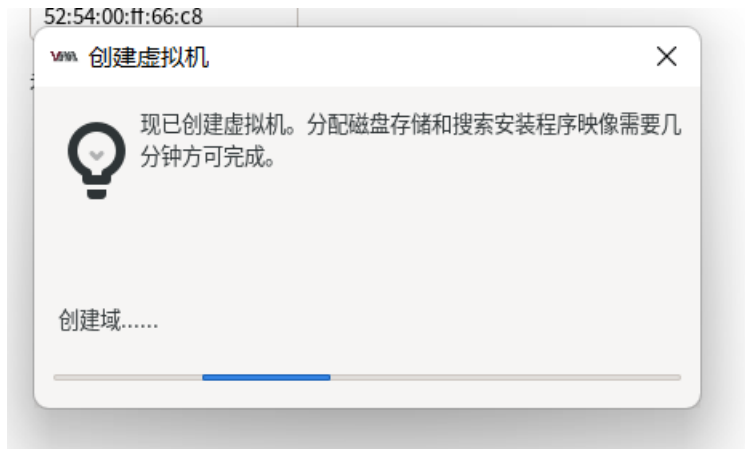
链接状态(S): ☒ 活动

添加硬件(D)

Remove

Cancel

Apply



openwrt - QEMU/KVM

文件(F) 虚拟机(M) 查看(V) 发送按钮(K)



```
[ 10.989228] usbcore: registered new interface driver carl9170
[ 10.993176] usbcore: registered new interface driver mt76x0u
[ 10.996565] usbcore: registered new interface driver rt2500usb
[ 11.001645] usbcore: registered new interface driver rt2800usb
[ 11.007138] usbcore: registered new interface driver rtl8192cu
[ 11.011498] usbcore: registered new interface driver ath9k_htc
[ 11.014490] kmodloader: done loading kernel modules from /etc/modules.d/*
[ 12.691011] xt_FULLCONENAT: RFC3489 Full Cone NAT module
[ 12.691011] xt_FULLCONENAT: Copyright (C) 2018 Chion Tang <tech@chionlab.moe>
[ 13.586476] 8021q: adding VLAN 0 to HW filter on device eth0
[ 35.424246] NFSD: Using /var/lib/nfs/v4recovery as the NFSv4 state recovery directory
[ 35.425532] NFSD: Using legacy client tracking operations.
[ 35.426248] NFSD: starting 10-second grace period (net f0000000)
```

```
Base on OpenWrt R22.6.16 by lean & lienol
Kernel 5.18.11-flippy-74+
Packaged by flippy on 2022-07-15
PLATFORM: qemu-aarch64 SOC: generic BOARD: vm
```

```
设备信息: KVM Virtual Machine
CPU 型号: ARMv8 Processor rev 2 (v8l) x 2
系统负载: 0.24 0.07 0.02          运行时间: 0分钟 54秒
内存已用: 7% of 1981MB            IP 地址: 192.168.3.27
启动存储: 56% of 1.0G             系统存储: 56% of 1.0G
```

```
root@vm27:/#
```





```
config interface 'loopback'
    option ifname 'lo'
    option proto 'static'
    option ipaddr '127.0.0.1'
    option netmask '255.0.0.0'

config globals 'globals'
    option ula_prefix 'fd3a:4d2a:6376::/48'

config interface 'lan'
    option ifname 'eth0'
    option proto 'static'
    option ipaddr '192.168.3.27'
    option netmask '255.255.255.0'
    option gateway '192.168.3.18'
    option broadcast '192.168.3.255'
    option dns '192.168.3.18'

config interface 'VPN'
    option ifname 'ipsec0'
    option proto 'static'
    option ipaddr '10.10.10.1'
```

13,22-29

Top

修改/etc/config/network  
然后运行service network restart即可联网

## 状态

## 系统

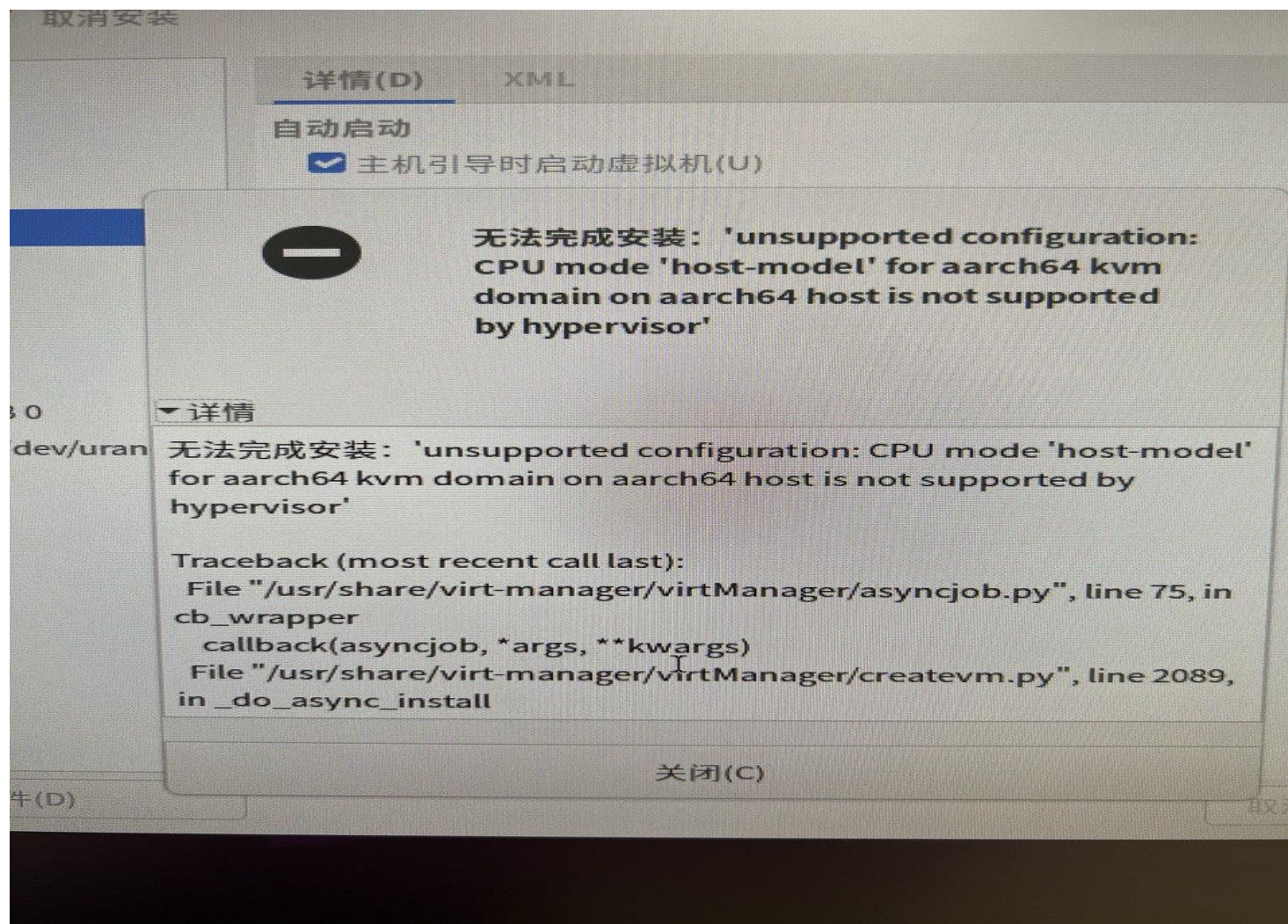
主机名	vm27
型号	KVM Virtual Machine
	CPU CoreMark 24876.008644
	aes-128-gcm(1K) 2274377.73k
	aes-256-gcm(1K) 1942573.06k
	chacha20-poly1305(1K) 732791.47k
架构	ARMv8 Processor rev 2 x 2
固件版本	OpenWrt R22.6.16 (2022-07-15 23:27:16 by flippy) / LuCI Master (git-22.193.59890-c3e15ff)
内核版本	5.18.11-flippy-74+
本地时间	Wed Jul 20 15:39:25 2022
运行时间	0h 2m 29s
平均负载	0.23, 0.10, 0.03
CPU 使用率 (%)	5 %

## 内存

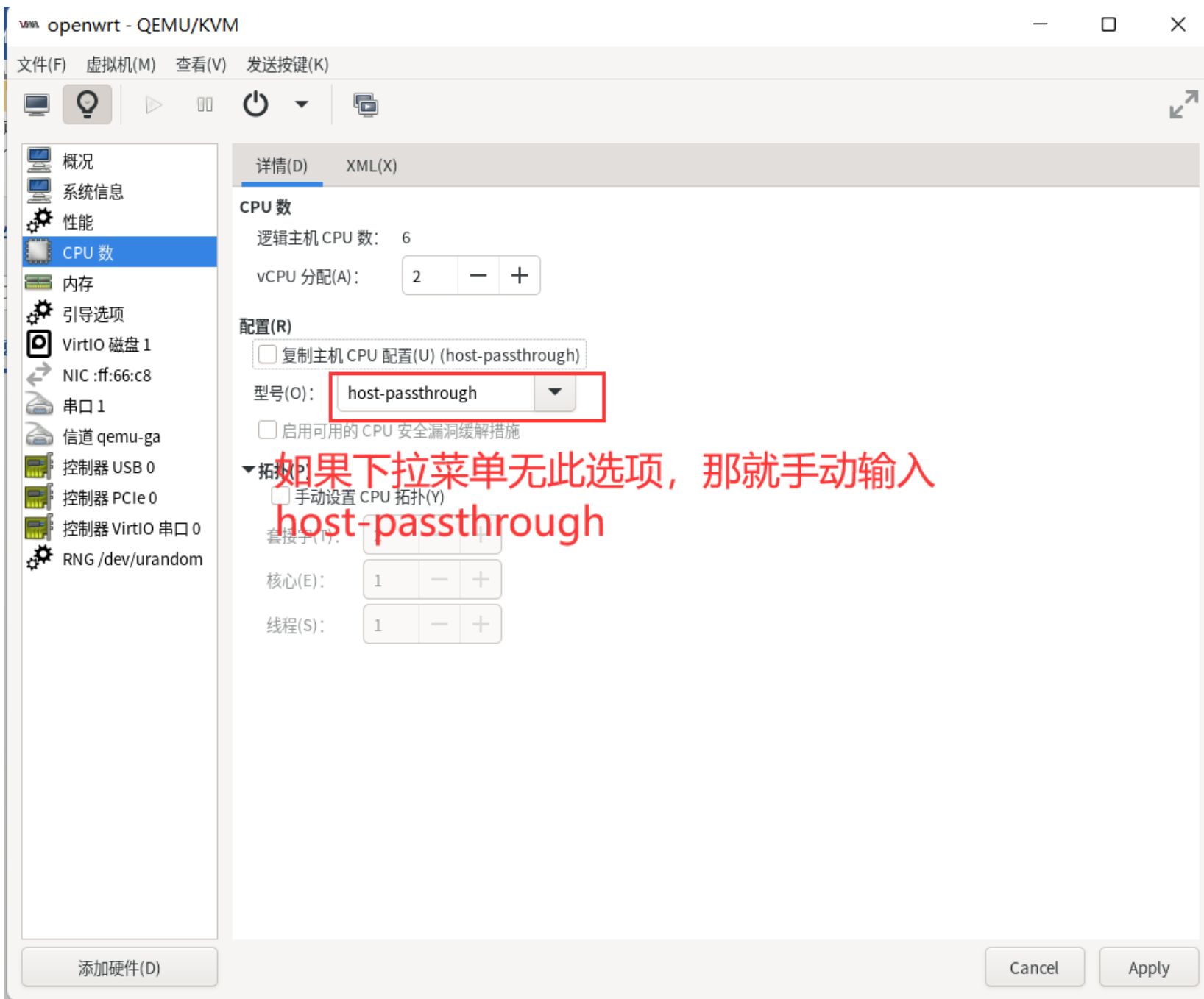
可用数	1801 MB / 1981 MB (90%)
已缓存	2 MB / 1981 MB (0%)

## 五、故障处理

### 5.1 cpu 模式不对：提示 cpu mode 'host-model' not supported



解决方法：把 cpu 模式手动改成 host-passthrough (如果下拉选项没这个，那就手动输入)



## 5.2 虚拟机服务未启动

systemctl status libvirt

正常情况应该这样：

```
root@gtk-pro:/etc/libvirt/qemu# systemctl status libvirt
● libvirtd.service - Virtualization daemon
   Loaded: loaded (/lib/systemd/system/libvirtd.service; enabled; vendor preset: enabled)
   Active: active (running) since Sat 2022-07-16 20:45:31 CST; 4 days ago
 TriggeredBy: ● libvirtd.socket
               ● libvirtd-admin.socket
               ● libvirtd-ro.socket
    Docs: man:libvirtd(8)
          https://libvirt.org
   Main PID: 2087 (libvirtd)
     Tasks: 23 (limit: 32768)
    Memory: 175.9M
       CPU: 13.546s
    CGroup: /system.slice/libvirtd.service
            └─2087 /usr/sbin/libvirtd
              └─2497 /usr/sbin/dnsmasq --conf-file=/var/lib/libvirt/dnsmasq/default.conf --leasefile-ro --dhcp-sc
                └─2503 /usr/sbin/dnsmasq --conf-file=/var/lib/libvirt/dnsmasq/default.conf --leasefile-ro --dhcp-sc
```

如果服务未激活，请手动激活并启动服务：

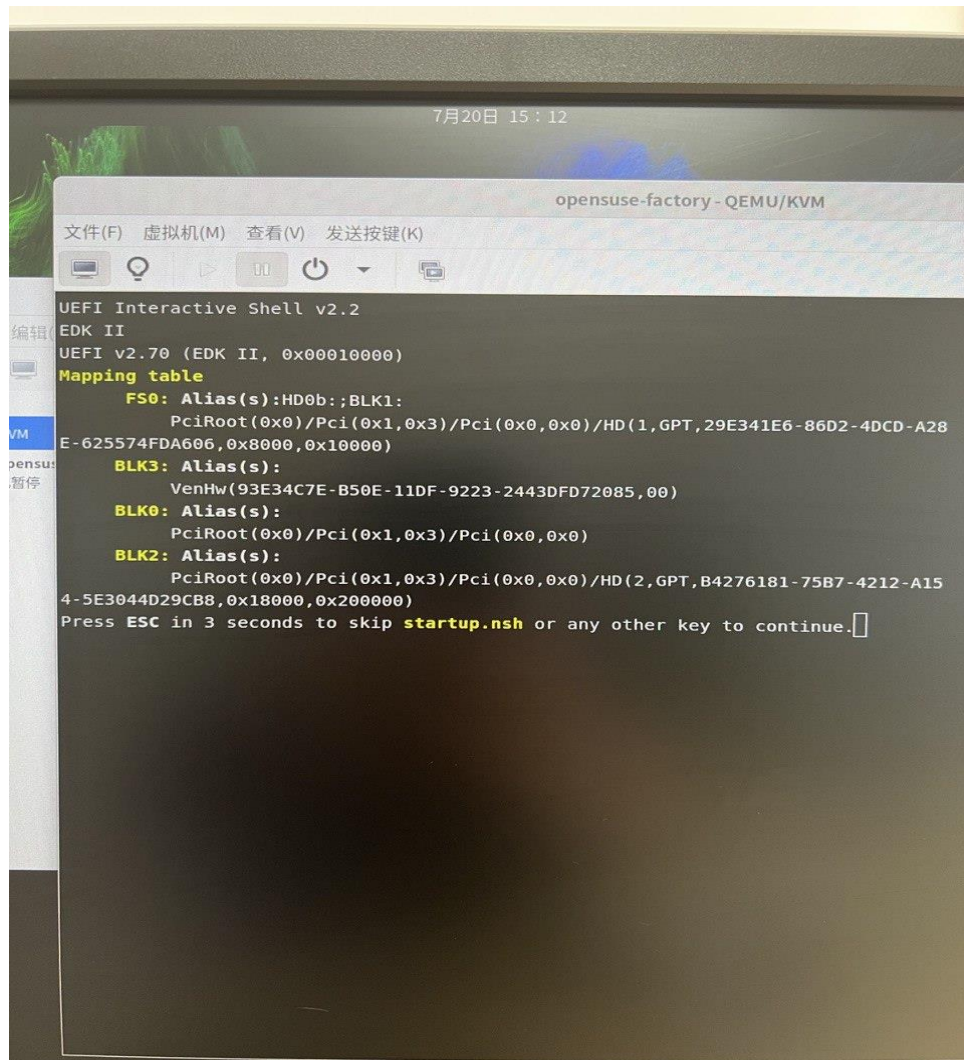
systemctl enable libvirt

systemctl start libvirt

systemctl status libvirt



## 5.3 EFI 启动失败



解决方法：删除虚拟机重建，多试几次，或者给虚拟机改个名

## 5.4 桥接模式下，虚拟机能 ping 通主机，主机也能 ping 通虚拟机，但虚机 ping 不通外网

解决方法：一般是物理机防火墙开着引起的，可以关掉防火墙，或者在物理机的 /etc/sysctl.conf 里添加以下内容：

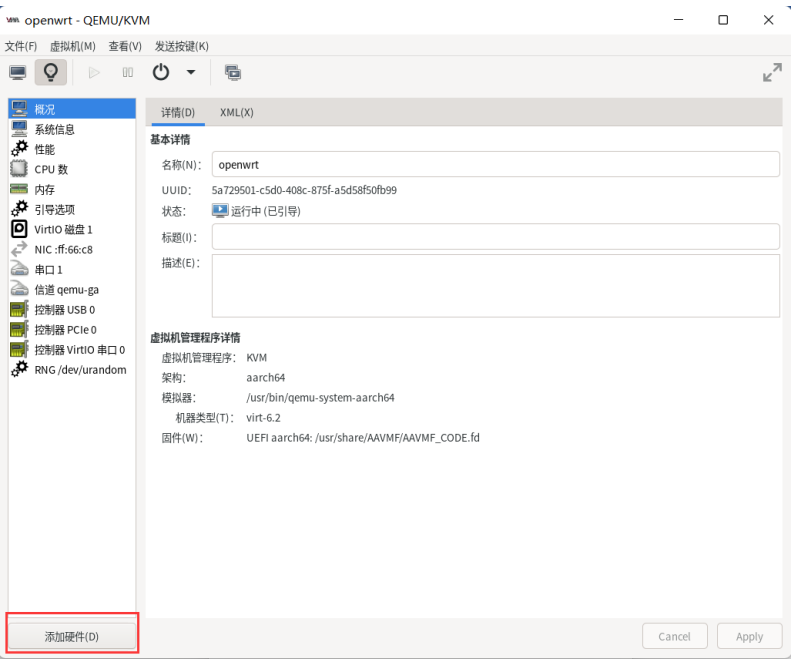
```
net.bridge.bridge-nf-call-ip6tables = 0
net.bridge.bridge-nf-call-iptables = 0
net.bridge.bridge-nf-call-arptables = 0
```

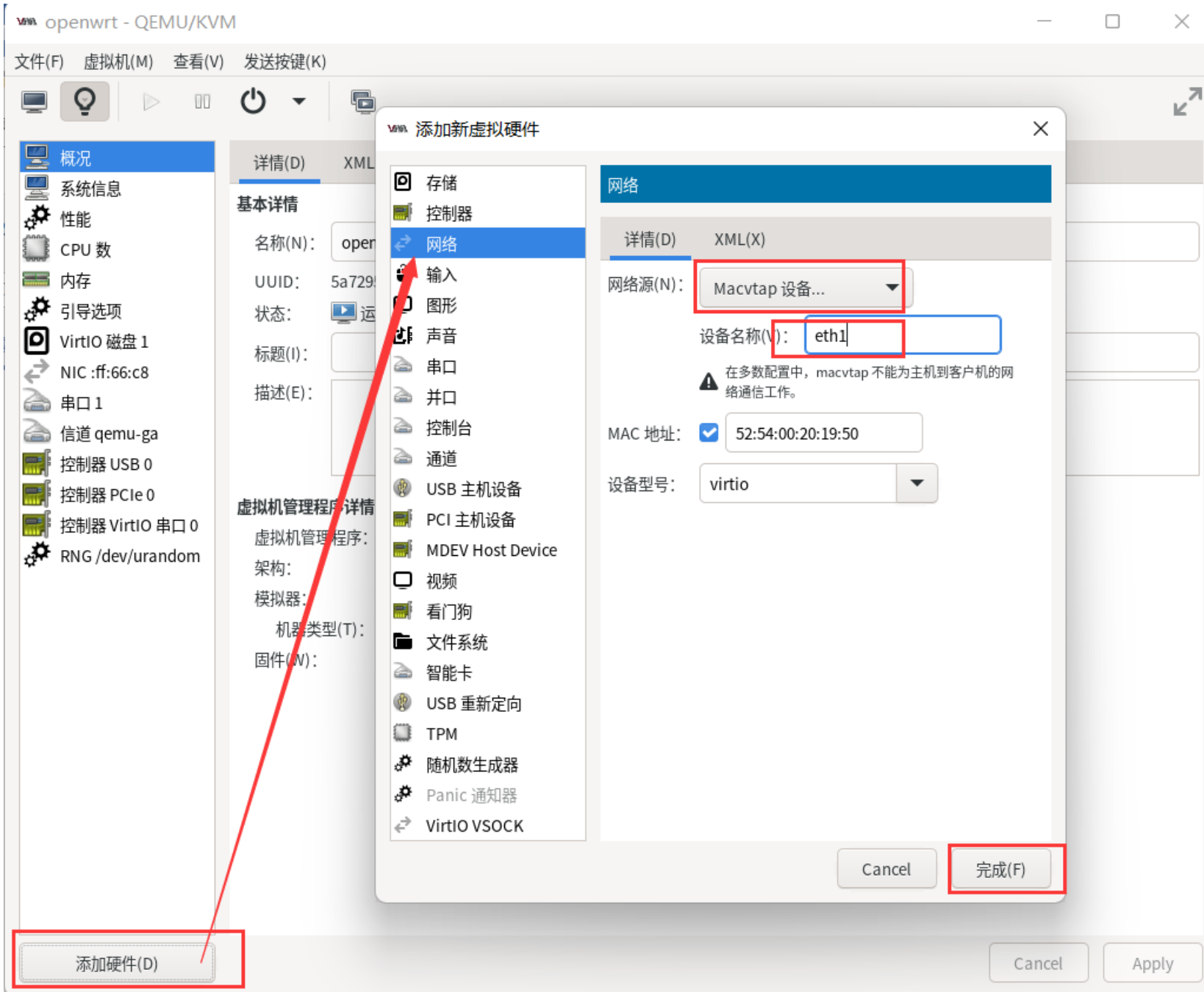
然后运行 `sysctl -p`

# 六、进阶用法

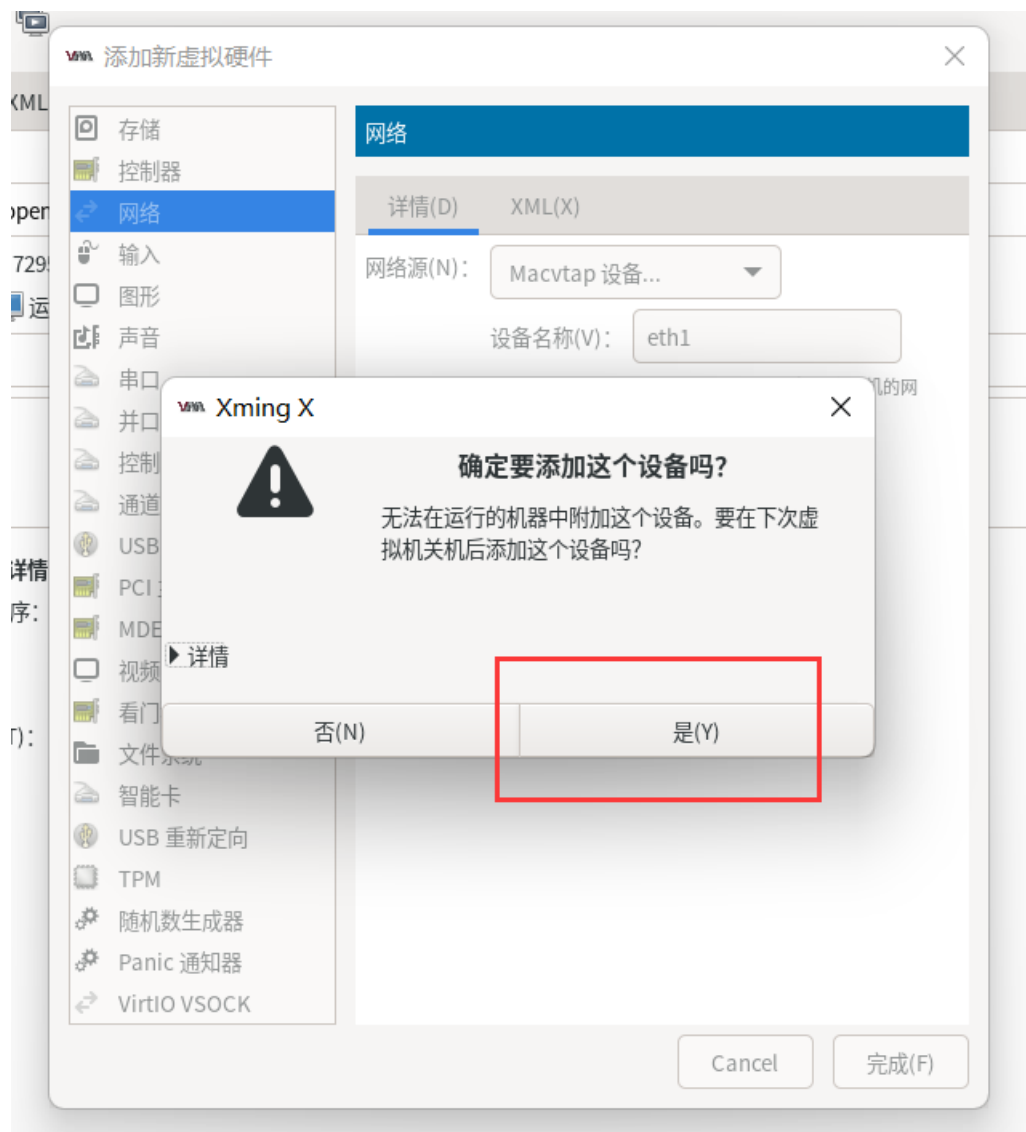
## 6.1 给虚拟机添加第 2 张网卡

前提是物理机有多余的网卡可用，无论是 usb 扩展的还是 pcie 扩展的都行，假设物理机的第二张网卡是 eth1，那么：







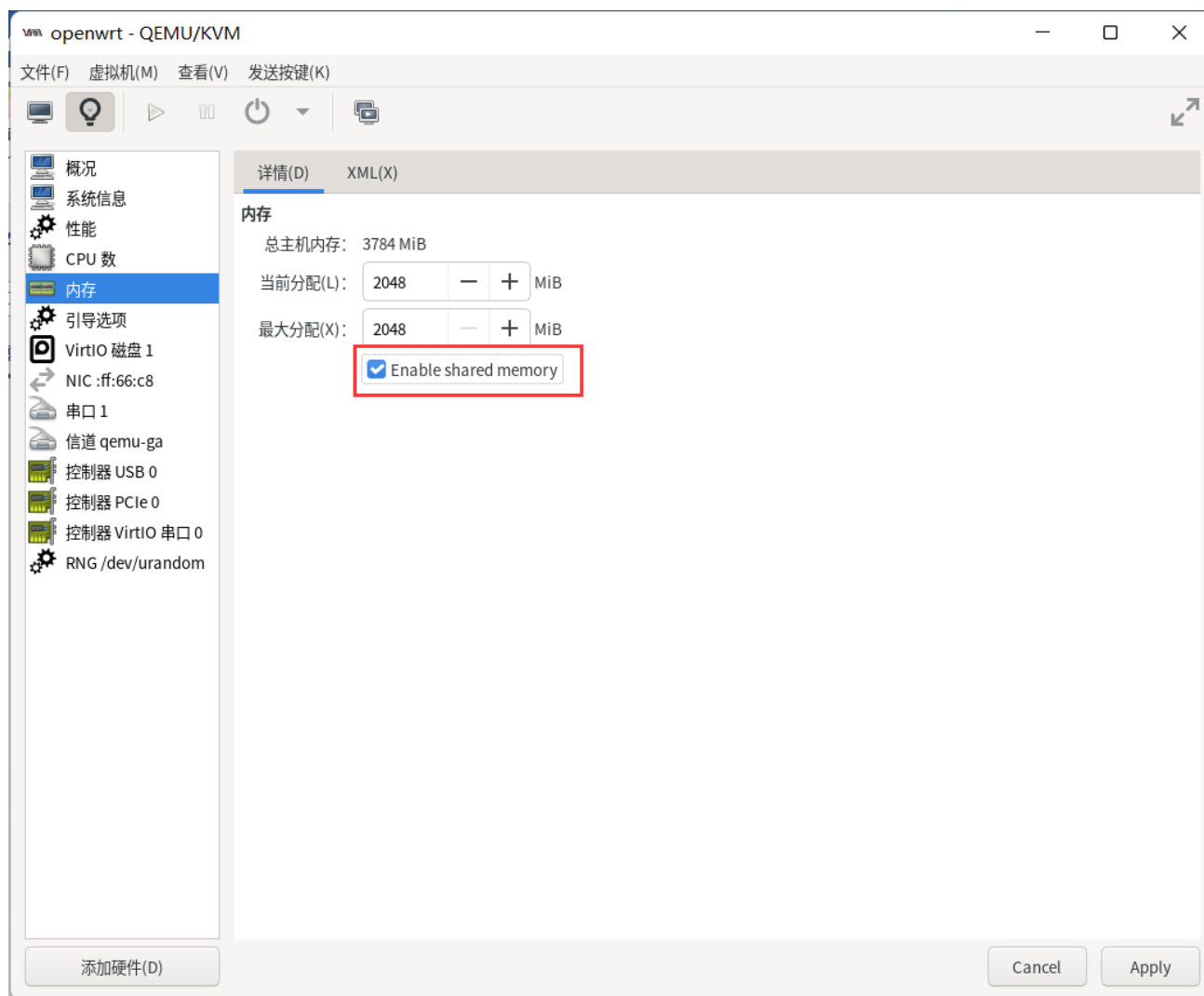


新添的网卡要关闭虚拟机之后才会出现，下次启动生效。

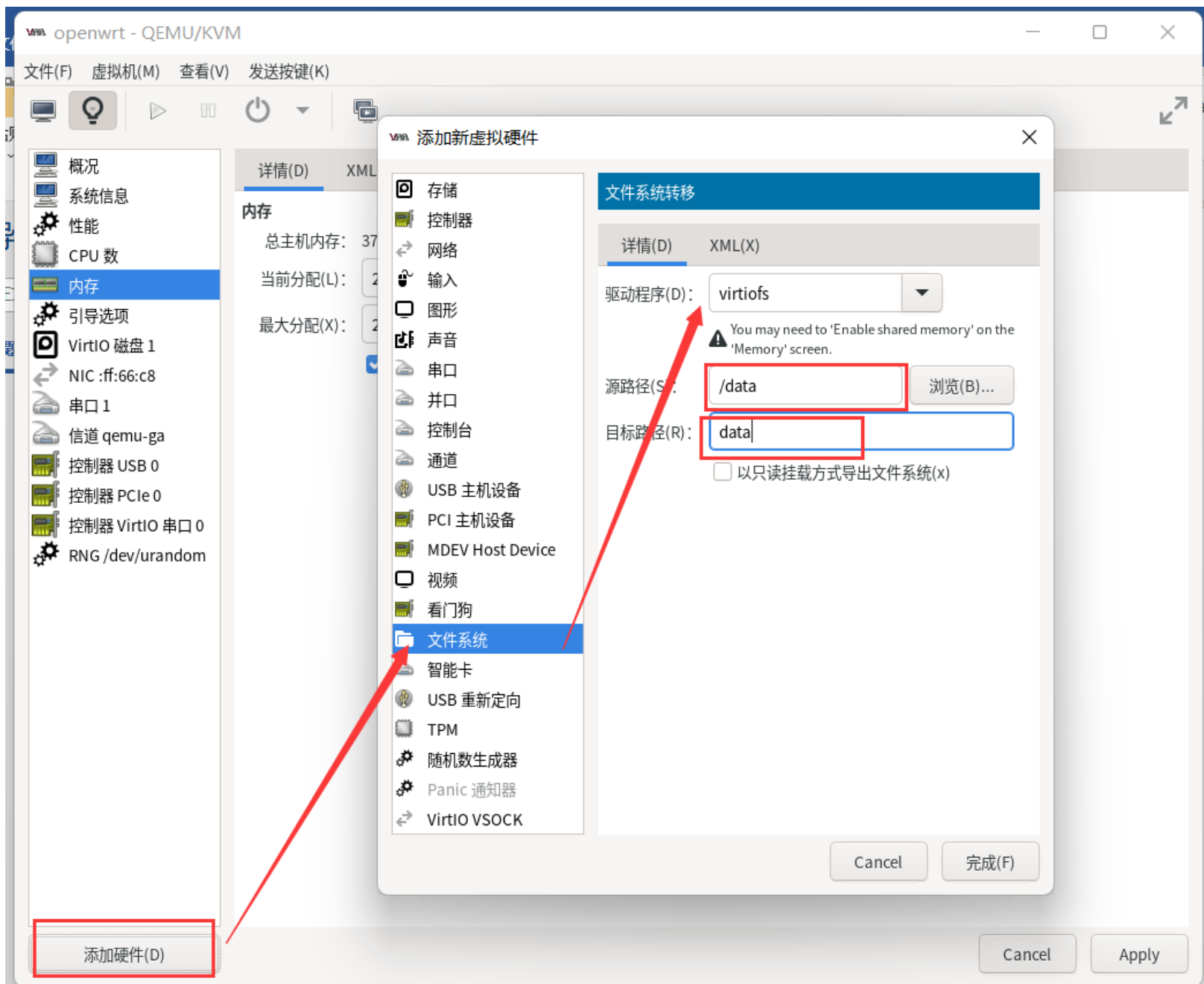
## 6.2 给虚拟机添加共享文件系统

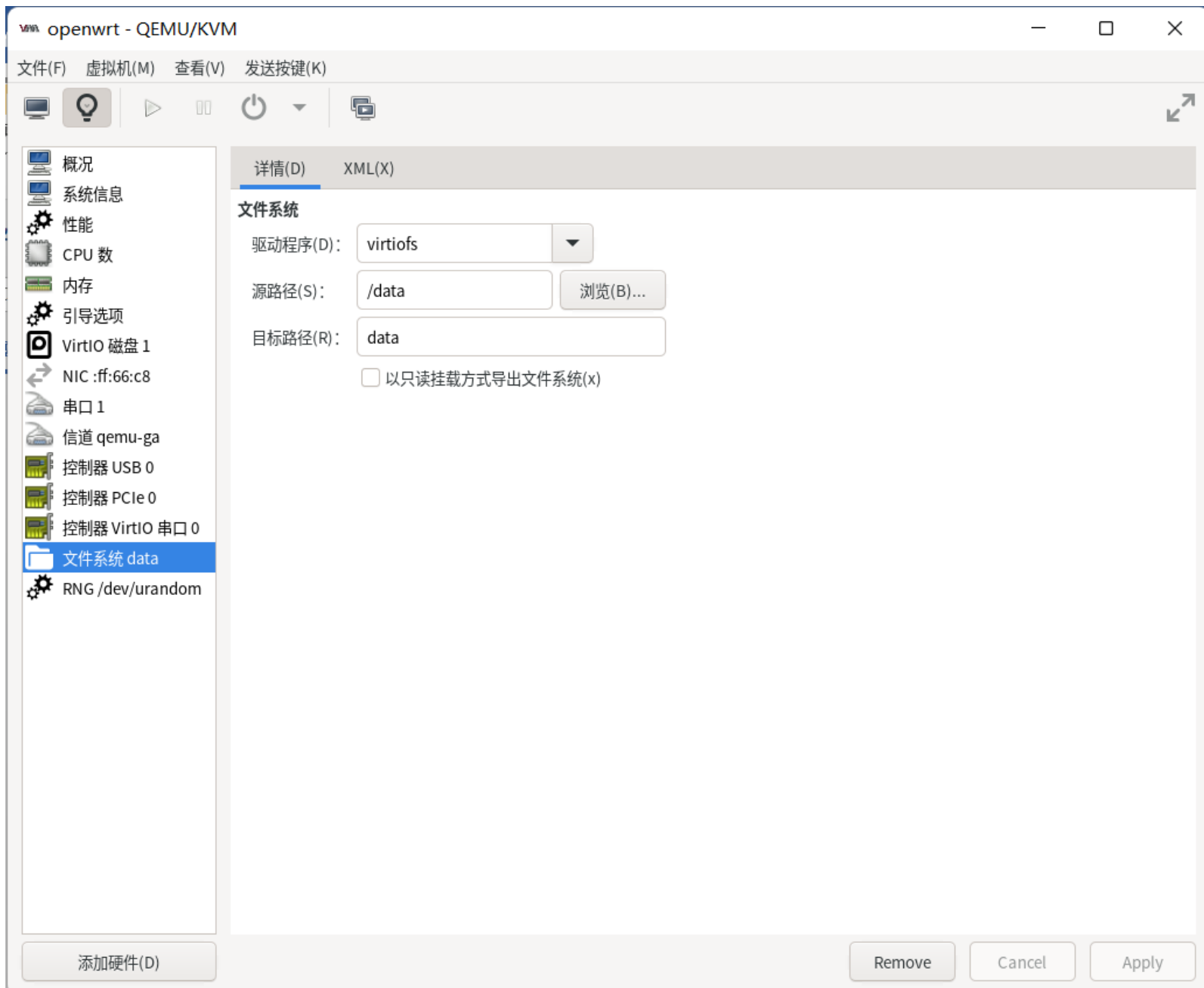
此功能可以把物理机的一个文件夹共享给虚拟机使用，实现物理机、虚拟机之间文件共享，或多个虚拟机之间文件共享，非常实用！

首先要关闭虚拟机，把内存的 shared memory 选项打开：



然后添加硬件，选择“文件系统”，驱动程序选择“virtiofs”，源路径选择物理机上已存在的某个文件夹，目标路径随便编个名字（例如 data）





然后启动虚拟机

在虚机中输入命令：

```
mkdir /mnt/data
```

```
mount -t virtiofs data /mnt/data
```

```
df -h
```

```
root@vm27:/# mkdir /mnt/data
mkdir: can't create directory '/mnt/data': File exists
root@vm27:/# mount -t virtiofs data /mnt/data
root@vm27:/# df -h
```

Filesystem	Size	Used	Available	Use%	Mounted on
udev	512.0K	0	512.0K	0%	/dev
tmpfs	198.1M	92.0K	198.0M	0%	/run
/dev/vda2	1.0G	512.0M	409.4M	56%	/
tmpfs	990.6M	17.2M	973.5M	2%	/tmp
tmpfs	512.0K	0	512.0K	0%	/dev
cgroup	990.6M	0	990.6M	0%	/sys/fs/cgroup
/dev/vda4	14.0G	4.0M	13.4G	0%	/mnt/vda4
/dev/vda3	1.0G	3.8M	904.6M	0%	/mnt/vda3
/dev/vda1	31.9M	1.4M	30.5M	4%	/boot/efi
/dev/vda4	14.0G	4.0M	13.4G	0%	/mnt/vda4/docker
/dev/vda4	14.0G	4.0M	13.4G	0%	/mnt/vda4/docker/btrfs
data	100.2G	89.6G	10.5G	89%	/mnt/data

```
root@vm27:/#
```

挂载成功，如果想要开机自动挂载的话，可以把挂载命令添加到 /etc/rc.local 里

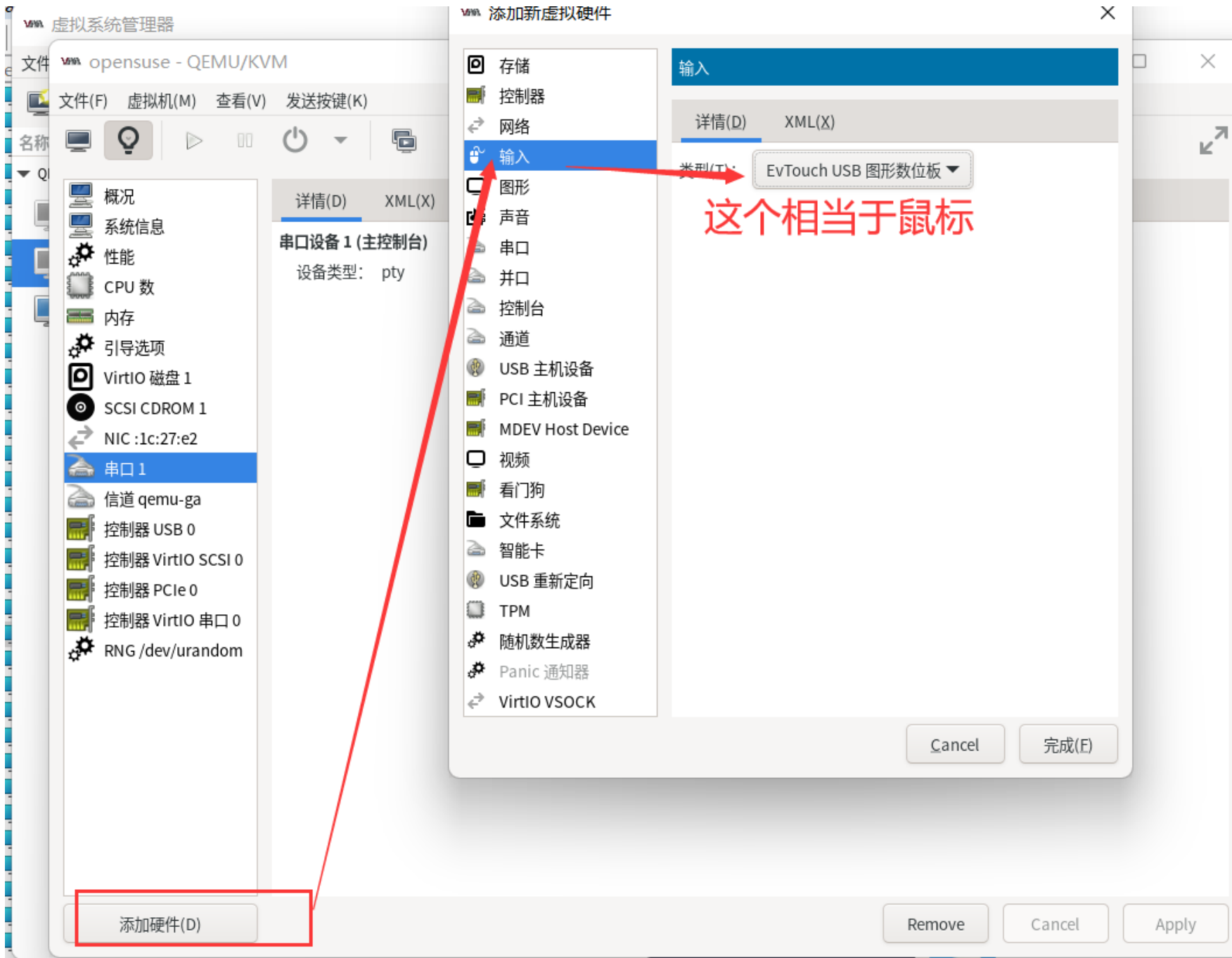
```
# Put your custom commands here that should be executed once
# the system init finished. By default this file does nothing.
mount -t virtiofs data /mnt/data
exit 0
~
```



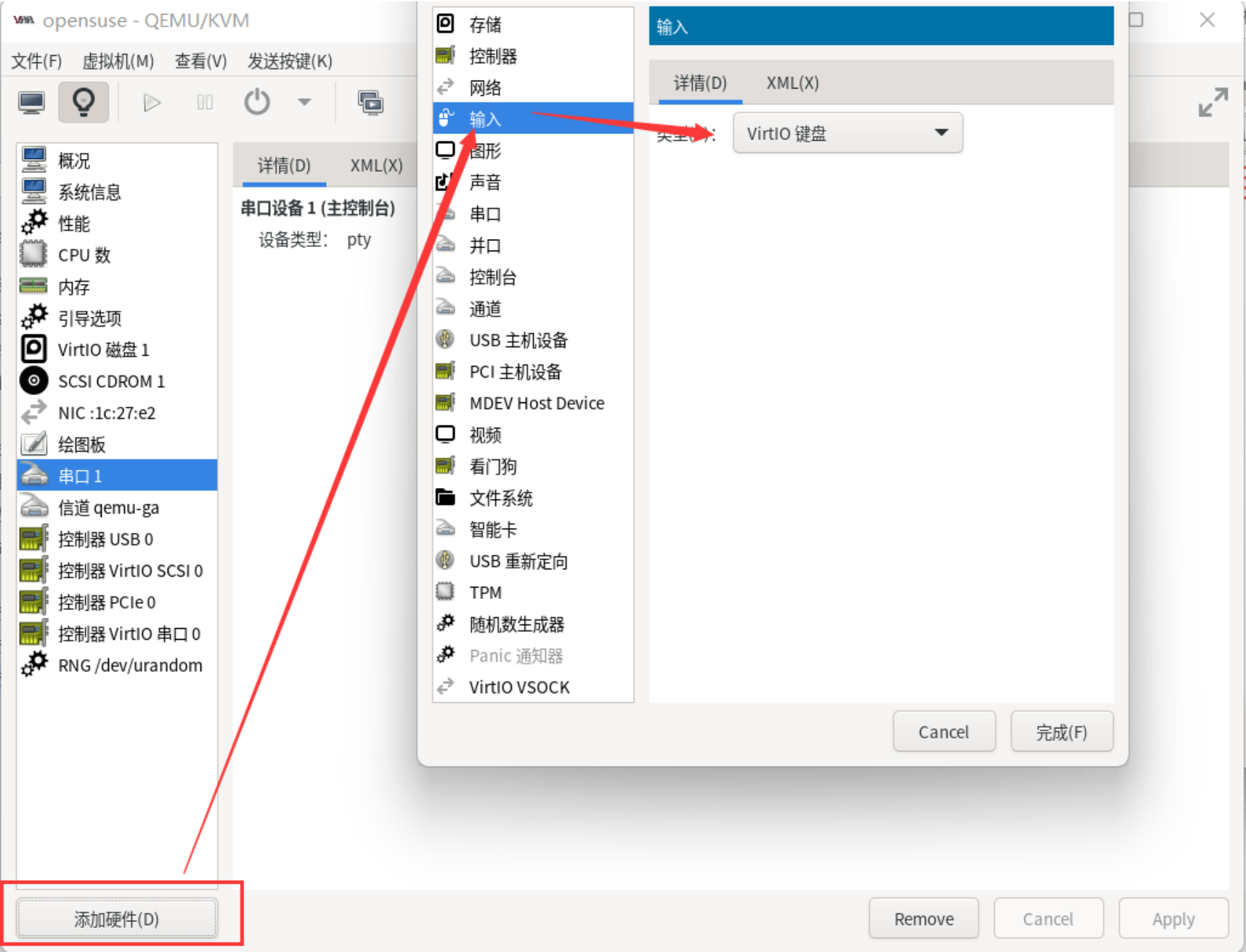
## 6.3 给虚拟机添加显卡（对于 openwrt 没啥用）

虽然对于 openwrt 没用，但对于其它 linux 发行版有用，如果想在 armbian 里运行另一个 linux (debian,ubuntu,openSUSE,archlinux,centos,gentoo,国产麒麟, 国产统信 uos 等等)，那这一步是必需的：

添加鼠标：

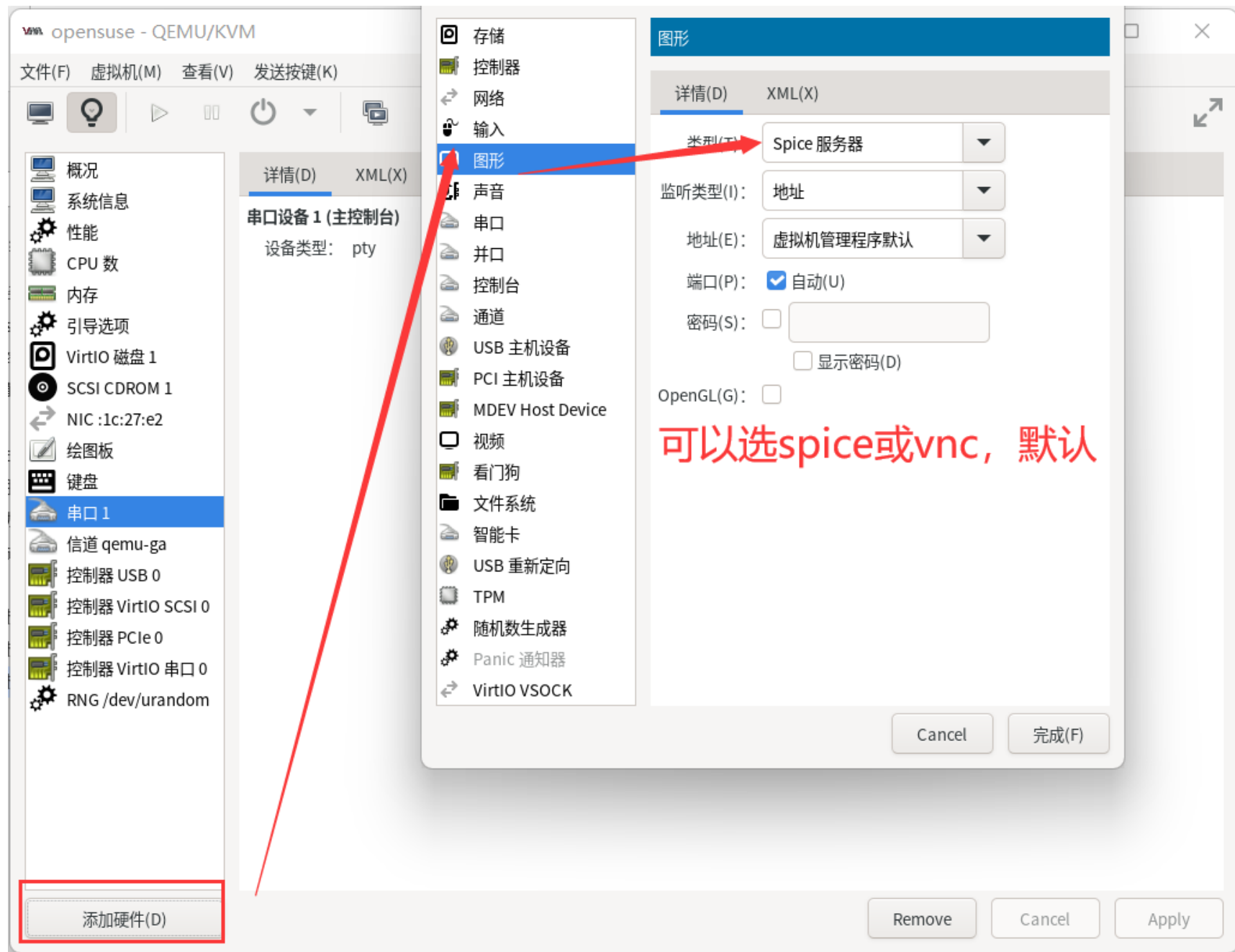


添加键盘：

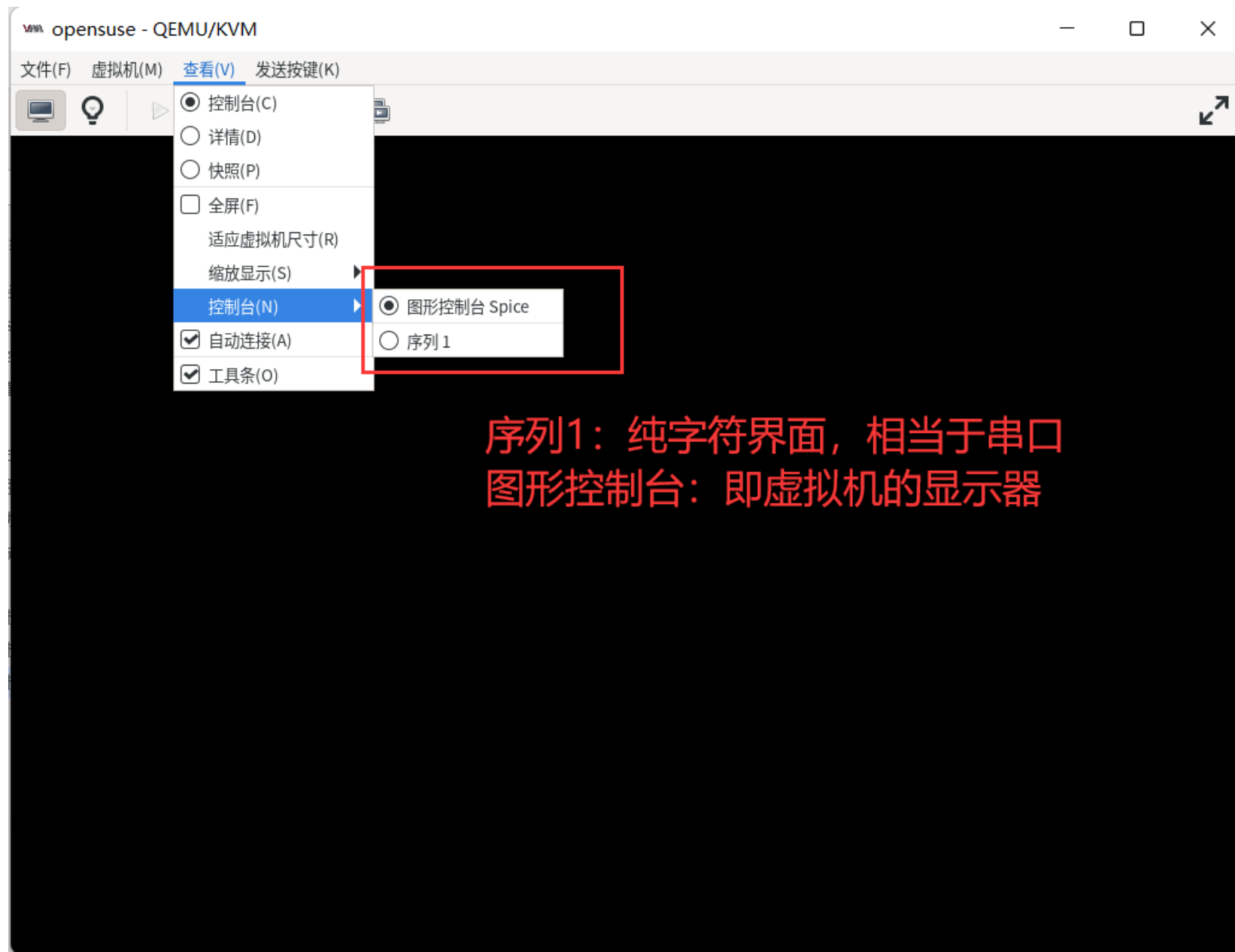




添加显卡:



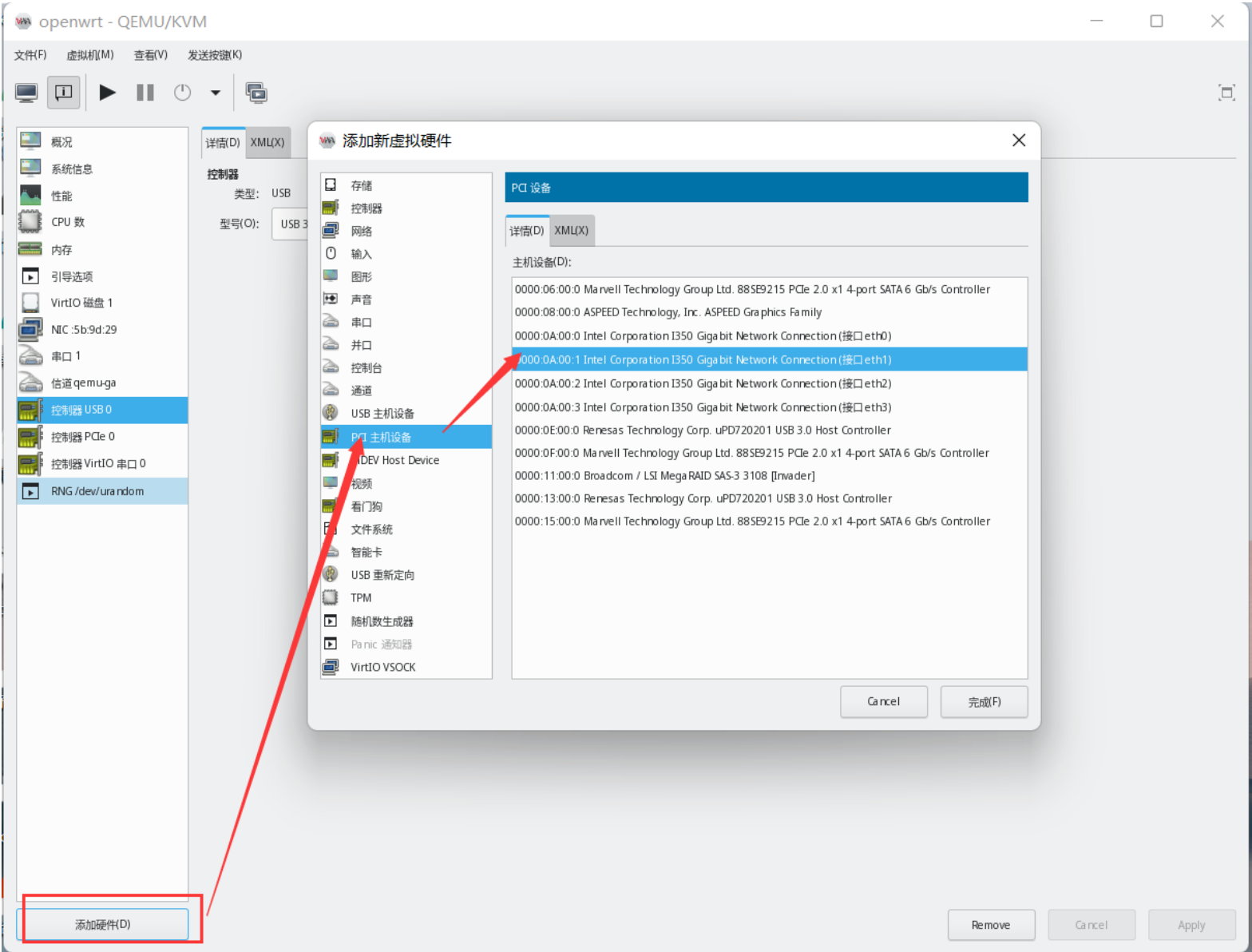
启动虚拟机：

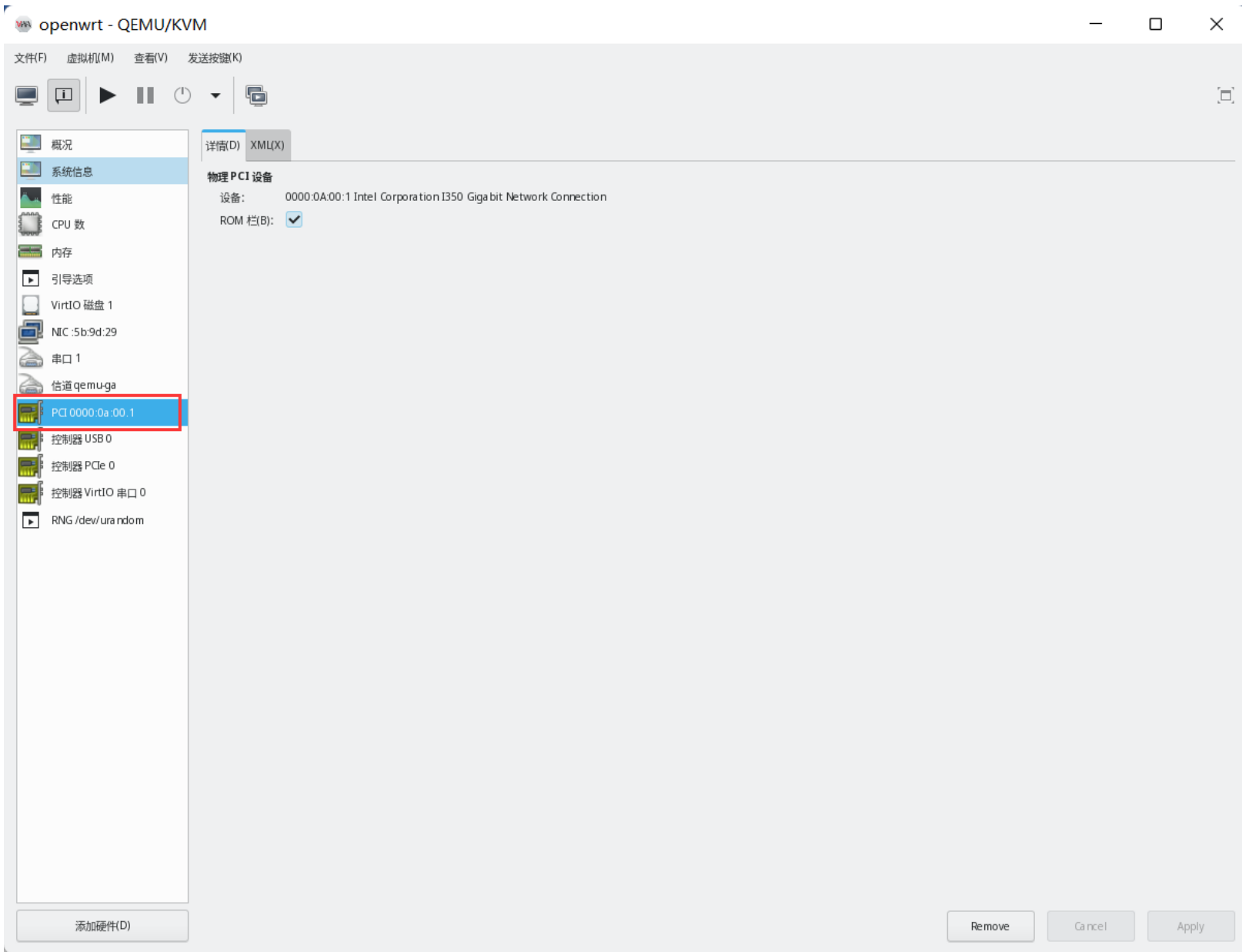


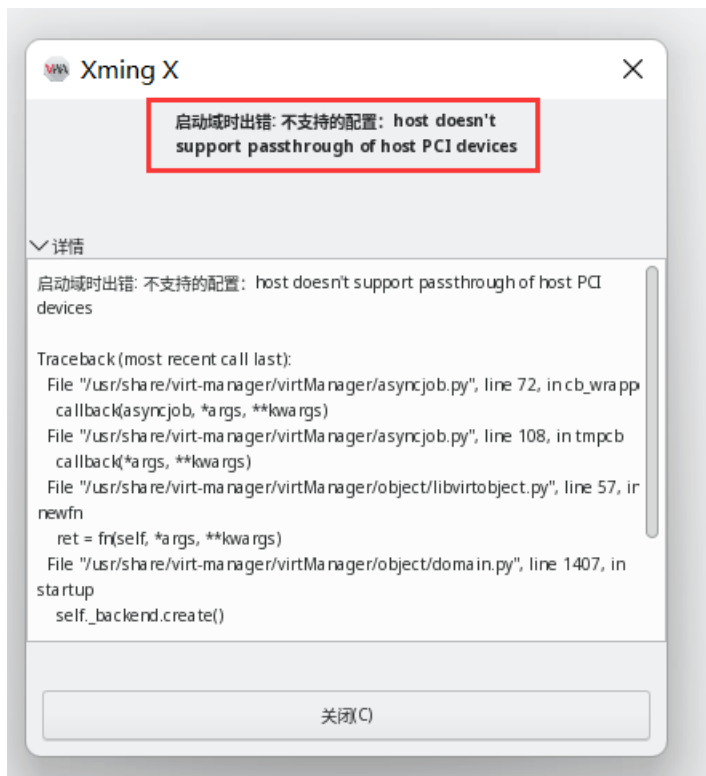
序列1：纯字符界面，相当于串口  
图形控制台：即虚拟机的显示器

## 6.4 给虚拟机添加直通设备

这需要物理机支持 iommu，一般的电视盒子就别想了，目前即使正规的 arm64 服务器也很少支持。







出现这个提示就是不支持直通 😞

## 七、固件升级

每次固件发布会有 2 个文件:

openwrt\_qemu-aarch64\_generic\_vm\_k5.18.13-flippy-75+.img

openwrt\_qemu-aarch64\_generic\_vm\_k5.18.13-flippy-75+.qcow2

其中, 后缀为.qcow2 的文件是首次创建虚拟机用的, 而另一个后缀为.img 的文件就用于升级的。

### 7.1 命令行升级方法:

1. 把 openwrt\_qemu-aarch64\_generic\_vm\_k5.18.13-flippy-75+.img 及附带的升级脚本上传至虚拟机的 /mnt/vda4 目录下

(7z 压缩包里也会同时包含一个升级脚本：update-kvm-openwrt.sh，与/usr/sbin/openwrt-update-kvm 是同一个文件，但版本可能更新一些)

2. `cd /mnt/vda4`
3. `/usr/sbin/openwrt-update-kvm openwrt_qemu-aarch64_generic_vm_k5.18.13-flippy-75+.img`  
或  
`./update-kvm-openwrt.sh openwrt_qemu-aarch64_generic_vm_k5.18.13-flippy-75+.img`

## 7.2. 用“虚拟宝盒”应用进行升级

使用方法基本与“晶晨宝盒”相同

# 八、内核升级

内核升级即：只升级 kernel，不升级 openwrt 的应用。

## 8.1. 命令行升级

把 boot-xxxx.tar.gz、modules-xxxx.tar.gz 两个内核压缩包上传至 /mnt/vda4，然后运行：  
openwrt-kernel-kvm

## 8.2. 用“虚拟宝盒”应用进行升级

使用方法基本与“晶晨宝盒”相同