

以上是基本拓扑图，在设备上标注的是他们各自接口的ip地址。

**一些知识**

我们首先需要知道的一些常识：

**关于ping:**

**如果没有路由器：**

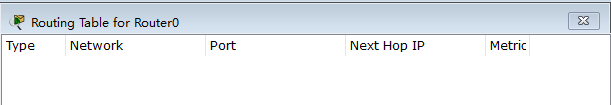
同一网段之间可以相互ping通，就是源设备发送一个数据报，目的设备能接收到，并且能够发回一个响应数据报，使源设备接收到相应数据报。如果两个ip地址之间没有ping通，不仅需要检查源设备是否能发送到目的设备，还需要检查目的设备是否能发送到源设备（这不是废话）。

**如果有路由器：**

**但是没有路由表信息（使用ip route命令添加的那个信息）：**

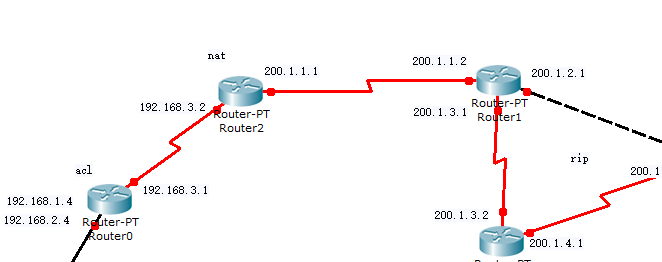
路由器只能转发到它的接口直接连接的网段，没有的话就只能超时失败，ping不通。

**可以添加路由表信息：**

路由表信息包括类型，网络，端口，下一跳地址，度量。

类型和度量暂时无关紧要。

如果转发的数据报要到network下的网络，如果有直连的接口，就通过port标记的接口转发，直连的网络也占用一条路由信息。没有下一跳IP的信息。



如果你想要到的网络地址没有直连的接口，比方说router0想要到上图中的200.1.2.0网络，但是转发到router2时，没有直连的接口，你可以手动添加一条路由信息：ip route 200.1.2.0 255.255.255.0 200.1.1.2，它就会添加一条路由信息到上图的路由表中，network显示200.1.2.0，port没有，下一跳IP显示200.1.1.2。

就是说，如果想要到200.1.2.0网络，就把数据报发送到200.1.1.2的ip地址，到router1之后，发现有直连200.1.2.0的接口，于是转发。假设有没有就会又查路由表，如果有路由信息就会再次转发。

**关于静态路由和动态路由：**

上面是我们手动添加的路由信息，叫做**静态路由**，而我们为路由器添加rip或者ospf的协议配置之后，路由器之间就会自己交换路由信息，叫做**动态路由**，我们的实验目的之一就是让我们实现动态路由，而这些协议有固定的激活命令，稍后谈到再说。

**关于交换机：**

路由器的接口可以配置ip地址，但是交换机的不能，它是数据链路层的设备，路由器是IP层的。

但是将交换机的端口划分到vlan中后，可以为vlan配置ip地址。这个ip地址通常作为pc的网关。

通常所有的端口都是access类型的，默认有一个vlan 1，所有端口都属于它，我们创建新vlan后，就可以将接口划分到这些vlan中去。

路由器和交换机的接口默认都是关闭的，所以配置完之后需要使用no shutdown命令来开启接口，防止意外，最好每退出一个接口就no shut一次。

示例：

vlan 2 //创建vlan 2

vlan 3 //创建vlan 3

int f0/1 //进入f0/1接口的配置

sw acc vlan 2 //划分到vlan 2

int f0/2 //进入到f0/2接口的配置

sw acc vlan 3 //划分到vlan 3

int vlan 2 //进入到vlan 2的配置，初次会有信息弹出，不要惊慌

ip add 192.168.1.1 255.255.255.0 //为vlan 2配置ip地址和掩码

no shut //开启vlan 2下所有接口

int vlan 3 //进入到vlan 3的配置

ip add 192.168.2.1 255.255.255.0 //为vlan 3配置ip地址和掩码

no shut //开启vlan 3下所有接口

**关于trunk:**

交换机接口默认是acces类型的，一般与PC连接，trunk接口就是传输数据用的，没有ip也不能划分vlan，标记成tr后它就不属于任何vlan了。跟交换机trunk口相连的数据报传输没有阻碍，不会识别数据包的ip。

int f0/3 \\进入到f0/3的接口配置

sw mode trunk \\把它标记为trunk接口

no shut

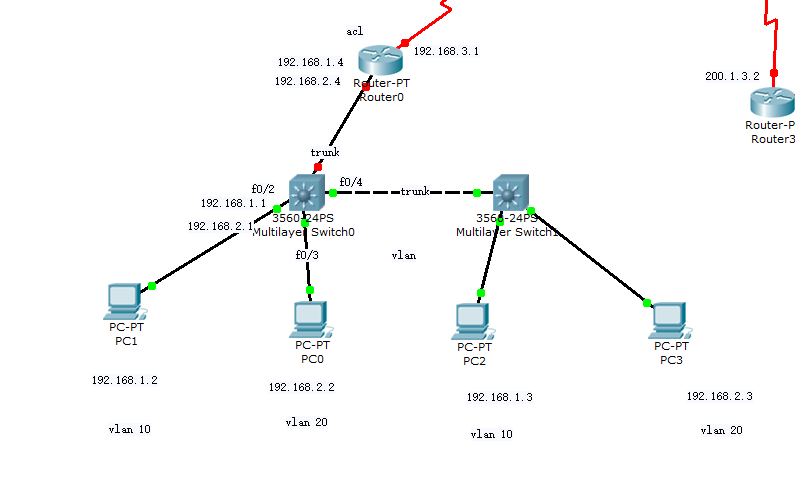
**关于命令：**

很多命令都是有缩写的，不必打全，比如int是interface的缩写，sw是switchport的缩写，add是address的缩写，acc是access的缩写。

如果你想打全又想省事，可以写了它的缩写之后使用tab键来补全它。

**单臂路由、VLAN、trunk**

首先在下面的2个交换机+4台PC+最上面1台路由器，是我们体现VLAN划分和trunk功能的地方。



首先我们**配置好4台PC的地址**：

通过PC电脑的设置来做，可以看出来PC1和PC2属于vlan10，是同一网段，PC0和PC3属于vlan20，是同一网段。PC只有一个f接口，没有数字标记。

然后**配置交换机**：

在switch0创建vlan 10和vlan 20

将f0/2划分到vlan 10，f0/3划分到vlan 20，f0/4设置成trunk。

设置vlan 10 和vlan 20的Ip地址，192.168.1.1和192.168.2.1并启用。

具体命令如下：

en

conf t

//创建vlan 10,20

vlan 10

vlan 20

//划分vlan

int f0/2

sw acc vlan 10

int f0/3

sw acc vlan 20

//把f0/4接口配置成trunk类型

int f0/4

sw mode trunk

//配置vlan

int vlan 10

ip add 192.168.1.1 255.255.255.0

no shut

int vlan 20

ip add 192.168.2.1 255.255.255.0

另一个交换机接口情况一样，所以命令也一样。

这样我们就完成了vlan的划分，实现了一个小功能。pC1可以ping通pC2,其余PING不通，PC0可以PING通PC3，其余PING不通。

然后是路由器和交换机之间的配置：

交换机上面的接口是f0/1，也配置成trunk，路由器连接的是f0/0接口，这里有两个网段，正常情况应该用两个接口配置两个ip地址来对应，但是使用了单臂路由的技术，将f0/0接口又划分成了f0/0.1和f0/0.2两个子接口，进而可以配置两个地址。

在sw0交换机上：

en

conf t

int f0/1

sw mode trunk

在Router0上：  
进入f0/0接口的配置，然后打开它

en

conf t

int f0/0

no shut

再来配置它的子接口

对f0/0.1：

int f0/0.1

encapsulation dotQ 10

ip add 192.168.1.4

encapsulation dotQ 10的命令是必要的，它表明将子接口f0/0.1用802.1q的标准配置到vlan 10，使用了它之后才能配置子接口，不然会出错。因为f0/0已经打开了，所以不用使用no shut。

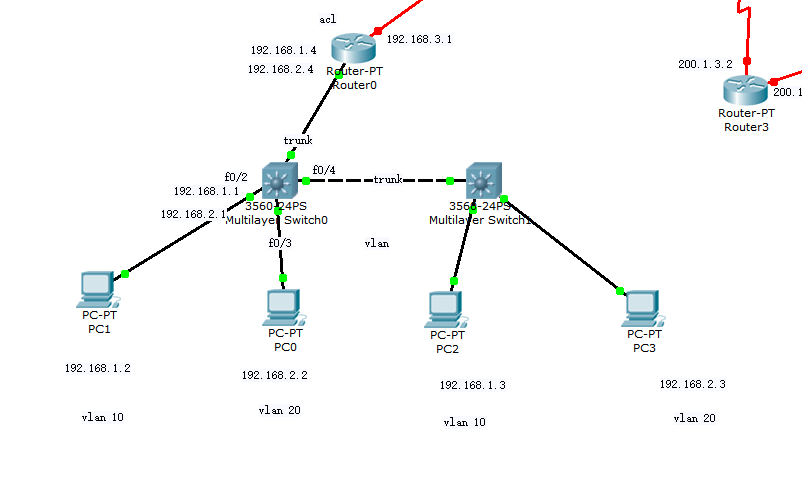
对于f0/0.2同理：

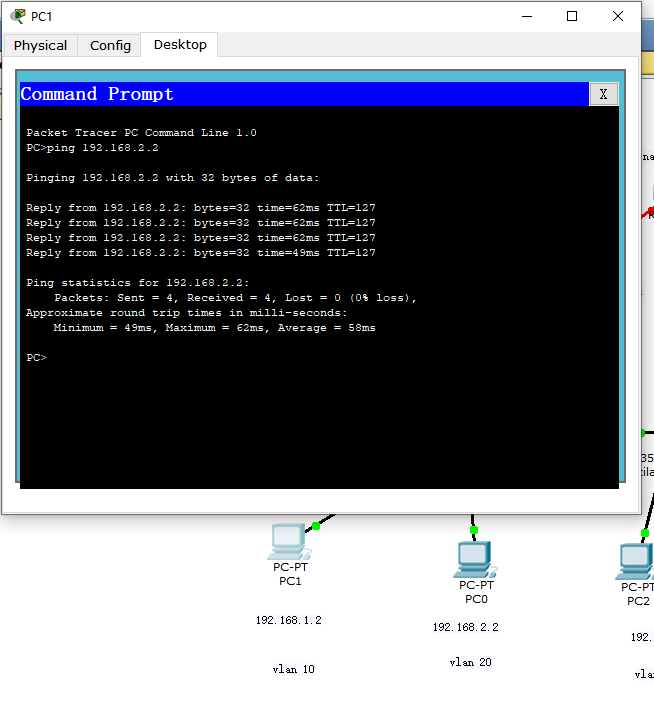
int f0/0.2

encapsulation dotQ 20

ip add 192.168.2.4

那么现在我们就可以ping通192.168.1.0网络和192.168.2.0网络了，也就是vlan 10和vlan 20。

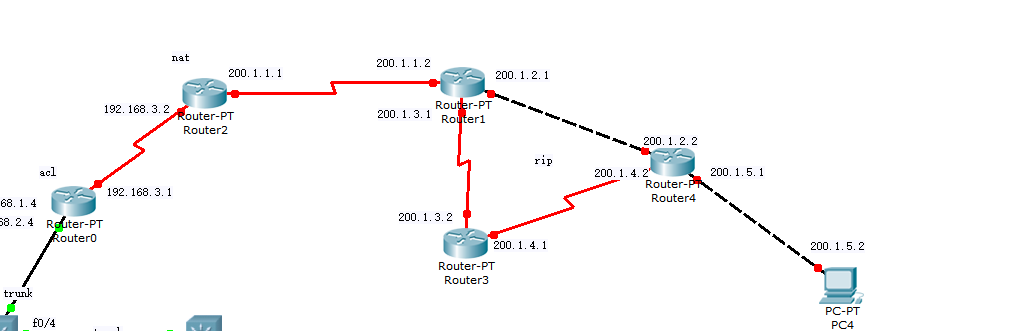




可以看到成功了。

**rip、ospf动态路由**

接下来我们来配置动态路由，这是要求的另一项基本功能。

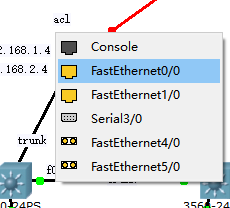


照例先配置PC的IP地址。

**关于路由器的一些事：**

然后是路由器，路由器每个端口必须连接不同的网络，因此每个路由器之间都创建一个网络。

模拟器上的路由器有两个F口，两个S口，两个F口?和一个console口。



电脑通过console口来配置路由器的端口。其余用作传输数据，S口使用DCE-DTE线来连接，F口用正常的交叉线和直通线来连接（就是黑实线和黑虚线，一般来说相同设备之间使用交叉线\黑虚线，不同设备之间使用直通线\黑实线，来自百科）。总之是用来传数据包的。一般使用上面的两个s口和F口。路由器之间优先使用S口。

S口之间使用DCE-DTE线。

DCE-DTE线是一条线，但是有DCE端和DTE端的区别，当使用它来连接路由器时，在DCE端的接口还要配置clock rate 64000，不然ping不通（暂时不知道为什么）。在虚拟软件上，有时钟标志的那个红闪电线是以DCE端开头的（第一个配置的接口连的就是DCE端），另一个没有时钟的红闪电是以DTE端开头的。

实体上机时，深蓝色扁头的线好像就是这个，不清楚哪一段是DCE端，配置路由器时可以两端都配一下。

**命令配置：**

那么从左至右依次配置ip，过程中再为所有路由器顺便配置rip或者ospf：

Router0：

此路由器已经配置过f0/0的子接口的ip了，我们来配置s2/0端口。

int s2/0

ip add 192.168.3.1 255.255.255.0

no shut

clock rate 64000

顺便配置rip或者ospf

ip routing

如果后面的router命令无法使用，就使用这个命令，表示启用路由。

router rip

network 192.168.1.0

network 192.168.2.0

network 192.168.3.0

其中router rip表示启用rip协议。

然后通过network把与此路由器直连的网络都加入rip协议中。宣告本路由器与这些网络直连。

如果ospf：

router ospf 1

network 192.168.1.0 0.0.0.255 area 0

network 192.168.2.0 0.0.0.255 area 0

network 192.168.3.0 0.0.0.255 area 0

router ospf 1表示启用ospf协议，数字表示进程，可以随便取，没什么意义。

network 192.168.1.0 0.0.0.255 area 0表示在area 0宣告192.168.1.0网络。

后面的是路由器通配符，会检查所有0位，正好跟掩码的作用相反，所以也称作反掩码。

ospf有划分区域的功能，分了很麻烦，也不加分，所以如果换的话就弄到同一个区域里，跟network命令差别不大。还有router-id正常来说也需要配置，但是不配置也可以，路由器会使用loopback地址最大的来作为router-id。很麻烦，没有意义，也不加分，所以我们如果要换的话也不额外弄了。这样也可以运行。

我们先以rip来完成实验拓扑。

Router2：

en

conf t

int s2/0

ip add 192.168.3.2 255.255.255.0

no shut

clock rate 64000

int s3/0

ip add 200.1.1.1 255.255.255.0

no shut

clock rate 64000

router rip

network 192.168.3.0

network 200.1.1.0

Router1:

en

conf t

int s3/0

ip add 200.1.1.2 255.255.255.0

no shut

clock rate 64000

int s2/0

ip add 200.1.3.1 255.255.255.0

no shut

clock rate 64000

int f0/0

ip add 200.1.2.1 255.255.255.0

no shut

router rip

network 200.1.1.0

network 200.1.2.0

network 200.1.3.0

Router3:

en

conf t

int s2/0

ip add 200.1.3.2 255.255.255.0

no shut

clock rate 64000

int s3/0

ip add 200.1.4.1 255.255.255.0

no shut

clock rate 64000

router rip

network 200.1.3.0

net work 200.1.4.0

Router4

en

conf t

int s3/0

ip add 200.1.4.2 255.255.255.0

no shut

clock rate 64000

int f0/0

ip add 200.1.2.2 255.255.255.0

no shut

int f1/0

ip add 200.1.5.1 255.255.255.0

no shut

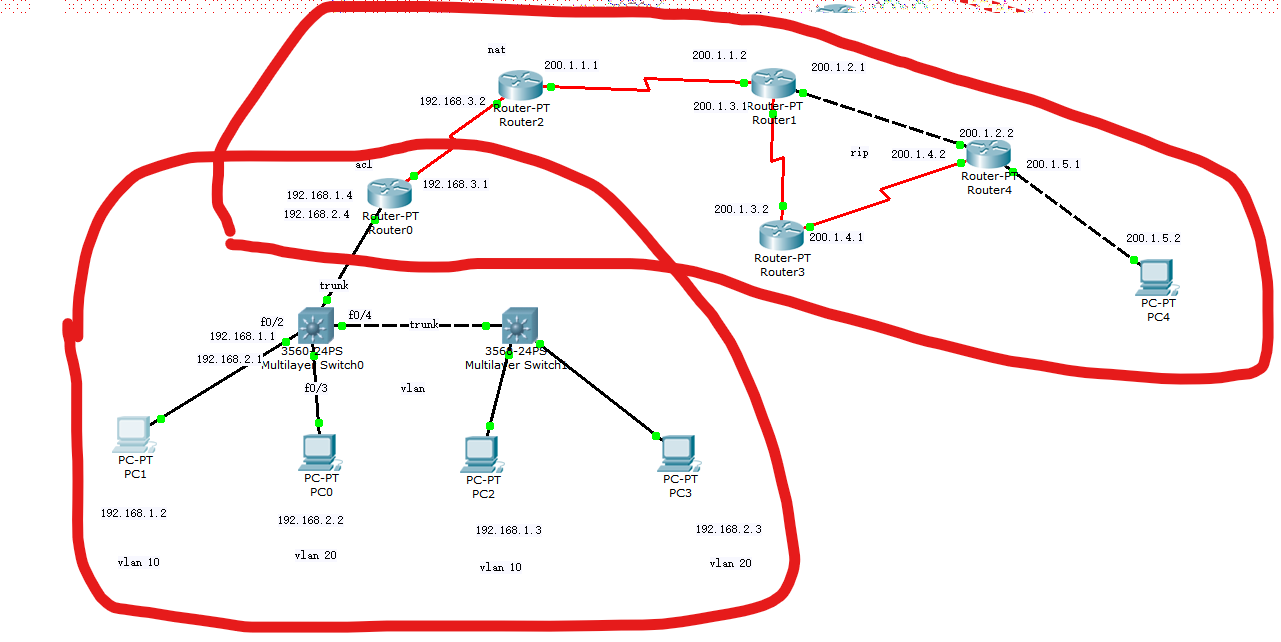
router rip

network 200.1.2.0

network 200.1.4.0

network 200.1.5.0

现在我们就配置完毕了rip动态路由。



路由器需要交换路由之后才可以ping通，所以开始时会失败几次，如果无论如何都无法ping通，请检查是否是忘记了配置接口的clock rate。

但是最终我们发现区域内的设备都是可以ping通的，但是PC0-3无法ping通router2，发现是交换机没有配置rip，于是我们配置两个交换机的rip协议：

en

conf t

router rip

network 192.168.1.0

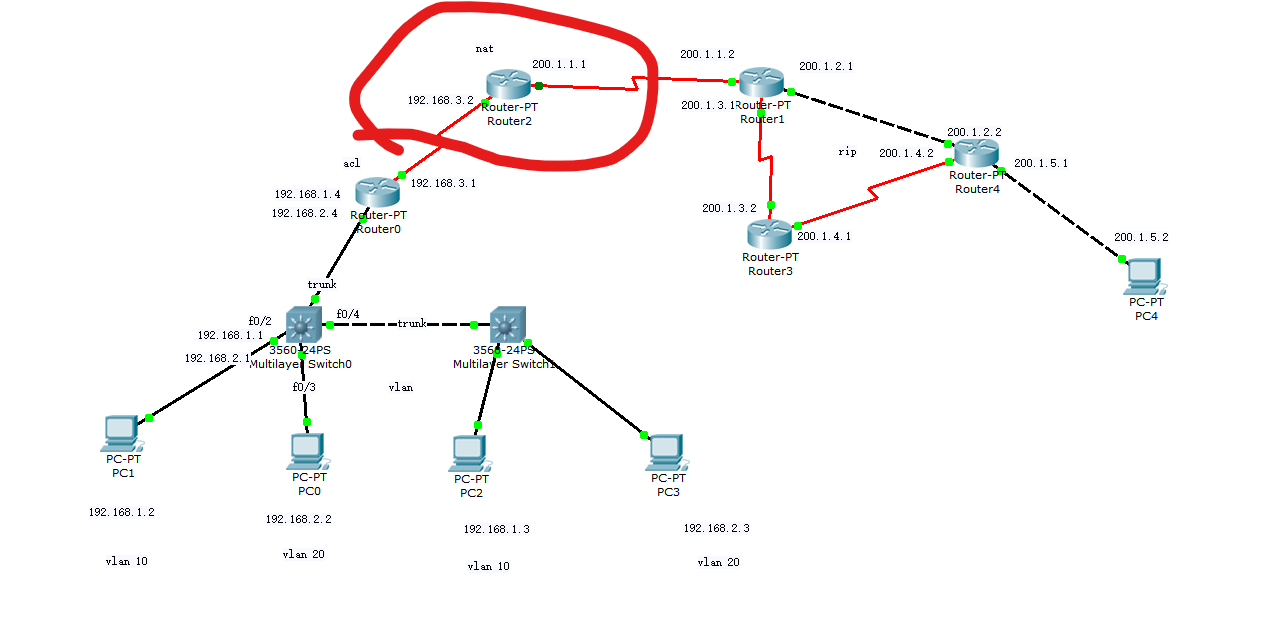
network 192.168.2.0

两台交换机的命令相同

简单配置之后又ping几次后，我们发现已经可以从PC0-3ping通PC4了。那么我们现在就实现了要求的基本的功能。

**NAT**

NAT就是转化ip地址，在经过有NAT配置的路由时，把数据包的ip地址替换成它要发送出去的IP地址。



我们在router2上实现nat的功能。

NAT是为了内部局域网与外部互联网沟通发明的，我们把下面的192.168.0.0系列定义为内部局域网，把200.1.0.0系列看成是外部互联网。

我们定义一下：

int s2/0

ip nat inside

int s3/0

ip nat outside

接下来可以有两种方法，静态nat配置和动态nat配置，差别不大，后者比较灵活简便。

先退出接口配置，回到config级。

exit

**静态nat:**

ip nat inside source static 192.168.1.2 200.1.1.1

就是说将内部IP地址192.168.1.2静态固定地转换为200.1.1.1。那么我们的192.168.1.2的PC发送的数据报在经过此路由器时，IP地址就会被替换成200.1.1.1。

**动态nat:**

定义一下外部IP池：

ip nat pool poolname 200.1.1.3 200.1.1.4 netmask 255.255.255.0

poolname是外部IP池地名字可以随便填，后面是IP和它们的掩码。IP至少需要两个，否则命令会出错。IP可以不必是路由器外部接口的地址，但一定要是同一个网络，不然路由器不会转发。

定义一下内部的：

access-list 1 permit 192.168.0.0 0.0.255.255

access-list后面的数字可以随便写，代表允许转化的内部表的编号，然后是内部IP和它们的反掩码，跟掩码作用一样但是反过来了。

然后匹配它们：

ip nat inside source list 1 pool poolname

跟静态路由的配置差不多，就是说符合list 1中的网络的IP地址匹配到poolname池，路由器会在其中自动选择外部IP地址进行转化。

我们使用动态nat配置一下。

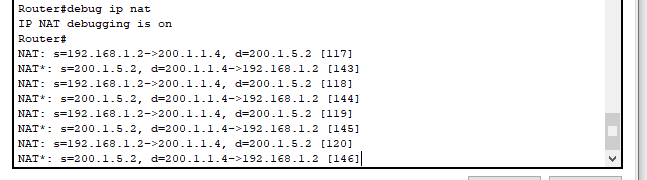
动态IP和静态IP不能兼容，若要删除静态IP，在原来添加的命令上加no就好。

no ip nat inside source static 192.168.1.2 200.1.1.1

配置完成后使用end命令退出，回到初始enable级别。

如果要看到nat的转化过程，可以使用debug ip nat命令。

然后我们从PC0-3ping一下PC4，发现router2上出现了内容：



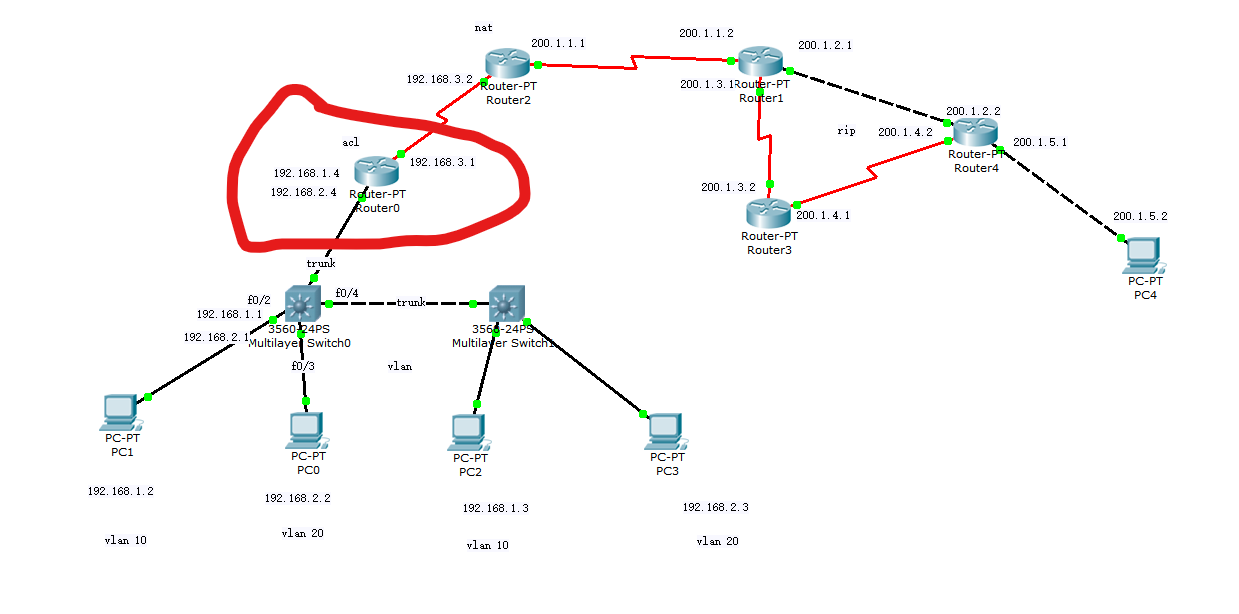
可以看到成功了。不带星号的是发出去时的转化过程，带星号的是响应的数据包发回来时的转化过程。

**ACL**

访问控制技术。

标准ACL可以根据数据报的源地址进行过滤，允许或禁止某个网段的数据报通过。

扩展ACL的功能更多，我们这里为了简便只简单使用标准ACL来配置一下。



我们配置在这个路由器中，阻止192.168.1.0网段的数据包通过，允许192.168.2.0网段的数据包通过。

在config模式下，输入命令：

access-list 1 deny 192.168.1.0 0.0.0.255

access-list 1 permit 192.168.2.0 0.0.0.255

即创建了一个编号为1的阻止通过的列表和一个编号为1的允许通过的列表。

数字可以任取1到99以内的，100及以上的属于扩展ACL。

然后进入上面的接口s2/0，如果下面四台PC发送的数据报要出去，一定会经过它。

int s2/0

ip access-group 1 out

即根据编号1的列表中的规定，控制数据报的输出。

现在我们再ping几下，发现vlan10的PC已经不能ping 通PC4了，但是vlan 20的仍然可以。

所有的PC仍然可以ping通Router0，体现了我们使用了访问控制技术。

现在我们就完成了实验要求的所有内容，在实际的实验室里交换机和路由器的端口名称较为不同，是g1/0/1和s1/0/0系列的，在实体机上show int来查看一下接口名称然后替换就好。

以上属于多方拼凑的理解，可能出现一些问题，如果有问题可以随时跟我交流，所有人都理解掉整个过程应该应对提问就没什么问题了。