课程目标

- 理解路由表的作用
- 能够读懂路由表信息
- 能够使用图形抓包工具wireshark进行数据包的抓取，如（TCP/IP的三次握手四次断开）

一、路由表

思考:

什么是交换,什么是路由,什么是路由表?

1. 交换是指**同网络访问**（两台机器连在同一个交换机上，配置同网段的不同ip就可以直接通迅)
2. 路由就是**跨网络访问**(路径选择)
3. 路由表是**记录路由信息的表**，在Linux中首先是一张可见的,可更改的表,它的作用就是当数据包发到Linux的时候，系统（或者说内核)就根据这张表中定义好的信息来决定这个数据包接下来该怎么走.

1. 查看路由表信息

```
route命令用来查看和设置路由表信息

[root@server ~]# route -n     //查看路由表信息
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.0.0     0.0.0.0         255.255.255.0   U     0      0        0 eth1
10.1.1.0        0.0.0.0         255.255.255.0   U     0      0        0 eth0
0.0.0.0         10.1.1.254      0.0.0.0         UG    0      0        0 eth0
目标网络            网关          子网掩码          路由标志                    网卡
                                              Up:启动状态
                                              UG:该网关为路由器

113.28.10.34
```

2. 读懂路由信息

```
讨论1:
按上面的路由表来看，如果我ping一个公网IP（如ping 14.200.151.38)，应该怎么走?

1）先看目标ip是否为本地ip，如果是，则直接访问本地;如果不是，则找路由表里是否有你访问的网段.
2）如果路由表有则从这个路由条目后面指定的网卡出去；如果路由表里没有你访问的网段，则会找默认路由
   （也就是网关）。
3）如果网关也没有的话，则会报错网络不可达。
connect: Network is unreachable


讨论2:
按上面的路由表来看，如果我ping一个局域网IP为10.1.1.10，会怎么走?

ping 10.1.1.10不会走网关，而是走本地路由从eth0网卡出去（因为路由表里有10.1.1.0/24的路由）。

讨论3:
```

```
如何加网关和删除网关，加网关有什么要求?
route add default gw  x.x.x.x        临时加网关，马上生效
route del default gw  x.x.x.x        临时删网关，马上生效
永久增加网关:
vim  /etc/sysconfig/network-scripts/ifcfg-eth0
GATEWAY=x.x.x.x
或者
vim /etc/rc.local    操作系统开机最后读取的一个文件
..
route add default gw xxxxx

注意事项:
1. 加网关只能加你已经有的路由网段里的一个IP才行（此IP不一定存在）
2. 加网关可以不用指定子网掩码（因为是已有的一个网段的ip，所以掩码已经确认了）

讨论4:
一个linux服务器上能有几个有效网关?
准确来说：一个路由表上可以加多个网关，但只有一个生效。

讨论5:
我一台linux上如果有双物理网卡，请问可不可以两个网卡配置同网段的不同IP呢?
eth0 10.1.1.1/24
eth1 10.1.1.2/24

如果两个网卡同网段，则会有下面两条路由
10.1.1.0        0.0.0.0         255.255.255.0   U     0      0        0 eth0
10.1.1.0        0.0.0.0         255.255.255.0   U     0      0        0 eth1

结果:
它会实现从两张网卡进来的包，却从一张网卡出去，问题就产生了。假设eth0网卡有问题时，路由表里匹配到第一条后，依然会走eth0网卡，而不会走eth1。
也有解决方法（比如多路由表或者双网卡绑定），这里不涉及。
```

二、路由选择实验

1. route命令介绍

```
route -n  查看路由,显示ip,不解析
route del default gw 10.1.1.254    删除默认路由
route add default gw 192.168.1.110    添加一个默认网关，把所有不知道的网络交给网关来转发
route add -net 192.168.2.0/24 dev eth0    对一个网络添加一个新的路由（另一个网段）
route del -net 192.168.2.0/24
```

2. 实验要求

前提条件：三台虚拟主机的网络模式都是仅主机模式!
环境准备：
node1:10.1.1.1/24
作为网关服务器，开启路由转发功能/proc/sys/net/ipv4/ip_forward
node2:192.168.0.254/24
node3:172.16.0.254/24
要求：
实现不同网络(172.16.0.0/24和192.168.0.0/24)之间的互通，使用第三方主机node1作为路由进行转发

3. 具体步骤

思路：

1. 中间人node1，必须开启路由转发功能；
2. 中间人node1即要有到达A网络，又能到达C网络的路
3. node2主机（A网络）必须设置中间人node1作为自己的网关
4. node3主机（C网络）必须设置中间人node1作为自己的网关

步骤：
1.node1服务器（中间人）完成以下任务
1）开启路由转发功能
临时开启：

```
[root@server ~]# cat /proc/sys/net/ipv4/ip_forward
0
[root@server ~]# echo 1 > /proc/sys/net/ipv4/ip_forward
[root@server ~]# cat /proc/sys/net/ipv4/ip_forward
1
```

永久开启：

```
[root@server ~]# vim /etc/sysctl.conf
# Controls IP packet forwarding
net.ipv4.ip_forward = 1
```

2）分别添加到node2和node3两台主机所在的网络

```
[root@node1 ~]# route add -net 192.168.0.0/24 dev eth0
[root@node1 ~]# route add -net 172.16.0.0/24 dev eth0
[root@node1 ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.16.0.0      0.0.0.0         255.255.255.0   U     0      0        0 eth0
192.168.0.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
```

2. 分别配置node2和node3的IP和网关

```
[root@node2 ~]# route add -net 10.1.1.0/24 dev eth0
[root@node2 ~]# route add default gw 10.1.1.1
[root@node3 ~]# route add -net 10.1.1.0/24 dev eth0
[root@node3 ~]# route add default gw 10.1.1.1
```

3. 测试验证
1）中间主机分别ping node2和node3

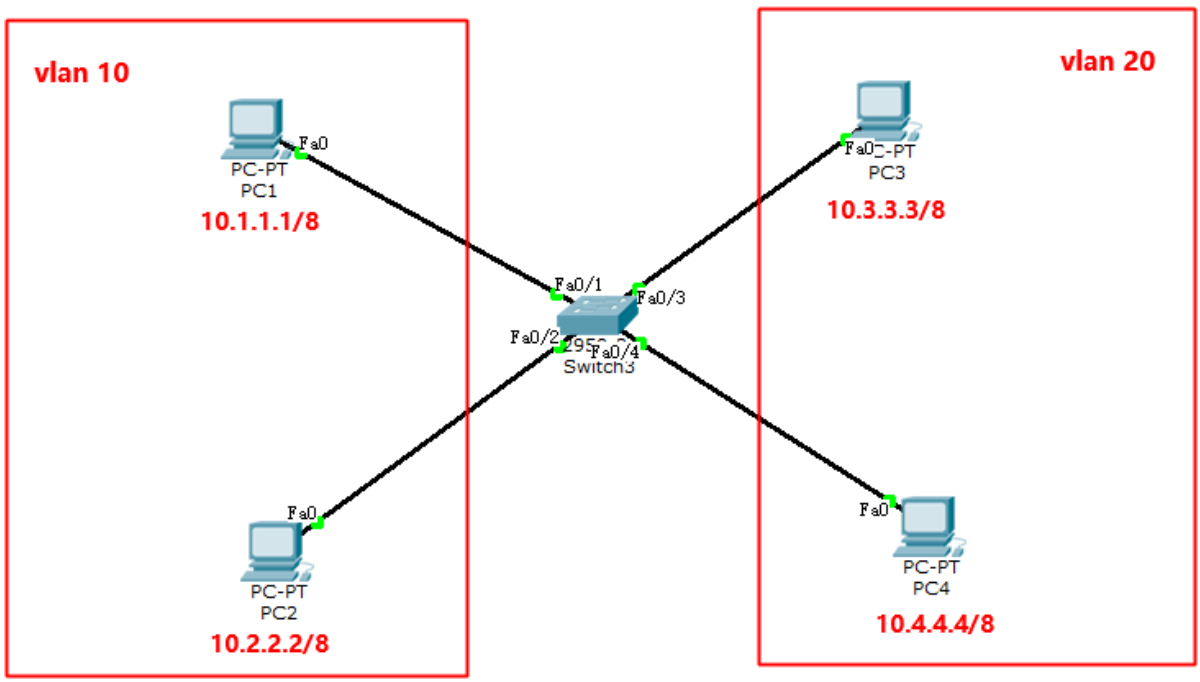
2）node2和node3相互ping

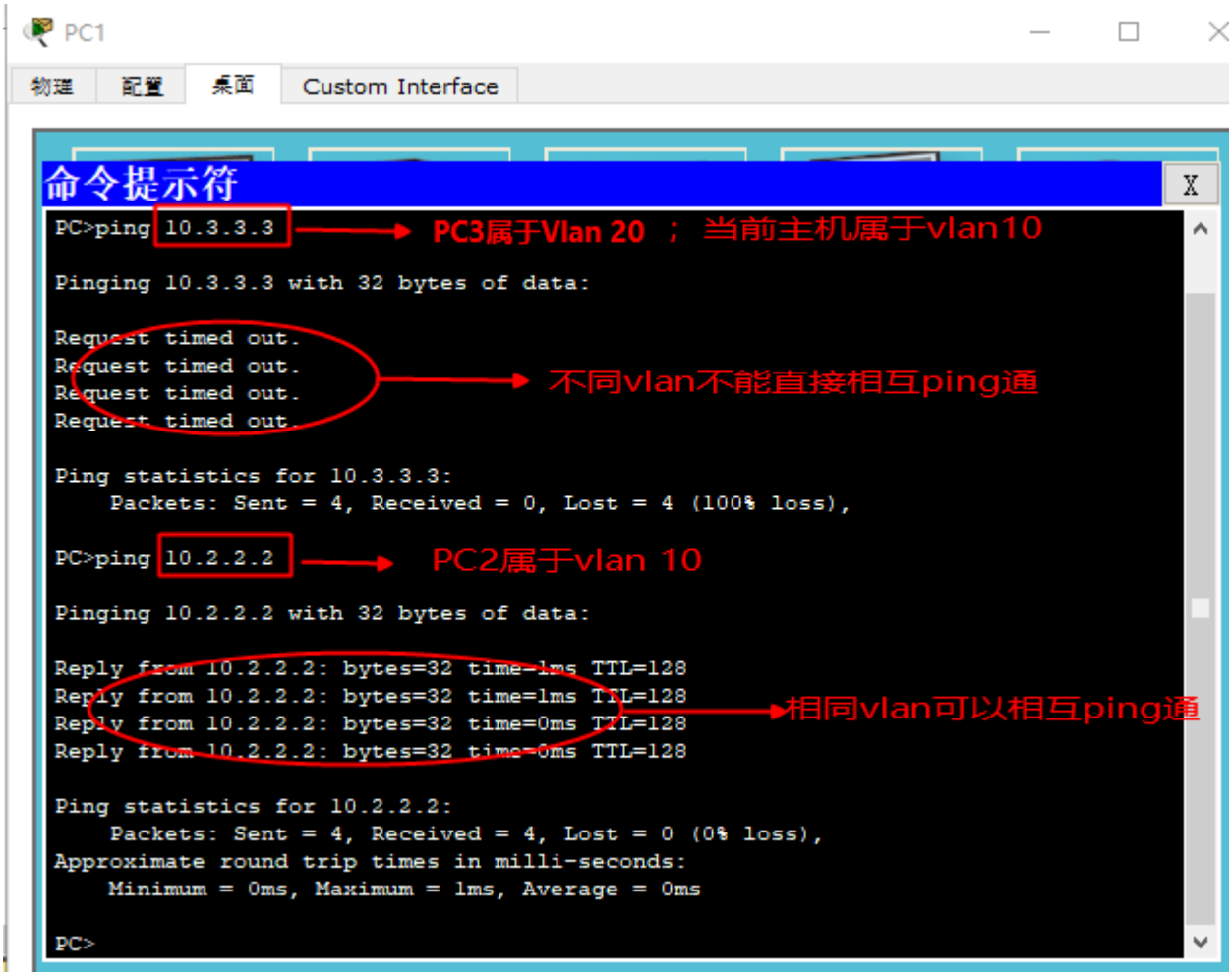
4. 实战组建局域网

思科模拟器交换机配置相关命令：

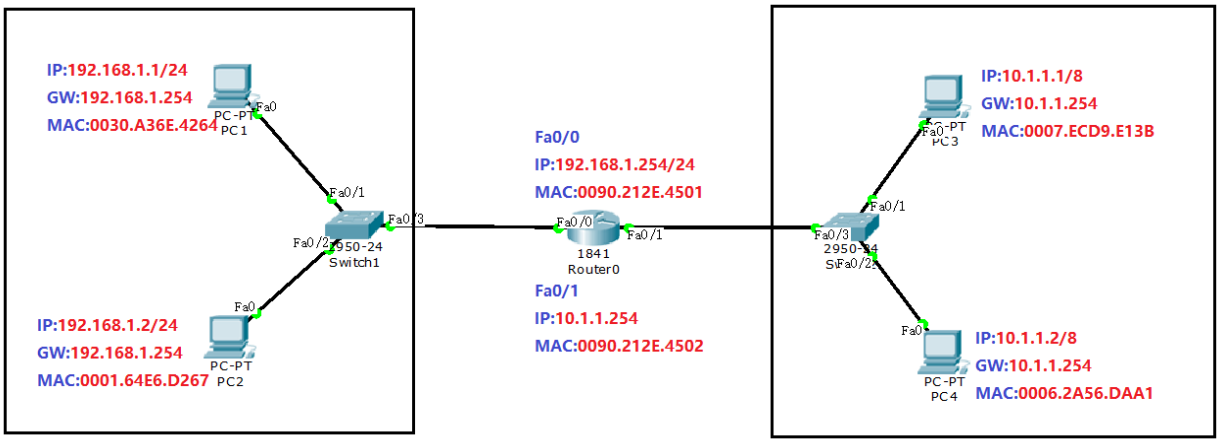
```
进入到使能模式
Switch>?
Switch>enable
进入配置终端
Switch#?
Switch#configure terminal
创建vlan
Switch(config)#vlan 10
删除vlan
Switch(config)#no vlan 10
进入指定的配置接口
Switch(config)#interface FastEthernet0/1
或者
Switch(config)#int fa0/1
将fa0/1接口加入到vlan中
Switch(config-if)#switchport access vlan 10
从vlan中删除接口
Switch(config-if)#no switchport access vlan 10
```

4.1 交换机vlan划分





4.2 路由器连接多个网络



三、抓包工具

1. wireshark工具

```
安装wireshark工具
# yum -y install wireshark*
```

抓telnet登录包：

telnet:远程管理服务 tcp/23

server：10.1.1.5 telnet-server

client：10.1.1.1 telnet 使用：telnet 10.1.1.1

```
Centos6.5系统：
1. 安装软件包
yum -y install telnet-server telnet

2. 启动telnet服务

vim /etc/xinetd.d/telnet
service telnet
{
        disable = yes      将yes变为no
        flags           = REUSE
        socket_type     = stream
        wait            = no
        user            = root
        server          = /usr/sbin/in.telnetd
        log_on_failure  += USERID
}

启动服务：
service xinetd start

检查telnet服务是否启动：
[root@node1 Desktop]# netstat -nltp|grep 23
tcp        0      0 :::23                       :::*                        LISTEN
 5546/xinetd

Centos7.5系统：
1. 安装软件包
yum -y install telnet-server
2. 启动服务
[root@centos7 ~]# systemctl start telnet.socket
[root@centos7 ~]# netstat -nltp|grep :23
tcp6       0      0 :::23                   :::*                    LISTEN      1/systemd
3. 测试验证
```

2. tcpdump工具

```
查看arp缓存  (同一网段主机的MAC地址)
[root@node1 Desktop]# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.1.1.254               ether   00:50:56:c0:00:01   C                     eth0
```

删除目标主机的MAC缓存

```
[root@node1 Desktop]# arp -d 10.1.1.254
[root@node1 Desktop]# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.1.1.254                       (incomplete)                              eth0
```

查看主机10.1.1.1收到的和发送的数据包

```
[root@node1 Desktop]# tcpdump -i eth0 -nn host 10.1.1.1
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
22:33:25.148389 ARP, Request who-has 10.1.1.254 tell 10.1.1.1, length 28
22:33:25.148896 ARP, Reply 10.1.1.254 is-at 00:50:56:c0:00:01, length 46
22:33:30.714605 IP 10.1.1.254 > 10.1.1.1: ICMP echo request, id 1, seq 237, length 40
22:33:30.714619 IP 10.1.1.1 > 10.1.1.254: ICMP echo reply, id 1, seq 237, length 40
```

-i 指定网络接口，对于多个网络接口有用
-n 显示IP地址，不查主机名。当DNS不起作用时常用到这个参数
-nn 不显示协议和端口名。即显示IP地址和端口

-w 将抓的包保存到指定的文件里方便后续分析

其他用法:

```
man tcpdump
TCP Packets
The general format of a tcp protocol line is:
              src > dst: flags data-seqno ack window urgent options
       Src and dst are the source and destination IP addresses and ports.  Flags are some
combination  of  S
       (SYN),  F  (FIN),  P (PUSH), R (RST), W (ECN CWR) or E (ECN-Echo), or a single ‘.’
(no flags).
```

1.获取主机10.1.1.1接收或发出的telnet包

```
#tcpdump tcp port 23 and host 10.1.1.1
```

2.对本机的udp协议的123端口进行监听（123是ntp服务端口）

```
# tcpdump udp port 123
```

3.只对hostname的主机的通信数据包进行监视。主机名可以是本地主机，也可以是网络上的任何一台计算机。
下面的命令可以查看主机hostname发送的所有数据:

```
#tcpdump -i eth0 src host hostname
#tcpdump -i eth0 src host 10.1.1.254
```

4.下面的命令可以查看所有送到主机hostname的数据包:

```
#tcpdump -i eth0 dst host hostname
#tcpdump -i eth0 dst host 10.1.1.1
```

5.监视通过指定网关的数据包:

```
#tcpdump -i eth0 gateway Gatewayname
#tcpdump -i eth0 gateway 10.1.1.254
```

6.其他
只需要列出送到80端口的数据包，用dst port;

```
#tcpdump -i eth0 host hostname and dst port 80   //目的端口是80
```

只需要看到返回80端口的数据包，用src port

```
#tcpdump -i eth0 host hostname and src port 80  //源端口是80,一般是提供http的服务的主机
```

如果条件很多的话要在条件之前加and 或 or 或 not

```
#tcpdump -i eth0 host ! 210.161.223.70 and ! 210.161.223.71 and dst port 80
```