计算机网络技术实践

实验报告

实验名称 实验三 RIP和OSPF路由配置协议流程分析

姓 名 陈朴炎 实 验 日 期：2023.11.05

学 号 2021211138 实验报告日期：2023.12.06

报 告 退 发：（ 订正 、 重做 ）

目录

[计算机网络技术实践 1](#_Toc153052729)

[实验报告 1](#_Toc153052730)

[一、环境 3](#_Toc153052731)

[1.1 运行的操作系统以及网络平台环境 3](#_Toc153052732)

[1.2 网络拓扑图及说明 3](#_Toc153052733)

[二、实验内容及目的 4](#_Toc153052734)

[2.1 实验内容 4](#_Toc153052735)

[2.2 实验目的 5](#_Toc153052736)

[三、实验步骤及分析 5](#_Toc153052737)

[3.1 ip配置实验步骤 5](#_Toc153052738)

[3.2 RIP协议 8](#_Toc153052739)

[3.2.1 RIP协议配置过程 8](#_Toc153052740)

[3.2.2 RIP协议路由构建过程分析 10](#_Toc153052741)

[3.2.3 RIP协议的水平分割 13](#_Toc153052742)

[3.3 OSPF协议 15](#_Toc153052743)

[3.3.1 OSPF协议配置过程 15](#_Toc153052744)

[3.3.2 OSPF链路状态信息更新过程 19](#_Toc153052745)

[3.4 RIP和OSPF协议的区别 21](#_Toc153052746)

[3.4.1 RIP协议特点 21](#_Toc153052747)

[3.4.2 RIP协议工作过程 21](#_Toc153052748)

[3.4.1 OSPF协议的特点 22](#_Toc153052749)

[3.4.2 OSPF协议的工作过程 23](#_Toc153052750)

[四、实验结果（包括最终实验结果，需要截图） 24](#_Toc153052751)

[4.1 RIP协议实验结果 24](#_Toc153052752)

[4.2 OSPF协议实验结果 27](#_Toc153052753)

[五、实验中的问题及心得 29](#_Toc153052754)

[5.1 实验中的问题 29](#_Toc153052755)

[5.2 心得体会 30](#_Toc153052756)

[六、实验思考 31](#_Toc153052757)

[6.1 不同方式配置路由丢包解析 31](#_Toc153052758)

[6.2 RIP和OSPF的协议工作过程 36](#_Toc153052759)

[6.3 数据包的发送过程 38](#_Toc153052760)

# 一、环境

## 1.1 运行的操作系统以及网络平台环境

**Windows 11Dynamips 0.2.7 BY N.L.F.EGNS3windows版本：22.44.1VM Version：0.15.0Ubuntu version focalQemu version 4.2.1**

## 1.2 网络拓扑图及说明

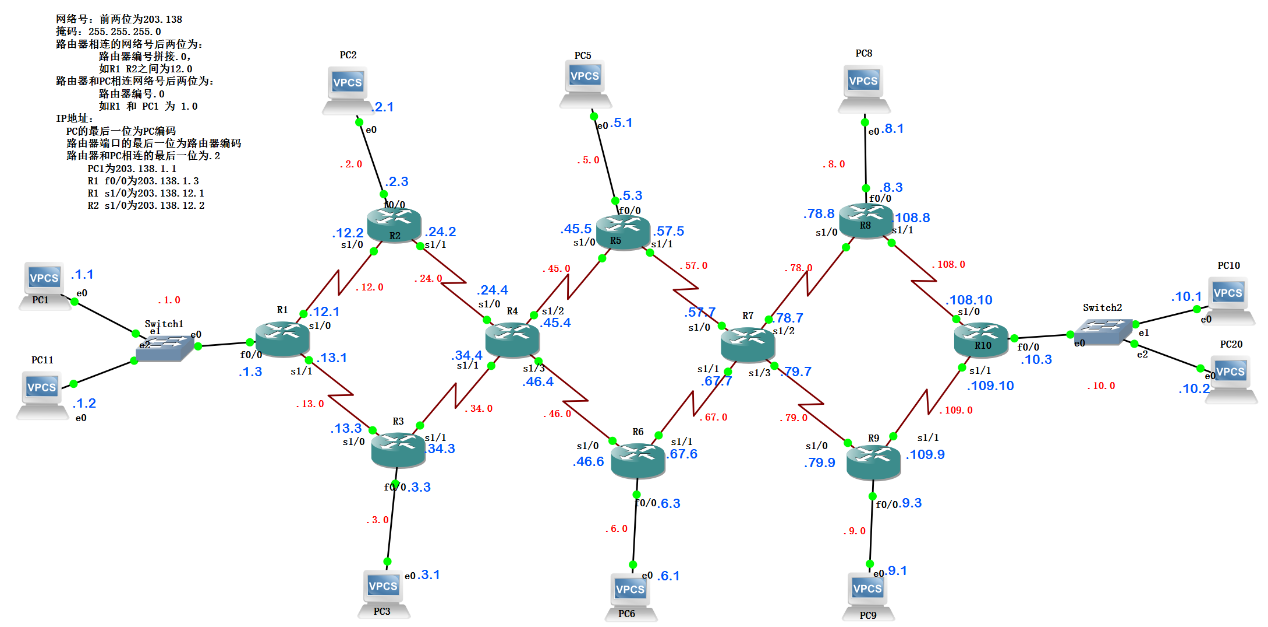


图1-1 网络拓扑图

拓扑图说明：

图1-1中，用红色标明的是网络号，用蓝色标明的是IP地址。

路由器10个，PC机10个，交换机2个，网段数量20个。

（1）网络号

所有的网络号前两位都是203.138，子网掩码都是255.255.255.0

路由器相连的网络号第3位为：路由器编号拼接的数字，如R1连R2，之间的网络就是203.138.12.0，R8和R10连接的网络就是203.138.108.0。

路由器和PC相连网络号后两位为：<路由器编号>.0。比如，R1和PC1的网络号就是203.138.1.0。

（2）IP地址：

所有的ip地址前三位都对应着它们所在的网络号，只有第四位有区别：

和PC相连的网络中，PC的最后一位是1或者2，路由器的端口最后一位是3，PC设置的默认网关是它们的网络号.3，即和路由器相连的端口IP。

在路由器和路由器相连的网络中，路由器端口的最后一位为路由器编码。

示例如下：

PC1为203.138.1.1，PC11 为203.138.1.2

R1 f0/0为203.138.1.3

R1 s1/0为203.138.12.1

R2 s1/0为203.138.12.2

# 二、实验内容及目的

## 2.1 实验内容

（1）在第二次实验的基础上实现RIP和OSPF路由协议

（2）自己设计网络物理拓扑和逻辑网段，并在其上实现RIP和OSPF协议（不能少于4台路由器，要求IP地址第一位是203，第二位是学号后三位%255）

（3）通过debug信息详细描述RIP和OSPF协议的工作过程，包括初始信息交互、路由计算、链路故障处理等部分。（要修改部分链路，观察工作过程）

（4）RIP协议中观察没有配置水平分割和配置水平分割后协议的工作流程，和路由消息传递方式；（要修改部分链路，观察区别，默认有水平分割）

（5）OSPF中数据库同步信息的格式和同步对象？链路改变信息如何发送，具体格式（要修改部分链路，观察消息传递过程）

## 2.2 实验目的

（1）理解和掌握RIP和OSPF路由协议的基本原理：

（2）设计和实现网络拓扑：

（3）实现RIP和OSPF协议：

（4）调试和观察协议工作流程：

（5）观察水平分割的影响：

（6）深入理解OSPF的数据库同步和链路改变信息：

# 三、实验步骤及分析

## 3.1 ip配置实验步骤

设计完网络拓扑后便开始配置ip和网络号。

首先将所有设备开启，并双击想要配置的设备。

（1）PC的配置方法如下：

|  |
| --- |
| ip <ip地址>/<子网掩码位数> <默认网关> |

比如，PC1的配置语句就是 ip 203.138.1.1/24 203.138.1.3，如图3-1所示，当终端报告该PC的ip、掩码、网关时便说明配置完成。

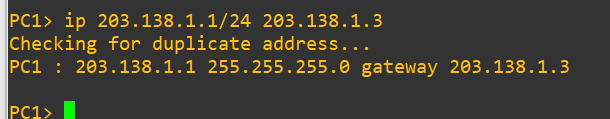


图3-1 PC1的ip配置示意图

（2）路由器和PC连接的接口ip配置如下：

首先点开路由器终端，输入conf t，进入配置命令模式，如图3-2。

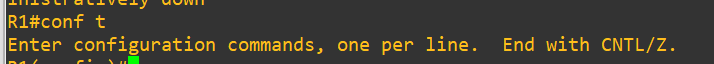


图3-2 R1的ip配置步骤1示意图

进入待配置的端口，比如我的R1要配置端口f0/0的IP地址，我就输入interface f0/0，如图3-3所示，进入接口配置。



图3-3 R1的ip配置步骤2示意图

在接口里，配置路由器该接口的ip，并将该接口开启，如下图3-4所示。

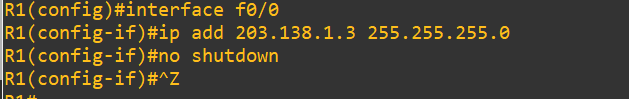


图3-4 R1的ip配置步骤3示意图

检查配置是否成功，用PC1来ping R1的f0/0接口，看是否ping通，如下图：

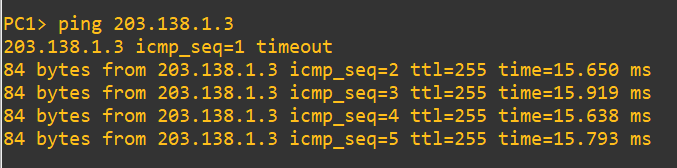


图3-5 检查ip配置正确性示意图

（3）路由器和路由器相连的接口ip配置：

路由器之间连接通过串口，属于广域网连接，一端是DCE，一端是DTE，DCE端配上时钟，数据链路层采用PPP协议。

我们在两个路由器之间选择一个路由器配DCE，比如我的R1和R2相连，我选择R1的S1/0端口配DCE，步骤如图3-6：

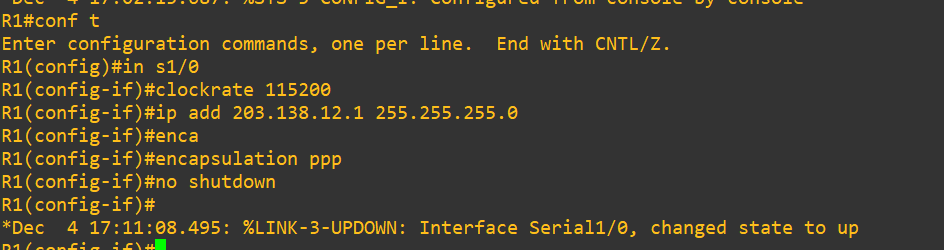


图3-6 路由器接口配DCE及IP示意图

首先进入相应接口，配DCE就是加上时钟，输入clockrate 115200，之后再配上ip地址，给数据链路层封装一个PPP协议，并启动该接口。当看到哦最后一行的提示 changed state to up时就说明接口启动完成。

给R2配DTE时，只需少掉配时钟的步骤，其余步骤一样，如图3-7：

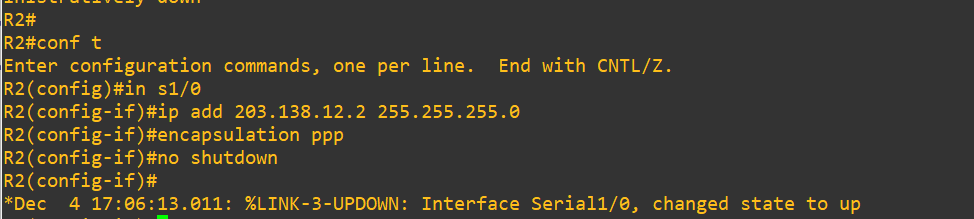


图3-7 R2路由器配DTE及IP示意图

首先进入串口，配上ip，并封装好ppp协议，开启该接口，便配置完成。

其余的路由器和PC配置和上述步骤相差不大，我就不罗列了。

## 3.2 RIP协议

### 3.2.1 RIP协议配置过程

首先，将路由器的debug信息打开，如图3-8。

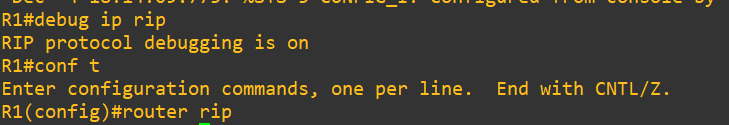


图3-8 R1开debug信息

在conf t下配置rip协议：

（1）router ip rip

（2）version 2

（3）network 203.138.1.0

（4）neighbor 203.138.12.2

具体步骤如图3-9、图3-10所示：

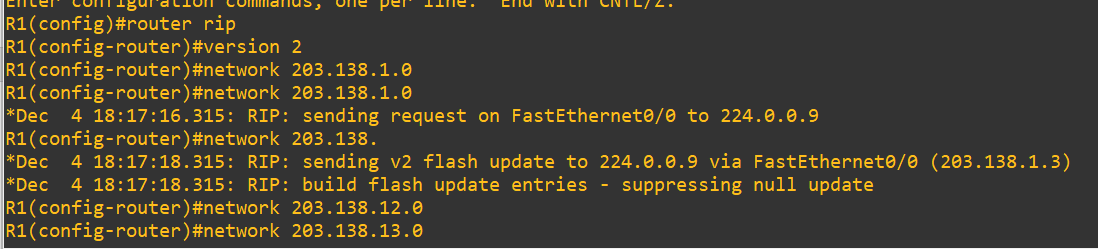


图3-9 rip协议配置图

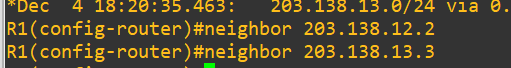


图3-10 rip协议配置图

network命令用来设置该路由器连接的网络号。

neighbor命令用来设置和该路由器相邻的路由器ip。

输入完上述命令后，会循环出现如下debug信息：

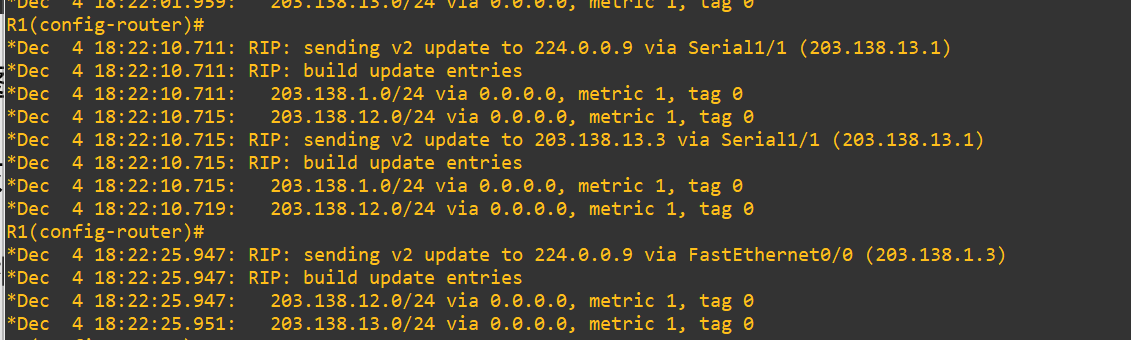


图3-11 R1debug信息示意图1

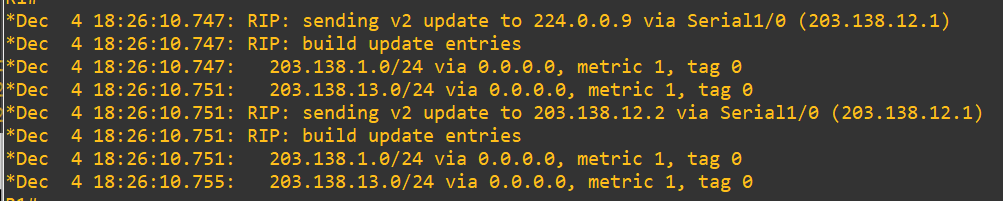


图3-12 R1debug信息示意图2

sending v2 update 表示路由器在使用RIP的version2。

224.0.0.9是RIP version2的组播地址，via Serial1/1（203.138.13.1）表示发送路由信息通过的串口是S1/1，该串口的ip地址为203.138.13.1。

build update entries表示路由器在创建路由信息，203.138.1.0/24 via 0.0.0.0, metric 1, tag 0 这句话表示网络203.138.1.0/24 是通过路由器直连的，度量为1，标记值为0。0.0.0.0表示的是路由器直连。

同理，配置其他的路由器，R2配置过程如下：

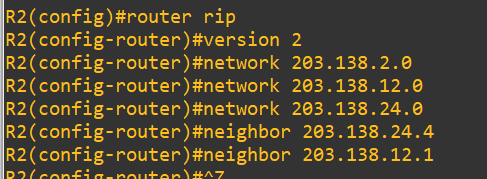


图3-13 R2的RIP协议配置过程图

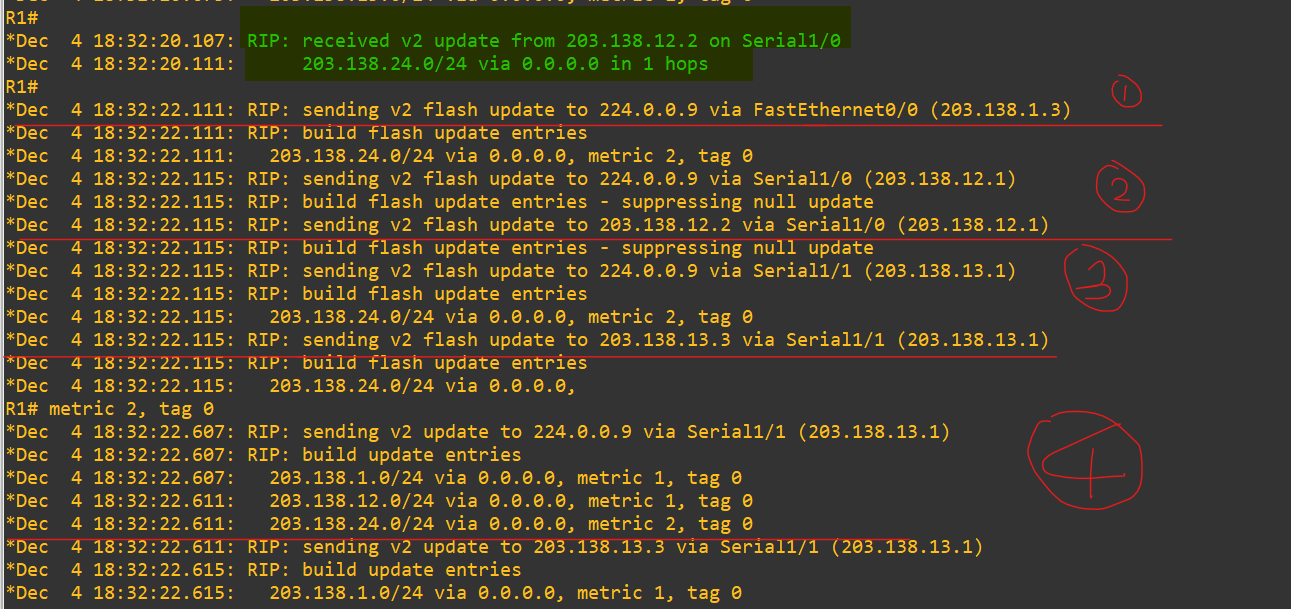


图3-14 R2配置好后R1的debug信息图

#### RIP协议的工作过程

首先，标出绿色的那两行表示R1收到了R2发来的update信息。R2告诉R1：从我这里到网络203.138.24.0/24 需要1跳。R1收到后，更新自己的路由表，将203.138.24.0/24加入到自己的路由表中，并将跳数从R2来的1跳加1变成2跳，也就是图3-14 ②中的2 metric。之后，R1通过组播，将该路由信息变化告知通过串口S1/0、S1/1，还有接口F0/0通知给它的邻居，如图3-14的①②③。在④中，R1已经构建好新的路由表，并将它们定期发送给自己的邻居。

### 3.2.2 RIP协议路由构建过程分析

以R10为例，因为R10我是最后配置的，信息比较完整。

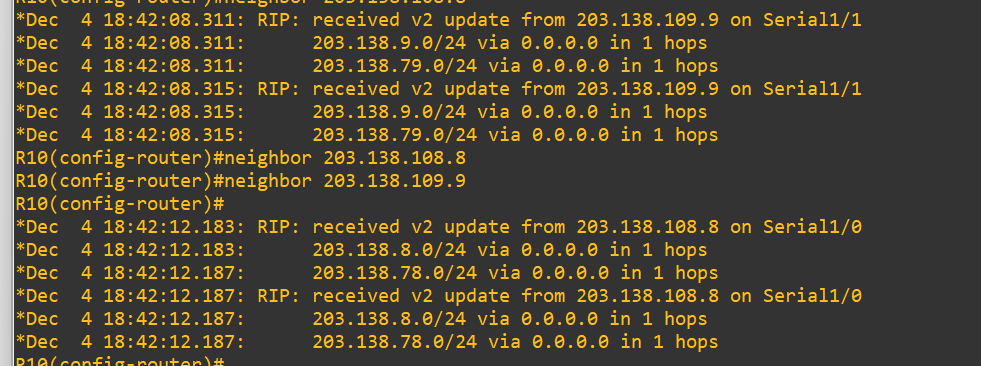


图3-15 R10构建路由表1

首先，R10收到从203.138.109.9和203.138.108.8发来的路由更新信息，告诉R10它们各自直连的网络有哪些，R10更新完路由表后，将新的路由表信息发送给和自己直连的各个接口，如图3-16。

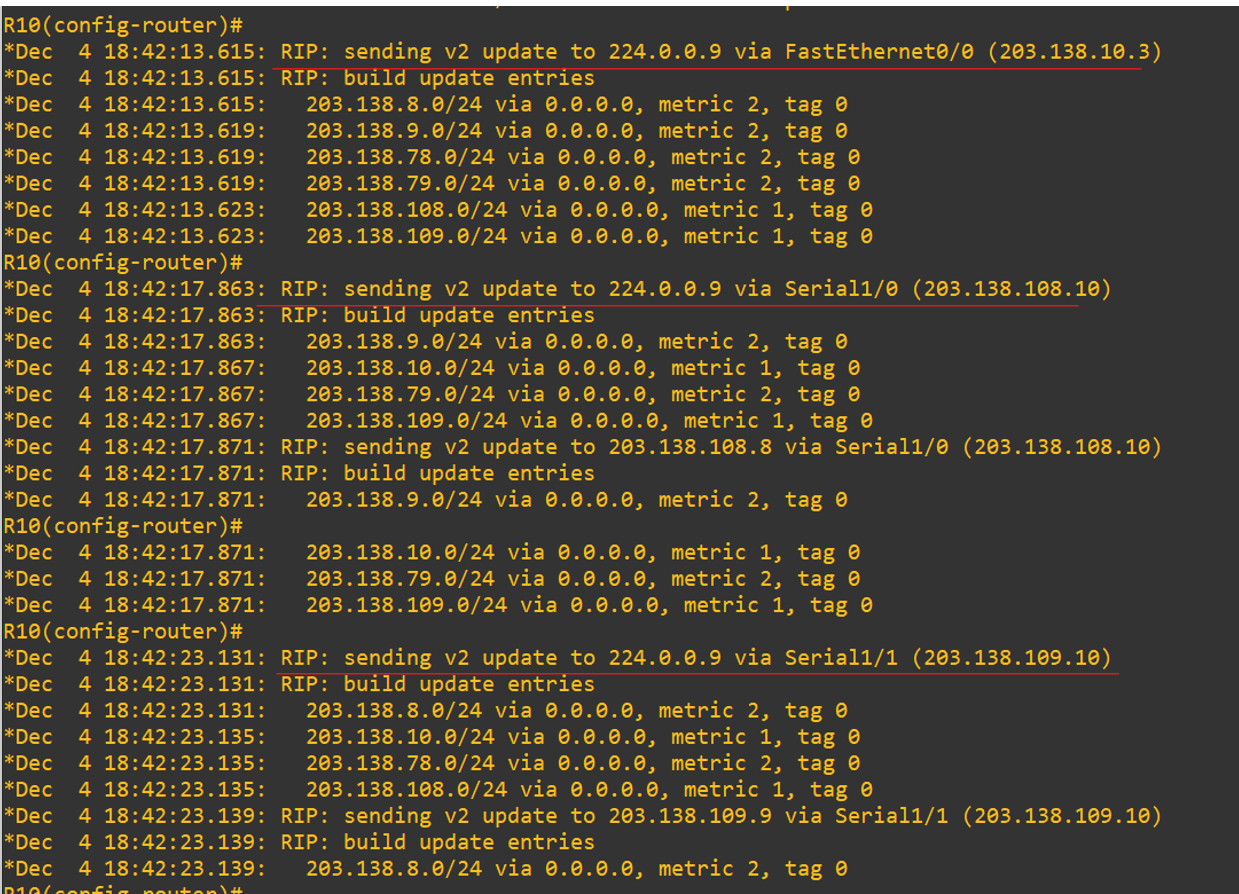


图3-16 R10发送路由表信息图1

作为R10的邻居R8或R9，在收到R10的消息后，将会更新各自的路由表信息，比如将R10连接的网络203.138.10.0记录到自己的路由表中，并设置跳数为2。更新完这个路由表信息后，它们会将更新的数据发送给自己的邻居。

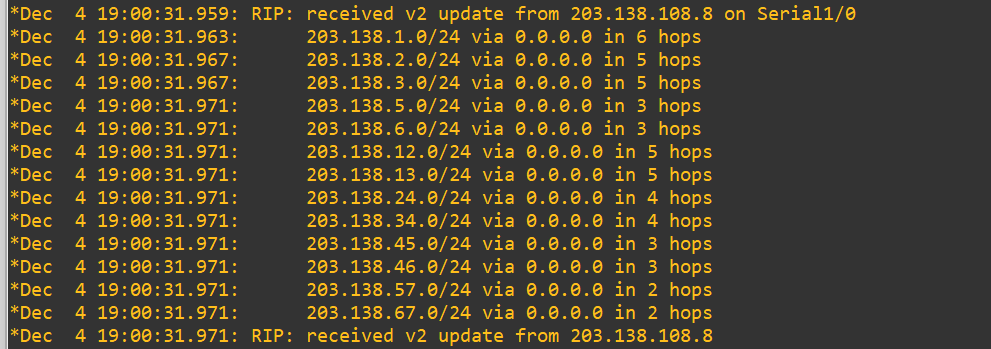


图3-17 R10收到R8路由表示意图

然后R10收到了来自R8的路由表信息，里面记载着R8能到达的网络号和所需要的跳数。如图2-18，从R8经过到203.138.1.0需要6跳，到达203.138.2.0需要5跳……到达203.138.67.0需要2跳。

查看路由表信息：在R10中输入show ip route，结果如下。其中C代表直连，R代表是通过RIP协议动态获取的路由信息。可以看到，这个通过RIP协议，整个网络拓扑的所有网络都包含在路由表里了。

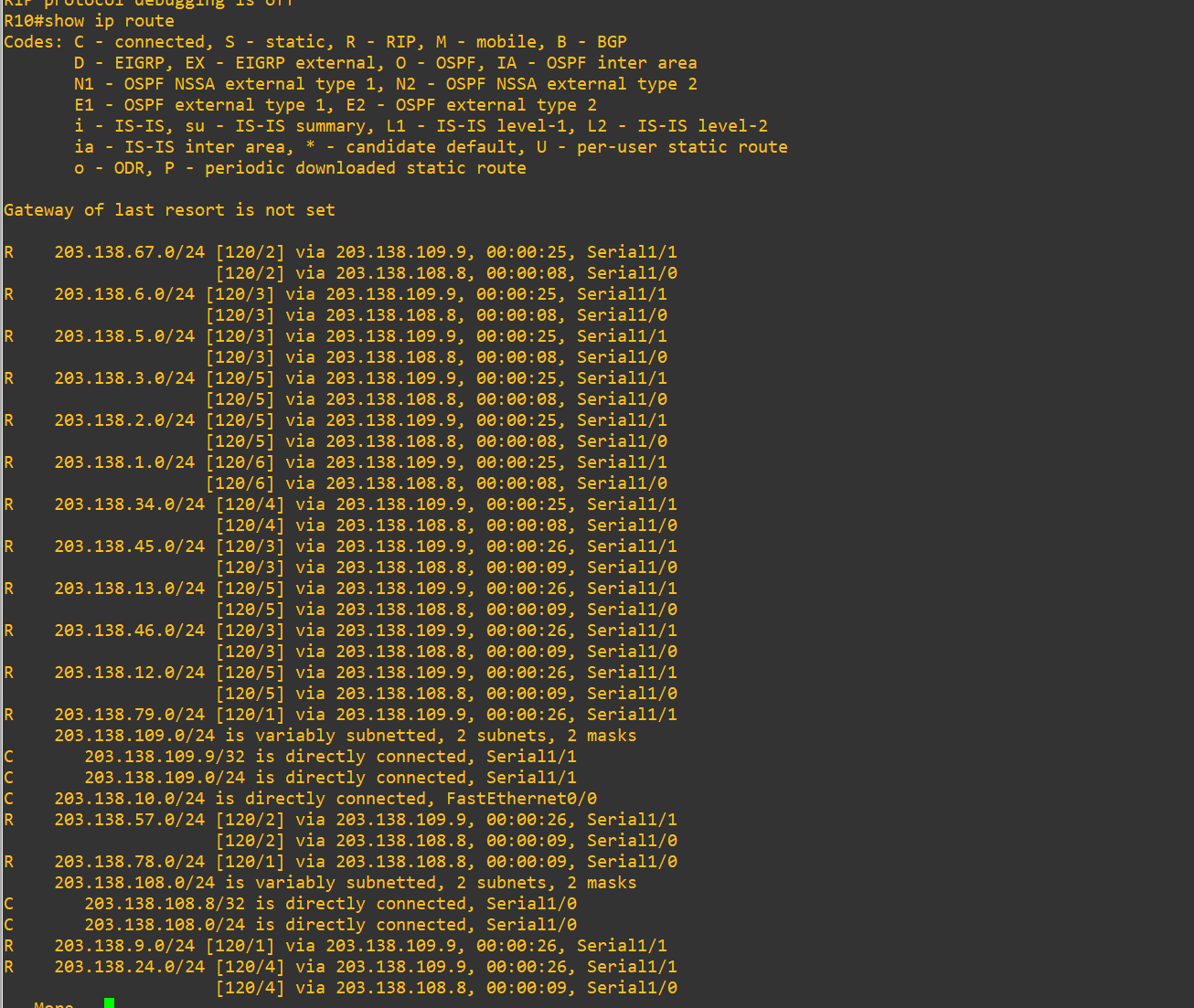


图3-18 R10路由信息图

最后检查网络之间能否互通，这里我选择使用相邻最远的PC1和PC10进行测试，测试结果如下：

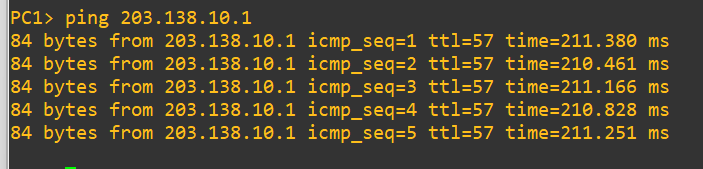


图3-19 PC1 ping通 PC10

### 3.2.3 RIP协议的水平分割

在R9的S1/1中，将水平分割关闭，步骤如下：

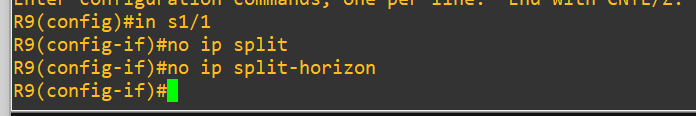


图3-20 关闭R9 S1/1的水平分割图

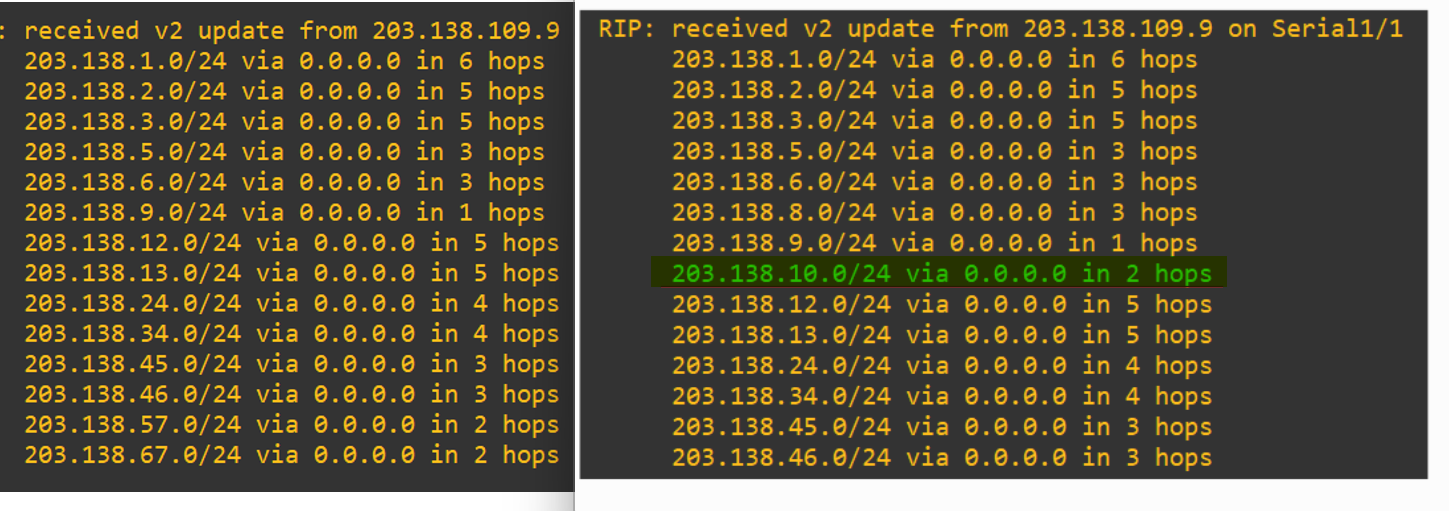
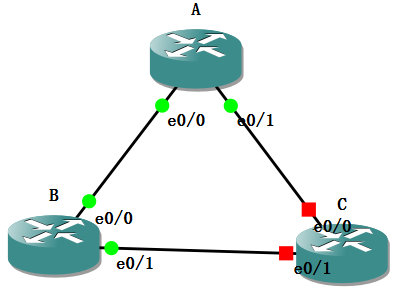
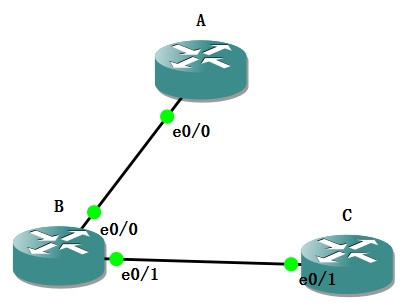
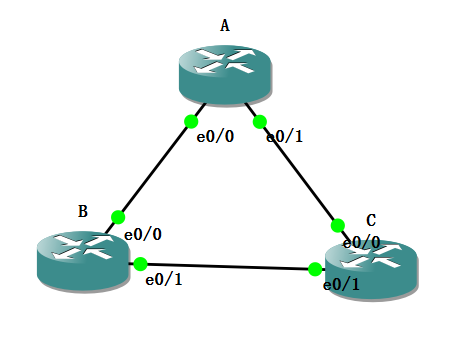


图3-21 R10 接收信息对比图

如图3-21，左边为关闭水平分割前从R9收到的信息，右边为关闭水平分割后从R9收到的信息。它们的差别就在于：关闭水平分割后，R9会向R10发送从R10发给R9的消息。水平分割的含义指的是RIP从某个接口接收到的路由信息，不会从该接口再发给邻居设备。它的好处是，不但减少了带宽消耗，还可以防止路由环路。

#### 路由环路解析



（a） （b） （c）

图3-22 路由环路示意图

路由环路是这样一种情况，拓扑图如图3-22。

在状态（a）时，A和B都直连C。当A和C之间断开后，如图状态（b），B通知A，到达C的距离是1，A此时只能通过B到达C。当B和C的连接也断开后，如图状态（c）。由于没有水平分割，A告诉B到达C需要2跳，因此B将自己到C的路由信息设置为3。B又告诉A到达C需要3跳，A将自己的路由信息修改成4跳并告诉B…….就这样形成了一个路由环路。这会大大影响拓扑的性能。

## 3.3 OSPF协议

### 3.3.1 OSPF协议配置过程

首先，将原来的RIP协议配置从所有路由器内删除干净，具体步骤为：conf t进入路由器的配置模式，输入no router rip，将rip协议删除。

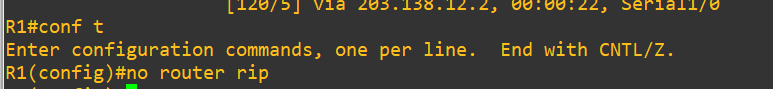


图3-23 删除rip协议示意图

在还没配协议前，打开OSPF的debug信息，用来观察网络中路由器建立路由信息的过程。打开debug信息命令为：

|  |
| --- |
| debug ip ospf events -- 打开事件  debug ip ospf flood -- 打开洪泛信息  sh ip ospf neighbor -- 展示邻居信息 |

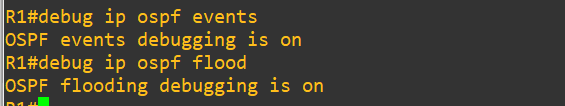


图3-24 打开R1 ospf的debug信息

由于打开debug信息后，配置过程难以截图，因此这一过程没有步骤图片。

打开R1的debug信息后，要配置R1的OSPF协议，步骤如下：

|  |
| --- |
| conf t  router ospf 1  network 203.138.1.0 255.255.255.0 area 0  network 203.138.12.0 255.255.255.0 area 0  network 203.138.13.0 255.255.255.0 area 0  in s1/0  ip ospf hello-interval 5  ip ospf dead-interval 20  in s1/1  ip ospf hello-interval 5  ip ospf dead-interval 20 |

步骤说明：

（1）router ospf 1 表示给R1配上ospf路由，放在进程1里

（2）network <网络号> <子网掩码> <区域号> 表示和R1直接相连的网络号，并将区域设置为0。在本次实验中，并不涉及区域的设计，所以都配置成area 0。

（3）进入相应的接口，给为该接口的hello包配置一个间隔。并设置一个间隔20，表示多久没收到这个邻居的消息就说明这个邻居宕机了。

配完R1之后就会显示如下debug信息：

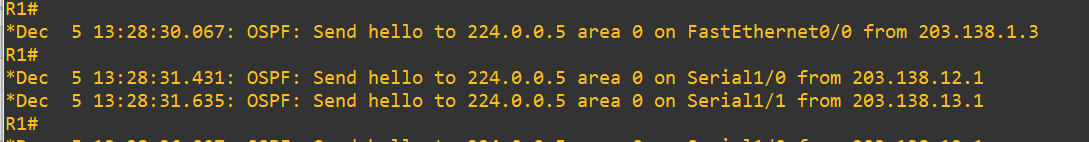


图3-25 R1刚配置好的debug信息图

同理，为R2配置OSPF协议，debug信息如下：

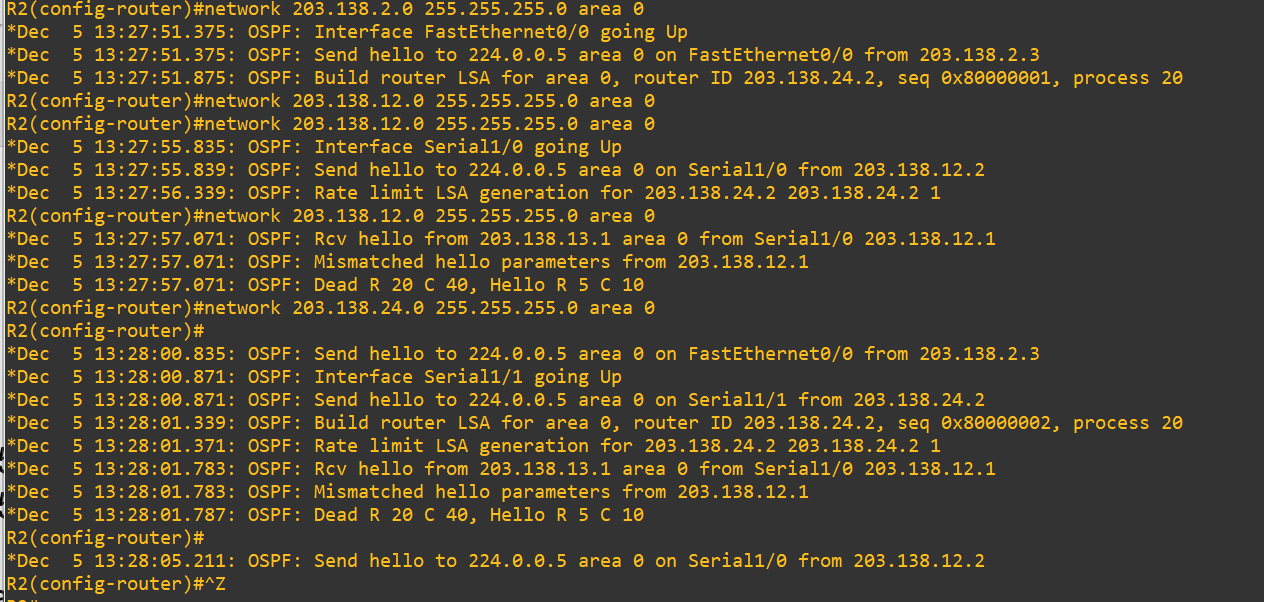


图3-26 R2配置OSPF的debug信息图

debug信息说明：

（1）OSPF: Interface FastEthernet0/0 going Up

OSPF: Send hello to 224.0.0.5 area 0 on FastEthernet0/0 from 203.138.2.3

这表示FastEthernet0/0接口状态变为Up，同时OSPF发送了Hello消息到224.0.0.5（OSPF的多播地址）以通知其他OSPF路由器。

（2）OSPF: Build router LSA for area 0, router ID 203.138.24.2, seg 0x80000001, process 2

表示OSPF正在为区域0构建Router LSA（Link State Advertisement）。

（3）\*Dec 5 13:35:21.039: OSPF: Rcv hello from 203.138.13.1 area 0 from Serial1/0 203.138.12.1

表示OSPF协议从Serial1/0接口收到了来自203.138.13.1的Hello消息。这是指有一个OSPF相邻路由器（203.138.13.1）通过Serial1/0接口发送了Hello消息。

在R1中输入 sh ip ospf neighbor，得到如下信息：

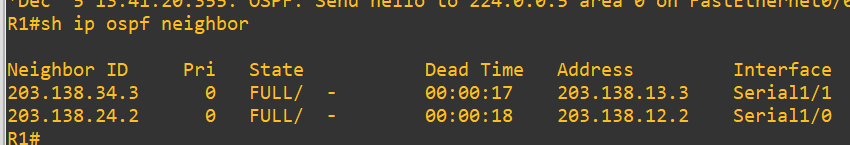


图3-27 R1的neighbor表

其中有两个neighbor信息，第一条表示 邻居203.138.34.3优先级是0，状态是FULL，还有17秒没收到邻居信息就说明它宕机了，邻居的路由ip地址是203.138.13.3，通过接口S1/1进行连接。

将所有路由的OSPF协议配置好后，检查所有路由器的路由表。

在R1下输入show ip route，显示R1的路由表，如下图所示，

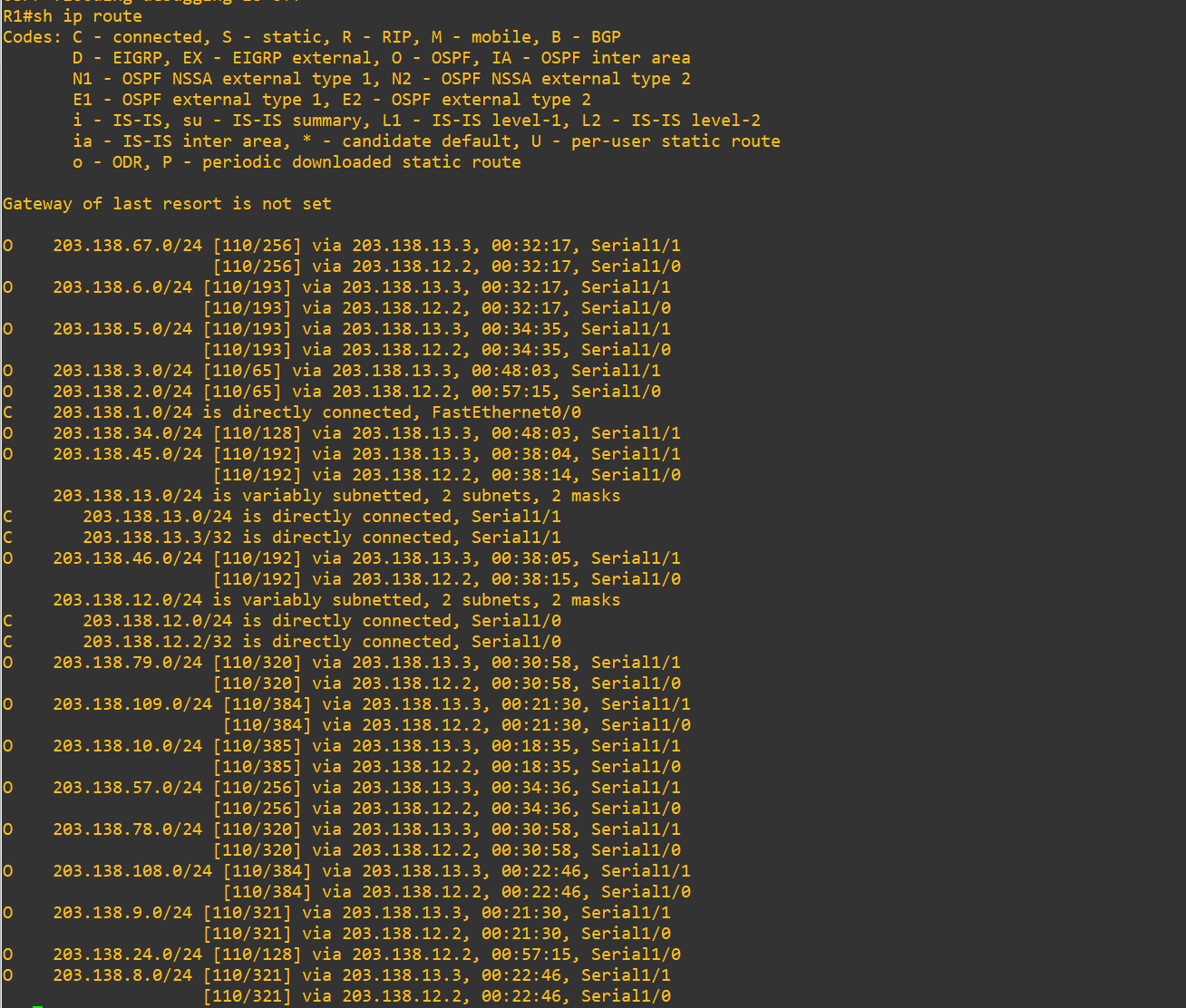


图3-28 R1的ospf路由表

其中，前缀为C的是直接连接的，前缀为O的是通过OSPF协议动态更新的路由信息。

尝试用网络中的每个PC来ping其他的PC，看看网络是否每个节点都能ping通。下面展示PC1 ping PC10，成功ping通，结果如下：

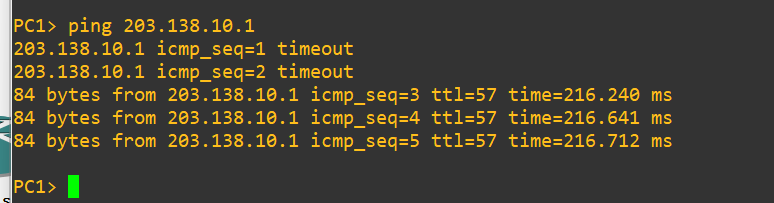


图3-29 PC1 ping PC10结果图

### 3.3.2 OSPF链路状态信息更新过程

将R1一直到R10配置好OSPF协议，观察R1的debug信息。当R10刚配置好时,R1收到的其中一条debug信息如下：

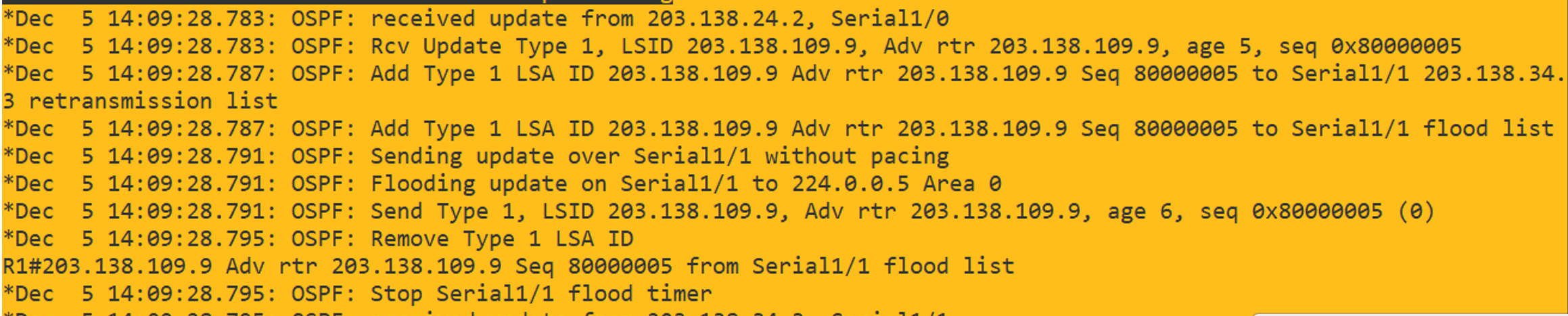


图3-30 R1的update debug信息

（1）OSPF: received update from 203.138.24.2, Serial1/0

这条语句表示从203.138.24.2，串口1/0收到了一条链路状态更新信息。

（2）OSPF: Rcv Update Type 1, LSID 203.138.109.9, Adv rtr 203.138.109.9, age 5, seq 0x80000005

表示收到了类型为1的OSPF路由更新，LSA的ID是203.138.109.9，广播路由器是203.138.109.9，路由信息的年龄是5秒，序列号是0x80000005。

（3）OSPF: Add Type 1 LSA ID 203.138.109.9 Adv rtr 203.138.109.9 Seq 80000005 to Serial1/1 203.138.34.3 retransmission list

表示将LSA ID为203.138.109.9的路由信息添加到Serial1/1接口的重传列表中。

（4）OSPF: Sending update over Serial1/1 without pacing

表示通过Serial1/1接口发送OSPF路由更新，不使用 pacing。

（5）OSPF: Flooding update on Serial1/1 to 224.0.0.5 Area 0

表示在Serial1/1接口向OSPF Area 0的多播地址224.0.0.5广播更新。

（6）OSPF: Send Type 1, LSID 203.138.109.9, Adv rtr 203.138.109.9, age 6, seq 0x80000005 (0)

表示发送了类型为1的OSPF路由更新，其中广播的消息来源于LSA ID 203.138.109.9的信息。

（7）OSPF: Remove Type 1 LSA ID 203.138.109.9 Adv rtr 203.138.109.9 Seq 80000005 from Serial1/1 flood list

表示将LSA ID为203.138.109.9的路由信息从Serial1/1接口的洪泛列表中移除。

（8）OSPF: Stop Serial1/1 flood timer

表示停止Serial1/1接口的洪泛计时器。

上述debug信息详细描述了OSPF协议在链路状态信息更改时一个路由器所作的工作。首先，路由器收到了一个状态更新信息，里面包含了是什么类型的更新，以及发送该路由信息的路由ip地址。接着，路由器更新自己的路由表。然后，路由器通过另一个串口发送更新信息的洪泛包，并设置洪泛计时器。当计时器到时的时候，就将该洪泛信息从洪泛表中删除，并停止该计时器。

## 3.4 RIP和OSPF协议的区别

### 3.4.1 RIP协议特点

RIP协议每隔一段时间就送出自己完整的路由表到所有激活的接口。（不管网络有没有变换，周期性的。将自己所连接的网段和学到的路由信息，通过这些口告诉其他的路由器。如果路由表条数多，那么可能需要发好几个包才能将路由表通告出去，如果条数较少，可能一个数据包就可以发送完毕） RIP协议选择最佳路径的标准就是跳数，认为到达目标网络经过的路由器最少的路径就是最佳路径。 默认它所允许的最大跳数为15跳，也就是说16跳的距离将被认为是不可达的。在小型网络中，RIP会运转良好，但是对于使用慢速WAN连接的大型网络或者安装有大量路由器的网络来说，它的效率就很低了。

### 3.4.2 RIP协议工作过程

路由器在刚刚开始工作时，只知道到直接连接的网络的距离（此距离定义为 0）。它的路由表是空的。 以后，每一个路由器也只和数目非常有限的相邻路由器交换并更新路由信息。 经过若干次更新后，所有的路由器最终都会知道到达本自治系统中任何一个网络的最短距离和下一跳路由器的地址。（最后运行RIP协议的路由器，经过一段时间更新之后，都知道网络当中有多少网段，下一跳给谁） 距离矢量算法：

路由器收到相邻路由器（其地址为 X）的一个 RIP 报文： (1) 先修改此 RIP 报文中的所有项目：把“下一跳”字段中的地址都改为 X，并把所有的“距离”字段的值加 1。 (2) 对修改后的 RIP 报文中的每一个项目，重复以下步骤： 若项目中的目的网络不在路由表中，则把该项目加到路由表中。

若在网络中，且下一跳字段给出的路由器地址是同样的，则把收到的项目替换原路由表中的项目。若收到项目中的距离小于路由表中的距离，则进行更新。 否则，什么也不做。 (3) 若3分钟还没有收到相邻路由器的更新路由表，则把此相邻路由器记为不可达路由器，即将距离置为 16，即不可达，并且将到该网段的路由删除。

### 3.4.1 OSPF协议的特点

OSPF协议使用 Dijkstra 算法来选择最短路径并将其加入到路由表中。 OSPF 在网络中建立一个联通域，该域中的所有路由器都互相交换信息，以便维护一张完整的拓扑图。当路径发生变化时，路由器会通过 LSA (Link State Advertisements) 报告给同一联通域内的其他路由器。 每个路由器都使用 LSA 来更新它的链路状态数据库并重新计算出最短路径。这些信息则会被同步到整个联通域内的其他路由器。

运行OSPF路由器之间交互的是LS（Link State，链路状态）信息，而不是直接交互路由。LS信息是OSPF能够正常进行拓扑及路由计算的关键信息。 OSPF路由器将网络中的LS信息收集起来，存储在LSDB中。路由器都清楚区域内的网络拓扑结构，这有助于路由器计算无环路径。

### 3.4.2 OSPF协议的工作过程

（1）OSPF路由器启动后，便会通过OSPF接口向外发送Hello报文用于发现邻居。收到Hello报文的OSPF路由器会检查报文中所定义的一些参数，如果双方的参数一致，就会彼此形成邻居关系，状态到达2-way 即可称为建立了邻居关系。 Hello报文用来发现和维持OSPF邻居关系。 Hello报文的作用： 1° 邻居发现：自动发现邻居路由器。 2° 邻居建立：完成Hello报文中的参数协商，建立邻居关系。 3° 邻居保持：通过Keepalive机制，检测邻居运行状态。 （2）建立邻接关系，形成链路状态数据库

形成邻居关系的双方不一定都能形成邻接关系，这要根据网络类型而定。当双方成功交换DD报文，并同步LSDB后，才形成真正意义上的邻接关系。路由器一旦建立了邻居关系，就可以创建链路状态数据包。

（3）交换链路信息

路由器将描述链路状态的LSA泛洪到邻居，最终形成包含网络完整链路状态信息的链路状态数据库。

（4）计算最短路径

路由区域内的每台路由器都可以使用SPF算法来独立计算路由。

（5）链路状态更新

当链路状态更新时，新加入的路由器或者宕机的路由器的邻居会察觉到链路状态更新，并先更新自己的链路状态数据库。然后构建链路状态改变的广播包，将该广播包放到自己的洪泛列表中。在接下去一段时间内朝自己的各个启动的邻居发送洪泛包。当洪泛时间到达后，该路由器会把该洪泛包从洪泛列表里删除，并停止洪泛计时器。（具体请看[3.3.2节图3-30以及下方解析过程](#_2.4.2_OSPF链路状态信息更新过程)）

# 四、实验结果（包括最终实验结果，需要截图）

## 4.1 RIP协议实验结果

以R10为例，因为R10我是最后配置的，信息比较完整。

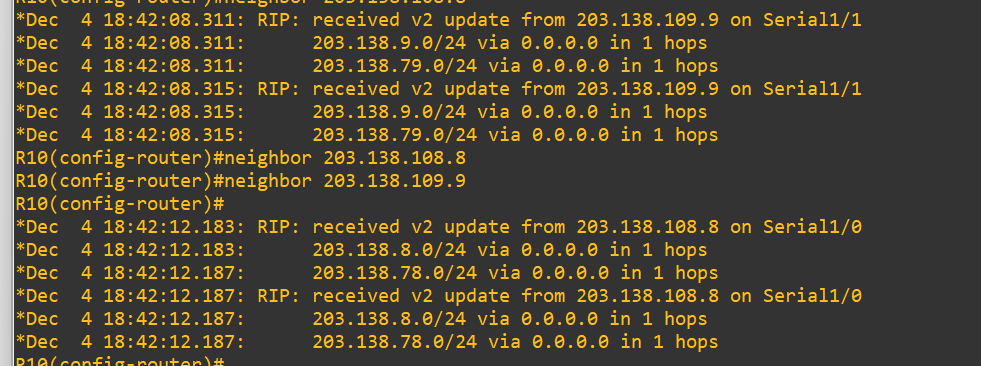


图4-1 R10构建路由表1

首先，R10收到从203.138.109.9和203.138.108.8发来的路由更新信息，告诉R10它们各自直连的网络有哪些，R10更新完路由表后，将新的路由表信息发送给和自己直连的各个接口，如图4-5。

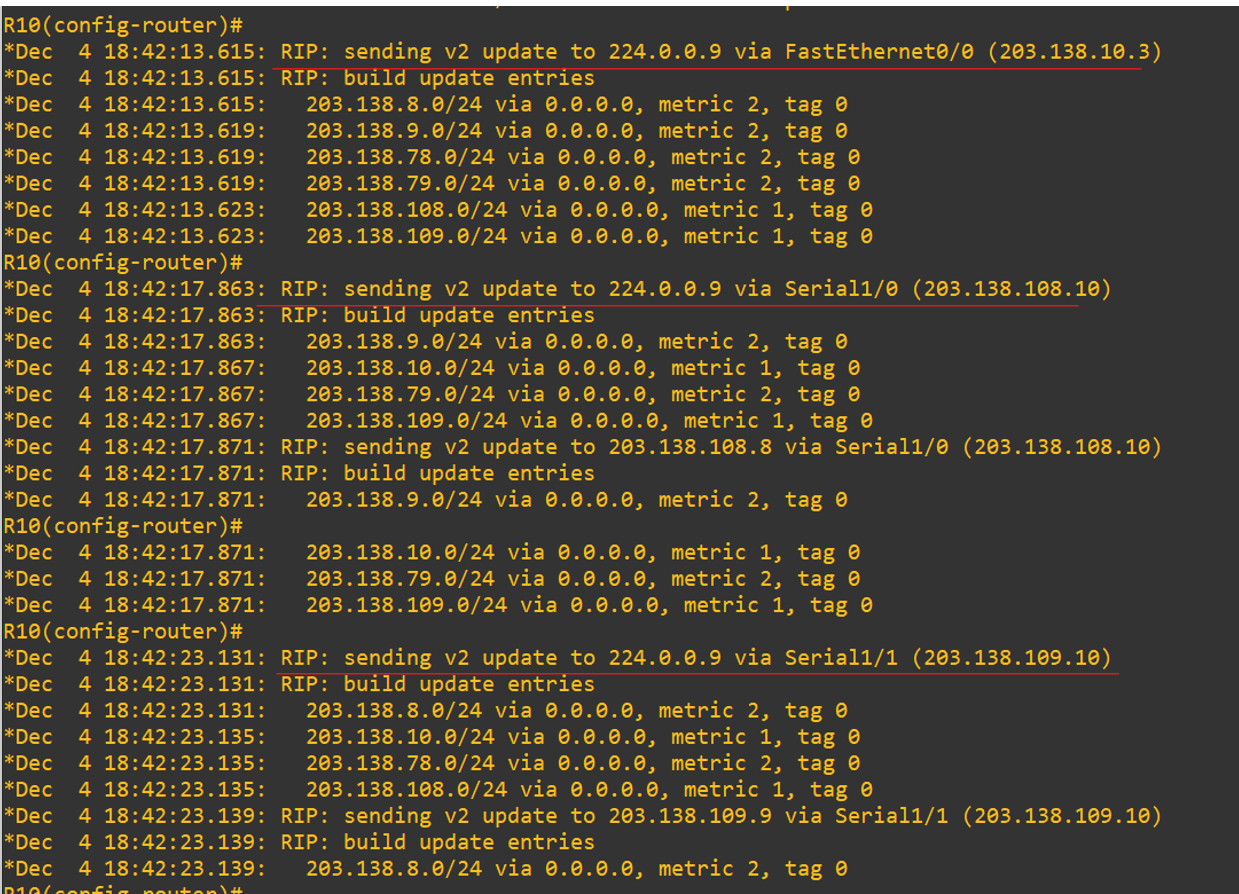


图4-5 R10发送路由表信息图1

作为R10的邻居R8或R9，在收到R10的消息后，将会更新各自的路由表信息，比如将R10连接的网络203.138.10.0记录到自己的路由表中，并设置跳数为2。更新完这个路由表信息后，它们会将更新的数据发送给自己的邻居。

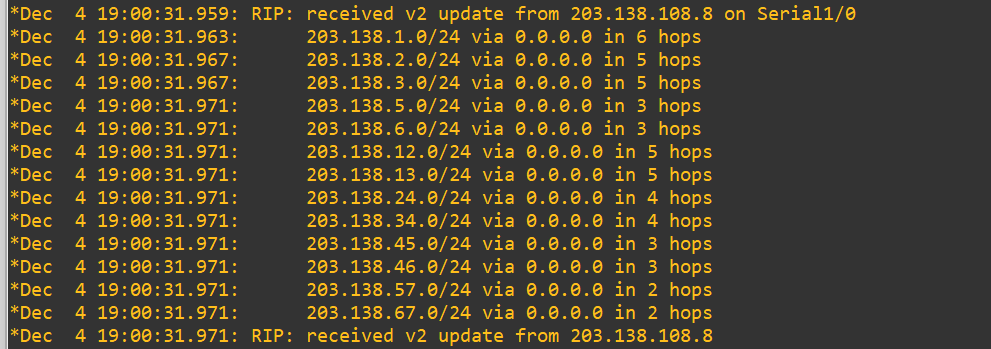


图4-3 R10收到R8路由表示意图

然后R10收到了来自R8的路由表信息，里面记载着R8能到达的网络号和所需要的跳数。如图4-4，从R8经过到203.138.1.0需要6跳，到达203.138.2.0需要5跳……到达203.138.67.0需要2跳。

查看路由表信息：在R10中输入show ip route，结果如下。其中C代表直连，R代表是通过RIP协议动态获取的路由信息。可以看到，这个通过RIP协议，整个网络拓扑的所有网络都包含在路由表里了。

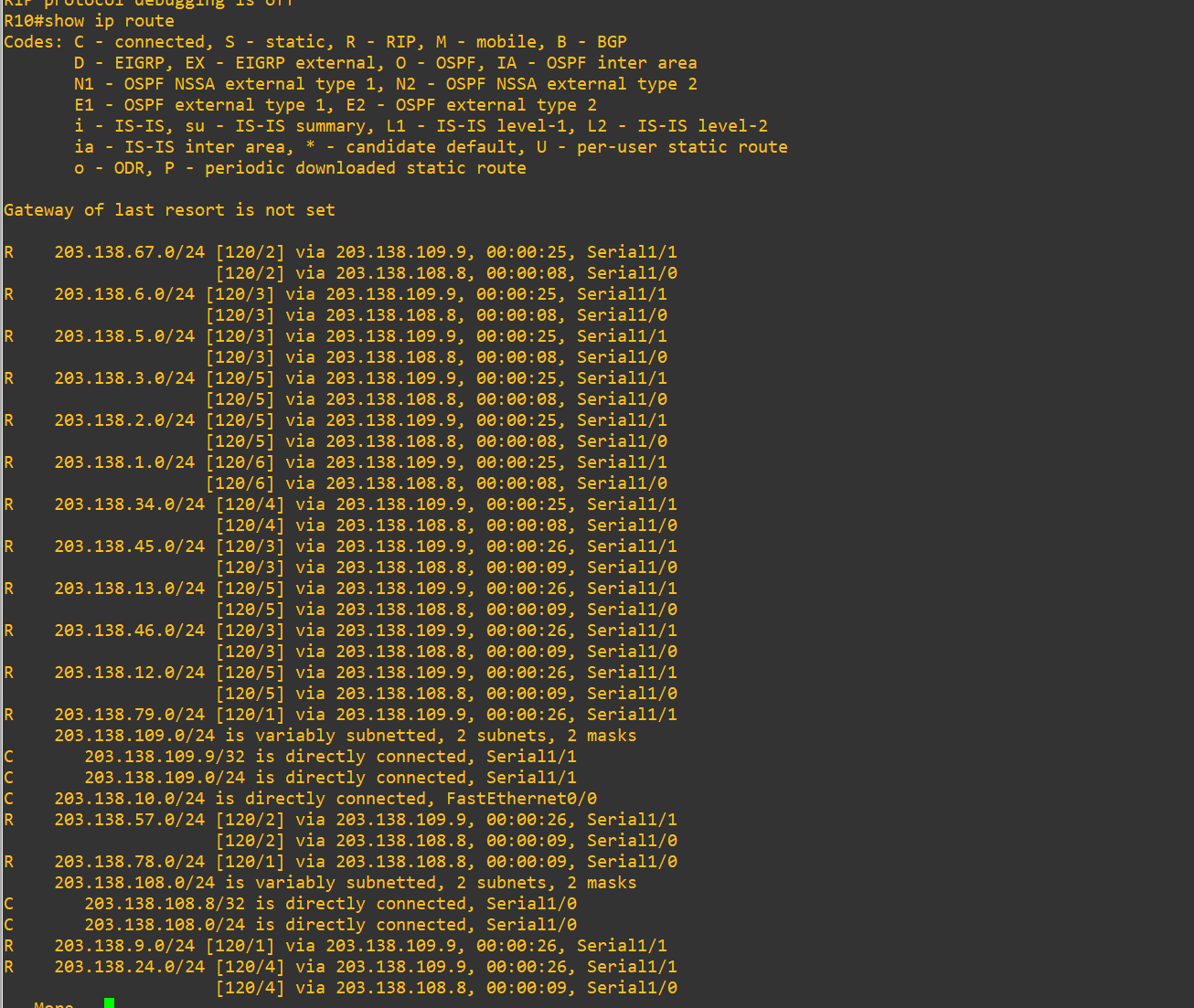


图4-4 R10路由信息图

最后检查网络之间能否互通，这里我选择使用相邻最远的PC1和PC10进行测试，测试结果如下：

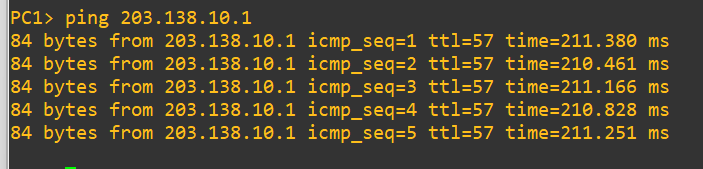


图4-5 PC1 ping通 PC10

对于水平分割协议，结果如下：

在R9的S1/1中，将水平分割关闭，步骤如下：

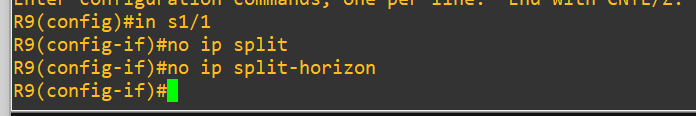


图4-6 关闭R9 S1/1的水平分割图

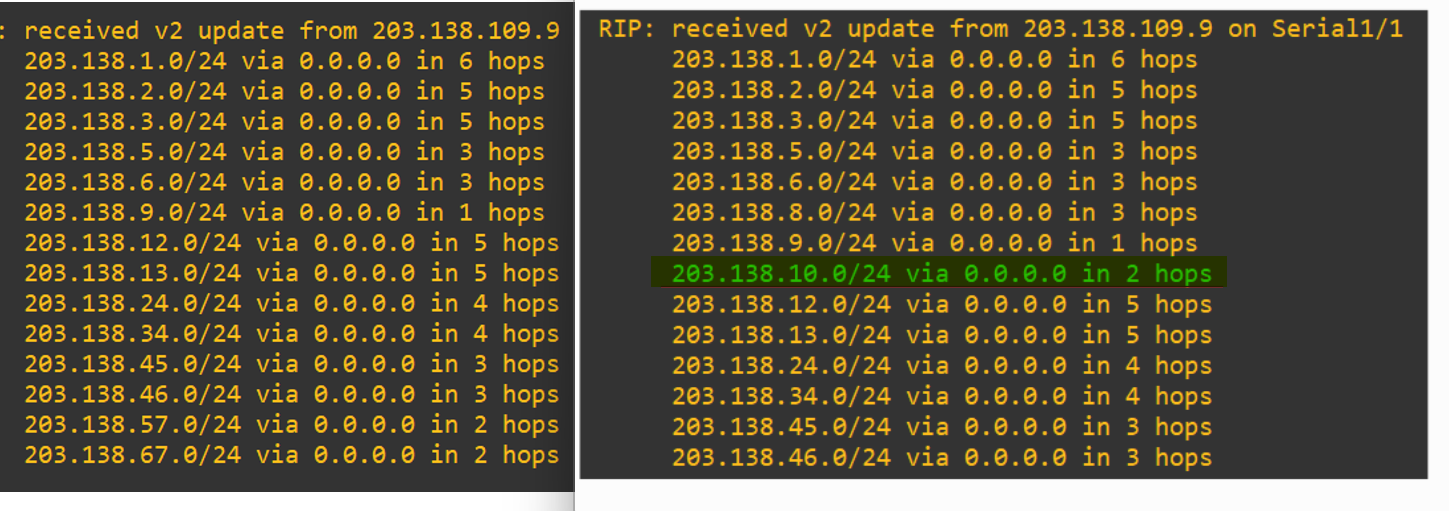


图4-7 R10 接收信息对比图

如图4-7，左边为关闭水平分割前从R9收到的信息，右边为关闭水平分割后从R9收到的信息。它们的差别就在于：关闭水平分割后，R9会向R10发送从R10发给R9的消息。水平分割的含义指的是RIP从某个接口接收到的路由信息，不会从该接口再发给邻居设备。它的好处是，不但减少了带宽消耗，还可以防止[路由环路](#_路由环路解析)（在实验报告3.2.3节有路由环路解析）。

RIP协议的debug信息可以看实验报告的 [RIP协议的工作流程](#_RIP协议的工作过程)

## 4.2 OSPF协议实验结果

将所有路由的OSPF协议配置好后，检查所有路由器的路由表。

在R1下输入show ip route，显示R1的路由表，如下图所示，

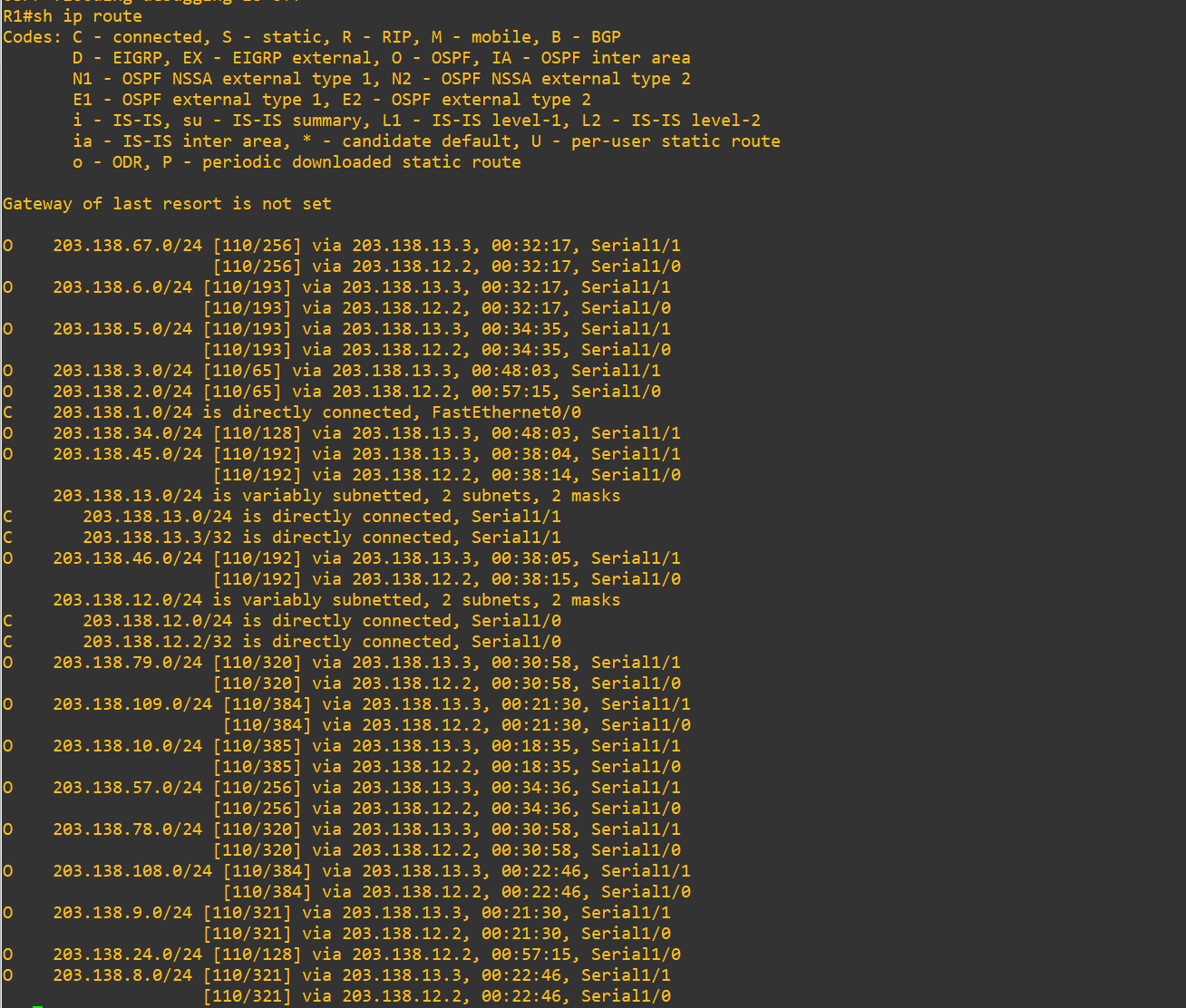


图4-8 R1的ospf路由表

其中，前缀为C的是直接连接的，前缀为O的是通过OSPF协议动态更新的路由信息。

尝试用网络中的每个PC来ping其他的PC，看看网络是否每个节点都能ping通。下面展示PC1 ping PC10，成功ping通，结果如下：

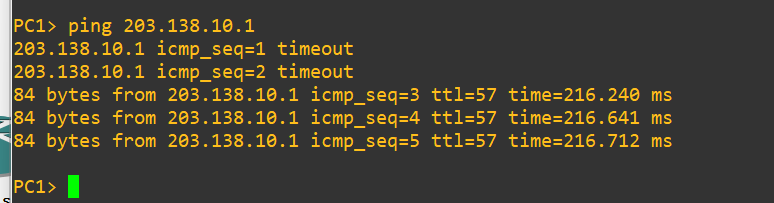


图4-9 PC1 ping PC10结果图

OSPF的链路状态更新过程及分析可以查看[3.3.2 OSPF链路状态更新过程](#_3.4.2_OSPF链路状态信息更新过程)

# 五、实验中的问题及心得

## 5.1 实验中的问题

（1）第一次做实验三时，我使用了Dynamips作为仿真工具。我的PC3死活ping不通别的设备，别的设备也ping不通PC3。我检查了我的ip地址设置，检查了我的默认路由设置，检查了我的配置文件编写，几乎用了所有的查看命令，比如show ip route、show run、show interface等等……但是都没能发现实际的问题所在，只有show interface显示出我的PC3 f0/0接口和R3的f0/0接口protocol 是down的状态，但是接口本身是up的状态。我用PC3自己ping自己倒是能ping通。我询问了老师原因，才知道原来Dynamips会有一些bug，出现连接不通的情况下可以去配置文件中，把原来R3 f0/0连接到PC3的配置改到PC3 f0/0 连到R3上。重新打开命令行和设备，才成功ping通。

（2）见识到Dynamips的险恶之后，我尝试使用GNS3作为我的仿真工具。但是在配GNS3的时候卡了半天。首先是GNS3的虚拟机老是装不成功，我又决定不装虚拟机，直接在本机上跑。但是老是出现wait for localhost …… 的弹窗，让我烦不胜烦。我删掉重装了好几次GNS3，最后沉下心来成功安装了虚拟机，并且设置好了环境，也导入了所需要的路由器，这才得以安心实验。

（3）在GNS3软件上，我被一个小bug折磨了好久。一开始，我想知道GNS3的简单工作流程，我设计了一个简单的拓扑，两台PC机和一台路由器，我设置好了IP和网关，但是两台PC怎么样也ping不通。我上网找也没找到有相关的问题，我还以为是我安装软件忘记配置什么东西了。

我想着抓包分析一下包在哪里被丢掉了，但是在抓包时我又犯难了。打开Wireshark，几乎所有的网口我都尝试了一遍，但是都没有找到我想看到的包。机缘巧合之下，我在GNS3界面中右键点击了一条相连的线，看到了start capture的选项，这回才成功抓包了。

对于一开始设计的简单网络，我看了从PC1到PC2的数据包和PC2到PC2的数据包，发现PC2去ping PC1的时候根本没发数据包出来。我检查了好几次PC2的配置，觉得都没问题，有理由怀疑PC2坏了。我又拉出了一个PC3，放到该网络里，这回PC1 ping PC3就成功了，PC3 ping PC1也成功了，但是PC2还是ping不通，那就说明是PC2坏了，应该是这个软件的小bug。

（4）做实验的时候我对于RIP协议和OSPF协议还是懵逼的状态，我忘记了什么是距离矢量算法，什么是链路状态信息，我也不记得RIP协议里路由器都发送什么消息，OSPF里路由器都发送什么消息，以及两个协议里路由表是怎么维护的。我看了老师发的ppt，找了网上的公开课，也看了好多博客，最后搭配着自己实验，才逐渐理清它们的工作过程。

## 5.2 心得体会

在遇到Dynamips时，遇到连接不通的情况，通过修改配置文件将连接反转是一种巧妙的解决方法。这种情况下，灵活运用实践经验和尝试不同的方法，最终找到解决方案是非常重要的。

在安装GNS3时，遇到虚拟机安装失败和本地连接问题，最终通过耐心多次尝试、重新安装以及不放弃的精神解决了问题。这强调了在面对技术挑战时需要具备坚韧不拔的毅力和解决问题的能力。

在解决实际网络问题中，我知道了常常可以采用以下的方法来找到问题：

（1）通过ping自己可以确认设备的本地网络配置是否正确，包括IP地址、子网掩码、默认网关等。如果设备无法ping通自己，说明可能存在本地配置错误或者接口状态问题。

（2）利用命令行工具查看各个设备各个端口的状态以及接口信息，比如使用命令行工具show命令，可以获取实时的设备状态信息，帮助定位问题。

（3）如果前两种还没定位到问题时，可以采用抓取每一个接口通过的数据包，通过抓包分析，可以确定数据包在网络中的流动情况，以及是否在某一特定接口上被丢弃。这有助于追踪问题发生的位置，确定是否是某个特定的链路或设备导致的问题。

同时，通过这次实验，我也掌握了RIP 和 OSPF 以及ARP协议的工作流程。这次实验花了我好多时间，配置环境花了一天半，剩下一天半大多数时间都在熟悉命令，检查问题、排除问题。现在，我已经熟悉了整个配置的流程以及网络的工作流程，我可以很快的为设备配置好这两个协议，并分析每个包的传输过程。

还有部分心得我写在了[3.4 RIP和OSPF协议的区别](#_3.5_RIP和OSPF协议的区别)

# 六、实验思考

## 6.1 不同方式配置路由丢包解析

实验中，采用下一跳和转发接口这两种方式配置PC11和PC10的静态路由有什么区别？会导致在你的拓扑结构中从PC11 ping PC10时的丢包数有什么变化？需要用你的拓扑中PC11和PC10的arp 表和路由表中的内容来解释，要附带截图。

采用下一跳的方式配置静态路由，步骤如下图6-1 ~ 图6-4：

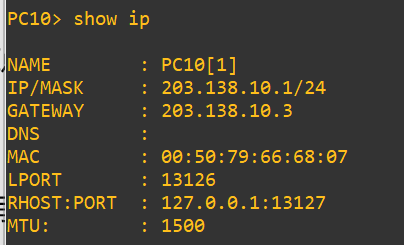
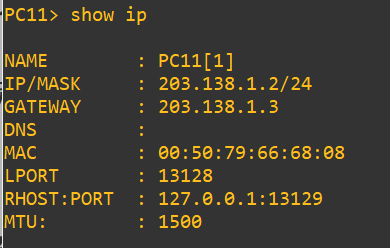


图6-1 PC11和PC10下一跳配置图

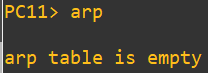
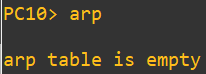


图6-2 PC11和PC10的初始arp表

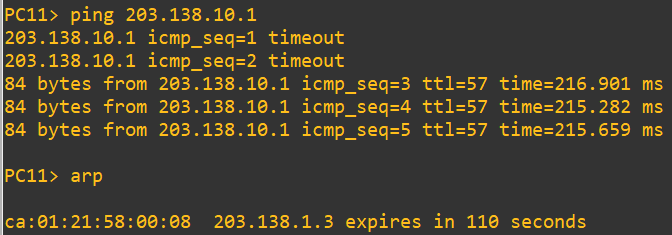


图6-3 PC11第一次ping PC10

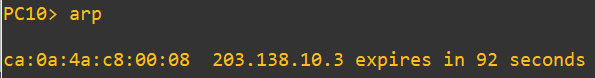


图6-4 PC10被ping通后的arp表

根据上图所示，当用下一跳的方式配置PC的静态路由时，两个PC的初始arp表都是空的。当PC11 ping PC10时，PC11先发给网关，但是PC11并不知道网关的MAC地址，这时候会丢一包。当网关返回它的MAC地址后，PC11会把ping的包发给网关。之后由于路由器之间是点对点网络，不是广播型网络，路由器传输不会丢包。但是当R10给PC10转发这个ping的包时，也会丢包，因为R10并不知道PC10的MAC地址，这是第二次丢包。PC10给PC11返回回应包的时候就不会出现丢包的情况了，因为这时候MAC地址都知道了，PC和路由器会根据自己的arp表进行相应的转发操作。

接下来分析用接口转发配置会出现什么情况。由于原先GNS3上我使用的是软件带的PC机，而不是用路由器模拟，所以我重新设置了一个拓扑图，在Dynamips仿真运行，里面的PC机用路由器来模拟，拓扑图如下：

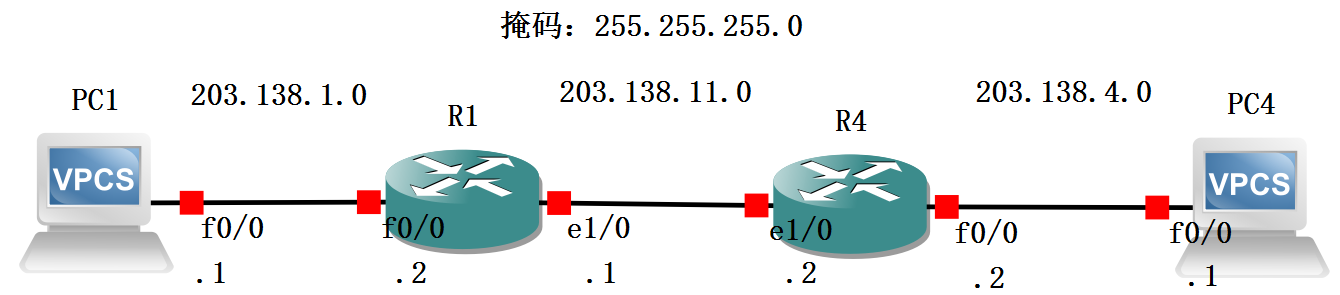


图6-5 接口转发拓扑图

采用转发接口的方式配置静态路由，步骤及结果如下，PC1 ping PC4丢了三次包。



图6-6 配置PC1和PC4的接口转发

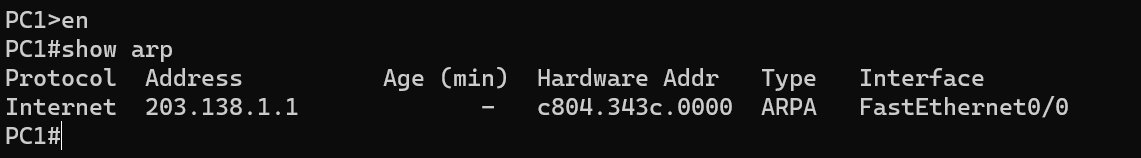


图6-7 查看PC1的arp表

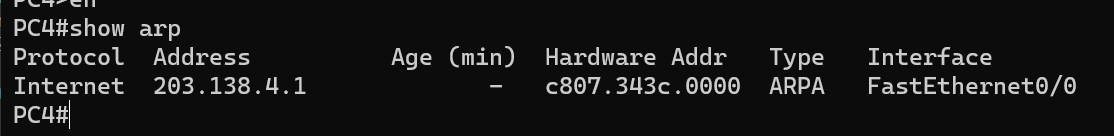


图6-8 查看PC4的arp表

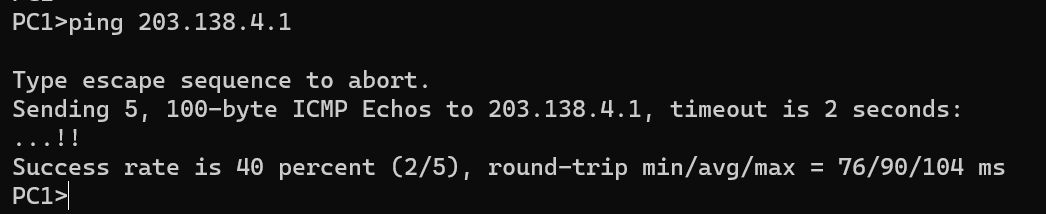


图6-9 PC1 ping PC4结果

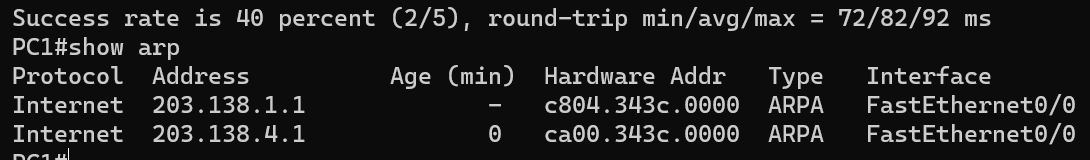


图6-10 查看ping通后PC1的arp表

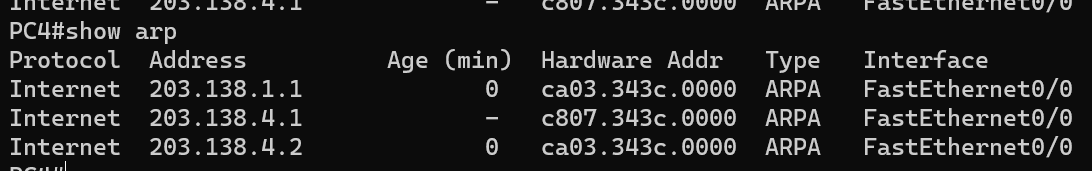


图6-11 查看ping通后的PC4的arp表

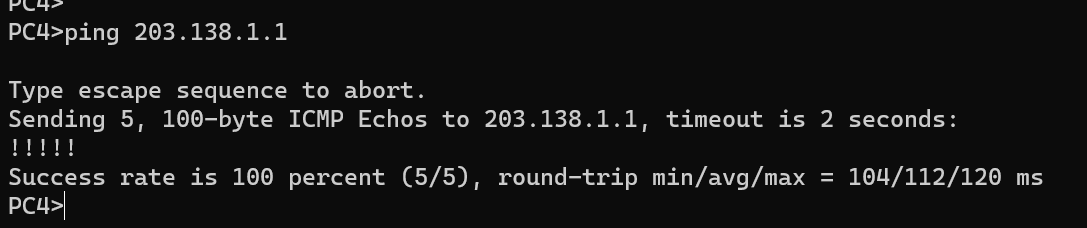


图6-12 第一次ping通后用PC4 ping通PC1

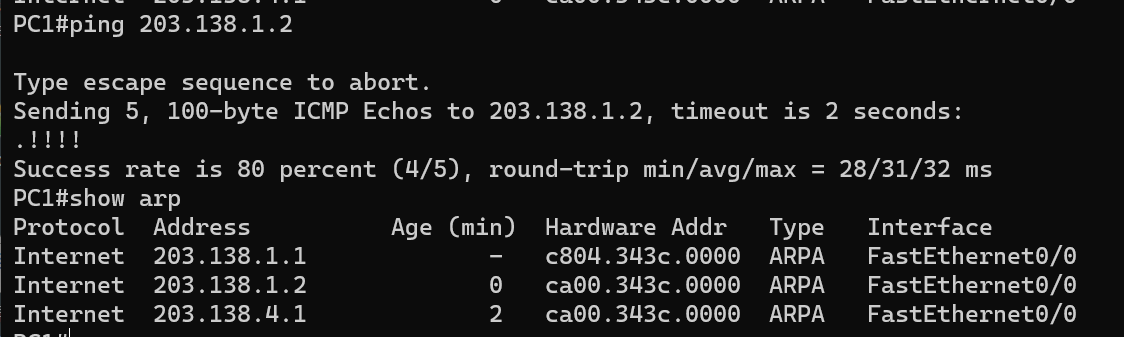


图6-13 用PC1 ping R1，arp表变化

在ping操作前，我查看了PC1和PC4的arp表，里面只有自己的表项，没有其他的表项。用PC1 ping PC4，丢包数为3，以下是丢包分析：

PC1 ping PC4，由于PC1没有网关IP地址， arp包解析出来只有PC4的IP地址，却没有网关的IP地址。所以第一次PC1在网络内广播，R1收到了PC1的arp请求，就返回一个回应，告诉PC1：要去PC4就要先发到我的f0/0接口，在这里第一次丢包。

因此，PC1的arp表中就多了一组对应关系，是PC4的IP地址对应R1的f0/0接口的MAC地址。同时，R1的arp表中也多了PC1和它的MAC地址的对应关系。然后，PC1在发包给R1。路由器中由于是点对点网络，不是广播型网络，所以R1给R4直接转发，不会丢包。R4给PC4时由于R4不知道PC4MAC地址所以这时候还会发一个广播包，询问PC4的MAC地址，得到PC4的MAC地址后才能有效转发，这里又丢一次包。

在PC4收到PC1的ping包后，PC4需要返回回应，但是PC4的arp表中并没有PC1的IP和MAC地址对应关系，所以它会在网络内先发一个广播包，询问PC1的MAC地址。R4收到PC4的arp请求后，告诉PC4，发到PC1的IP地址需要走我的f0/0接口。所以PC4的arp表中多了一组对应关系，PC1的IP对应R4的f0/0接口MAC地址。这是第三次丢包。然后PC4才能返回一个回应包，之后就不会再丢包了。因此是丢包三次。

在PC1 ping 通PC4之后，两个主机都有对方的IP地址和MAC地址的对应关系，虽然是PC的IP和Router的接口MAC，但是双方都知道到达对方那里需要怎么走了。所以再尝试PC4 ping PC1时，不会丢包。

最后我尝试了PC1 ping R1的f0/0接口。由于PC1的arp表项中并没有R1 f0/0的ip地址和MAC地址对应关系，所以它第一次会在网络内发一个arp广播包，询问R1 f0/0 的IP地址对应的MAC地址，所以会丢一次包。而在R1返回给PC1时，由于在R1中已经建立了关于PC1 的arp表对应关系，所以返回时并不会丢包。

## 6.2 RIP和OSPF的协议工作过程

对照所截获的消息，说明RIP协议有无水平分割的工作流程；以及OSPF协议在点对点网络和广播型网络上工作流程。附截图和对消息的说明。

说明RIP协议有无水平分割的工作流程：请看[3.2.3 RIP协议水平分割](#_2.3.3_RIP协议的水平分割)。

说明OSPF在点对点网络和广播型网络上工作流程。详细信息请看[3.3.2 OSPF链路状态信息更新过程](#_2.4.2_OSPF链路状态信息更新过程)以及[3.4.2 OSPF协议的工作过程](#_2.5.2_OSPF协议的工作过程)。下面我也给出了简要的说明：

1、OSPF在点对点网络上的工作流程：

（1）邻居发现：

在点对点网络中，OSPF路由器直接连接到另一个路由器，因此邻居关系相对简单。当两个路由器相互连接时，它们通过发送Hello消息来发现彼此，建立邻居关系。

（2）邻居关系建立：

通过交换Hello消息，路由器确认彼此的OSPF参数，包括区域号、Hello间隔、路由器ID等。如果相邻路由器的OSPF参数匹配，它们就建立邻居关系。

（3）链路状态数据库同步：

一旦建立邻居关系，路由器会交换链路状态数据库信息。路由器将自己的链路状态数据库信息发送给邻居，使两者保持同步。

（4）最短路径计算：

每个路由器使用链路状态数据库中的信息计算最短路径树。SPF算法应用于链路状态数据库，确定最短路径并生成路由表。

（5）路由表更新：

路由器根据计算得到的最短路径更新其路由表。每个路由器的路由表都反映了网络拓扑的当前链路状态。

2、OSPF在广播型网络上的工作流程：

（1）邻居发现：

在广播型网络中，路由器通过多播Hello消息来发现潜在的邻居。路由器会定期发送Hello消息，而其他路由器监听这些消息以识别潜在的邻居。

（2）邻居关系建立：

当一个路由器收到另一个路由器的Hello消息时，它会回应以建立邻居关系。邻居关系的建立还包括交换OSPF参数和验证邻居身份。

（3）链路状态数据库同步：

一旦建立邻居关系，路由器通过发送LSA来同步链路状态数据库。这种同步确保网络中的所有路由器都有相同的链路状态数据库。

（4）最短路径计算：

在广播型网络中，每个路由器都能够收到其他路由器的LSA，从而构建完整的链路状态数据库。SPF算法再次用于计算最短路径树。

（5）路由表更新：

路由器更新其路由表，以反映最新的最短路径信息。这确保了在广播型网络中路由器都具有相同的最短路径信息。

总体而言，OSPF在点对点网络和广播型网络上的基本工作流程相似，都涉及邻居发现、邻居关系建立、链路状态数据库同步、最短路径计算和路由表更新。不同之处在于邻居发现的方式和如何同步链路状态数据库。在点对点网络中，直接连接的两个路由器可以直接交换Hello消息；而在广播型网络中，路由器通过广播Hello消息来发现邻居。

## 6.3 数据包的发送过程

写出在你的拓扑中，数据包从某台PC机A发送给其他PC机B完整过程（这两台PC机间要跨越两台以上的路由器），包括A在网络内发送数据包的过程和跨越网络时路由器的处理过程，附相关截图。

在[6.1不同方式配置路由丢包解析](#_6.1_不同方式配置路由丢包解析)中，我们探讨了不同的路由配置方式导致的丢包数量，以及不同的路由方式数据包的收发过程和arp表的建立过程。但是我们只说明了PC到路由和路由到PC的过程。

而当数据包通过PC传到路由器之后，路由器会解析该数据包。我用图1-1的网络拓扑中PC5 ping PC20，结果和在PC5的f0/0接口的抓包分析如下：

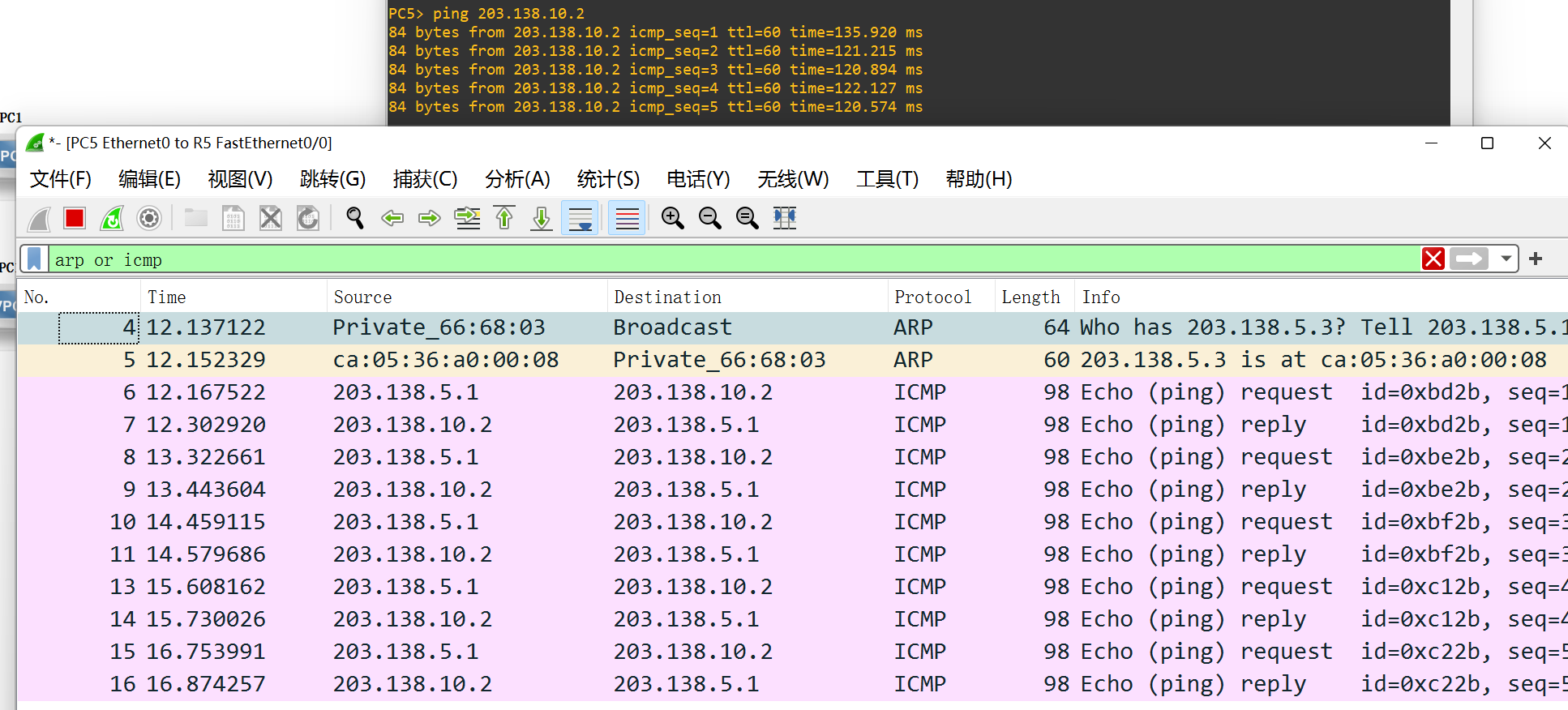


图6-14 PC5 ping PC20 分析

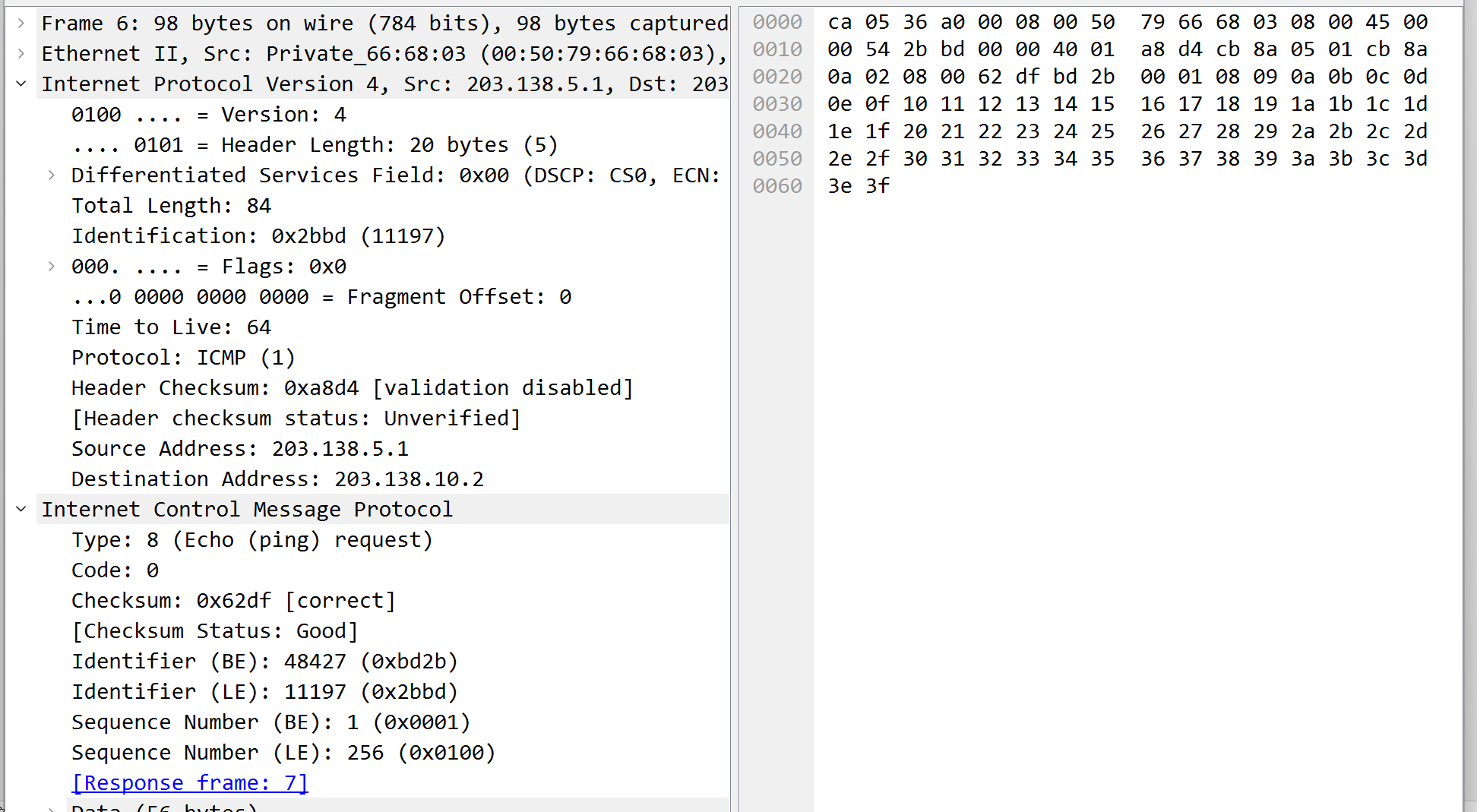


图6-15 IP包拆解图

（1）一个路由器在收到ICMP包时， 由于ICMP 报文通常是与 IP 数据报一起传输的，路由器的网络层负责将其从链路层中提取出来。

（2）检查报文类型和代码，路由器检查 ICMP 报文的类型和代码字段，以确定报文的目的和所包含的信息类型。

（3）校验 ICMP 报文的校验和，路由器会对 ICMP 报文进行校验和的验证，确保报文的完整性。

（4）查找路由表：路由器根据 ICMP 报文中的目的 IP 地址查找路由表，确定下一跳的接口。如果目的地址是路由器本身，路由器将 ICMP 报文传递给本地进程进行处理。

（5）转发 ICMP 报文，路由器根据路由表找到的下一跳信息，将 ICMP 报文转发到相应的出接口。

（6）生成 ICMP 响应，如果 ICMP 报文需要路由器生成响应（例如 Destination Unreachable 或 Time Exceeded），路由器会生成相应的 ICMP 响应报文。响应报文中包含了原始数据报文的一部分，以便通知源主机或路由器有关问题的详细信息。

（7）发送 ICMP 响应，生成的 ICMP 响应通过相应的出接口发送到下一跳或目的地。

（8）更新 TTL（Time to Live）或 Hop Limit，如果路由器是中间路由器，它会更新 TTL 或 Hop Limit 字段，并重新计算 ICMP 报文的校验和，然后将其传递到下一个路由器。

（9）当目的地址为本地时，路由器会将 ICMP 报文传递给本地协议栈，以便本地协议栈处理相应的 ICMP 消息。