# 云计算

## 网络配置

```
ln -s /usr/libexec/qemu-kvm /usr/bin

qemu-kvm -net nic,model=?
virt-install --network=?
```

### 网卡类型

- **rtl8139**　Realtek公司10/100M网卡(qemu-kvm)
- **e1000** Intel e1000(纯qemu)

```
lspci | grep Ethernet
ethtool -i eth0

(qemu) info network


virsh # qemu-monitor-command Z_D --hmp info
```

### QEMU支持的网络模式

- 基于网桥(bridge)的虚拟网卡
- 基到NAT(Network Address Translation)的虚拟网络
- 内置用户模式网络(user mode networking)
- 直接分配网络设备的网络(VT-d,SR-IOV)

#### 网桥模式

```
lsmod | grep tun


brctl addbr br0
brctl addif br0 eth0
brctl stp br0 on
ifconfig eth0 0
dhclient br0
route
ping -c 3 8.8.8.8
dmesg | grep eth0


/etc/sysconfig/network-scripts/ifcfg-br0

TYPE=Bridge
UUID=d2d68553-f97e-7549-7a26-b34a26f29318
BOOTPROTO=none
NAME=br0
DEVICE=br0
ONBOOT=yes
IPADDR=172.19.9.252
PREFIX=16
GATEWAY=172.19.0.1
DNS1=172.19.0.1
DNS2=119.29.29.29
STP=yes
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
IPV6INIT=no
```

/etc/sysconfig/network-scripts/ifcfg-em1

```
TYPE=Ethernet
NAME=em1
DEVICE=em1
ONBOOT=yes
BRIDGE=br0
```

```
nmcli connection add type bridge con-name br0 ifname br0
nmcli connection add type bridge-slave con-name br0-eth0 ifname eth0 master br0
nmcli connection modify br0 ipv4.method manual
nmcli connection mpdify br0 ipv4.address "192.168.1.1/24" 192.168.1.254
nmcli connection modify br0 ipv4.dns 192.168.1.10
nmcli connection modify br0 ipv4.gateway 192.168.1.10
nmcli connection modify br0 connection.autoconnect yes
nmcli connection up br0
```

```
#!/bin/bash
#This is a qemu-ifup script for bridging.
#You can use it when starting a KVM guest with bridge mode network.

# set you bridge name
switch=br0

if [ -n "$1" ]; then
  #create a TAP interace; qemu will handle it automatically.
  #tunctl -u $(whoami) -t $1

  #start up the TAP interface
  ip link set $1 up
  sleep 1
  #add TAP interface to the bridge
  brctl addif ${switch} $1
  exit 0
else
  echo "Error: no interface specified"
  exit 1
fi
```

```
#!/bin/bash
#This is a qemu-ifdown script for bridging.
#You can use it when starting a KVM guest with bridge mode network.
#Don't use this script in most cases; QEMU will handle it automatically.

#set your bridge name
switch=br0

if [ -n "$1" ]; then
  #Delete the specified interfacename
  tunctl -d $1
  #release TAP interface from bridge
  brctl delif ${switch} $1
  #shutdown the TAP interface
  ip link set $1 down
  exit 0
else
  echo "Error: no interface specified"
  exit 1
fi
```

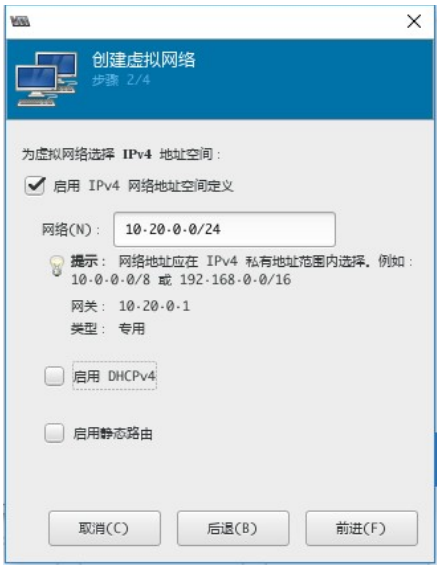
```
qemu-kvm cirros.img -smp 2 -m 1024 -net nic -net tap,ifname=tap1,script=/etc/qemu-ifup,downscript=no -vnc :0 -daemonize
```

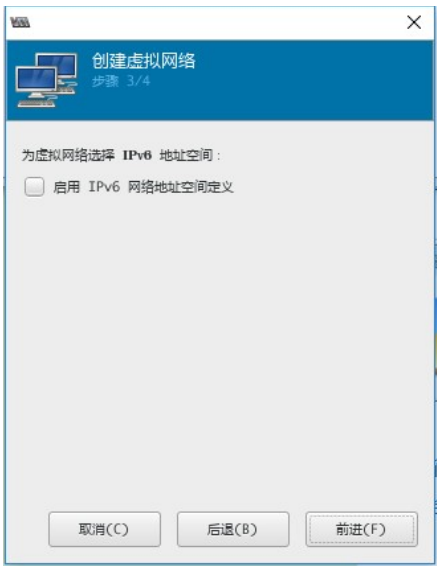
**NAT模式**

- **伪装**
- SNAT
- DNAT

```
ps aux  | grep dnsmasq
ps -elf | grep dnsmasq
```


创建虚拟网络
步骤 1/4
为虚拟网络选择名称：
网络名称(N)：
virbr1
示例： network1
取消(C)
后退(B)
前进(F)


创建虚拟网络
步骤 2/4
为虚拟网络选择 IPv4 地址空间：
启用 IPv4 网络地址空间定义
网络(N)：
10.20.0.0/24
提示： 网络地址应在 IPv4 私有地址范围内选择。例如：
10.0.0.0/8 或 192.168.0.0/16
网关： 10.20.0.1
类型： 专用
启用 DHCPv4
启用静态路由
取消(C)
后退(B)
前进(F)


创建虚拟网络
步骤 3/4
为虚拟网络选择 IPv6 地址空间：
启用 IPv6 网络地址空间定义
取消(C)
后退(B)
前进(F)


创建虚拟网络
步骤 4/4
连接到 物理网络：


/etc/libvirt/qemu/networks/virbr1.xml

```
<!--
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 net-edit virbr2
or other application using the libvirt API.
-->

<network>
  <name>virbr2</name>
  <uuid>f3b4de5e-8ad5-4117-9f66-9335d2188ecb</uuid>
  <forward mode='nat'/>
  <bridge name='virbr2' stp='on' delay='0'/>
  <mac address='52:54:00:f2:50:ec'/>
  <ip address='172.16.0.1' netmask='255.255.255.0'>
  <dhcp>
    <range start='172.16.0.128' end='172.16.0.254'/>
  </dhcp>
  </ip>
</network>
```

```
virsh net-list
virsh net-dumpxml
virsh domiflist CentOS
```