LINUX的网络管理

一、基本概念

1. 物理网卡（NIC, Network Interface Card）

- **硬件实体**：插在主板或扩展槽上的设备（如 Intel千兆网卡、Realtek 无线网卡）。
- 功能：
  - 负责物理层（PHY）信号传输（如电信号、光信号）。
  - 支持数据链路层（MAC 地址、帧封装）的基础操作。

2. 网络适配器（Adapter）

- **软件抽象**：Windows 系统中驱动程序与服务的集合，管理网卡硬件。
- 功能：
  - 加载网卡驱动，提供 API 接口供操作系统调用。
  - 处理网络协议（如 TCP/IP）、虚拟化支持（如 Hyper-V 的虚拟网卡）。
  - 在设备管理器中显示为“网络适配器”条目。

3. DHCP（Dynamic Host Configuration Protocol）动态主机配置协议

1. 核心功能

- **自动分配 IP 地址**：在局域网内自动为设备分配 `IP/MAC` 绑定信息（如 `192.168.1.100`）。
- **配置同步**：自动设置子网掩码、网关、DNS 服务器等参数。
- **租约管理**：IP 地址租用期限（如 `86400 秒`），到期后自动续租或回收。

2. 典型应用场景

- **动态网络**：办公室、家庭 Wi-Fi，设备频繁变动（如手机、平板接入）。
- **虚拟化环境**：VMware/Kubernetes 中的虚拟机自动获取 IP。
- **服务部署**：快速部署临时服务器，无需手动配置 IP。

4. 静态地址（Static IP Address）

1. 核心特征

- **手动配置**：固定 IP 地址（如 `192.168.1.100`），不会因网络变化而改变。
- **高稳定性**：适用于需要长期稳定访问的服务（如 Web 服务器、数据库）。

2. 典型应用场景

- **生产服务器**：确保外部用户始终通过固定 IP 访问服务。
- **网络设备**：路由器、防火墙等需要固定 IP 的设备。
- **内网穿透**：在 NAT 环境中，为特定服务分配静态内网 IP。

5. DNS（Domain Name System）域名系统

1. 核心功能

- **域名解析**：将人类可读的域名（如 `google.com`）转换为 IP 地址（如 `172.217.14.104）。
- 支持正向/反向解析：
  - 正向解析：`域名 → IP`
  - 反向解析：`IP → 域名`

2. 典型应用场景

- **访问互联网服务**：浏览器、邮件客户端依赖 DNS 解析域名。
- **内部网络管理**：企业内网域名解析（如 `mail.example.com`）。
- **负载均衡**：通过 DNS 轮询实现流量分发。

6.linux中常见网络接口类型及缩写标识解析

标识符	含义	典型接口	配置场景
`en`	有线以太网	`ens160`、`eth0`	服务器局域网接入
`wl`	无线LAN	`wlan0`	笔记本无线网络连接
`ww`	WAN	`ppp0`	拨号上网
`o`	主板集成网卡	`enp0s3`	服务器硬件加速网络
`p`	PCI 接口网卡	`enic0`	高性能网络适配器
`v`	虚拟接口	`virbr0`	容器网络、虚拟机桥接

7.、Windows 中的“网络适配器”分类

类型	描述	示例
物理网卡适配器	对应硬件网卡，如 `Intel(R) Ethernet Connection I219-V`。	设备管理器中的“以太网适配器”
虚拟网卡适配器	软件模拟的网卡，用于虚拟机、容器或隔离网络环境。	`VMnet8 (NAT)`、
系统服务适配器	提供底层网络协议支持（如 `TCP/IP` 协议栈）。	`Microsoft TCP/IP Adapter`
无线局域网适配器	集成或外接的 Wi-Fi 硬件管理模块。	`Intel(R) Wireless-AC 9480`

8.路由转发

路由转发（IP Forwarding）是计算机网络中的核心机制，指 **路由器或具备路由功能的设备根据路由表将接收到的数据包转发到目标网络。**以下是详细解析：

---

1. 核心概念

- **路由表（Routing Table）**：
  存储网络地址与下一跳信息的表格，决定数据包的转发路径。
  示例条目：

```
目标网络        下一跳IP        接口        标记
192.168.1.0/24  10.0.0.1      eth0       UG
```

- **下一跳（Next Hop）**：
  数据包转发的下一个设备或网关的 IP 地址。
- **路由协议（如 OSPF、BGP）**：
  动态更新路由表的协议，适用于大型网络（如企业网或互联网）。

---

2. 路由转发的工作流程

1. **接收数据包**：
   设备收到来自本地网络的数据包（如 `192.168.1.5` → `8.8.8.8`）。
2. 路由表查询：
   检查目标地址 `8.8.8.8` 是否属于本地网络。
   - **若是**：直接交付给目标主机。
   - **若否**：查找路由表中匹配的条目。
3. 数据包转发：
   - 修改数据包的 **TTL（生存时间）** 和 **校验和**。
   - 将数据包发送到下一跳设备（如网关 `10.0.0.1`）。
4. **递归转发**：
   下一跳设备重复上述过程，直至数据包到达最终目的地。

---

3. 路由转发的关键场景

(1) 内部网络访问外部网络

- **场景**：公司内网（`192.168.1.0/24`）通过路由器（`10.0.0.1`）访问互联网（`8.8.8.8`）。
- 路由表配置：

```
# 添加默认网关（互联网出口）
sudo ip route add default via 10.0.0.1 dev eth0
```

(2) 数据中心流量调度

- **场景**：多个服务器（`10.0.1.1`、`10.0.1.2`）通过负载均衡器（`10.0.0.1`）分发流量。
- 路由策略：
  - 基于流量权重或服务器状态动态选择下一跳。

(3) VPN 隧道

- **场景**：远程办公用户通过 VPN 隧道（如 `172.16.0.0/16`）访问公司内网。
- 路由表配置：

```
# 添加 VPN 路径
sudo ip route add 172.16.0.0/16 via 10.0.0.1 dev tun0
```

---

4. 路由转发的配置方法

(1) Linux 系统（使用 `ip` 命令）

```
# 查看路由表
ip route show

# 添加静态路由（目标网络→下一跳→接口）
sudo ip route add 192.168.2.0/24 via 10.0.0.2 dev eth1

# 删除路由
sudo ip route del 192.168.2.0/24
```

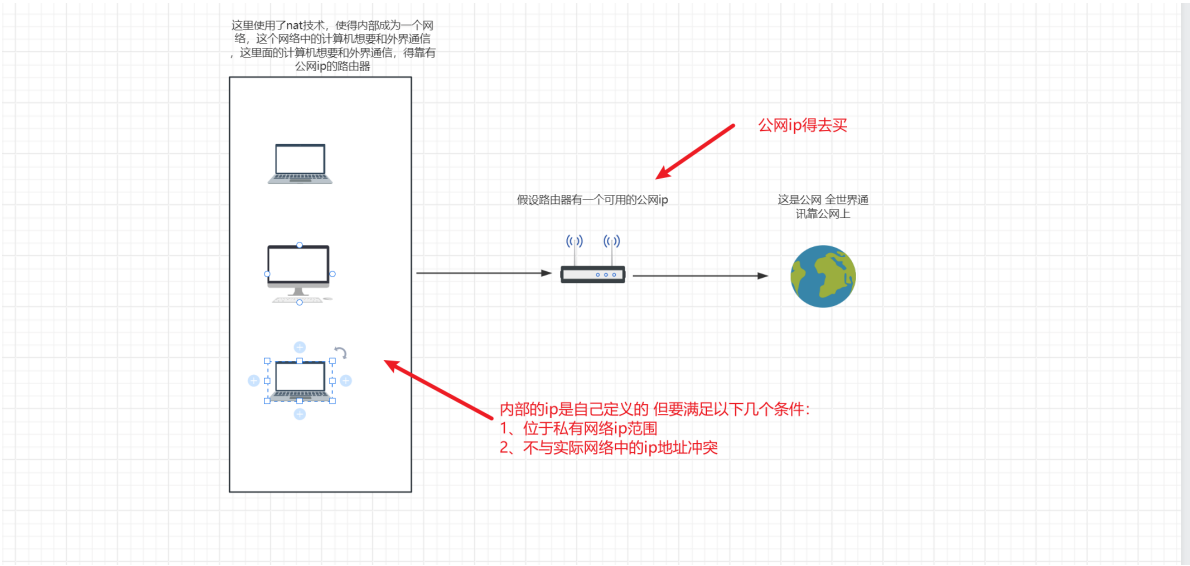
(2) 配置文件持久化

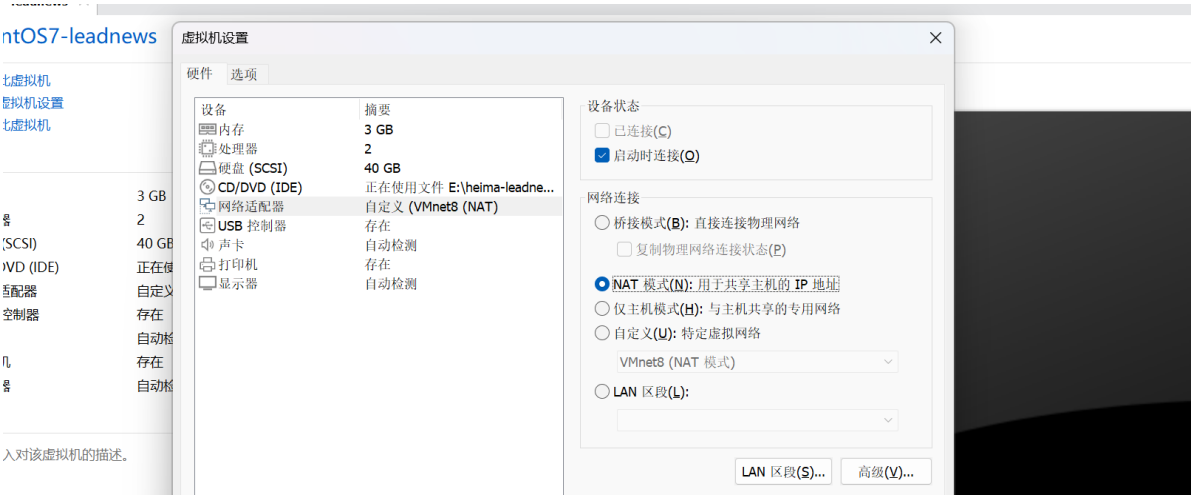
- **路径**：`/etc/sysconfig/network-scripts/route-eth0`（针对接口 `eth0`）。
- /etc/sysconfig/network-scripts/ifcfg-ensxxx
- 示例内容：

```
to default via 10.0.0.1 dev eth0
to 192.168.2.0/24 via 10.0.0.2 dev eth1
```

二、虚拟机的网络连接方式

1.NAT（网络地址转换）





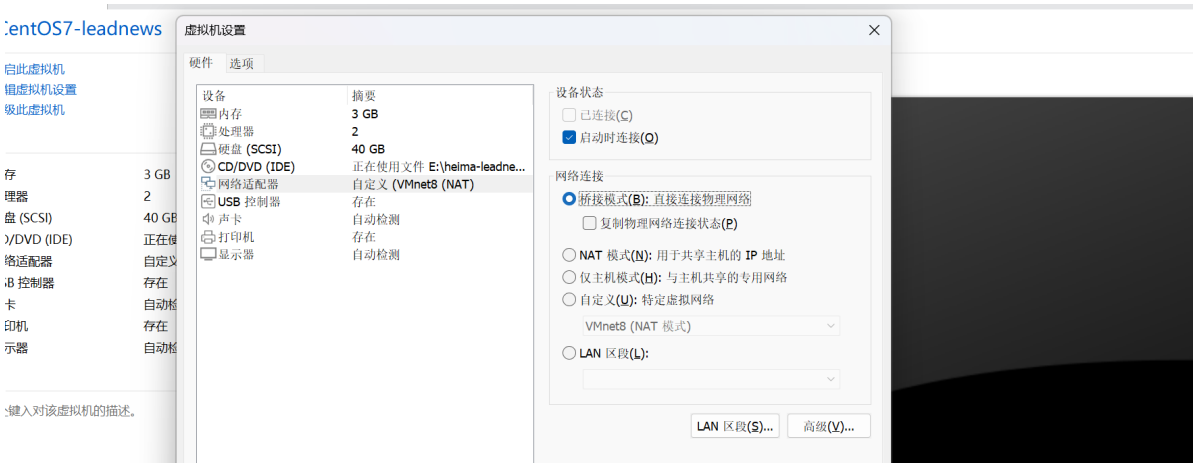
工作原理

- **NAT模式**（网络地址转换模式，Network Address Translation）允许虚拟机通过主机访问外部网络（如互联网），但外部网络无法直接访问虚拟机。其工作原理主要是通过主机充当虚拟机的“网关”来实现网络地址转换。

1. 虚拟机的**私有网络**：
    - 在 NAT 模式下，虚拟机会被分配一个**私有 IP 地址**（如 192.168.x.x 或 10.x.x.x），这个 IP 地址只能在虚拟机和主机之间有效，并且与外部网络隔离。
2. 虚拟机请求外部网络：
    - 当虚拟机需要访问外部网络（如互联网）时，它会把数据包通过虚拟网卡发送给主机。
    - 数据包的源 IP 地址是虚拟机的私有 IP 地址，而目标地址是外部网络的某个设备（例如某个网站的服务器 IP）。
3. 主机进行 NAT 转换：
    - 主机充当，NAT 路由器，将来自虚拟机的数据包进行，网络地址转换。
        - 将数据包中的源 IP 地址从虚拟机的私有 IP 地址转换为主机的公网 IP 地址。
        - 保留虚拟机的端口信息，以便将返回数据正确转发给虚拟机。
4. 数据包发送至外部网络：
    - 主机将已经替换源 IP 的数据包发送到外部网络（例如互联网）。
    - 外部服务器只看到数据包的源地址为主机的公网 IP，而不知道虚拟机的私有 IP。
5. 外部网络响应数据：
    - 外部服务器将响应数据包发送回主机的公网 IP。
    - 主机接收到响应后，使用 NAT 表将数据包中的目标 IP 地址重新转换回虚拟机的私有 IP 地址。
6. 主机将数据包转发给虚拟机：
    - 主机将响应数据包转发给对应的虚拟机，虚拟机接收到外部网络的数据包后完成通信。
        - NAT模式下，主机是虚拟的路由器，而虚拟机是一台私有网络中的机器。而既然是私有网络中的机器，那就和私有网络IP地址有关

2.桥接模式（独立主机）

桥接模式允许虚拟机与主机网络以及外部网络完全独立地通信，虚拟机就像是网络中的一台独立主机，具有自己的 IP 地址。



工作原理

1.**网络连接**:

- 在桥接模式下，虚拟机的网络适配器与主机的物理网卡（例如以太网适配器或无线适配器）相连接。这种连接模拟了虚拟机与局域网中的一台独立设备直接相连的情况。

2.**获取 IP 地址**:

- 虚拟机启动后，会通过 DHCP（动态主机配置协议）从网络中的 DHCP 服务器获取一个 **独立的 IP 地址**。这个 IP 地址与局域网中的其他设备位于同一子网内，确保虚拟机可以与其他设备直接通信。

3.**数据链路层通信**:

- 虚拟机和其他网络设备之间的通信是在数据链路层（Layer 2）进行的。虚拟机使用 MAC 地址进行识别和通信。
- 当虚拟机发送数据包时，它会将数据包发送到主机的物理网卡。由于虚拟机的网络适配器与主机的网卡桥接，数据包将直接转发到局域网。

3. 主机模式（子主机）

主机模式将虚拟机完全隔离在主机的网络中，虚拟机只能与主机通信，无法访问外部网络，也无法与其他局域网设备通信。

工作原理

1.**虚拟网络适配器**：

- 在主机模式下，虚拟机通过一个**虚拟网络适配器**与主机进行连接。这个虚拟适配器形成了一个独立的局域网，仅包含主机和虚拟机，完全隔离于外部网络。

2.**IP 地址分配**：

- 主机和虚拟机在这个私有网络中都有各自的 IP 地址。主机的虚拟网络适配器通常会为虚拟机分配一个私有 IP 地址（例如 192.168.x.x）。这个 IP 地址只在主机和虚拟机之间有效，外部网络无法访问。

3.**数据传输**：

- 当虚拟机需要与主机通信时，它通过虚拟网络适配器发送数据包。这些数据包只在主机和虚拟机之间进行转发。
- 主机和虚拟机之间的通信是在数据链路层（Layer 2）进行的，使用 MAC 地址进行识别和处理。

三、网络配置

怎么配ip！！！

1.网络服务-网络管理器（NetworkManager）

网络管理器（NetworkManager）是一个动态网络的控制器与配置系统，它用于当网络设备可用时保持设备和连接开启并激活

默认情况下，CentOS8 已安装网络管理器，并处于启用状态。

- 查看网络管理程序的状态

```
systemctl    status    NetworkManager
```

- 查看网络子管理程序的状态

```
systemctl    status    network    老版（centos 6之前）
```

2.配置网络的方法

配置文件：vim

- [root@localhost ~]# vim /etc/sysconfig/network-scripts/ifcfg-ens160

命令行： nmcli

- 如果没有这个命令，可以执行安装 yum -y install NetworkManager

图形配置

- 简易图形： nmtui
- 图形界面：nm-connection-editor

3.配置网络参数

实验：将虚拟机网卡设置为桥接模式，配置静态地址后能够访问外网

3.1 纯命令行配置静态 IP（无需依赖 `NetworkManager` 或 `systemd-networkd`，重启失效）

1. 查看当前网络状态

```
[root@xnha ~]# ip addr                      # 查看 IP、子网、网关等信息
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: ens160: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group
default qlen 1000
    link/ether 00:0c:29:5c:a4:ed brd ff:ff:ff:ff:ff:ff
    inet 192.168.119.128/24 brd 192.168.119.255 scope global dynamic
noprefixroute ens160
       valid_lft 1522sec preferred_lft 1522sec
    inet6 fe80::ac23:8439:4329:9fe2/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: ens192: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group
default qlen 1000
    link/ether 00:0c:29:5c:a4:f7 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.23/24 brd 192.168.1.255 scope global dynamic noprefixroute
ens192
       valid_lft 258922sec preferred_lft 258922sec
    inet6 fe80::4d93:c232:2460:e4f3/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
[root@xnha ~]# ip route                     # 查看路由表
default via 192.168.119.2 dev ens160 proto dhcp metric 100
default via 192.168.1.1 dev ens192 proto dhcp metric 101
192.168.1.0/24 dev ens192 proto kernel scope link src 192.168.1.23 metric 101
192.168.119.0/24 dev ens160 proto kernel scope link src 192.168.119.128 metric
100

#ens160 的网关 192.168.119.2 是 NAT 模式下的宿主机网关。
#ens192 的网关 192.168.1.1 是桥接模式下的宿主机网关
[root@xnha ~]# cat /etc/resolv.conf       # 查看 DNS 配置
```

```
# Generated by NetworkManager
nameserver 114.114.114.114
nameserver 192.168.1.1
```

---

2. 手动分配静态 IP

查看物理机IP，确保虚拟机和物理机处于同一网段

```
[root@xnha ~]# ip addr add 192.168.1.100/24 dev ens192  # 分配静态 IP + 子网掩码
```

验证：ping 物理机地址

```
[root@xnha ~]# ip addr show ens192 |grep 192.168.1.100
inet 192.168.1.100/24 scope global secondary ens192  # 确认 IP 已分配
```

---

3. 设置默认网关

```
[root@xnha ~]# ip route add default via 192.168.1.1 dev ens192  # 添加默认路由
```

验证：

```
[root@xnha ~]# ping -c4 8.8.8.8                          #谷歌的dns
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=53 time=127 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=53 time=50.7 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=53 time=50.0 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=53 time=50.7 ms
```

---

4. 配置 DNS 服务器

```
[root@xnha ~]# vim /etc/resolv.conf
# Generated by NetworkManager
# nameserver 192.168.1.1
nameserver 114.114.114.114
```

验证：

```
[root@xnha ~]# ping www.qq.com            # 测试 DNS 解析
PING ins-r23tsuuf.ias.tencent-cloud.net (111.30.178.240) 56(84) bytes of data.
64 bytes from 111.30.178.240 (111.30.178.240): icmp_seq=1 ttl=249 time=6.22 ms
64 bytes from 111.30.178.240 (111.30.178.240): icmp_seq=2 ttl=249 time=6.74 ms
64 bytes from 111.30.178.240 (111.30.178.240): icmp_seq=3 ttl=249 time=25.1 ms
64 bytes from 111.30.178.240 (111.30.178.240): icmp_seq=4 ttl=249 time=6.15 ms
```

5.重启不失效的解决方案

将命令写进 `/etc/rc.d/rc.loca` 配置文件

`etc/rc.d` 配置文件 是Linux 系统中与 **初始化脚本（Init Scripts）** 和 **服务管理** 相关的核心目录。

`etc/rc.d/rc.rclocal` 是自定义启动命令的文件(开机启动脚本，基本已弃用)

```
[root@xnha rc.d]# cat rc.local
#!/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
```

```
[root@xnha ~]# vim /etc/rc.d/rc.local
```

```
ip addr add 192.168.1.100/24 dev ens192
ip route add default via 192.168.1.1
echo "namesever 114.114.114.114">> /etc/resolv.conf
```

```
文件(F)  编辑(E)  查看(V)  搜索(S)  终端(T)  帮助(H)
#!/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
ip addr add 192.168.1.100/24 dev ens192
ip route add default via 192.168.1.1
echo "namesever 114.114.114.114">> /etc/resolv.conf
~
~
~
~
~
~
~
~
~
~
-- 插入 --                                          16,52         全部
```

保存退出后，给该文件添加执行权限

```
[root@xnha ~]# chmod +x /etc/rc.d/rc.local
```

重启后，配置成功，删除执行权限

```
[root@xnha ~]# chmod -x /etc/rc.d/rc.local
```

```
[root@xnha ~]# ip addr
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: ens160: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default q
len 1000
    link/ether 00:0c:29:5c:a4:ed brd ff:ff:ff:ff:ff:ff
3: ens192: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 00:0c:29:5c:a4:f7 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.100/24 scope global ens192
       valid_lft forever preferred_lft forever
[root@xnha ~]# chmod -x /etc/rc.d/rc.local
[root@xnha ~]#
```

3.2 修改网卡的配置文件`/etc/sysconfig/network-scripts/ifcfg-ensxx`

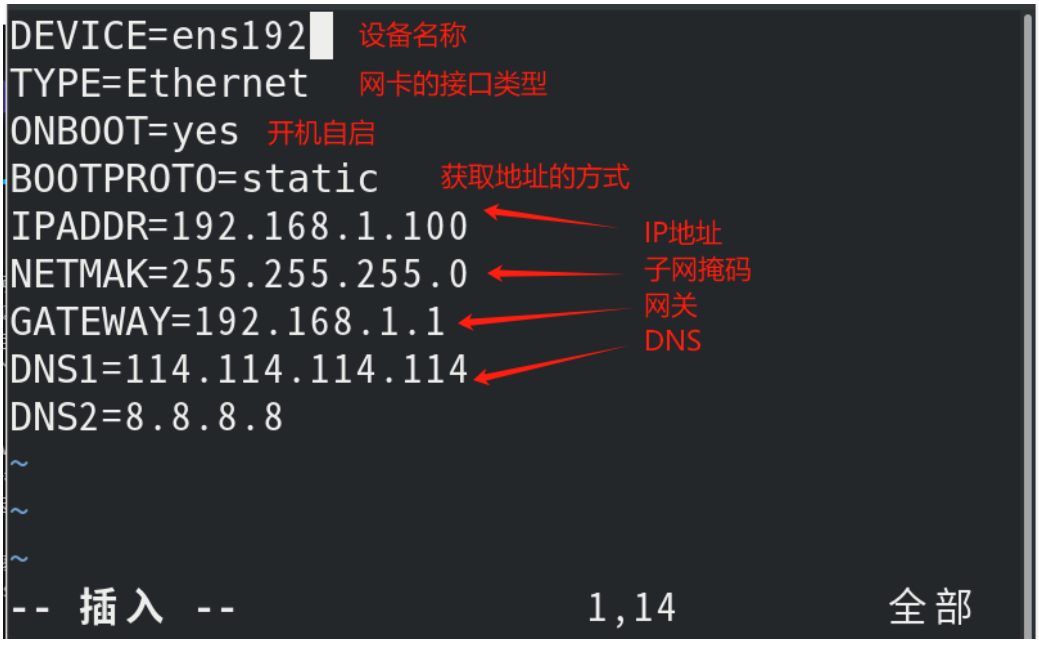
1.先备份网卡配置信息，在修改

- 先备份网卡配置文件，在修改

```
[root@xnha network-scripts]# ls
ifcfg-ens160  ifcfg-ens192
[root@xnha network-scripts]# mv ifcfg-ens192 ifcfg-ens192.bak
[root@xnha network-scripts]# touch ifcfg-ens192
[root@xnha network-scripts]# ls
ifcfg-ens160  ifcfg-ens192  ifcfg-ens192.bak
```

2.编辑配置文件，每个参数都要背下来！！！

- 最简化配置参数



3.重启后验证

```
[root@xnha ~]# ip addr
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: ens160: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group
default qlen 1000
    link/ether 00:0c:29:5c:a4:ed brd ff:ff:ff:ff:ff:ff
3: ens192: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group
default qlen 1000
    link/ether 00:0c:29:5c:a4:f7 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.100/24 brd 192.168.1.255 scope global noprefixroute ens192
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe5c:a4f7/64 scope link
       valid_lft forever preferred_lft forever
[root@xnha ~]# ping www.baidu.com
PING www.a.shifen.com (39.156.70.239) 56(84) bytes of data.
64 bytes from 39.156.70.239 (39.156.70.239): icmp_seq=1 ttl=53 time=10.8 ms
64 bytes from 39.156.70.239 (39.156.70.239): icmp_seq=2 ttl=53 time=12.1 ms
64 bytes from 39.156.70.239 (39.156.70.239): icmp_seq=3 ttl=53 time=10.7 ms
64 bytes from 39.156.70.239 (39.156.70.239): icmp_seq=4 ttl=53 time=10.8 ms
```

3.3 nmcli 命令配置IP（基于NetworkManager的管理工具） network /etc/sysconfig/network-scripts/ifcfg-ensxx

1. 查看所有网络连接

```
[root@xnha ~]# nmcli connection show
```

输出示例:

```
NAME           UUID                                  TYPE      DEVICE
System ens192  03da7500-2101-c722-2438-d0d006c28c73  ethernet  ens192
ens160         aaca87e6-f62a-4693-a09b-1ac25d77b8f2  ethernet  --
```

2. 查看特定连接详情

```
nmcli connection show "ens192"
```

3. 启用/禁用接口

```
# 启用 eth0
nmcli connection up eth0

# 禁用 wlan0
nmcli connection down wlan0
```

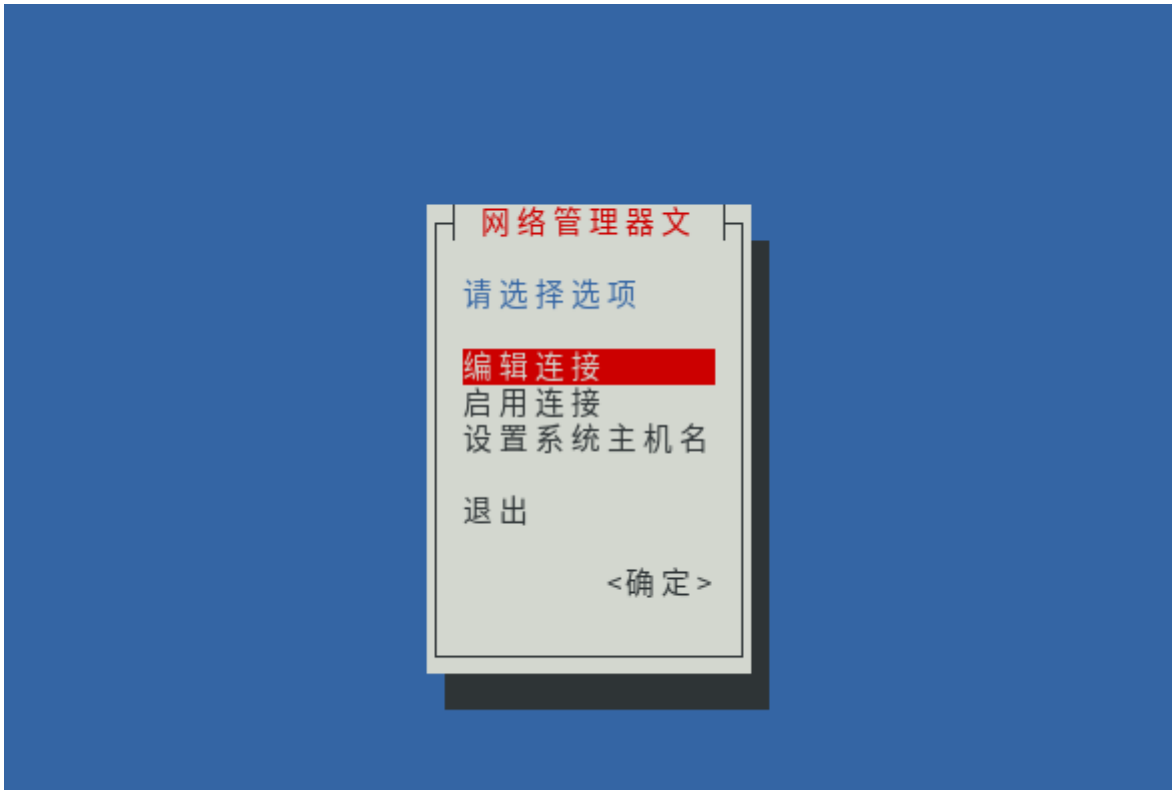
4. 配置静态 IP

```
# 设置 eth0 为静态 IP 192.168.1.100/24，网关 192.168.1.1
nmcli connection modify eth0 \
  type ethernet \
  ip4 192.168.1.100/24 \
  gateway4 192.168.1.1 \
  dns "8.8.8.8;8.8.4.4"
```

5. 配置 DHCP

```
# 设置 eth0 为 DHCP 自动获取 IP
nmcli connection modify eth0 \
  type ethernet \
  ip4 auto dhcp
```

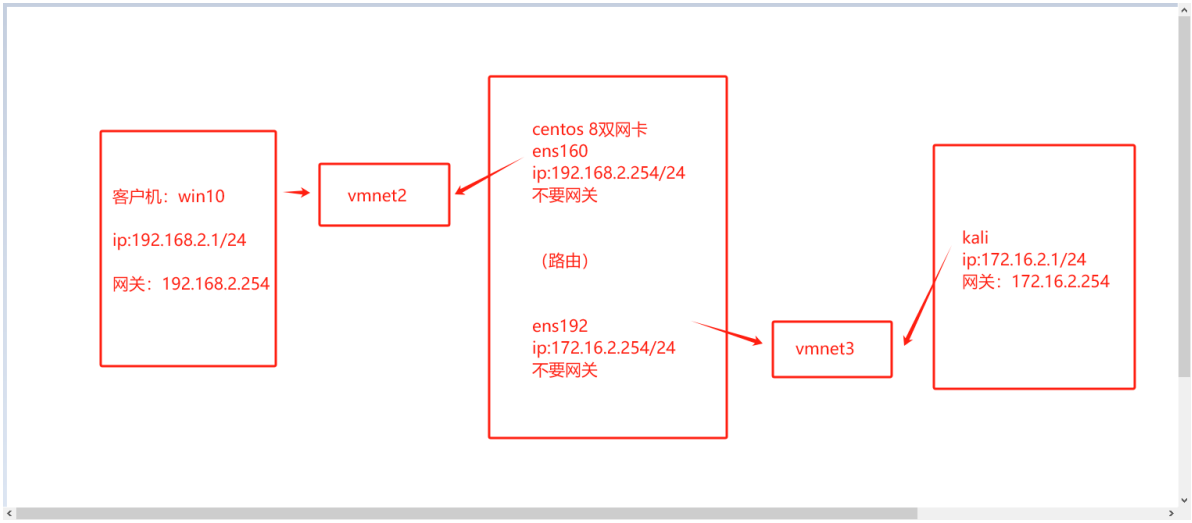
3.4 nmtui-图形化配置



四、练习

1.拓扑说明

- 客户机为windows 10操作系统，选择vmnet2 网卡
- centos 8 为双网卡，分别连接vmnet2和vmnet3
- kali 选择vmnet3网卡



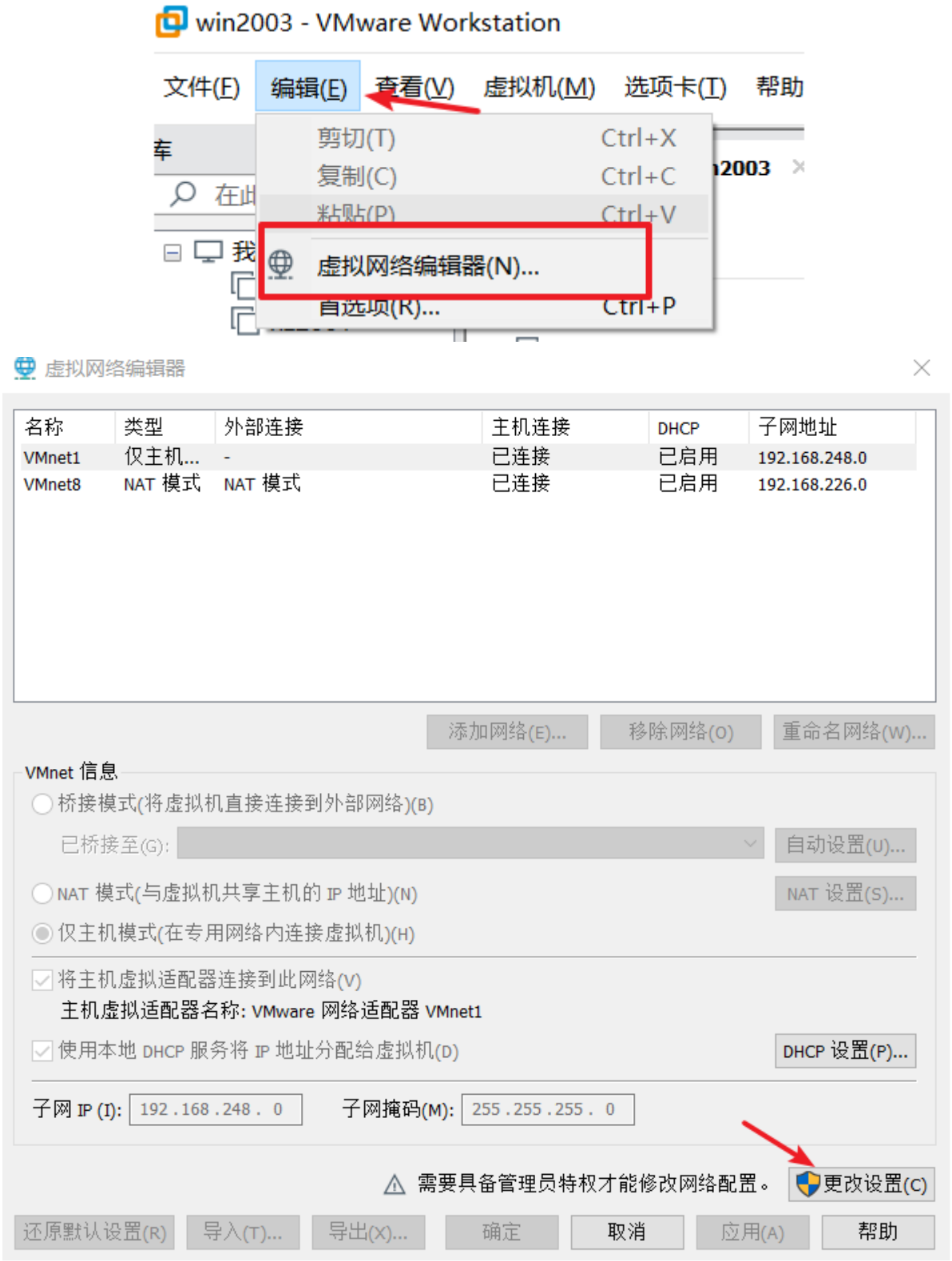
2.目标

192.168.2.1和172.16.2.1能够互相Ping通

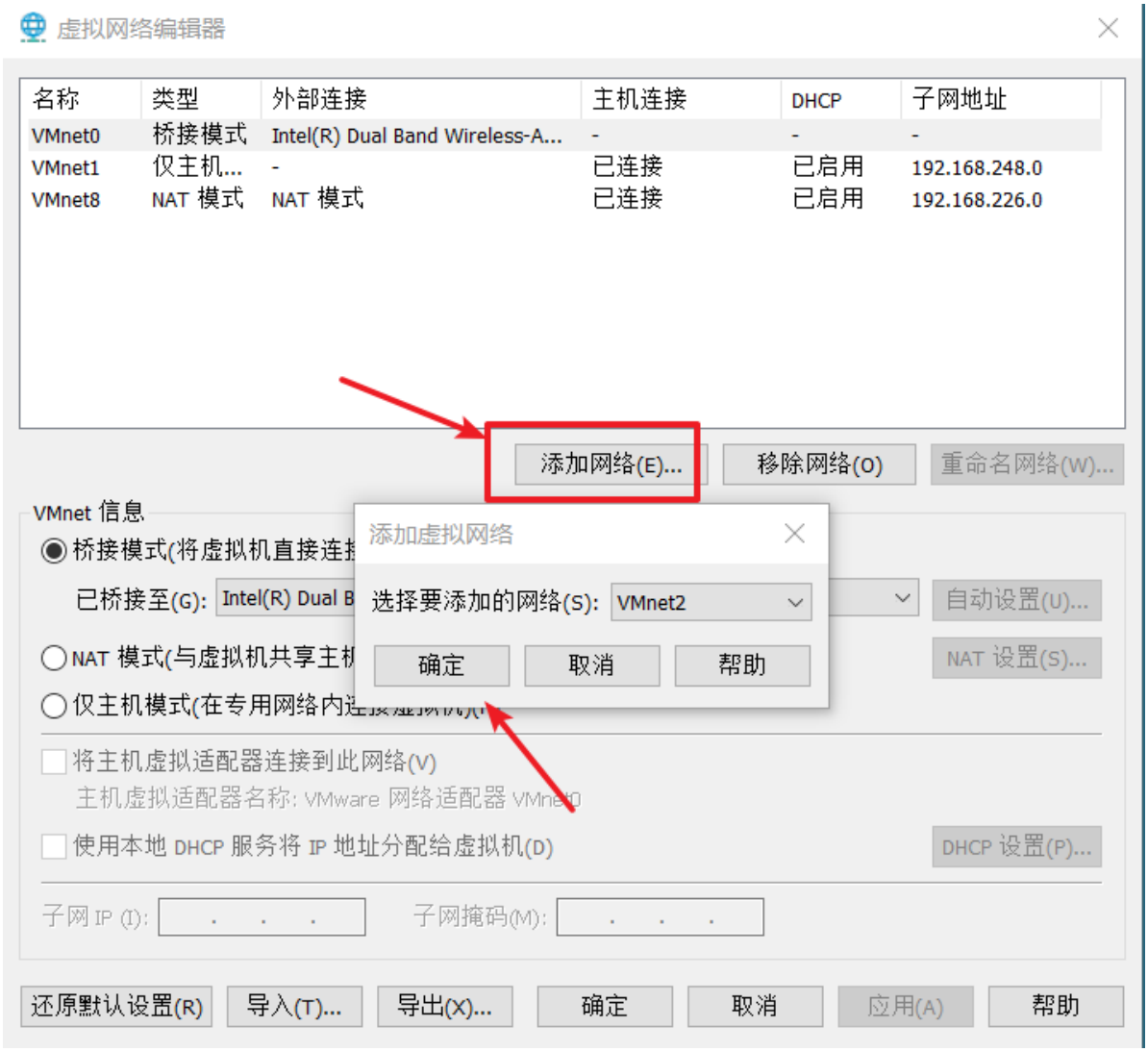
3.操作

1.实验之前先确定一下自己虚拟机是否存在vment2和vmnet3

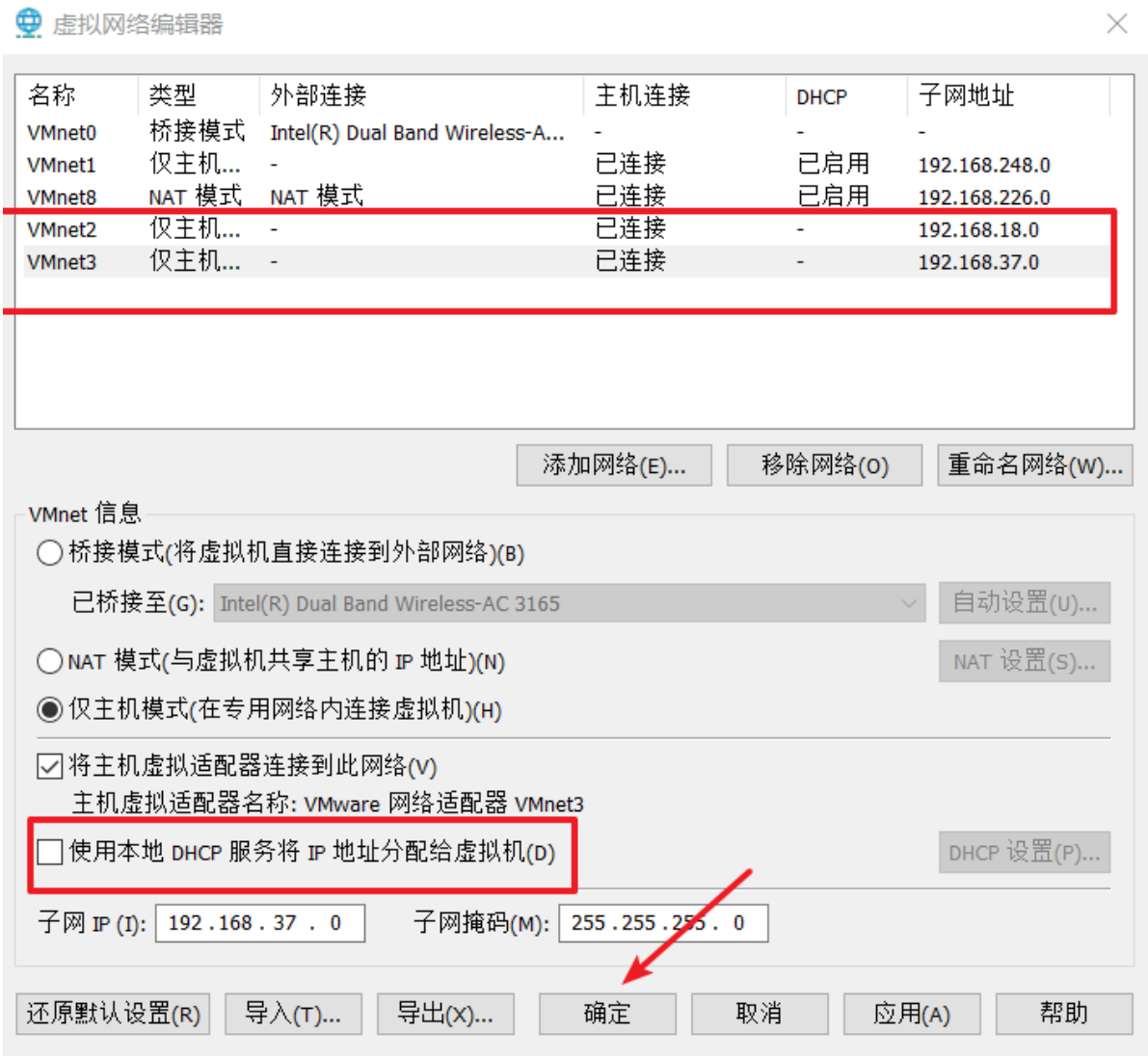
- 手动添加vmnet2 和 3


win2003 - VMware Workstation
文件(F)
编辑(E)
查看(V)
虚拟机(M)
选项卡(T)
帮助
剪切(T) Ctrl+X
复制(C) Ctrl+C
粘贴(P) Ctrl+V
虚拟网络编辑器(N)...
首选项(R)... Ctrl+P
虚拟网络编辑器
名称 类型 外部连接 主机连接 DHCP 子网地址
VMnet1 仅主机... - 已连接 已启用 192.168.248.0
VMnet8 NAT 模式 NAT 模式 已连接 已启用 192.168.226.0
添加网络(E)...
移除网络(O)
重命名网络(W)...
VMnet 信息
桥接模式(将虚拟机直接连接到外部网络)(B)
已桥接至(G):
自动设置(U)...
NAT 模式(与虚拟机共享主机的 IP 地址)(N)
NAT 设置(S)...
仅主机模式(在专用网络内连接虚拟机)(H)
将主机虚拟适配器连接到此网络(V)
主机虚拟适配器名称: VMware 网络适配器 VMnet1
使用本地 DHCP 服务将 IP 地址分配给虚拟机(D)
DHCP 设置(P)...
子网 IP (I): 192 . 168 . 248 . 0
子网掩码(M): 255 . 255 . 255 . 0
需要具备管理员特权才能修改网络配置。
更改设置(C)
还原默认设置(R)
导入(T)...
导出(X)...
确定
取消
应用(A)
帮助

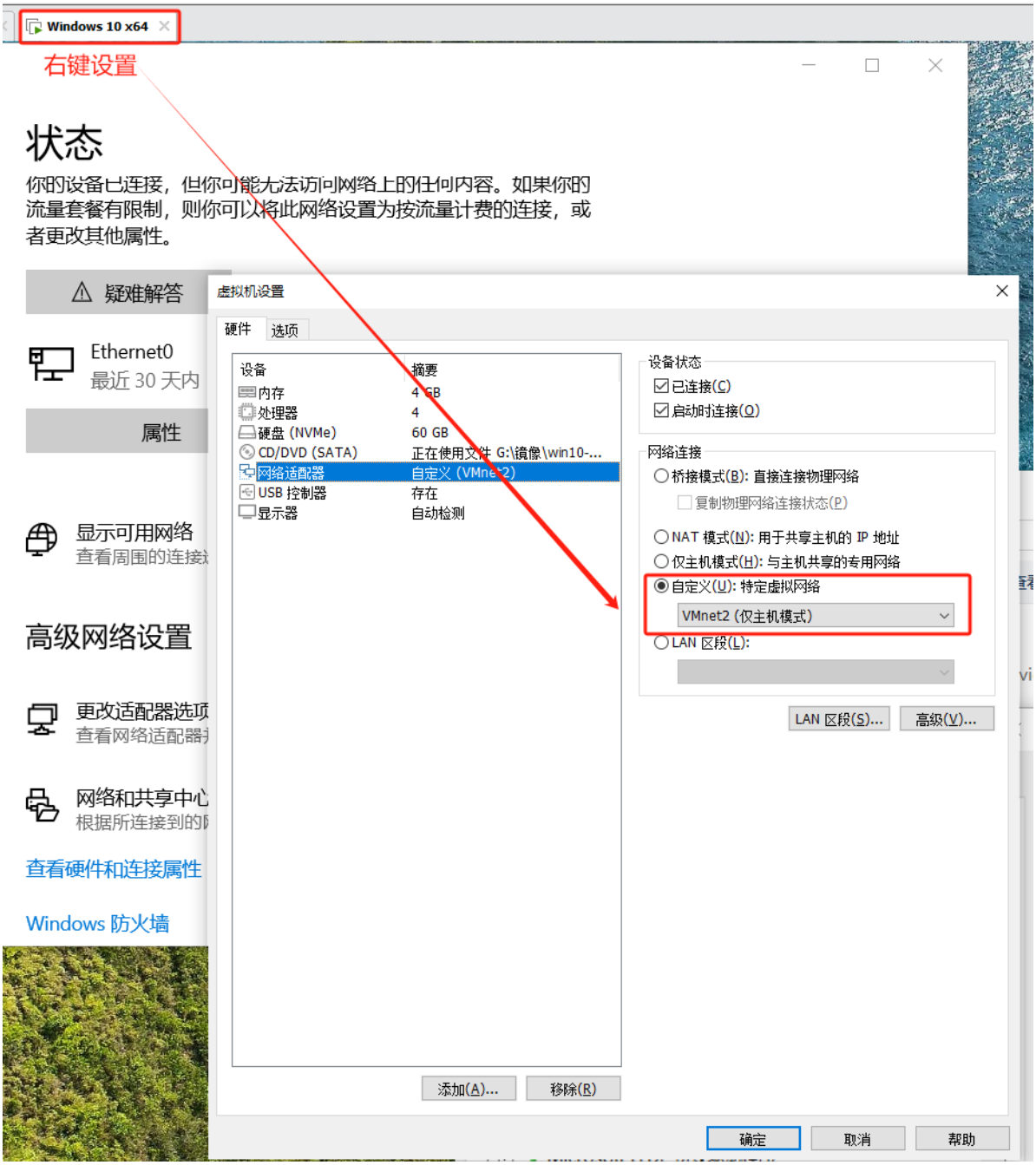
- 选择添加

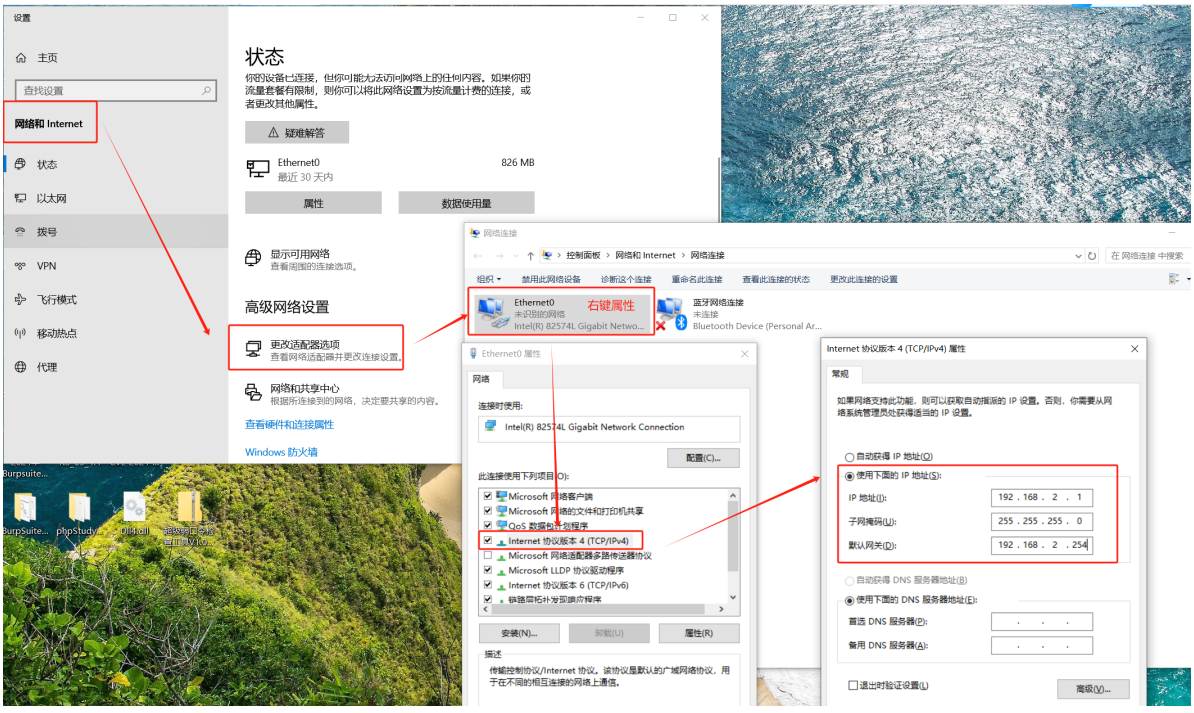


- 以相同的配置建立vmnet3



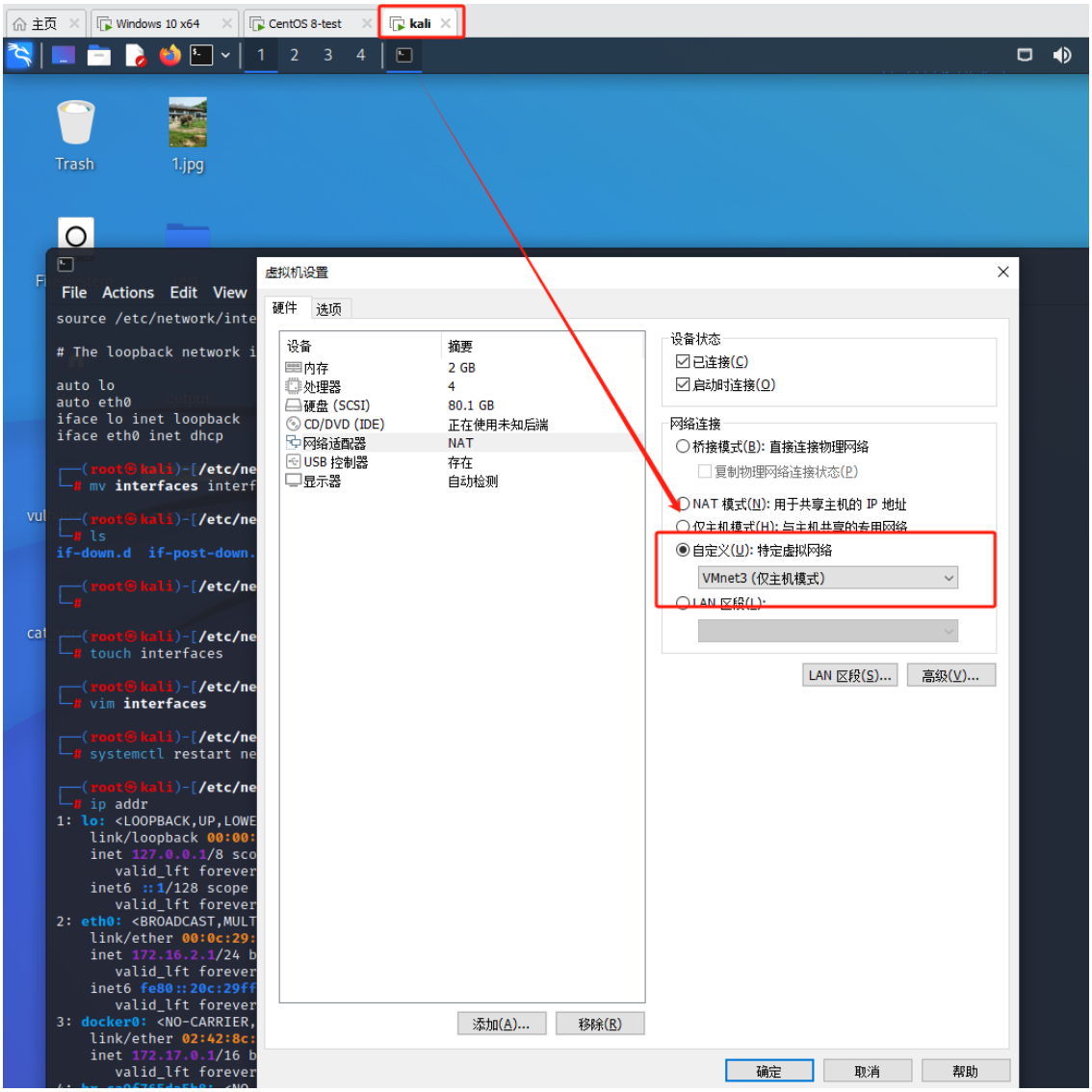
2.针对客户机win10确认连接的交换机为vmnet2同时配置ip


Windows 10 x64
右键设置
状态
你的设备已连接，但你可能无法访问网络上的任何内容。如果你的流量套餐有限制，则你可以将此网络设置为按流量计费的连接，或者更改其他属性。
疑难解答
Ethernet0
最近 30 天内
属性
显示可用网络
查看周围的连接
高级网络设置
更改适配器选项
查看网络适配器
网络和共享中心
根据所连接到的
查看硬件和连接属性
Windows 防火墙
虚拟机设置
硬件
选项
设备
摘要
内存
4 GB
处理器
4
硬盘 (NVMe)
60 GB
CD/DVD (SATA)
正在使用文件 G:\镜像\win10-...
网络适配器
自定义 (VMnet2)
USB 控制器
存在
显示器
自动检测
添加(A)...
移除(R)
设备状态
已连接(C)
启动时连接(O)
网络连接
桥接模式(B): 直接连接物理网络
复制物理网络连接状态(P)
NAT 模式(N): 用于共享主机的 IP 地址
仅主机模式(H): 与主机共享的专用网络
自定义(U): 特定虚拟网络
VMnet2 (仅主机模式)
LAN 区段(L):
LAN 区段(S)...
高级(V)...
确定
取消
帮助


设置
主页
查找设置
网络和 Internet
状态
以太网
拨号
VPN
飞行模式
移动热点
代理
状态
你的设备已连接，但你可能无法访问网络上的任何内容。如果你的流量套餐有限制，则你可以将此网络设置为按流量计费的连接，或者更改其他属性。
疑难解答
Ethernet0
最近 30 天内
826 MB
属性
数据使用量
显示可用网络
查看周围的连接选项。
高级网络设置
更改适配器选项
查看网络适配器并更改连接设置。
网络和共享中心
根据所连接到的网络，决定要共享的内容。
查看硬件和连接属性
Windows 防火墙
网络连接
控制面板 › 网络和 Internet › 网络连接
Ethernet0
右键属性
Intel(R) 82574L Gigabit Netwo...
蓝牙网络连接
未连接
Bluetooth Device (Personal Ar...
Ethernet0 属性
网络
连接时使用:
Intel(R) 82574L Gigabit Network Connection
配置(C)...
此连接使用下列项目(O):
Internet 协议版本 4 (TCP/IPv4)
安装(N)...
卸载(U)
属性(R)
描述
Internet 协议版本 4 (TCP/IPv4) 属性
常规
自动获得 IP 地址(O)
使用下面的 IP 地址(S):
IP 地址(I):
192 . 168 . 2 . 1
子网掩码(U):
255 . 255 . 255 . 0
默认网关(D):
192 . 168 . 2 . 254
自动获得 DNS 服务器地址(B)
使用下面的 DNS 服务器地址(E):
首选 DNS 服务器(P):
备用 DNS 服务器(A):
退出时验证设置(L)
高级(V)...

3.kali 配置IP

- 将kali网络连接方式修改为vmnet3



- 将kali原本的配置文件`/etc/network/interfaces`备份为`/etc/network/interfaces.bak

```
┌──(root㉿kali)-[/etc/network]
└─# mv interfaces interfaces.bak

┌──(root㉿kali)-[/etc/network]
└─# ls
if-down.d  if-post-down.d  if-pre-up.d  if-up.d  interfaces.bak  interfaces.d

┌──(root㉿kali)-[/etc/network]
└─#
```

- 新建interfaces

```
┌──(root㉿kali)-[/etc/network]
└─# touch interfaces

┌──(root㉿kali)-[/etc/network]
└─# vim interfaces
```

- 编辑文件，写入IP配置，保存退出

```
File  Actions  Edit  View  Help

# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface

auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
address 172.16.2.1
netmask 255.255.255.0
gateway 172.16.2.254
~
~
~
~
```

- 重启服务

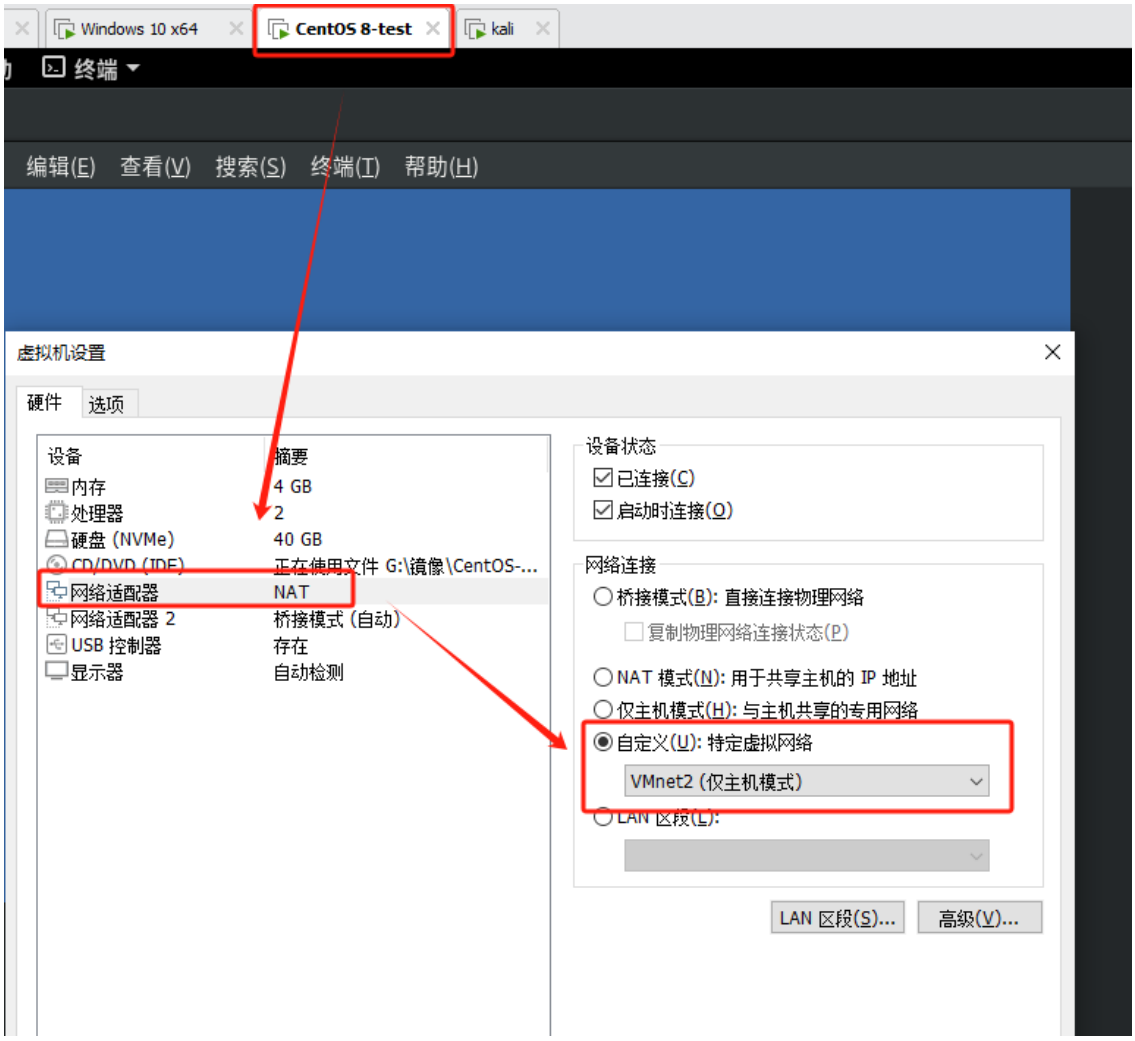
```
┌──(root㉿kali)-[/etc/network]
└─# systemctl restart networking

┌──(root㉿kali)-[/etc/network]
└─#
```

- 查看配置

```
┌──(root㉿kali)-[/etc/network]
└─# ip addr
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 noprefixroute
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:cd:6c:d6 brd ff:ff:ff:ff:ff:ff
    inet 172.16.2.1/24 brd 172.16.2.255 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fecd:6cd6/64 scope link
       valid_lft forever preferred_lft forever
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:8c:3e:84:84 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
       valid_lft forever preferred_lft forever
4: br-ca9f765da5b8: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:a8:8e:34:dd brd ff:ff:ff:ff:ff:ff
    inet 172.19.0.1/16 brd 172.19.255.255 scope global br-ca9f765da5b8
       valid_lft forever preferred_lft forever
```

4.centos8双网卡配置

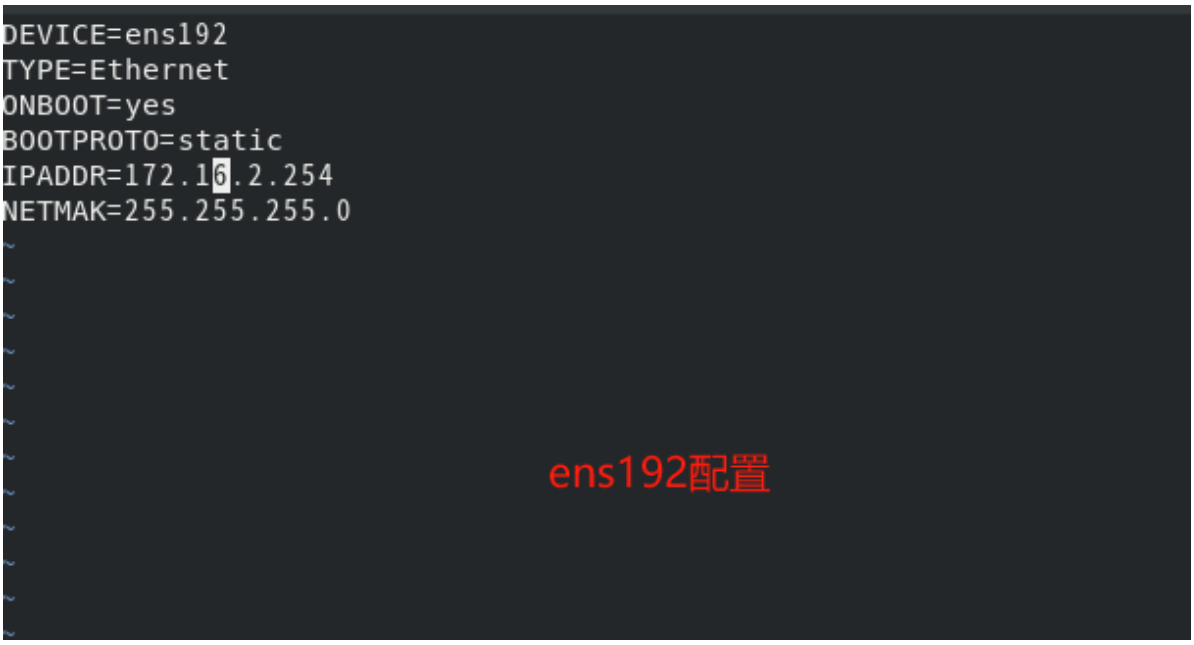
- 将ens160设置为vmnet2,将ens192设置为vmnet3



- 将两个网卡的配置文件备份，并创建新的配置文件

```
[root@xnha system-connections]# cd /etc/sysconfig/network-scripts/
[root@xnha network-scripts]# ls
ifcfg-ens160  ifcfg-ens192  ifcfg-ens192.bak
[root@xnha network-scripts]# mv ifcfg-ens160 ifcfg-ens160.bak
[root@xnha network-scripts]# touch ifcfg-ens160
[root@xnha network-scripts]# vim ifcfg-ens160
```

- 编辑网卡配置文件为目标要求



5.测试各自网卡连通性

- kali和ens192能够互相ping通

```
┌──(root㉿kali)-[/etc/network]
└─# ping 172.16.2.254
PING 172.16.2.254 (172.16.2.254) 56(84) bytes of data.
64 bytes from 172.16.2.254: icmp_seq=1 ttl=64 time=0.596 ms
64 bytes from 172.16.2.254: icmp_seq=2 ttl=64 time=0.542 ms
64 bytes from 172.16.2.254: icmp_seq=3 ttl=64 time=0.352 ms
64 bytes from 172.16.2.254: icmp_seq=4 ttl=64 time=0.472 ms
64 bytes from 172.16.2.254: icmp_seq=5 ttl=64 time=0.538 ms
^C
── 172.16.2.254 ping statistics ──
5 packets transmitted, 5 received, 0% packet loss, time 4092ms
rtt min/avg/max/mdev = 0.352/0.500/0.596/0.083 ms
```

```
[root@xnha ~]# ping 172.16.2.1
PING 172.16.2.1 (172.16.2.1) 56(84) bytes of data.
64 bytes from 172.16.2.1: icmp_seq=1 ttl=64 time=0.359 ms
64 bytes from 172.16.2.1: icmp_seq=2 ttl=64 time=0.328 ms
64 bytes from 172.16.2.1: icmp_seq=3 ttl=64 time=0.315 ms
^C
```

- win10和ens160能够互相ping通

```
C:\Users\freeplus>ping 192.168.2.254

正在 Ping 192.168.2.254 具有 32 字节的数据:
来自 192.168.2.254 的回复: 字节=32 时间<1ms TTL=64
来自 192.168.2.254 的回复: 字节=32 时间<1ms TTL=64
来自 192.168.2.254 的回复: 字节=32 时间<1ms TTL=64
来自 192.168.2.254 的回复: 字节=32 时间<1ms TTL=64

192.168.2.254 的 Ping 统计信息:
    数据包: 已发送 = 4，已接收 = 4，丢失 = 0 (0% 丢失)，
往返行程的估计时间(以毫秒为单位):
    最短 = 0ms，最长 = 0ms，平均 = 0ms
```

```
[root@xnha ~]# ping 192.168.2.1
PING 192.168.2.1 (192.168.2.1) 56(84) bytes of data.
64 bytes from 192.168.2.1: icmp_seq=1 ttl=128 time=0.384 ms
64 bytes from 192.168.2.1: icmp_seq=2 ttl=128 time=0.368 ms
64 bytes from 192.168.2.1: icmp_seq=3 ttl=128 time=0.317 ms
64 bytes from 192.168.2.1: icmp_seq=4 ttl=128 time=0.372 ms
^C
```

- kali 和 win10 无法 互相ping通

6.配置路由转发

6.1临时开启

- 修改内核配置 `/proc/sys/net/ipv4/ip_forward`

```
[root@xnha ~]# cat /proc/sys/net/ipv4/ip_forward
0
[root@xnha ~]#
```

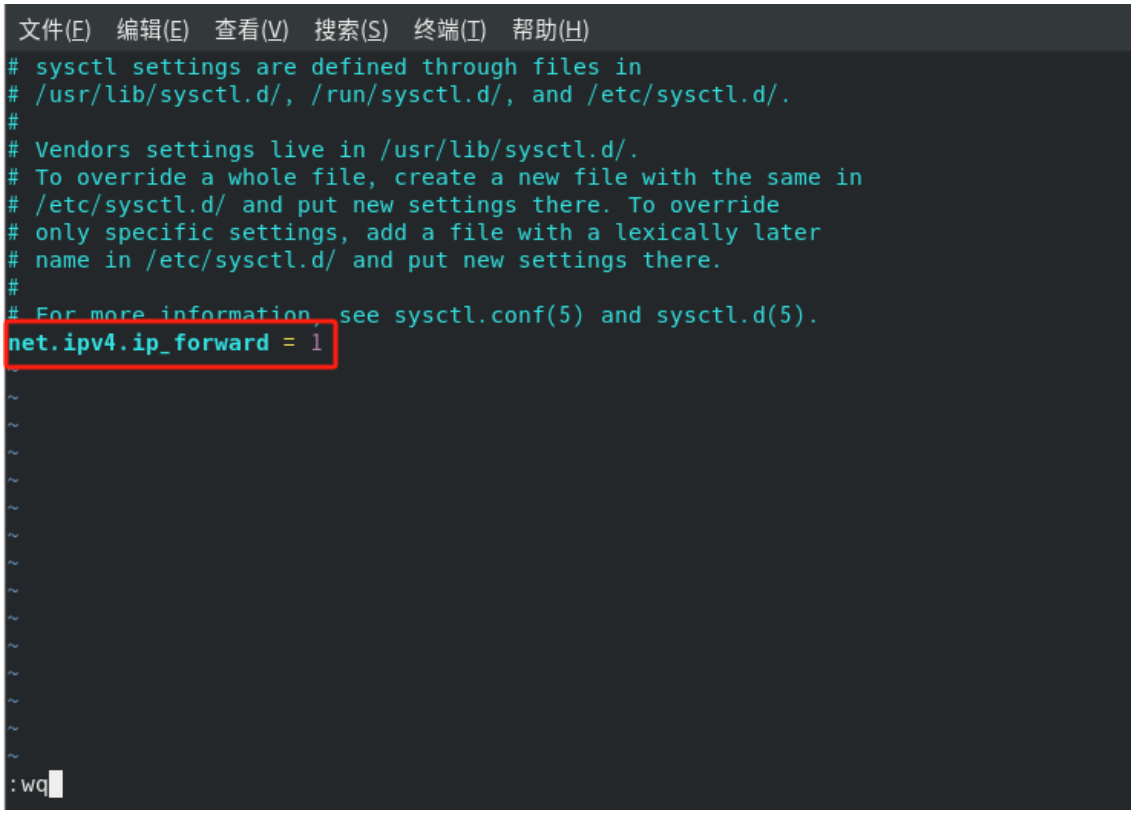
- 通过echo编辑该文件

```
[root@xnha ~]# echo 1 > /proc/sys/net/ipv4/ip_forward
[root@xnha ~]# cat /proc/sys/net/ipv4/ip_forward
1
[root@xnha ~]#
```

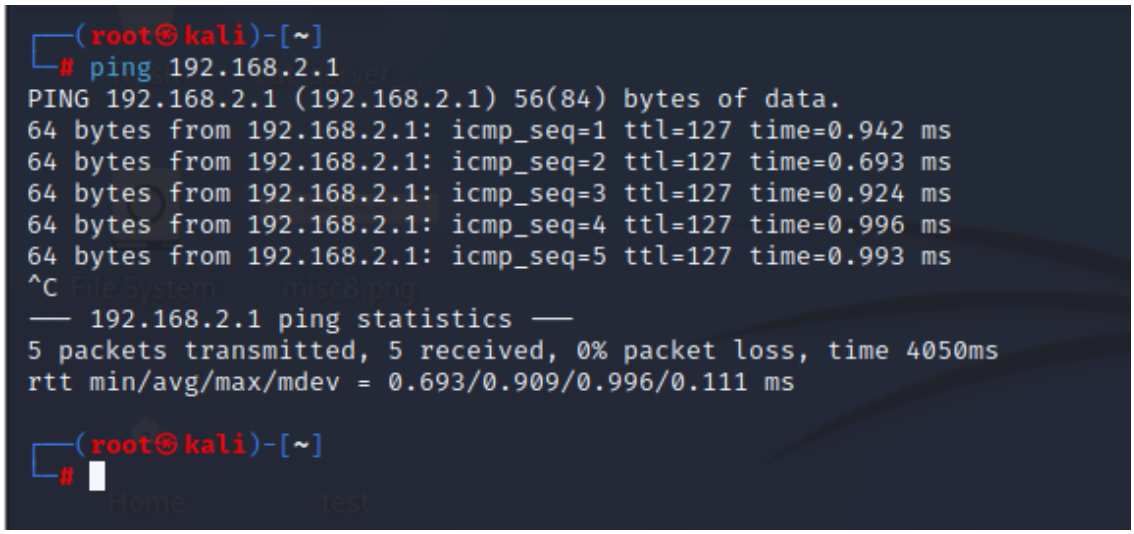
- 此时kali能够ping通win10,但重启后失效

6.2 永久生效

- 修改配置文件 `/etc/sysctl.conf`


文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
sysctl settings are defined through files in
/usr/lib/sysctl.d/, /run/sysctl.d/, and /etc/sysctl.d/.
#
Vendors settings live in /usr/lib/sysctl.d/.
To override a whole file, create a new file with the same in
/etc/sysctl.d/ and put new settings there. To override
only specific settings, add a file with a lexically later
name in /etc/sysctl.d/ and put new settings there.
#
# For more information, see sysctl.conf(5) and sysctl.d(5).
net.ipv4.ip_forward = 1
:wq

- 测试


(root㉿kali)-[~]
ping 192.168.2.1
PING 192.168.2.1 (192.168.2.1) 56(84) bytes of data.
64 bytes from 192.168.2.1: icmp_seq=1 ttl=127 time=0.942 ms
64 bytes from 192.168.2.1: icmp_seq=2 ttl=127 time=0.693 ms
64 bytes from 192.168.2.1: icmp_seq=3 ttl=127 time=0.924 ms
64 bytes from 192.168.2.1: icmp_seq=4 ttl=127 time=0.996 ms
64 bytes from 192.168.2.1: icmp_seq=5 ttl=127 time=0.993 ms
^C
— 192.168.2.1 ping statistics —
5 packets transmitted, 5 received, 0% packet loss, time 4050ms
rtt min/avg/max/mdev = 0.693/0.909/0.996/0.111 ms
(root㉿kali)-[~]
#