

# 实验4

## 静态路由配置

主讲：王信博

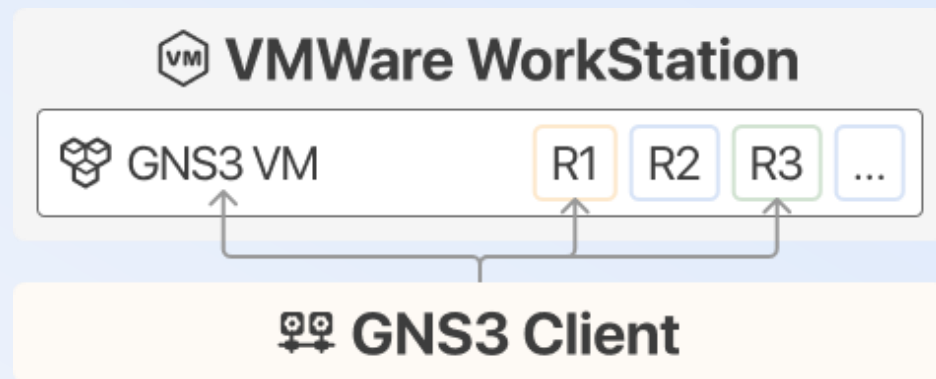


# PART 01

## 模拟环境准备

ENVIRONMENT PREPARATION

# GNS3 环境准备



## 必选安装:

**[客户端部分]** [zjucomp.net/docs/GNS3/client-install](http://zjucomp.net/docs/GNS3/client-install)

- GNS3 Client

**[虚拟机部分]** [zjucomp.net/docs/GNS3/VM-VMware](http://zjucomp.net/docs/GNS3/VM-VMware)

- VMWare WorkStation + GNS3 VM

**可选安装:** Tabby终端+Putty+实验终端辅助插件

**[辅助插件]** [zjucomp.net/docs/terminal\\_helper](http://zjucomp.net/docs/terminal_helper)

# GNS3 虚拟机导入

## 启用虚拟化平台

同学们DB/OS课程实验都会用到WSL，且较多设备默认启用“基于虚拟化的安全”，此时Hyper-V特性会自动启动，为确保VMWare兼容性，需要启用**虚拟化平台**特性

## 配置GNS3虚拟机

- 一般已有“仅主机模式”、“NAT”模式2个网络适配器，请再添加1个**桥接模式**网络适配器
- 内存建议增加到4G左右（如内存充裕可增加更多），CPU核心建议增加到4个左右
- **处理器-取消勾选“虚拟化引擎”中全部项目！取消勾选“虚拟化引擎”中全部项目！！**

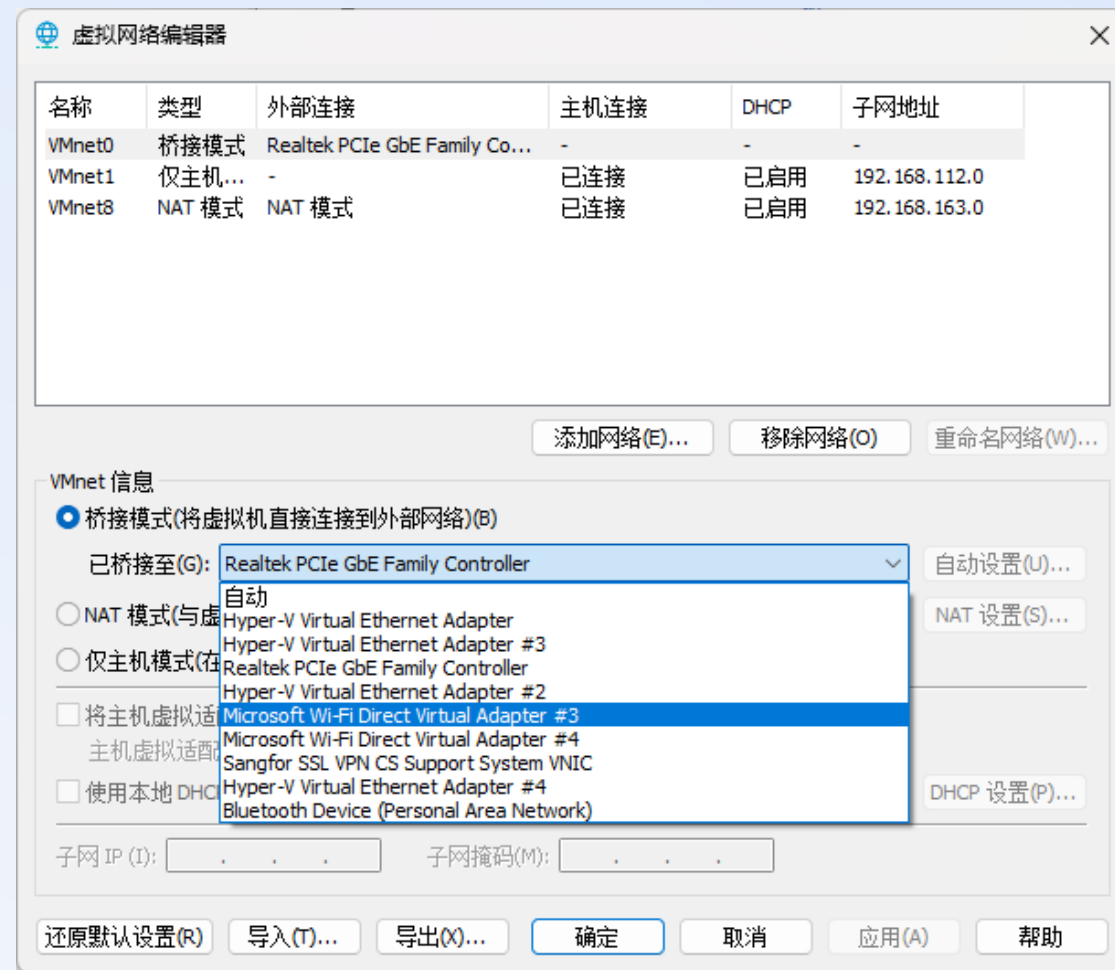
内存	4 GB
处理器	4
硬盘 (SCSI)	19.5 GB
硬盘 2 (SCSI)	488.3 GB
CD/DVD (IDE)	正在使用未知后端
网络适配器	仅主机模式
网络适配器 2	NAT
网络适配器 3	桥接模式 (自动)
显示器	自动检测

处理器	
处理器数量(P):	4
每个处理器的内核数量(C):	1
处理器内核总数:	4
虚拟化引擎	
<input type="checkbox"/> 虚拟化 Intel VT-x/EPT 或 AMD-V/RVI(V)	
<input type="checkbox"/> 虚拟化 CPU 性能计数器(U)	

# GNS3 虚拟机导入

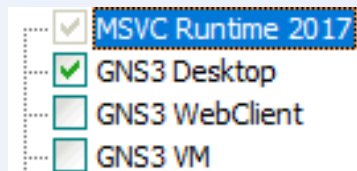
## 配置虚拟网络适配器

- VMware Workstation-编辑-虚拟网络编辑器
- 打开后点选更改设置，并授予管理员权限
- 选择VMnet0，将桥接的网络适配器修改为**实际使用的网卡**
- 如VMnet0/1/8对应的模式不对/子网地址为169.\*.\*.\*，请点还原默认设置进行重置
- 如找不到要桥接的网卡，请参考QA步骤



# GNS3 客户端安装

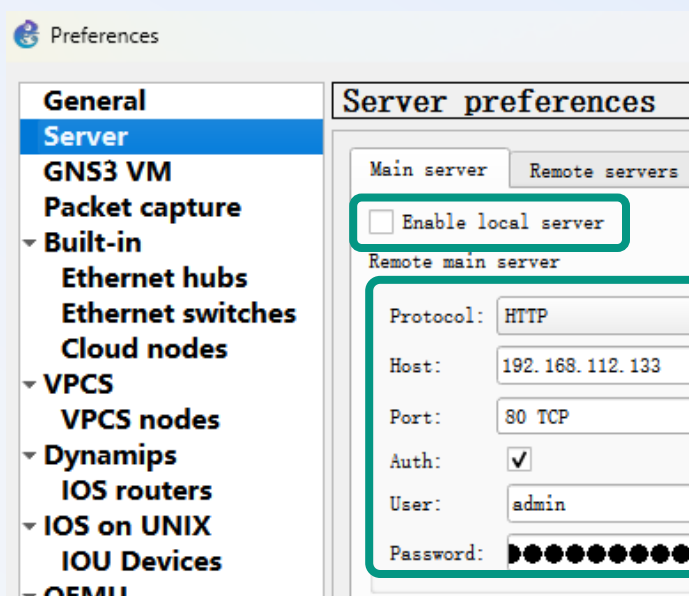
- 客户端和VM版本必须完全一致，不要选择3.0.0-3.0.3版本（存在Bug会导致系统卡死）
- 安装时取消勾选GNS3 VM，确保有GNS3 Desktop即可；不需要安装SolarWinds



- 安装完成后，打开Edit-Preference-Server，取消勾选“Enable Local Server”，并填入虚拟机中展示的IP与端口，密码通常保持默认即可

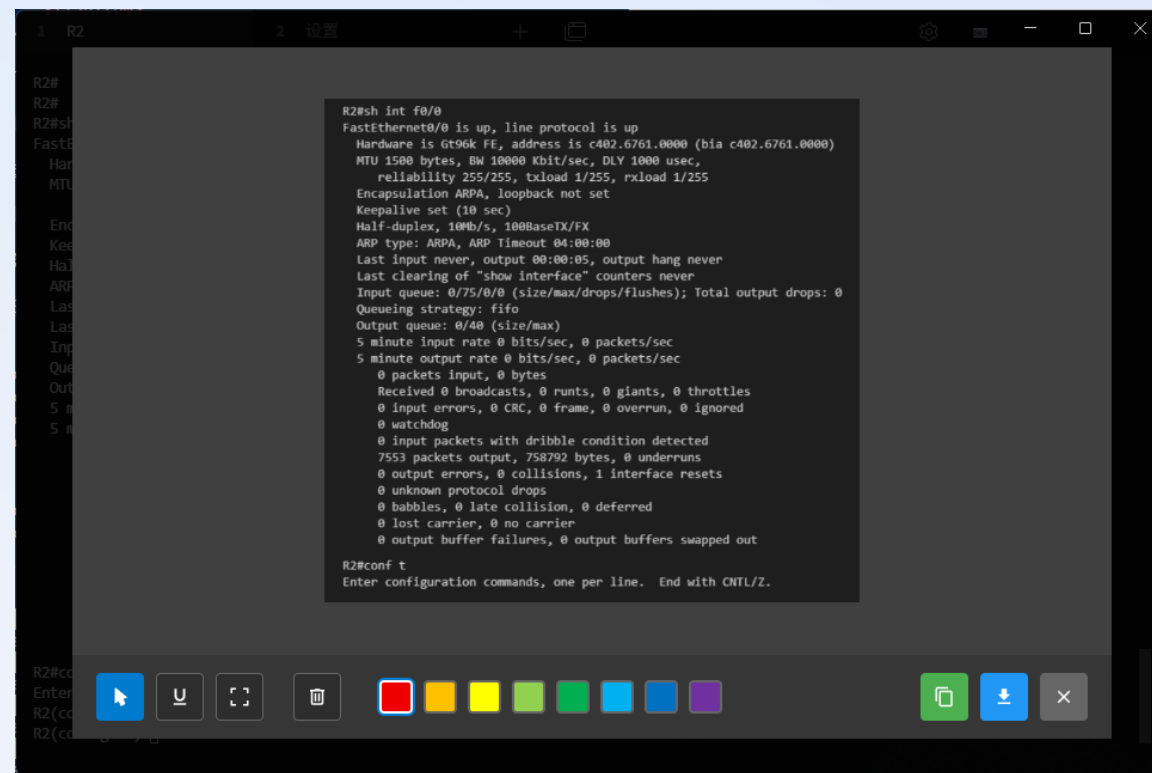
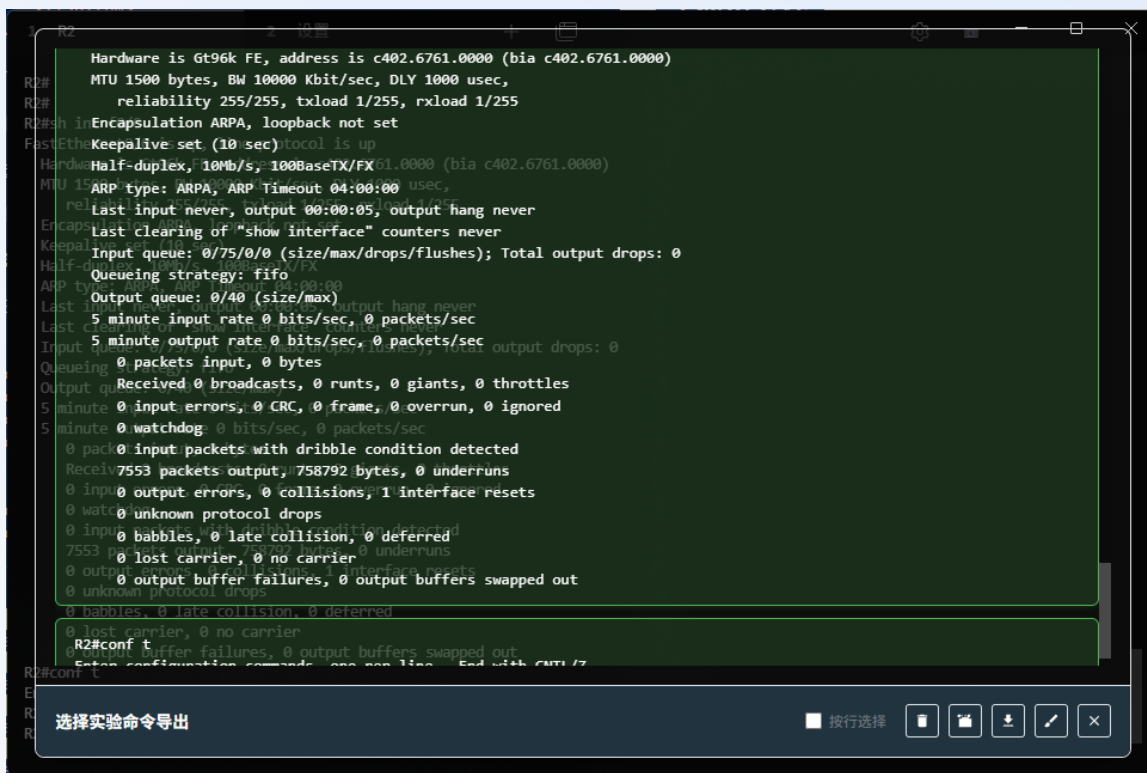
IP: 172.29.106.212 PORT: 80

To log in using SSH: ssh gns3@172.29.106.212  
Password: gns3



# 实验终端辅助插件

点击插件图标，会自动将终端输出按命令单独划分成块，点选可以直接复制/导出/标记





## 实验终端辅助插件

- 安装Putty、Tabby终端；从Github Release下载最新版本的插件压缩包并进行解压
- 打开Tabby终端，进入设置 (Settings) → 插件 (Plugins)，点击"插件目录"按钮，将解压好的插件复制到插件目录下的node\_modules目录（如果不存在，请先创建一个）
- 重启Tabby终端，此时应该能看到插件的图标



- 创建辅助脚本：以Windows为例，先找到PuTTY的安装目录，并找到plink.exe路径
  - 创建一个辅助脚本，保存在一个不含非法字符的路径：

```
@echo off  
"C:\Program Files\PuTTY\plink.exe" -telnet %1 -P %2
```
  - 打开GNS3的首选项，选择General - Console Applications，点击Edit
  - 选择Custom，填写"Tabby可执行文件路径" run "辅助脚本路径" %h %p"



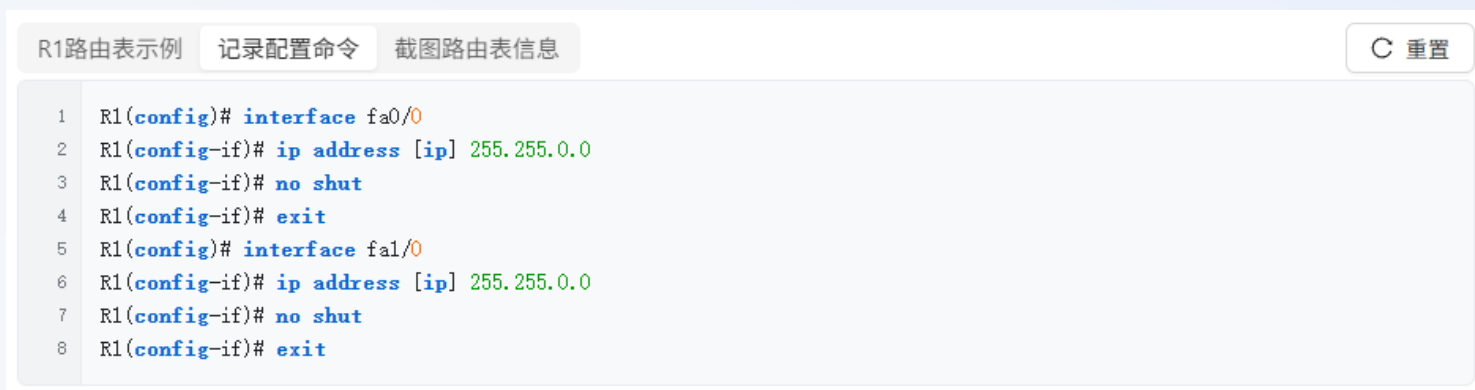
# 实验文档工具

可以在文档网页直接填写命令/粘贴截图，网页刷新后数据仍然保存  
完成后点击“导出实验报告”即可一键导出，无需反复切换窗口

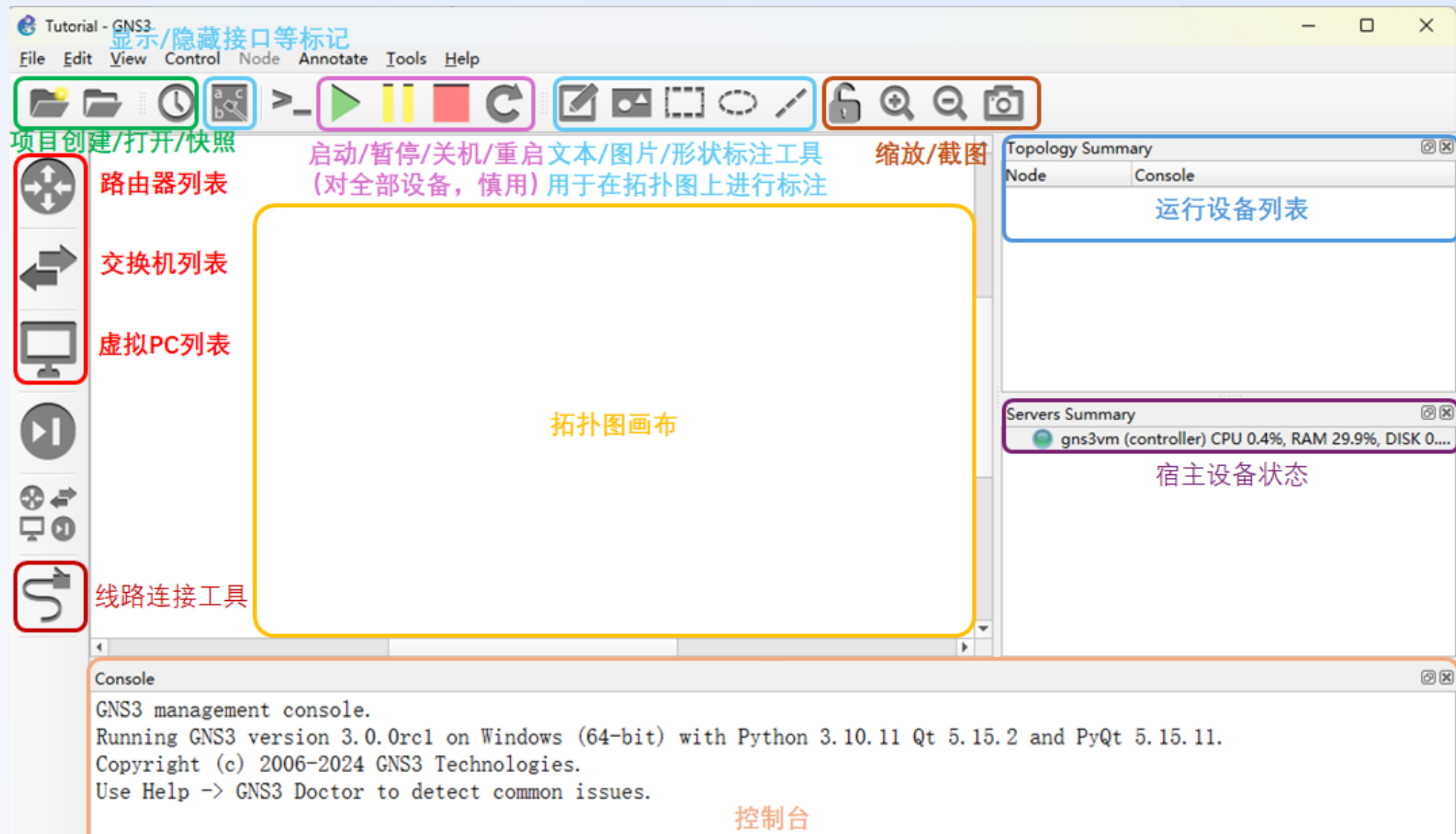


## 5 导出实验报告

导出实验报告

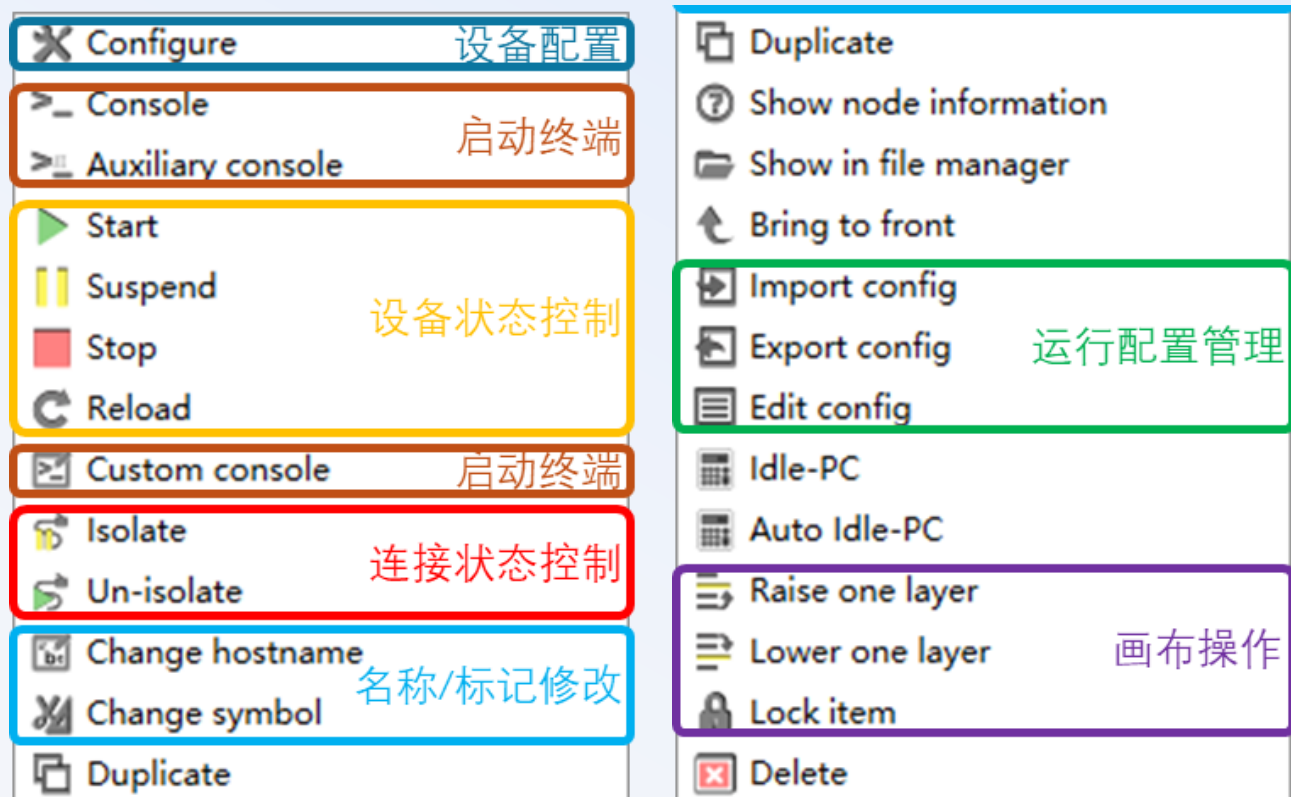


# GNS3 客户端使用



# GNS3 客户端使用

## 设备操作

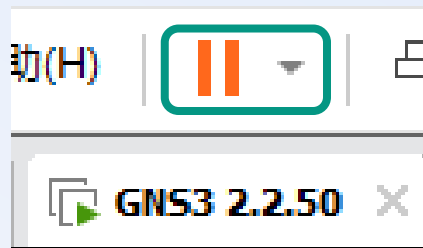


## 配置持久化

**一失足成“千古”恨！！！！！！**

未持久化时网络设备配置保存在内存  
断电 = 恢复出厂设置 = **全部白做了**

- vPC: save
- Router: write
- VMWare: 挂起VM



## 运行前注意:

关闭VPN (Clash/Atrust等), 关闭防火墙

# GNS3 客户端使用

## PC配置命令

仅配置IP: `ip [address]/[mask-length]`

`ip 10.0.0.11/16` 配置IP为10.0.0.11、16位子网掩码(255.255.0.0)

IP+网关: `ip [address]/[mask-length] [default-gateway]`

`ip 10.0.0.11/16 10.0.0.1` 配置IP为10.0.0.11、16位子网掩码(255.255.0.0), 默认网关10.0.0.1

**不要使用点分十进制的子网掩码**, 否则子网掩码会被当成默认网关

例如`ip 10.0.0.11 255.0.0.0`的效果是配置IP为10.0.0.11、默认的24位子网掩码(255.255.255.0), **默认网关255.0.0.0**

## PC查看配置好的IP

`show ip`

# PART 02

## 实验原理与背景

PRINCIPLE & BACKGROUND

# 名词解释

## 网线速率

Gigabit Ethernet: 1000Mbps

Fast Ethernet: 100Mbps

Ethernet: 10Mbps



## 串口

一种常用于广域网连接的接口类型，与以太网不同，通常用于**点对点**的长距离连接  
常用于连接地理位置分散的路由器，例如通过电信运营商提供的专线（如T1/E1线路）将不同城市的分支机构网络连接起来

串口的**速率通常较低**（如64Kbps或1.544Mbps），侧重于长距离的稳定传输  
如今在大多数广域网场景中已基本被速度更快、更可靠的光纤所取代



# HDLC协议

## HDLC协议

HDLC (High-Level Data Link Control) 是Cisco路由器串口上默认的数据链路层封装协议，是一个简单的、低开销的专有协议，用于连接两台Cisco设备的点对点链路

## 时钟速率 Clock Rate

串行通信中数据是一位一位地传输的，为了让接收端能够正确地同步和解码这些比特流，需要一个时钟信号来规定传输速率

物理连接中，提供时钟信号的一端被称为DCE (Data Communications Equipment, 数据通信设备)，而接收时钟信号进行同步的一端被称为DTE (Data Terminal Equipment, 数据终端设备)，`clock rate`命令必须在DCE端配置

实验中选取的64000是一个串行链路中常用的标准速率（64kbps）



# PPP配置

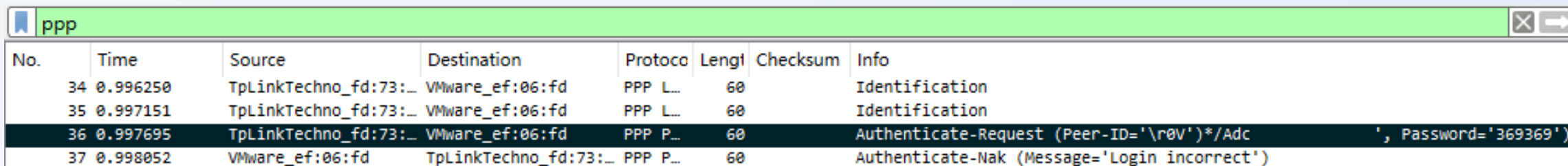
## PPP协议

PPP是另一种常用的广域网数据链路层协议，支持两种主要的认证协议：

- PAP (Password Authentication Protocol)：简单的两次握手协议，明文发送用户名和密码，安全性较低
- CHAP (Challenge-Handshake Authentication Protocol)：质询握手认证协议，采用三次握手和MD5哈希算法来验证身份，密码本身不会在链路上明文传输，安全性更高

国内运营商使用PAP较多，因此能轻松截获拨号上网账号密码，可以参考以下步骤尝试

[Wireshark捕捉宽带密码] <https://zjucomp.net/blog/adsl>



No.	Time	Source	Destination	Protocol	Length	Checksum	Info
34	0.996250	TpLinkTechno_fd:73:...	VMware_ef:06:fd	PPP L...	60		Identification
35	0.997151	TpLinkTechno_fd:73:...	VMware_ef:06:fd	PPP L...	60		Identification
36	0.997695	TpLinkTechno_fd:73:...	VMware_ef:06:fd	PPP P...	60		Authenticate-Request (Peer-ID='\r0V')*/Adc', Password='369369')
37	0.998052	VMware_ef:06:fd	TpLinkTechno_fd:73:...	PPP P...	60		Authenticate-Nak (Message='Login incorrect')

# PPP配置 - CHAP

## 配置R4、R2路由器之间的串口

- 设置IP地址
- 设置数据链路层协议为PPP（命令：encapsulation ppp）
- 设置PPP认证模式为CHAP（命令：ppp authentication chap）
- 分别为对方设置认证用户名和密码（命令：username [hostname] password [pswd]），用户名填写**对端路由器Hostname**（区分大小写），密码保持一致

CHAP握手过程中认证方会根据**挑战者（对端路由器）**报上来的名字（即它的Hostname）去**本地的Username数据库**里查找对应的密码，然后用这个密码和随机值进行哈希计算验证对方的身份，因此配置时需要配置对端的Hostname而不是自己的

仅配置R2时，R2-R4互联串口的LCP状态为“LCP Listen”，此时正在监听但尚未连接上R4完成后，LCP状态为“LCP Open”，表明PPP的LCP已经协商完成，身份验证通过

Encapsulation PPP, LCP Listen

Encapsulation PPP, LCP Open

# DHCP 动态主机地址协议

**想象一个场景：**192.168.0.0/24网络，去掉网络/网关/广播地址有253个地址可以分配  
实验室每周有20节课，平均每个班60人，一个人1-2台设备，IP怎么分配能足够使用？  
给每个人一个固定IP→人数>>可分配IP数量，且学生一般每周只来上1-2次课，非常浪费  
是否可以让IP地址资源给每个同学的设备轮流使用呢？

**想象一个场景：**10.0.0.0/8网络，有16,777,213个地址可以分配  
如果网络管理员给每个设备配置IP需要3分钟，那么总共需要配置95.7年！  
是否可以连接上网络后，自动给设备分配IP呢？

## DHCP 动态主机地址协议

一种客户端-服务端架构协议，用于自动为网络中设备分配IP、子网掩码、默认网关以及DNS服务器地址等关键网络配置参数

引入了IP地址租约（Lease）概念，设备只有在需要接入网络时才会从DHCP服务器租用一个IP地址，并在离开网络（或租约到期）后释放它，让这个地址能被其他用户重新租用

# DHCP配置

## 为R4 fa0/0接口配置DHCP

- 配置IP地址
- 定义DHCP地址池**1（序号不能重复，否则相当于修改指定序号地址池要分配的地址）**  
配置分配子网172.16.0.0:
  - R4#config terminal
  - R4(config)#ip dhcp pool 1 ← 指定DHCP地址池序号
  - R4(dhcp-config)#network 172.16.0.0 /24 ← 网络地址/掩码长度间有空格
  - R4(dhcp-config)#default-router 172.16.0.1
- 启动DHCP服务: service dhcp
- 另一个接口上操作类似，但**不要使用相同地址池序号**
- 在PC机上运行ip dhcp获取动态IP地址（重新启动PC后需要再次执行）

# 回顾Ping / ICMP

**Ping：通过发送和接受数据包，检测网络是否可达**

**Ping是一个双向的过程！ Ping是一个双向的过程！ Ping是一个双向的过程！**

源主机构建ICMP Echo Request，目标主机收到后**返回**ICMP Echo Reply

要成功Ping通一个主机，数据包不仅需要能顺利到达目标主机，还需要能成功从目标主机返回给源主机，往返中任意位置出现问题均会导致无法Ping通

因此：路径上的路由器**必须知道往返的路由**，才能确保两主机间的连通性

**ICMP差错报文：**路由器无法将IP数据报发送给目的地址时，会给发送端主机返回目标不可达ICMP消息，并在这个消息中显示不可达的具体原因

我们实验中常见差错：Type3 Destination host unreachable 目标主机不可达  
Code 1-直连目标主机网络的路由器找不到目标主机（ARP发现网络内不存在对应MAC）  
目标主机没有路由

# 静态路由

路由器对于直连网络（会直接加入路由表，标识：C）可以直接转发到出接口，那么对于其他网络，路由器要怎样知道如何到达呢？

```
192.168.24.0/24 is variably subnetted, 2 subnets, 2 masks
C      192.168.24.0/24 is directly connected, Serial0/1
C      192.168.24.2/32 is directly connected, Serial0/1
172.16.0.0/24 is subnetted, 2 subnets
C      172.16.0.0 is directly connected, FastEthernet0/0
C      172.16.1.0 is directly connected, FastEthernet0/1
C      192.168.34.0/24 is directly connected, FastEthernet1/0
```

## 静态路由

由网络管理员手动配置的路由条目，为路由器提供固定的路径指引，告诉路由器：“要去往[目标网络]，请将数据包发给[下一跳地址]”

## 动态路由

路由器自动地学习、共享和更新路由信息



# 路由转发过程

## 对于非直连子网目标主机：

- [网络层] 匹配选择路由表项（从最长子网掩码项开始，逐一比较，直到找到匹配）
  - [网络层] 找到该项记录的下一跳IP，将数据包向该IP转发
- 交付数据给数据链路层，通过ARP找到对应MAC，封装从出接口发送

## 路由表项：下一跳IP地址 + 出接口（从哪个物理接口发出）

下一跳一定与路由器直连，中间没有可转发的路由器→两端是相同子网→与下一跳在相同子网的即为出接口

## 路由是一种去中心化的过程，路由器在网络层：

- 不知道PC1等的概念，只关注IP、网络
- 不会像交换机一样，能学习到网络上所有主机的IP/找到某个设备在哪个接口上
- 不会指挥下一跳路由器应该怎么做，只关注转发给下一节点，而不关心路径
- 同理，在不使用动态路由协议的情况下，也不会知道其他路由器知道什么



# 静态路由

由网络管理员手动配置的路由条目，为路由器提供固定的路径指引，告诉路由器：“要去往[目标网络]，请将数据包发给[下一跳地址]”；直连网络会直接加入路由表（标识：C）

## 配置命令

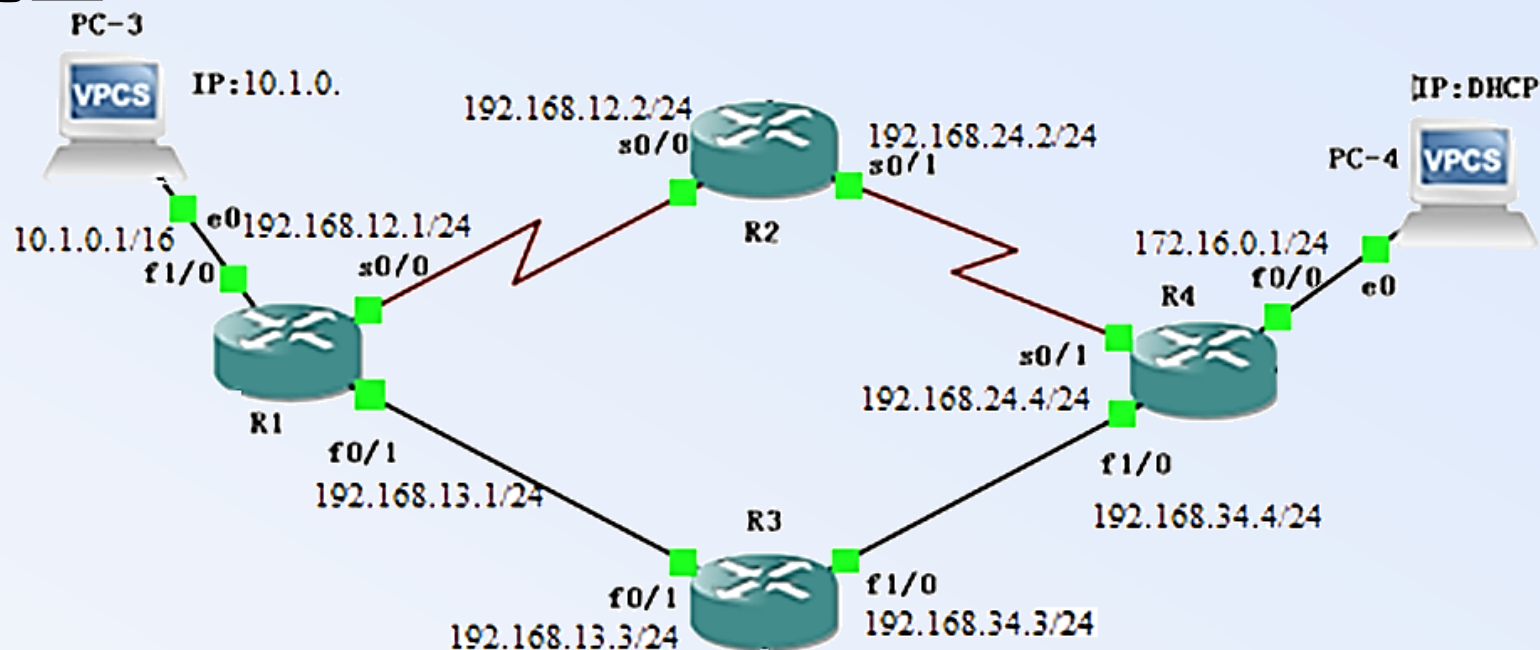
`ip route [目标网络地址] [目标网络子网掩码] [下一跳IP地址]`

如: `ip route 172.16.0.0 255.255.255.0 192.168.13.3`

## 如何确定需要增加哪些静态路由？

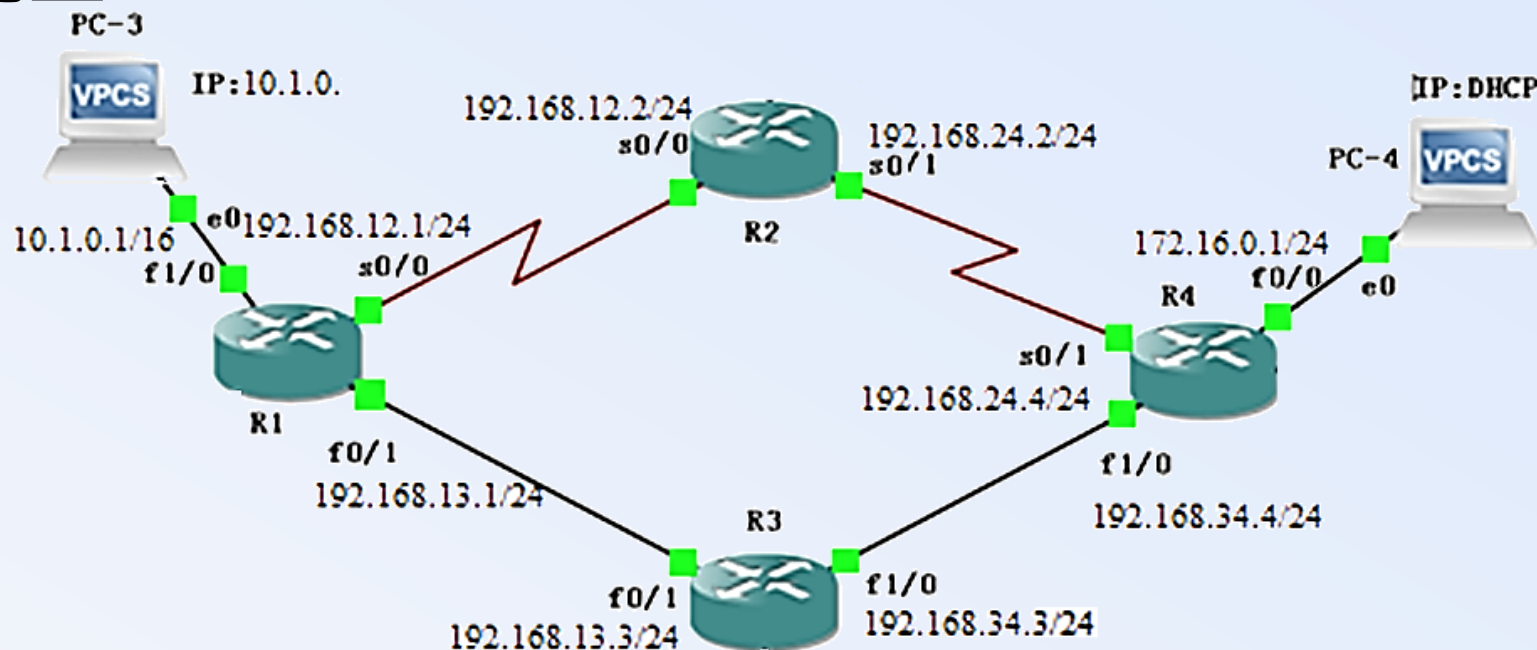
- 分析通信需求与路径（不妨设两台要通信主机分别为A、B）
  - 我们需要确定A/B所属网络，并分析它们间通信要经过哪些路由器转发
- 对路径上路由器节点逐个分析
  - 由于Ping是一个双向的过程，我们需要对途径的所有路由器分析：
    - 该路由器是否有前往A所属网络的静态路由？如果没有，需要补充一条
    - 该路由器是否有前往B所属网络的静态路由？如果没有，需要补充一条

# 静态路由配置



分析通信需求与路径：PC3: 10.1.0.0/16, PC4: 172.16.0.0/24, 优先以太网线路→路径R1 - 3 - 4

# 静态路由配置



分析通信需求与路径：PC3: 10.1.0.0/16, PC4: 172.16.0.0/24, **优先以太网线路**→路径R1 - 3 - 4

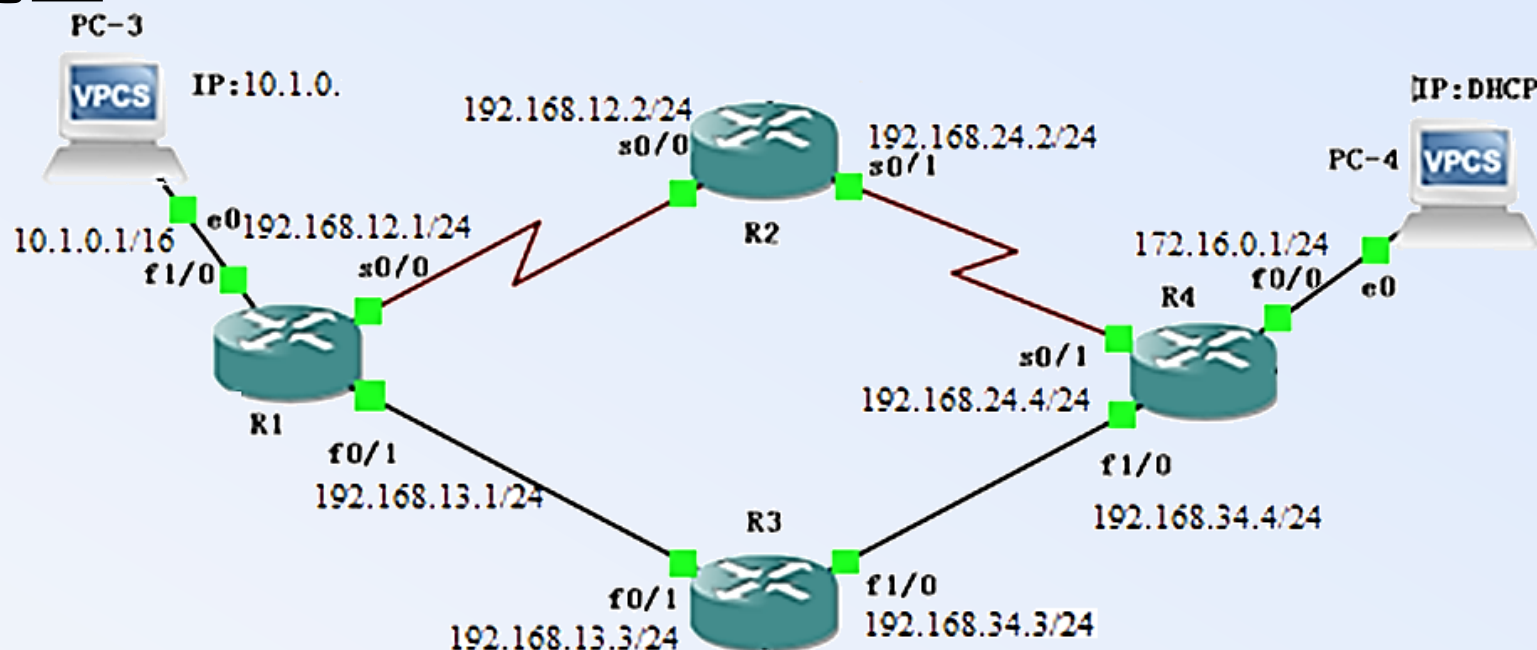
## 逐个节点分析：

R1：[去] 172.16.0.0/24 无表项，需要配置 [回] 10.1.0.0/16 直连，无需配置

R3：[去] 172.16.0.0/24 无表项，需要配置 [回] 10.1.0.0/16 无表项，需要配置

R4：[去] 172.16.0.0/24 直连，无需配置 [回] 10.1.0.0/16 无表项，需要配置

# 静态路由配置



## 最终配置命令:

R1: ip route 172.16.0.0 255.255.255.0 192.168.13.3

R3: ip route 172.16.0.0 255.255.255.0 192.168.34.4

ip route 10.1.0.0 255.255.0.0 192.168.13.1

R4: ip route 10.1.0.0 255.255.0.0 192.168.34.3

## 配置类型与撤销

**开关型：** `no+命令`即可撤销之前命令

例： `no shutdown`即可撤销`shutdown`

**选项型：** 相同命令，配置参数不同可以直接覆盖原有命令

例： `switchport mode access` → `switchport mode trunk` 即可改为`trunk`

**表项型：** 配置后为某个表添加1项，仅`no+完整原命令`才能删除，再次配置=再增加1项

例： `ip route 172.16.0.0 255.255.255.0 192.168.13.3`

`ip route 172.16.0.0 255.255.255.0 192.168.12.2`

## 配置类型与撤销

**开关型：** `no+命令`即可撤销之前命令

例： `no shutdown`即可撤销`shutdown`

**选项型：** 相同命令，配置参数不同可以直接覆盖原有命令

例： `switchport mode access` → `switchport mode trunk` 即可改为`trunk`

**表项型：** 配置后为某个表添加1项，仅`no+完整原命令`才能删除，再次配置=再增加1项

例： `ip route 172.16.0.0 255.255.255.0 192.168.13.3`

`ip route 172.16.0.0 255.255.255.0 192.168.12.2`

**结果是增加了两条静态路由**，而不是将第一次配置修改为下一跳192.168.12.2！

撤销第1条命令必须`no ip route 172.16.0.0 255.255.255.0 192.168.13.3`

# 私有网络

IPv4理论上可以提供43亿个地址用于分配，然而随着互联网迅猛发展、IoT兴起等的影响，这些公网资源已经**枯竭**，不再有新的空闲地址块可供分配

为了应对这样的挑战，RFC 1918定义了**3段特定地址空间**作为**私有地址**，专门作为组织/家庭的内部网络，允许不同网络间重复使用，这3段地址范围分别是：

- A类： 10. 0.0.0 - 10.255.255.255
- B类： 172. 16.0.0 - 172. 31.255.255
- C类： 192.168.0.0 - 192.168.255.255

然而允许重复使用也就意味着这些地址**绝对不能出现在公网**，否则会引起严重冲突和路由混乱，为了在节约IP地址资源的同时，让私有网络内的设备也能访问公网，NAT应运而生



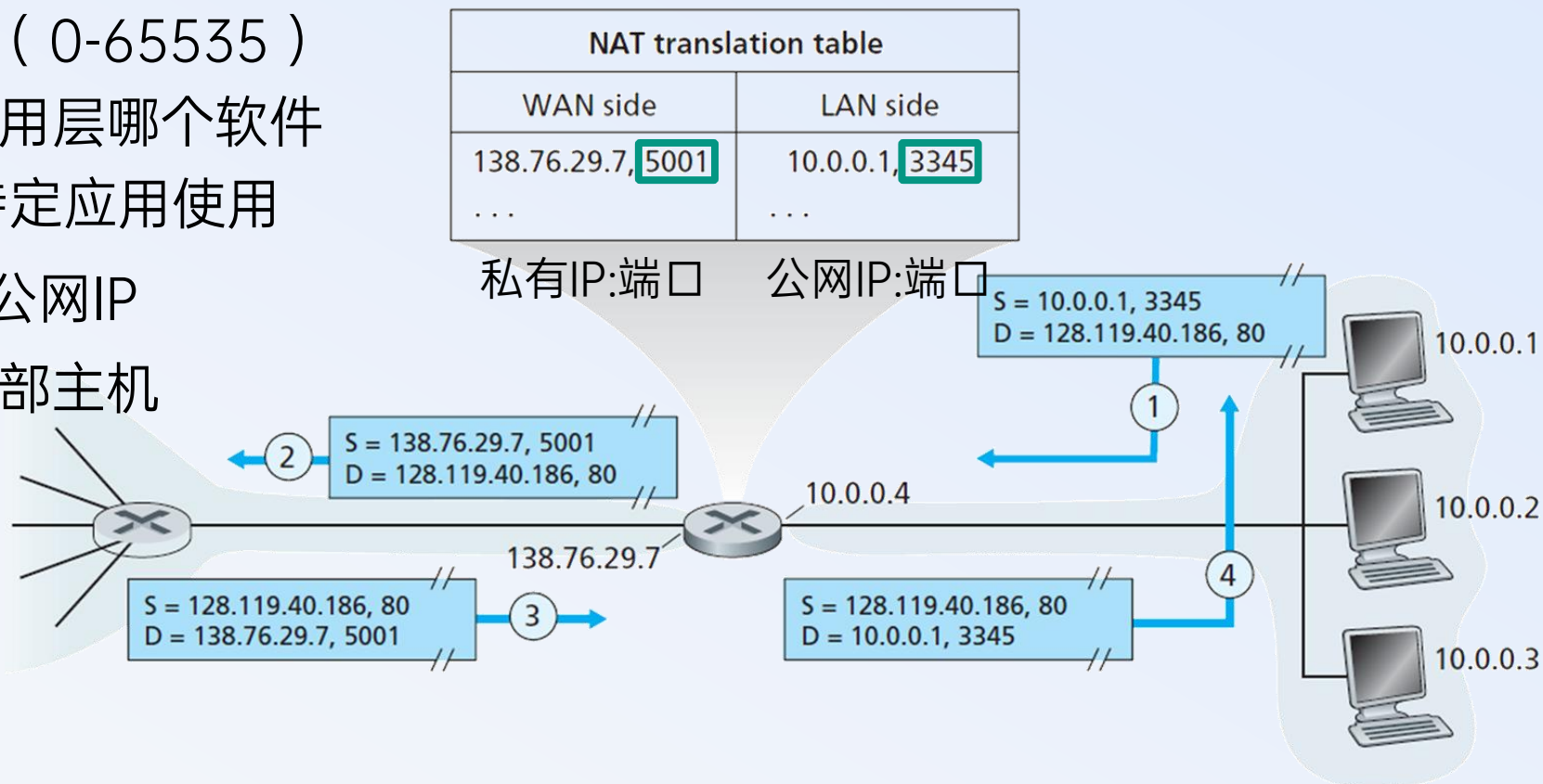
# NAT 网络地址转换

一种在IP数据包通过路由器/防火墙时，重写源/目的IP地址与端口号的技术  
允许多台使用私有地址的主机通过**共享1个或少数几个公网IP**地址来访问互联网，从而极大缓解了IPv4地址枯竭的压力，同时也能一定程度上隐藏内部网络结构

**端口号**：16b无符号整数（0-65535）  
用于传输层区分交付给应用层哪个软件  
熟知端口0-1023保留给特定应用使用

**共用身份**：共用1个/几个公网IP

**相互区分**：端口号标注内部主机



# NAT配置

**捉虫：** **作业系统内**实验报告模板 步骤25：应为定义fa0/1接口为内部接口，定义fa0/0接口为外部接口，模板内写反； **文档网站上**Word/Markdown版本已更正，无需再行修改

- 配置内（私有网络）/外（公开网络）接口：

```
R5(config)# interface fa0/1
R5(config-if)# ip nat inside
```

```
R5(config)# interface fa0/0
R5(config-if)# ip nat outside
```

- 配置访问控制列表，指定将内部哪个网络进行转换：

由于只配置1个内部网络转换，我们创建一个ACL即可，这里选取序号1

```
R5(config)# access-list 1 permit 192.168.0.0 0.0.0.255 ←通配符掩码！
```

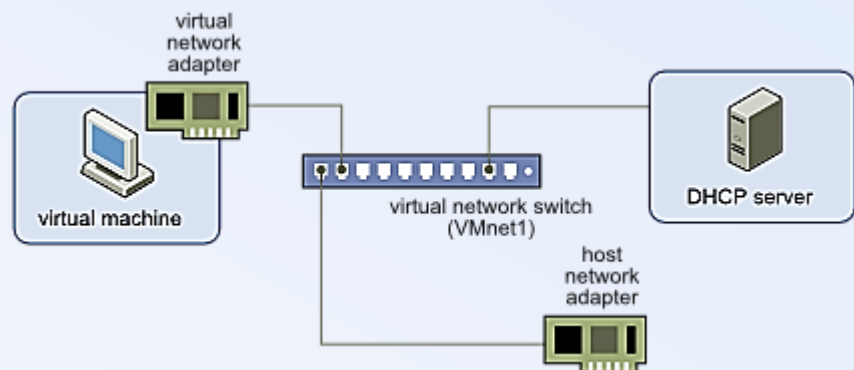
```
R5(config)# ip nat inside source list 1 interface fa0/0 overload
```

**终止持续Ping/NAT转换输出：**

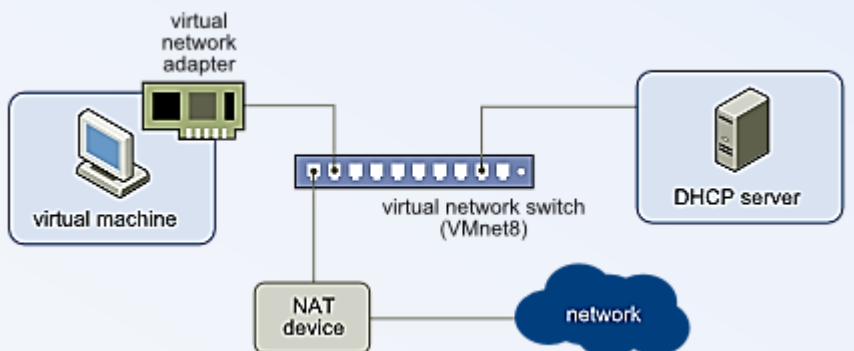
vPC：关闭相应vPC后重新启动

路由器：Ctrl+Shift+6

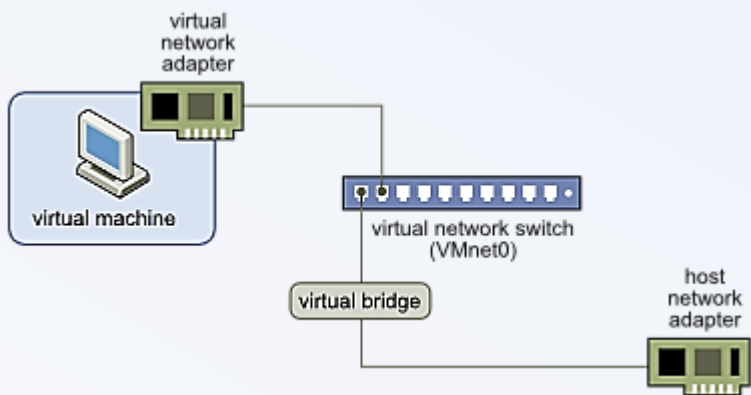
# 虚拟网络适配器



**仅主机模式：**所有虚拟机+宿主机在一个**隔离**私有网络VMWare创建了一个和外部网络完全隔离的虚拟交换机，并提供内置的DHCP服务为虚拟机分配地址，此时虚拟机只能和宿主机/其他虚拟机通信



**NAT模式：**虚拟机+宿主机在一个私有网络下宿主机为虚拟机提供NAT服务以访问外网，并提供内置的DHCP服务，虚拟机可访问宿主机能访问的任何网络



**桥接模式：**虚拟机绕过主机网络协议栈直连物理网络此时虚拟机就像物理网络中与宿主机平等的独立设备，能访问外部网络，虚拟机/宿主机共用网卡但不共享IP等

# 连接真实网络测试

## 模拟Ping物理网络的主机H:

- 一台手机开启热点
- 将电脑连接到该热点
- 选择另一台设备（如：平板/另一台手机/电脑）连接到该热点，作为主机H
- 在R2 / PC1上尝试Ping主机H

## 常见问题:

- 为什么连接到热点后，完全没有获取到IP?  
考虑是不是桥接到了错误的网络设备上
- 为什么Ping什么都通/Ping什么都不通?  
考虑Clash等VPN是否关闭
- 为什么Ping一台Windows电脑完全Ping不通，但是该电脑能Ping通R2?  
请关闭防火墙

# PART 03

## 调试技巧与建议

DEBUG TRICK & ADVICE

# 调试技巧

## 分而治之——缩小检查范围

从Lab4开始，我们将面临非常复杂的网络拓扑环境（幸运的是至少比实际网络简单很多）如果在调试时像无头苍蝇一样乱撞，效率会非常低，必须考虑怎样有章法地寻找问题根源同学们应该都学习过二分查找，这是一种非常有效的查找方式，复杂度低至  $\log N$  在网络中，我们也可以采用类似的思想，逐步将问题限定在较小的范围

**例：**PC3 Ping PC5 (PC3  $\rightarrow$  R1  $\rightarrow$  R3  $\rightarrow$  R4  $\rightarrow$  PC5)

### 方案A：

- 检查PC3 IP配置/默认网关配置
- 检查R1/R3/R4接口IP等是否正确
- 检查R1/R3/R4路由表配置是否正确
- 检查PC5 IP配置/默认网关配置

### 方案B：

- `tracert`看看哪里中断了，检查断点



# 调试技巧

## 不要想，而要看——观察失败，查看细节

一个常见的情况是，同学们遇到问题——“我觉得问题原因应该是A”——尝试修复问题A，然后不幸地发现在进行修复后，问题非但没有解决，还产生了一批新问题

这里核心的**误区**是**对问题的认知完全来自于想象**，而没有进行实际的验证

如果我们按照这样的流程呢？遇到问题——“觉得问题原因可能是A”——尝试验证/观察，分析系统状态——发现实际上是问题B——尝试修复问题B——问题解决

**例：**PC1 Ping PC2，提示Host(255.255.255.0) not reachable.

### 方案A：

- 我不是配置网关了吗，怎么还是不通？  
是辣鸡GNS3卡了的原因吗？我重新配置一次看看→怎么还不行

### 方案B：

- 我不是配置网关了吗，我需要确认一下
- PC1> show ip
- 哦我命令打错把默认网关配成子网掩码了



# 调试技巧

## “检查插头”——怀疑自己的假设，从头开始检查

尽管这个技巧的命名很抽象，但却很能反映上届同学们踩到的坑——当你认为你的网络当前处于怎样的状态时，真的是这样吗？

我们在完成实验的时候，难免会出现一些typo等粗心问题，如果我们没有发现这些问题，系统的**实际状态**与配置就可能**与我们的认知不同步**

此时我们遇到的一切问题都会像闹鬼一样：我路由该配置的都配置了，怎么就是不通？

在遇到这种古怪的问题的时候，请务必show running-config，看看路由器实际运行配置是怎样的，是不是符合你的预期

**例：**Lab2/3当你在其他同学已经做过实验的交换机/路由器上做实验遇到的各种奇怪现象

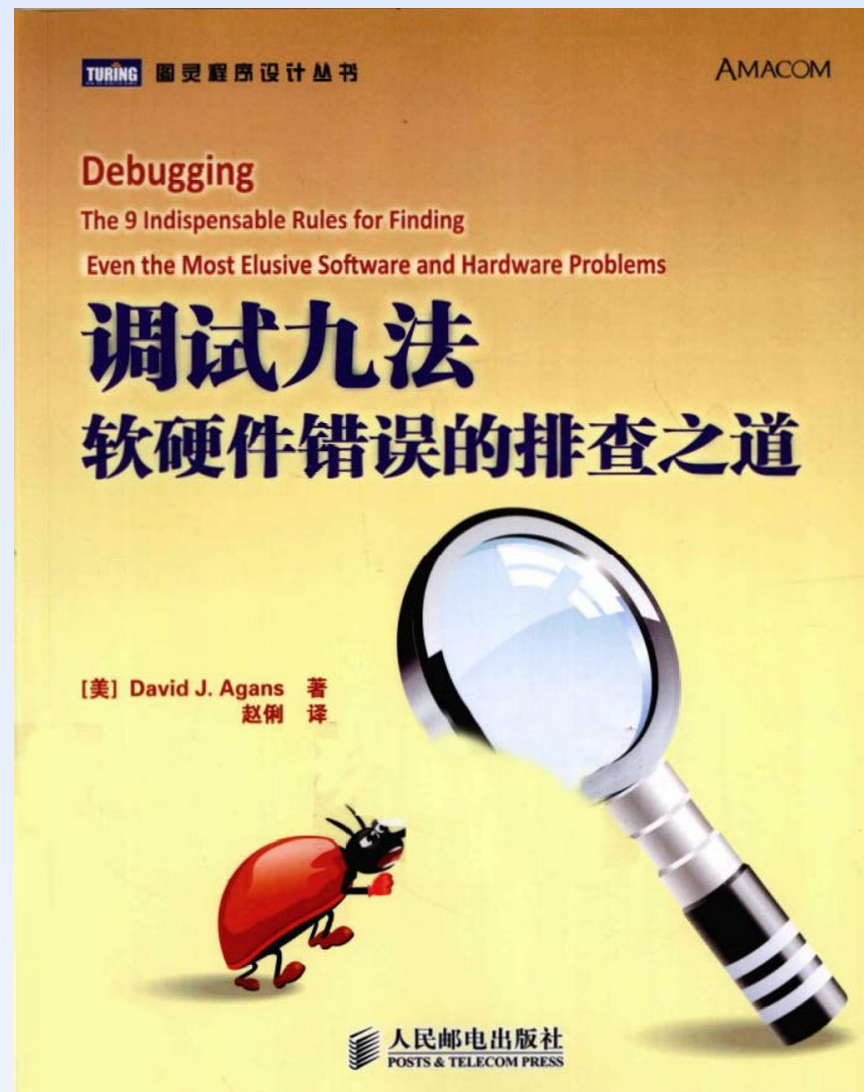
# 《调试九法》

## 调试九法 软硬件错误的排查之道

David J. Agans 著

调试领域非常经典的一本著作，对软硬件错误的排查与解决提供了成体系的最佳实践原则

作者结合自己工作经历，以实际情景引入了调试方面的建议



# 敢于提问

**同学们遇到难以解决的问题不要闭门造车，请随时来找我求助**

即使感觉问题可能很蠢/问题模糊不知道具体该问些什么/问题可能很麻烦，也欢迎你遇到困难时随时来寻求帮助

**为了更快帮你解决问题，提问时可以尽量：**

- 直接截图，避免手机拍屏
- 描述遇到问题的步骤、问题的表现，提供标注好IP/接口的拓扑图  
(Lab5/6还请提供运行配置)