

# IP组播总结

## 一、组播概述

### 1、IP组播介绍

#### →IP组播简介

组播就是源向多个接收者发送消息，其中，源可以是多个；

有效的节省了带宽；

减少主机和路由器的处理进程，减轻工作量；

接收者的地址未知；

实时性；

\*单播，广播和组播的区别：

- 1、单播发向一个特定的目标节点；
- 2、广播发向网络上的所有节点；
- 3、组播发向网络上的一组特定的用户（发送一份，复制多份，这克服了单播的不足；只有特定的接收者接收到消息，这克服了广播的缺点）

#### →常见的组播应用及共有特点

一对多：远程教育，企业新闻的视频和音频传送，基于网络的娱乐节目、新闻和股票更新等

多对多：视频会议，共享白板，多用户游戏等

多对一：比如我加入了多个组，需要接收多个组的消息

#### →组播优点

- 1、更好的利用带宽；
- 2、节约设备的处理资源；
- 3、实时性；
- 4、可以实现一对多的特性，适用于分布式的应用；
- 5、接收者未知（避免接收者受到攻击）？

#### →组播缺点

组播是基于UDP的，所以继承了UDP的缺点

UDP的缺点：

- 1、只能是很尽力而为的传输，不能保证可靠传输；
- 2、没有拥塞避免机制，不像TCP有滑动窗口机制；
- 3、会产生重分的报文；
- 4、无序性（因UDP没有序列号）→实时传输协议RTP来解决UDP的无序性

## <RTP>

实时传输协议，用来解决UDP的无序性

端口号：16384-32767

RTP-实时传输协议real-time

RCTP-实时控制协议

先封装RTP，再封装UDP

20byte	8byte	12byte	
IP	UDP	RTP	Vlice payload

## 2、组播地址

→链路本地地址（224.0.0.0-224.255.255.255）

\*几种常见的链路本地组播地址

224.0.0.1-----这是所有的所有路由器和主机需要监听的

224.0.0.2-----这是所有路由器需要监听的

224.0.0.4-----DVMRP路由器

224.0.0.5-----OSPF路由器

224.0.0.6-----OSPF中DR和BDR

224.0.0.9-----RIPv2

224.0.0.10-----IGRP、EIGRP路由器

224.0.0.12-----DHCP server/Relay Agent

224.0.0.13-----PIM路由器

224.0.0.18-----VRRP

224.0.0.22-----IGMP

224.0.1.39-----Cisco--RP宣告

224.0.1.40-----Cisco--RP发现

→全局地址范围(224.0.1.0--238.255.255.255)

\*保留地址

224.0.1.0-----224.0.1.255 ——分给特定的路由协议使用

239.0.0.0/8 ( 239.0.0.0--239.255.255.255 ) 有限范围, 这段地址不能在公网上路由

\*GLOP地址

ISP 233.0.0.0/8 ( 这段地址和AS号对应 )

例如: AS 62010的16进制表示为F23A, F2的十进制是242, 3A的十进制是58, 所以子网233.242.58.0/24被全局保留, 供AS 62010使用

\*SSM地址 ( SSM特定组播源协议 )

232.0.0.0/8, 全球唯一应用

→管理地址范围

\*前256个范围相对地址

→组播Mac地址

\*32:1的原因

组播地址的个数是 $2^{28}$ , 而保留前缀后的MAC地址数量是 $2^{23}$ , 那么 $2^{28}/2^{23}=32:1$

**组播MAC地址的前25位固定  
IP地址的最后23位被映射到MAC地址的最后23位**

**0000, 0001 0000, 0000 0101, 1110 0000, 0000 0000, 0000 0000, 1010**  
24位 0 23位

**01 00 5E 00 00 0A**

### 3、组播组管理协议

→IGMP的作用简介

不管组播网络中运行了多少种路由协议, 要求在主机和路由器之间运行IGMP, 所有希望加入到组播组的主机和所有接口连接的子网上存在多播路由器都必须运行IGMP, IGMP在IP头部协议号为2, 由于该消息仅限于在本地数据链路, 所以IP头部的TTL被设置为1, 这样路由器就不能转发IGMP消息。

主要作用: 组查询和组报告; 以及报告抑制

注意: IGMP版本一没有特定组query消息和leave group消息, 无查询路由器选举进程, 依赖于多播路由协议在子网中选举一个指定路由器, 由于不同的多播路由协议采取不同的选举机制, 因而在IGMPv1下, 同一个子网可能存在多个DR路由器。

### 4、组播转发机制

→组播转发路径与单播的差异

单播需要一对一的传输, 但是组播就是可以一对多的传输, 同时, 组播的转发方式是逆向路径转发RPF。

RPF规则: 收到组播流量的接口, 与沿单播最优路径查找组播源的出接口是一致的, 那么则为RPF接口。这一过程又称RPF检测。

单播: 先有路由表, 再转发;

组播: 先用流量推动形成组播表。

→S,D为止如何形成转发路径

SPT向下推; RPT向上拉

RPF, 确保无环和数据重份

→两种思想, 对应两种不同的树

推的思想: SPT---源树

拉的思想: RPT---共享树

→组播分发树及其特点

\*SPT

推, 树根为源, 较为耗路由器内存资源, 可形成最短路径, 可能存在潜在环路

\*RPT

拉, 树的根为RP ( 聚合点 ), 节省路由器资源, 可能存在次优路径

→RPF机制

\*RPF规则

收到组播流量的接口, 与沿着单播最优路径寻找组播源的出接口是一致的, 那么RPF检查成功。从RPF接口收到的流量才转发。

\*RPF防环实质

查找单播路由表到源的最佳路径, 单播无环, 那么组播就无环了

\* ( S,G ) 的RPF

根据S来做RPF

\* ( \*,G ) 的RPF

根据RP来做RPF

\*等假负载均衡RPF接收的选择

不能同时使用, 使用下一跳IP高的

RPF选接口的比较原则:

- 1、lower AD 同样的路由, 选最小AD值的路由所用的接口为RPF接口
- 2、longest match 同样的路由, 比最长掩码
- 3、lower metric 如果IGP是负载均衡, 同样的路由, 掩码一样长, 比metric
- 4、higher ip 以上都一样, 比接口IP地址

通过写多播静态路由来指定自己的信源接口

ip mroute 1.1.1.0 255.255.255.0 12.1.1.2 写去到源从哪个接口发出, 1.1.1.0是源, 12.1.1.2是下一跳  
show ip mroute static 查看静态多播路由, 注意多播静态路由的AD是0

RPF默认5S/次

## 5、组播路由协议概述

→组播路由协议分类

\*域内

组播协议: DVMRP, MOSPF, PIM

\*域间

需要使用域间协议, 例如MSDP, MBGP

→PIM-DM模式简介

\*推模型

→PIM-SM简介

\*拉模式

## 二、PIM-DM详解

### 1、邻居发现机制

通过PIMv2协议224.0.0.13周期30s的发送hello包给所有运行PIM协议的路由器, hold-time时间是105S

### 2、扩散过程(泛洪)

经过的每个路由器均会建立一个(S,G), 每3Min一次, 以便维护组播条目

### 3、剪枝/加入过程

假如我接收者退出了组播组, 那么这一条发送路径就会被修剪, 也是3Min一次, (泛洪过去, 修剪过来) 假如我有新的接收者假如, 假如加入, 向S发送加入信息

### 4、嫁接过程

嫁接这个过程是被修剪的树枝, 还存在于组播表中, 只是接口为prune状态, 在未超时的时候, 有新的接收者加入, 那么直接发送嫁接消息, 不需要等待。

### 5、宣称机制

自身收到相同组播流量, 就会触发宣称机制。比哪一接口到源更优, 先比AD, 小的优先, AD相同比Metric, 小的优。Metric相同再比较接口的IP地大的成为转发者, 差的接口停止转发, 向转发者发修剪。

### 6、PIM-DM状态刷新机制

产生这一机制的原因: 重复的扩散-剪枝行为将会浪费大量的带宽, 增加了路由器的负担(因为扩散-剪枝是周期进行的, 每个一段时间, 网络中就会充斥了组播报文, 而不需要接收组播报文的路由器就需要再次申请剪枝)。

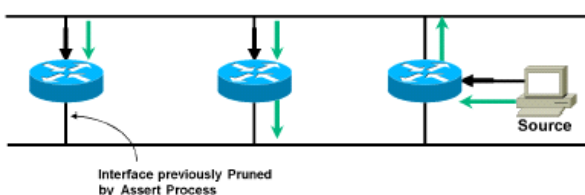
原理: 故PIM-DM有了state refresh机制, 和组播源直接连接的路由器发出state refresh消息, 其他的路由器收到这一消息后会重置剪枝超时计时器, 同时向入接口之外的所有连接PIM邻居的接口发送state refresh消息, 其中, 当前处于转发状态的端口, state refresh消息中的剪枝标识位置为0, 对于当前处于剪枝状态的接口, state refresh消息中的剪枝标识置为1。

好处: 通过周期的发送state refresh消息, 可以使得处于剪枝状态的接口一直维持这一状态。从而减少了不必要的扩散。

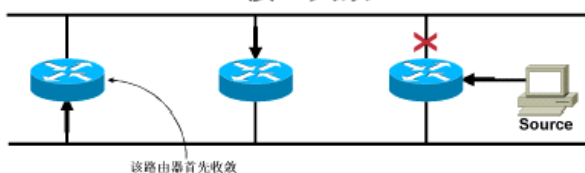
这一机制, 使用周期性的协议报文代替周期性的组播数据扩散, 可以减少网络消耗, 优化网络资源。

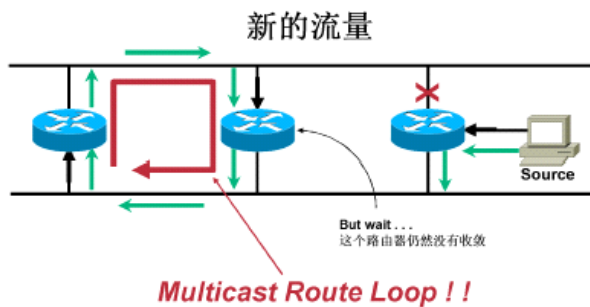
### 7、收敛时引发的报文重份和环路问题

#### 普通的恒定流量



#### 接口失效





↓ RPF Interface

### 三、组播组管理协议详解

#### 1、IGMPv1

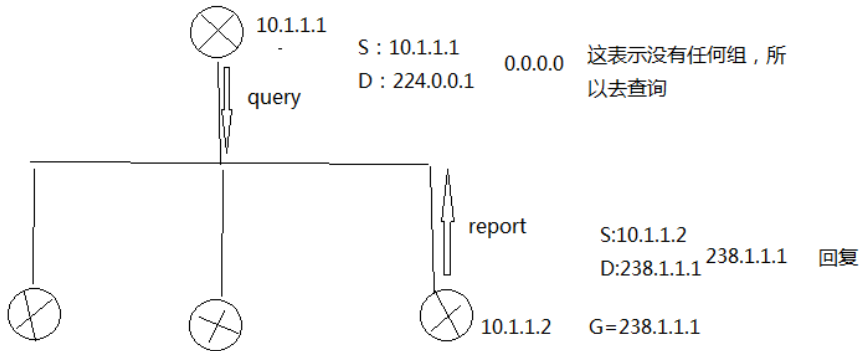
IGMP版本一是DR充当查询者（依赖于PIM选的），没有相应的查询机制选举查询者。

→报文格式



→组查询及报文封装

→成员报告及报文封装



假如有新的成员需要加入，直接发送成员report报文，不必等待查询者发送query，这样节省了主机加入某组播组的时间。

→报告抑制机制

当主机要发送成员报告时，首先会等0-10s的随机时间，假如收到其他发来的报告和自己的一致就取消自己的report（这样减少网段上重复报文的发送）。

→离开组的方式

IGMP版本一没有离开组消息，这就意味着从最后一台主机离开组播组到路由器停止转发组流量的时间周期将会更长。

#### 2、IGMPv2

→IGMPv2报文格式



成员查询：

```
40 151.367000 10.1.1.1 224.0.0.1 IGMP 60 V2 Membership Query, general
```

离开组：

```
55 210.020000 10.1.1.3 224.0.0.2 IGMP 60 V2 Leave Group 238.1.1.1
```

成员报告：

```
.962000 10.1.1.3 238.1.1.1 IGMP 60 V2 Membership Report / Join group 238.1.1.1
```

→相对IGMPv1，新增的关键特性

\*查询器选举过程

将报文的源IP和自己接口的IP比较，IP小的成为查询者

\*最大响应时间字段

最大延迟时间精确到0.1秒，这是对于接收者多的时候的优势，可以指定相应的报告抑制时间。

\*特定组的查询机制

用于确认该组播组在网段中是否还有成员存在，当查询者收到离开消息之后，就发送specific-query

\*离开组机制

向所有的组播路由器发送（目的224.0.0.2），报文中有要离开的组播组地址信息

→IGMPv3

\*仅需要了解增加了对源的过滤

进一步增强了主机的控制能力，并增强了成员查询报文和成员报告报文的功能。

\*增加了源过滤的支持；

\*定义了新的报文类型和格式；

\*report报文目的组播地址变为224.0.0.22

\*取消了成员报告抑制机制。

#### 四、二层组播帧交换

→IGMP snooping

\*为什么需要使用IGMP snooping

为了解决组播报文在二层被广播发送的问题（设备一定要支持ASIC芯片，因为检测IGMP消息就意味着必须检查每个IP包，靠软件来实现，则会严重影响交换机的性能）

\*工作机制

利用set IGMP enable即可在交换机上启用IP snooping，此后交换机软件将会检查IGMP消息，并知道多播路由器和组播成员所处的位置。

→CGMP

\*工作机制

将多播会话局限于组成员所在的交换端口。

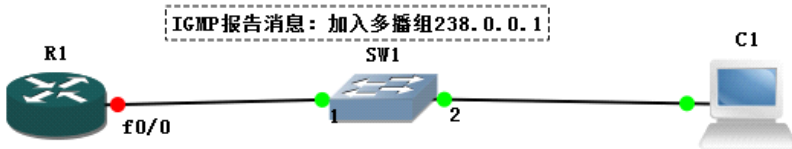
\*交换机和路由器都需要被配置为运行CGMP，但是只有路由器会产生CGMP包，交换机只是读取。

\*CGMP包类型：加入包，告知交换机将一个或者多个成员加入到组播组中。

离开包，告知交换机将一个或者多个成员从组播组中移除，或者删除同时删除组播组。

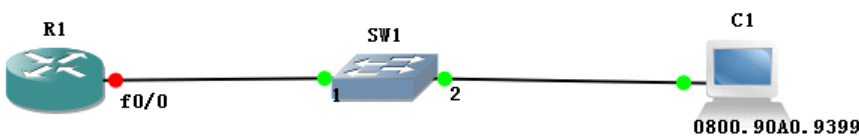
两种报文格式一样，目的地址始终是保留的Mac地址0100.0cdd.dddd，包中的基本信息就是一对或者多对Mac地址（GDA组目的地址和USA单播源地址）。

\*工作过程：当CGMP被激活时，会发送一个GDA被设置为0（0000.0000.0000），USA被设置为自身MAC地址CGMP加入包，（路由器每隔60秒发送一个这样的数据包，以保持激活状态）。



当路由器收到IGMP成员关系报告消息以后，会发送一条GDA被设置为组MAC，USA被设置为主机MAC地址的加入包

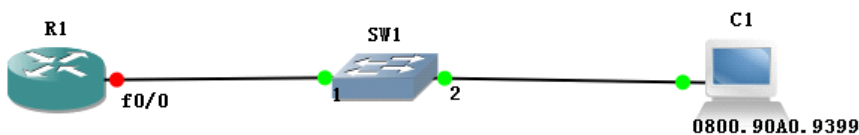
CGMP加入：  
 将GDA设置为 0100.5e10.1210  
 映射为USA: 0800.90A0.9399



交换机会将数据拷贝发送给与该组播组相关的所有端口（路由器端口除外）。只要交换网络中仍然有组成员，路由器就会每60S发送一次IGMP查询，并被交换机转发给组成员，之后由交换机将IGMP报告消息（查询消息的响应）转发给路由器。

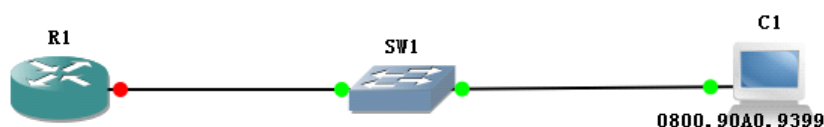
当路由器主机发送IGMPv2离开消息时，被消息转发给路由器，接着路由器会发送两条IGMP特定组查询消息，并且被交换机转发给所有组端口，

IGMP离开消息：离开组播组 238.0.0.1



如果其他的组成员响应了该特定组查询消息，那么路由器就会向交换机发送一个GDA被设置为MAC地址，USA被设置为即将离开的组成员MAC地址的CGMP离开包。（如果没有成员响应特定组查询消息，那路由器就断定该子网中没有组成员了，从而会向交换机发送一个GDA被设置为组MAC地址，USA被设置为0的CGMP离开包，以告知交换机从其CAM表中删除该组播组）。

CGMP离开：  
 GDA: 0100-5e10-1210  
 USA: 0800-90A0-9399



下面是CGMP包的类型和功能：

类型	GDA	USA	功能
----	-----	-----	----

join	0	路由器MAC	将该端口标识为所播路由器端口
join	组MAC	成员MAC	标识多播组并将成员的端口加入到组播组中
leave	组MAC	0	从CAM表中删除该组播组
leave	0	路由器MAC	从CAM表中删除所有组播组以及与路由器相关的端口-----触发条件：路由器关闭了CGMP功能
leave	0	0	从交换机中删除所有组播组和相关端口-----触发条件：clear IP CGMP

## 五、PIM-SM模式详解

### 1、PIM概述

→了解PIM协议工作思想和原理

\*RP的概念及作用

RP叫做聚合点，这是sparse mode中的核心，它的作用在于：接收者向其发送加入信息，源向其发送注册信息。

3种方式来获取RP：

1、静态的方式：IP PIM RP-address 1.1.1.1

2、BSR来收集RP信息，让C-RP自己选举

3、Auto-RP (Cisco) 向MA (RP Mapping Agent) 发送宣告信息，然后MA比较之后，得出RP再泛洪

\*接收者向RP逐跳加入

当接收者需要接收某组播消息的时候，就会发出加入消息

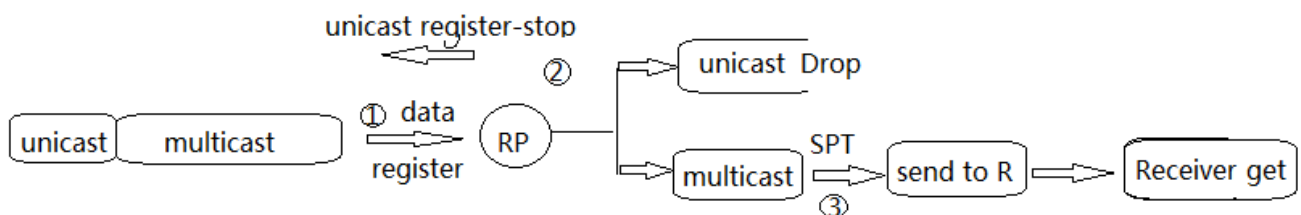
接收者首先通过IGMP report报文通知DR加入某组播组，那么DR会在本地建立一个\*组，然后向RPF邻居发送join消息。Join消息将沿着DR指向RP的单播路由逐跳转发，最后到达RP，沿途每一台路由器都会建立对应的\*组，包括RP在内。只要DR本地有组播组中的接收者，那么DR就会周期的向上游发送加入消息，保证上游的组播报文到达本地。假如没有接收者了，那么DR就会向上游发送剪切消息。

——由近RP的接收者的DR逐跳发送给RP，沿途所有路由器建立\*组，路径的反向形成RPT。

\*源向RP注册

近源的DR收到组播报文，封装在register消息中，单播空投给RP，RP中存在所有的\*组条目，如果有相应的组播组就顺着SPT转发，若不存在的话，RP会为源创建源组，并逆向逐跳加入至DR（近源），沿途各路由器都是产生源组条目了，路径形成SPT。

\*注册停止：



\*SPT切换

为了获得更小的延迟，接收者侧的DR会发起从RPT到SPT的切换。

当DR收到了第一个组播报文的时候，得到了源的IP，那么多久会向组播源方向发送特定源组的join信息，join消息将沿着DR指向组播源的单播路由逐跳转发，最后到达组播源侧的DR或者已经存在源组表项的路由器，沿途中的每一个路由器都会建立一个源组表项。

First Packet ⇌ Send join ⇌ be SPT ⇌ send the second packet ⇌ get come to SPT ⇌ TO be SPT

和PIM-Dm相比，SM中的SPT形成结余接收者的主动加入，所以不用泛洪报文到全网，这样就节省了大量的带宽，和RPT相比，组播报文经过最优路径到达接收者，不需要RP的中转，提高了报文的转发效率。

Drop come to RPT&Send prune to

### 2、邻居发现和RP的选举

→PIM邻居建立过程及使用地址

PIM路由器周期性的以组播的方式发送hello消息，目的组播地址是224.0.0.13，所有的PIM路由器都是该组播组的成员，通过hello消息，路由器之间可以发现PIM邻居，并建立和维护邻居关系，（PIMv2的hello和PIM v1的query都是30 S一次；超时时间是105 S）。

→DR选举规则及其作用

在连接组播接收者的共享网段上有多台路由器，且路由器都运行了IGMPv1的时候，路由器可以通过PIM hello消息为该网段的接收者主机选举DR，DR充当IGMPv1的查询者器，PIM-DM只有此时才选举DR，选举DR，优先级大的成为DR，优先级一致，那么路由器接口的IP大的成为DR。

当DR出现故障，其余路由器在超时后还没有收到DR发送的hello消息，那么就会触发新的DR选举过程。

### 3、PIM加入过程

→join消息触发条件

路由器从RPF接口收到prune消息，且路由器的下游存在接收者，那么路由器就会向RPF邻居发送join消息，使得上游RPF邻居继续维持组播报文在该共享网段继续的转发。

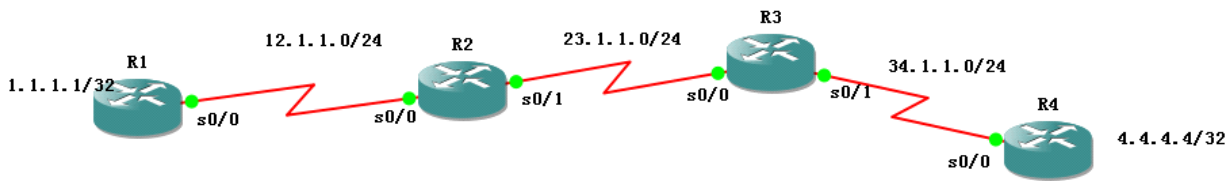
→加入过程中（\*, G）路由的形成过程

接收者通过IGMP report报文通知DR加入某组播组，DR会在本地建立\*组，然后向RPF邻居发送join消息，此时DR会将连接接收者的接口加入到\*组的outgoing口中，同时将指向RP的单播路由的下一跳出接口作为\*组的incoming接口，由于接收者没有指定组播源，故这里的\*组表项中的组播源是任意组播源。

join消息会沿着DR指向RP的单播路由逐跳转发，最后到达RP，沿途中的每一个路由器都会建立一个\*组，网络中的每一个接收者的DR都是这么做，最终形成以RP为根的RPT。

#### 4、组播源注册

默认基本配置完成



→接收者先出现

→源先出现

1、在R4上加入组238.1.1.1：

R4#

```
*Mar 1 00:11:56.439: IGMP(0): WAVL Insert group: 238.1.1.1 interface: Loopback0Successful
*Mar 1 00:11:56.439: IGMP(0): Send v2 Report for 238.1.1.1 on Loopback0
*Mar 1 00:11:56.439: IGMP(0): Received v2 Report on Loopback0 from 4.4.4.4 for 238.1.1.1
*Mar 1 00:11:56.439: IGMP(0): Received Group record for group 238.1.1.1, mode 2 from 4.4.4.4 for 0 sources
*Mar 1 00:11:56.443: IGMP(0): Switching to EXCLUDE mode for 238.1.1.1 on Loopback0
*Mar 1 00:11:56.443: IGMP(0): Updating EXCLUDE group timer for 238.1.1.1
*Mar 1 00:11:56.443: IGMP(0): MRT Add/Update Loopback0 for (*,238.1.1.1) by 0
*Mar 1 00:11:56.447: PIM(0): Building Triggered Join/Prune message for 238.1.1.1
*Mar 1 00:11:56.447: PIM(0): Insert (*,238.1.1.1) join in nbr 34.1.1.3's queue
*Mar 1 00:11:56.451: IGMP(0): MRT Add/Update Loopback0 for (*,238.1.1.1) by 4
*Mar 1 00:11:56.455: PIM(0): Building Join/Prune packet for nbr 34.1.1.3
*Mar 1 00:11:56.455: PIM(0): Adding v2 (3.3.3.3/32, 238.1.1.1), WC-bit, RPT-bit, S-bit Join
*Mar 1 00:11:56.459: PIM(0): Send v2 join/prune to 34.1.1.3 (Serial0/0)
*Mar 1 00:11:56.463: IGMP(0): Received v2 Report on Loopback0 from 4.4.4.4 for 238.1.1.1
*Mar 1 00:11:56.463: IGMP(0): Received Group record for group 238.1.1.1, mode 2 from 4.4.4.4 for 0 sources
*Mar 1 00:11:56.463: IGMP(0): Updating EXCLUDE group timer for 238.1.1.1
*Mar 1 00:11:56.463: IGMP(0): MRT Add/Update Loopback0 for (*,238.1.1.1) by 0
*Mar 1 00:11:56.955: %SYS-5-CONFIG_I: Configured from console by console
```

在R3上看收到的R4的消息：

R3#

```
*Mar 1 00:11:49.415: PIM(0): Received v2 Join/Prune on Serial0/1 from 34.1.1.4, to us
*Mar 1 00:11:49.415: PIM(0): Join-list: (*, 224.0.1.40), RPT-bit set, WC-bit set, S-bit set
*Mar 1 00:11:49.419: PIM(0): Update Serial0/1/34.1.1.4 to (*, 224.0.1.40), Forward state, by PIM *G Join
```

R3#

```
*Mar 1 00:11:57.771: PIM(0): Received v2 Join/Prune on Serial0/1 from 34.1.1.4, to us
*Mar 1 00:11:57.771: PIM(0): Join-list: (*, 238.1.1.1), RPT-bit set, WC-bit set, S-bit set
*Mar 1 00:11:57.775: PIM(0): Add Serial0/1/34.1.1.4 to (*, 238.1.1.1), Forward state, by PIM *G Join
```

R3#

```
*Mar 1 00:12:02.803: PIM(0): Building Periodic Join/Prune message for 238.1.1.1
*Mar 1 00:12:02.899: IGMP(0): Send v2 general Query on Loopback0
```

**分析情况1：接收者先出现的情况，源后出现。**接收者会给RP逐跳发送加入信息，在RP上完成加入。而源后出现，当源发送组播数据时，到第一跳路由器的时候没有接收者，所以其Outgoing口是空的，此时要将数据封装在注册信息上，然后再在外面封装一层单播SIP和DIP，此时应该从Outgoing口发送注册（假如注册238.1.1.1这一组播组：ping 238.1.1.1 repeat 1，之后show ip mroute就可以看见相应的注册标志。或者出接口写成loopback 0）

R1#ping

Protocol [ip]:

Target IP address: 238.1.1.1

Repeat count [1]:

Datagram size [100]:

Timeout in seconds [2]:

Extended commands [n]: y

Interface [All]: lo 0

Time to live [255]:

Source address:

Type of service [0]:

Set DF bit in IP header? [no]:  
Validate reply data? [no]:  
Data pattern [0xABCD]:  
Loose, Strict, Record, Timestamp, Verbose[none]:  
Sweep range of sizes [n]:  
Type escape sequence to abort.  
Sending 1, 100-byte ICMP Echos to 238.1.1.1, timeout is 2 seconds:

Reply to request 0 from 34.1.1.4, 168 ms

而到达RP后就会取出组播数据，沿着RPT转发给接收者。同时会向源发送加入（S，G）信息，最后在第一跳路由器上形成一个源组信息，并且相应的出现一个\*组条目，之后RP会收到的是两份组播数据，一份是从单播注册的路径来的，一份是沿SPT发来的，但是RP会默认丢弃沿注册路径来的，会接收沿SPT来的组播数据，再沿RPT转发下去，之后就会沿着这一路径完成组播数据的传输。第二个包发来才会发送注册停止信息。

R3#

```
*Mar 1 01:22:04.275: PIM(0): Received v2 Register on Serial0/0 from 12.1.1.1
*Mar 1 01:22:04.275:   for 1.1.1.1, group 238.1.1.1
*Mar 1 01:22:04.279: PIM(0): Insert (1.1.1.1,238.1.1.1) join in nbr 23.1.1.2's queue
*Mar 1 01:22:04.279: PIM(0): Forward decapsulated data packet for 238.1.1.1 on Serial0/1
*Mar 1 01:22:04.279: PIM(0): Building Join/Prune packet for nbr 23.1.1.2
*Mar 1 01:22:04.283: PIM(0): Adding v2 (1.1.1.1/32, 238.1.1.1), S-bit Join
*Mar 1 01:22:04.283: PIM(0): Send v2 join/prune to 23.1.1.2 (Serial0/0)
*Mar 1 01:22:04.379: PIM(0): Received v2 Join/Prune on Serial0/1 from 34.1.1.4, to us
```

R2#

```
*Mar 1 01:22:46.527: PIM(0): Received v2 Join/Prune on Serial0/1 from 23.1.1.3, to us
*Mar 1 01:22:46.527: PIM(0): Join-list: (1.1.1.1/32, 238.1.1.1), S-bit set
*Mar 1 01:22:46.527: PIM(0): Update Serial0/1/23.1.1.3 to (1.1.1.1, 238.1.1.1), Forward state, by PIM SG Join
```

R1#sho ip mroute

IP Multicast Routing Table

Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C - Connected,  
L - Local, P - Pruned, R - RP-bit set, F - Register flag,  
T - SPT-bit set, J - Join SPT, M - MSDP created entry,  
X - Proxy Join Timer Running, A - Candidate for MSDP Advertisement,  
U - URD, I - Received Source Specific Host Report, Z - Multicast Tunnel  
Y - Joined MDT-data group, y - Sending to MDT-data group

Outgoing interface flags: H - Hardware switched, A - Assert winner

Timers: Uptime/Expires

Interface state: Interface, Next-Hop or VCD, State/Mode

(\*, 238.1.1.1), 00:00:07/stopped, RP 3.3.3.3, flags: SPF

Incoming interface: Serial0/0, RPF nbr 12.1.1.2

Outgoing interface list: Null

(1.1.1.1, 238.1.1.1), 00:00:07/00:02:52, flags: FT

Incoming interface: Loopback0, RPF nbr 0.0.0.0, **Registering**

Outgoing interface list:

Serial0/0, Forward/Sparse, 00:00:07/00:03:22

(\*, 224.0.1.40), 00:02:34/00:02:58, RP 3.3.3.3, flags: SJCL

Incoming interface: Serial0/0, RPF nbr 12.1.1.2

Outgoing interface list:

Loopback0, Forward/Sparse, 00:02:36/00:02:56

这样等待一段时间（这段时间源会不断的给RP发送空的注册消息，提醒RP发送注册停止消息），注册消息就会停止。

**分析情况2：接收者后出现，源先出现。**源发送给第一跳路由器，这个路由器就发送注册信息（此时Outgoing口是空的），到RP上的时候，RP会创建一个\*组，其Outgoing口也是空的，RP会回复注册停止的消息，之后每间隔三分钟就会发送注册信息，以维护相应的源组信息（维护组播条目），以防接收者的突然出现，如果没有接收者的话，就会重复的每隔三分钟去注册。（问题1）之后接收者出现之后，就会沿着接收者加入的反向路径RPT转发下去，并且向源发送（S,G）加入消息，形成SPT，之后的组播流量就这样形成了。

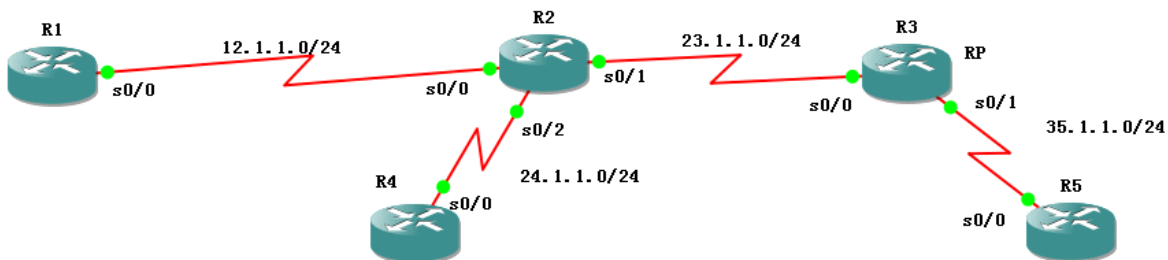


\*配置R1(config)#ip pim spt-threshold ? (建议配置为0, 或者最大, 中间的不建议配置)  
 <0-4294967> Traffic rate in kilobits per second  
 infinity Never switch to source-tree  
 配置成infinity时, 是不会切换成SPT的, 这样就会一直维持RPT树的转发树。(问题2)

R3#

```
*Mar 1 01:57:18.991: PIM(0): Received v2 Join/Prune on Serial0/1 from 34.1.1.4, to us
*Mar 1 01:57:18.991: PIM(0): Prune-list: (*, 238.1.1.1) RP 3.3.3.3
*Mar 1 01:57:18.991: PIM(0): Prune Serial0/1/224.0.0.2 from (*, 238.1.1.1) - deleted
R3#
*Mar 1 01:58:26.015: PIM(0): Received v2 Register on Serial0/0 from 12.1.1.1
*Mar 1 01:58:26.015: for 1.1.1.1, group 238.1.1.1
*Mar 1 01:58:26.019: PIM(0): Check RP 3.3.3.3 into the (*, 238.1.1.1) entry
*Mar 1 01:58:26.019: PIM(0): Send v2 Register-Stop to 12.1.1.1 for 1.1.1.1, group 238.1.1.1
```

分析情况3: 接收者在源和RP之间。



3.1、如果转发路径已经形成的话, 由于RP的Incoming接口不会是Outgoing口, 所以接收者的加入到RP就结束了, 不会给接收者copy Outgoing口, 而之后的组播数据就会直接给中间的路由器, 转发给接收者。

R3#sho ip mroute

IP Multicast Routing Table

Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C - Connected,  
 L - Local, P - Pruned, R - RP-bit set, F - Register flag,  
 T - SPT-bit set, J - Join SPT, M - MSDP created entry,  
 X - Proxy Join Timer Running, A - Candidate for MSDP Advertisement,  
 U - URD, I - Received Source Specific Host Report, Z - Multicast Tunnel  
 Y - Joined MDT-data group, y - Sending to MDT-data group

Outgoing interface flags: H - Hardware switched, A - Assert winner

Timers: Uptime/Expires

Interface state: Interface, Next-Hop or VCD, State/Mode

(\*, 238.1.1.1), 00:02:28/00:03:07, RP 3.3.3.3, flags: S

Incoming interface: Null, RPF nbr 0.0.0.0

Outgoing interface list:

Serial0/0, Forward/Sparse, 00:01:21/00:03:07

Serial0/1, Forward/Sparse, 00:02:28/00:02:59

(1.1.1.1, 238.1.1.1), 00:02:18/00:02:42, flags: T

Incoming interface: Serial0/0, RPF nbr 23.1.1.2

Outgoing interface list:

**Serial0/1, Forward/Sparse, 00:02:18/00:02:59**

(\*, 224.0.1.40), 00:02:43/00:02:57, RP 3.3.3.3, flags: SJCL

Incoming interface: Null, RPF nbr 0.0.0.0

Outgoing interface list:

Serial0/1, Forward/Sparse, 00:02:30/00:02:57

Serial0/0, Forward/Sparse, 00:02:43/00:02:40

这是之前形成的到R3到R5的Outgoing口, 当R4的Lo0加入组的时候, 我们看下面的debug信息:

R2#

```
*Mar 1 00:11:07.971: PIM(0): Received v2 Join/Prune on Serial0/2 from 24.1.1.4, to us
```

```
*Mar 1 00:11:07.971: PIM(0): Join-list: (*, 238.1.1.1), RPT-bit set, WC-bit set, S-bit set
```

```
*Mar 1 00:11:07.975: PIM(0): Add Serial0/2/24.1.1.4 to (*, 238.1.1.1), Forward state, by PIM *G Join
*Mar 1 00:11:07.975: PIM(0): Building Triggered Join/Prune message for 238.1.1.1
*Mar 1 00:11:07.975: PIM(0): Insert (*,238.1.1.1) join in nbr 23.1.1.3's queue
*Mar 1 00:11:07.975: PIM(0): Insert (1.1.1.1,238.1.1.1) sgr prune in nbr 23.1.1.3's queue
*Mar 1 00:11:07.979: PIM(0): Insert (1.1.1.1,238.1.1.1) join in nbr 12.1.1.1's queue
```

R2#

```
*Mar 1 00:11:07.979: PIM(0): Add Serial0/2/24.1.1.4 to (1.1.1.1, 238.1.1.1), Forward state, by PIM *G Join
*Mar 1 00:11:07.983: PIM(0): Building Join/Prune packet for nbr 12.1.1.1
*Mar 1 00:11:07.983: PIM(0): Adding v2 (1.1.1.1/32, 238.1.1.1), S-bit Join
*Mar 1 00:11:07.983: PIM(0): Send v2 join/prune to 12.1.1.1 (Serial0/0)——这给R1发的加入信息
*Mar 1 00:11:07.987: PIM(0): Building Join/Prune packet for nbr 23.1.1.3
*Mar 1 00:11:07.987: PIM(0): Adding v2 (3.3.3.3/32, 238.1.1.1), WC-bit, RPT-bit, S-bit Join
*Mar 1 00:11:07.991: PIM(0): Adding v2 (1.1.1.1/32, 238.1.1.1), RPT-bit, S-bit Prune
*Mar 1 00:11:07.995: PIM(0): Send v2 join/prune to 23.1.1.3 (Serial0/1)——给R3发的加入信息
```

但是R3是不会将S0/0作为Outgoing口的，因为这是之前的Incoming接口，可以看R3上的debug信息：

R3#

```
*Mar 1 00:11:06.651: PIM(0): Received v2 Join/Prune on Serial0/0 from 23.1.1.2, to us
*Mar 1 00:11:06.651: PIM(0): Join-list: (*, 238.1.1.1), RPT-bit set, WC-bit set, S-bit set
*Mar 1 00:11:06.655: PIM(0): Add Serial0/0/23.1.1.2 to (*, 238.1.1.1), Forward state, by PIM *G Join
*Mar 1 00:11:06.655: PIM(0): Prune-list: (1.1.1.1/32, 238.1.1.1) RPT-bit set
```

R3收到了但是没有加入Outgoing口，因为这加入Outgoing口的话本来就是不对的，可以看R1是也可以收到这给加入信息的，那么之后的到R4的转发路径就是R1-R2-R4了

R1#

```
*Mar 1 00:11:09.415: PIM(0): Received v2 Join/Prune on Serial0/0 from 12.1.1.2, to us
*Mar 1 00:11:09.415: PIM(0): Join-list: (1.1.1.1/32, 238.1.1.1), S-bit set
*Mar 1 00:11:09.415: PIM(0): Update Serial0/0/12.1.1.2 to (1.1.1.1, 238.1.1.1), Forward state, by PIM SG Join
*Mar 1 00:11:11.827: PIM(0): Building Periodic Join/Prune message for 238.1.1.1
```

3.2、如果转发路径还没有形成的话，此时假如说在R5上没有接收者，那R1 会每三分钟去给R3（RP）发送注册，以维持源组条目的存在。当在这一个转发路径没有形成的情况下，我们看一下它的情形是怎么样的：

R4(config-if)#ip igmp join-group 238.1.1.1

```
*Mar 1 00:40:59.539: PIM(0): Building Triggered ( 触发 ) Join/Prune message for 238.1.1.1
*Mar 1 00:40:59.539: PIM(0): ( 插入 ) Insert (*,238.1.1.1) join in nbr 24.1.1.2's queue
*Mar 1 00:40:59.543: PIM(0): Building Join/Prune packet for nbr 24.1.1.2
*Mar 1 00:40:59.547: PIM(0): Adding v2 (3.3.3.3/32, 238.1.1.1), WC-bit, RPT-bit, S-bit Join
*Mar 1 00:40:59.547: PIM(0): Send v2 join/prune to 24.1.1.2 (Serial0/0)
*Mar 1 00:41:00.695: PIM(0): Insert (1.1.1.1,238.1.1.1) join in nbr 24.1.1.2's queue
*Mar 1 00:41:00.699: PIM(0): Building Join/Prune packet for nbr 24.1.1.2
*Mar 1 00:41:00.699: PIM(0): Adding v2 (1.1.1.1/32, 238.1.1.1), S-bit Join
*Mar 1 00:41:00.699: PIM(0): Send v2 join/prune to 24.1.1.2 (Serial0/0)
```

R1#ping

Protocol [ip]:

Target IP address: 238.1.1.1

Repeat count [1]: 100

Datagram size [100]:

Timeout in seconds [2]:

Extended commands [n]: y

Interface [All]: lo 0

Time to live [255]:

Source address:

Type of service [0]:

Set DF bit in IP header? [no]:

Validate reply data? [no]:

Data pattern [0xABCD]:

Loose, Strict, Record, Timestamp, Verbose[none]:

Sweep range of sizes [n]:

Type escape sequence to abort.

Sending 100, 100-byte ICMP Echos to 238.1.1.1, timeout is 2 seconds:

```
*Mar 1 00:40:54.747: PIM(0): Check RP 3.3.3.3 into the (*, 238.1.1.1) entry
*Mar 1 00:40:54.747: PIM(0): Send v2 Register to 3.3.3.3 for 1.1.1.1, group 238.1.1.1
*Mar 1 00:40:54.871: PIM(0): Received v2 Register-Stop on Serial0/0 from 3.3.3.3
*Mar 1 00:40:54.871: PIM(0): for source 1.1.1.1, group 238.1.1.1
*Mar 1 00:40:54.871: PIM(0): Clear Registering flag to 3.3.3.3 for (1.1.1.1/32, 238.1.1.1).....
*Mar 1 00:41:03.807: PIM(0): Received v2 Join/Prune on Serial0/0 from 12.1.1.2, to us
*Mar 1 00:41:03.807: PIM(0): Join-list: (1.1.1.1/32, 238.1.1.1), S-bit set
*Mar 1 00:41:03.811: PIM(0): Add Serial0/0/12.1.1.2 to (1.1.1.1, 238.1.1.1), Forward state, by PIM SG Join
Reply to request 5 from 24.1.1.4, 156 ms
Reply to request 6 from 24.1.1.4, 76 ms
Reply to request 7 from 24.1.1.4, 56 ms
```

R2#

```
*Mar 1 00:41:02.303: PIM(0): Received v2 Join/Prune on Serial0/2 from 24.1.1.4, to us
*Mar 1 00:41:02.303: PIM(0): Join-list: (*, 238.1.1.1), RPT-bit set, WC-bit set, S-bit set
*Mar 1 00:41:02.307: PIM(0): Check RP 3.3.3.3 into the (*, 238.1.1.1) entry
*Mar 1 00:41:02.307: PIM(0): Add Serial0/2/24.1.1.4 to (*, 238.1.1.1), Forward state, by PIM *G Join
*Mar 1 00:41:02.307: PIM(0): Building Triggered Join/Prune message for 238.1.1.1
*Mar 1 00:41:02.311: PIM(0): Insert (*,238.1.1.1) join in nbr 23.1.1.3's queue
*Mar 1 00:41:02.311: PIM(0): Building Join/Prune packet for nbr 23.1.1.3
*Mar 1 00:41:02.311: PIM(0): Adding v2 (3.3.3.3/32, 238.1.1.1), WC-bit, RPT-bit, S-bit
```

R2# Join

```
*Mar 1 00:41:02.315: PIM(0): Send v2 join/prune to 23.1.1.3 (Serial0/1)
*Mar 1 00:41:02.399: PIM(0): Received v2 Join/Prune on Serial0/1 from 23.1.1.3, to us
*Mar 1 00:41:02.399: PIM(0): Join-list: (1.1.1.1/32, 238.1.1.1), S-bit set
*Mar 1 00:41:02.403: PIM(0): Add Serial0/1/23.1.1.3 to (1.1.1.1, 238.1.1.1), Forward state, by PIM SG Join
*Mar 1 00:41:02.403: PIM(0): Insert (1.1.1.1,238.1.1.1) join in nbr 12.1.1.1's queue
*Mar 1 00:41:02.403: PIM(0): Building Join/Prune packet for nbr 12.1.1.1
*Mar 1 00:41:02.407: PIM(0): Adding v2 (1.1.1.1/32, 238.1.1.1), S-bit Join
*Mar 1 00:41:02.407: PIM(0): Send v2 join/prune to 12.1.1.1 (Serial0/0)
```

R2#

```
*Mar 1 00:41:03.395: PIM(0): Insert (1.1.1.1,238.1.1.1) sgr prune in nbr 23.1.1.3's queue
*Mar 1 00:41:03.399: PIM(0): Building Join/Prune packet for nbr 23.1.1.3
*Mar 1 00:41:03.399: PIM(0): Adding v2 (1.1.1.1/32, 238.1.1.1), RPT-bit, S-bit Prune
*Mar 1 00:41:03.399: PIM(0): Send v2 join/prune to 23.1.1.3 (Serial0/1)
*Mar 1 00:41:03.491: PIM(0): Received v2 Join/Prune on Serial0/2 from 24.1.1.4, to us
*Mar 1 00:41:03.491: PIM(0): Join-list: (1.1.1.1/32, 238.1.1.1), S-bit set
*Mar 1 00:41:03.491: PIM(0): Update Serial0/2/24.1.1.4 to (1.1.1.1, 238.1.1.1), Forward state, by PIM SG Join
```

R2#

```
*Mar 1 00:41:08.015: PIM(0): Received v2 Join/Prune on Serial0/1 from 23.1.1.3, to us
*Mar 1 00:41:08.015: PIM(0): Prune-list: (1.1.1.1/32, 238.1.1.1)
*Mar 1 00:41:08.015: PIM(0): Prune Serial0/1/224.0.0.2 from (1.1.1.1/32, 238.1.1.1) - deleted
```

R3#

```
*Mar 1 00:40:52.023: PIM(0): Received v2 Register on Serial0/0 from 12.1.1.1
*Mar 1 00:40:52.023: for 1.1.1.1, group 238.1.1.1
*Mar 1 00:40:52.027: PIM(0): Check RP 3.3.3.3 into the (*, 238.1.1.1) entry
*Mar 1 00:40:52.027: PIM(0): Send v2 Register-Stop to 12.1.1.1 for 1.1.1.1, group 238.1.1.1
```

R3#

```
*Mar 1 00:41:00.975: PIM(0): Received v2 Join/Prune on Serial0/0 from 23.1.1.2, to us
*Mar 1 00:41:00.975: PIM(0): Join-list: (*, 238.1.1.1), RPT-bit set, WC-bit set, S-bit set
*Mar 1 00:41:00.979: PIM(0): Add Serial0/0/23.1.1.2 to (*, 238.1.1.1), Forward state, by PIM *G Join
*Mar 1 00:41:00.979: PIM(0): Insert (1.1.1.1,238.1.1.1) join in nbr 23.1.1.2's queue
*Mar 1 00:41:00.979: PIM(0): Building Join/Prune packet for nbr 23.1.1.2
*Mar 1 00:41:00.983: PIM(0): Adding v2 (1.1.1.1/32, 238.1.1.1), S-bit Join
```

\*Mar 1 00:41:00.983: PIM(0): Send v2 join/prune to 23.1.1.2 (Serial0/0)

R3#

\*Mar 1 00:41:02.055: PIM(0): Received v2 Join/Prune on Serial0/0 from 23.1.1.2, to us

\*Mar 1 00:41:02.055: PIM(0): Prune-list: (1.1.1.1/32, 238.1.1.1) RPT-bit set

R3#

\*Mar 1 00:41:06.595: PIM(0): Insert (1.1.1.1,238.1.1.1) prune in nbr 23.1.1.2's queue

\*Mar 1 00:41:06.595: PIM(0): Building Join/Prune packet for nbr 23.1.1.2

\*Mar 1 00:41:06.595: PIM(0): Adding v2 (1.1.1.1/32, 238.1.1.1), S-bit Prune

\*Mar 1 00:41:06.599: PIM(0): Send v2 join/prune to 23.1.1.2 (Serial0/0)

R4的Lo0会给R4发送IGMP加入，R4会给R2发送加入，同时R2会给R1和R3加入，但是是直接给R1的，由R1直接到R4形成SPT，不会走R3的。

→接收者沿着SPT

\*重点要掌握注册过程中组播分发树是如何形成的

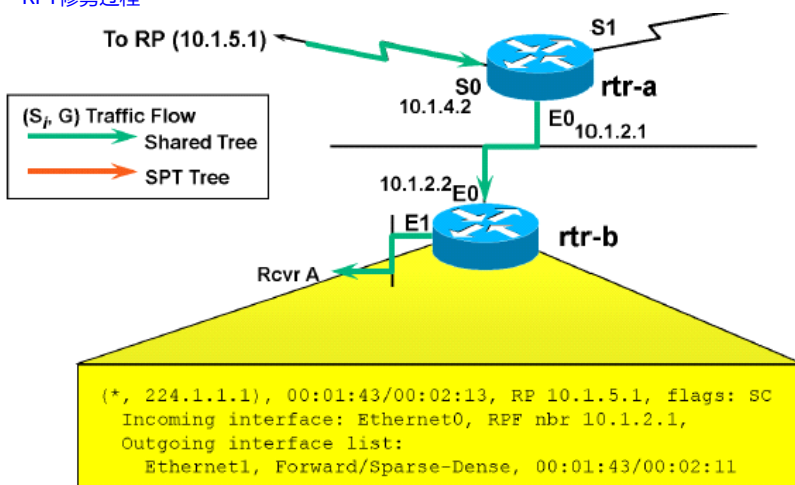
接收者收到第一个组播包的时候，知道了源的地址，于是，向源加入形成SPT

## 5、RPT向SPT的切换

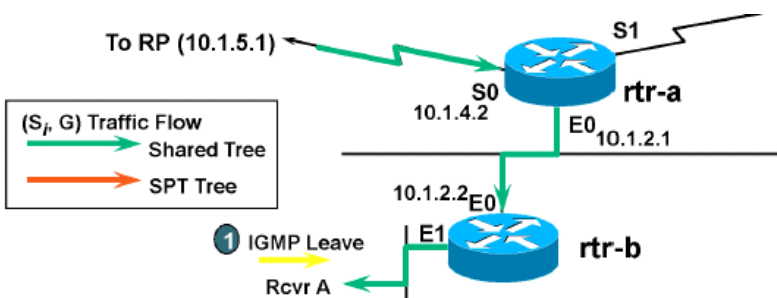
→触发条件

收到一个 (S,G) 信息

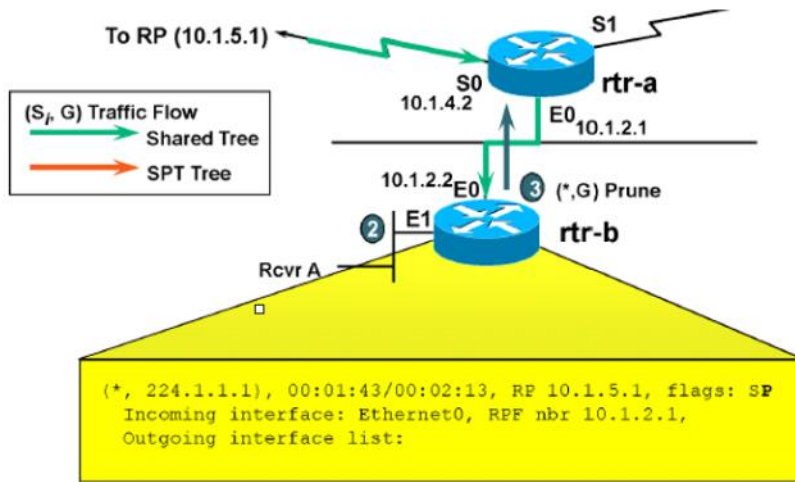
→RPT修剪过程



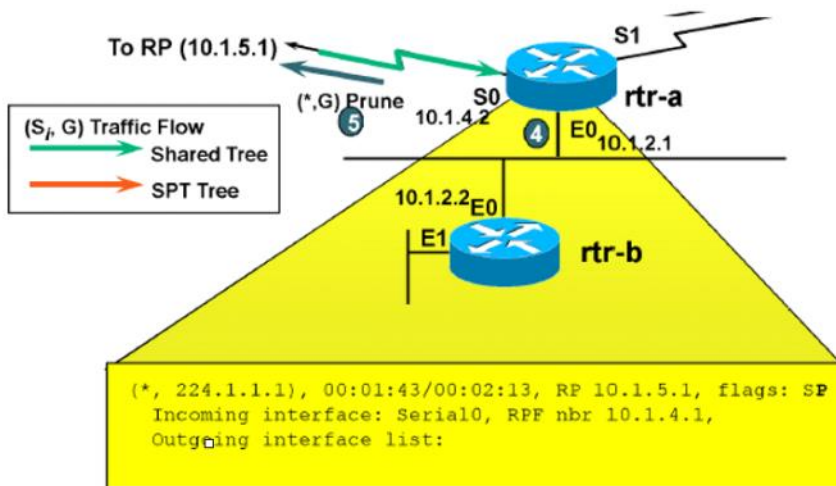
“rtr-b” 修剪前状态



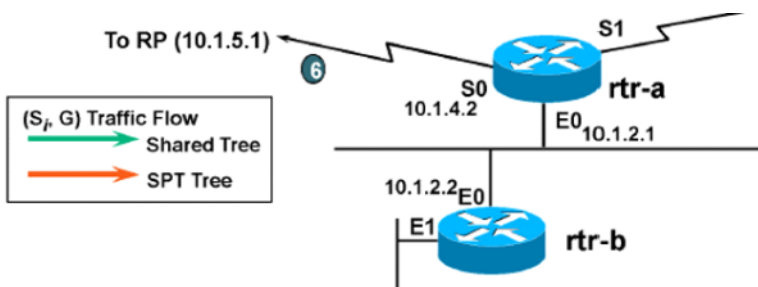
① “rtr-b” 是1台叶子路由器。最后“接收者 A”，离开组 G。



- ② “rtr-b” 从 (\*,G)和全部(Si,G) “oilists”.条目中移除E1接口
- ③ “rtr-b” (\*,G) “oilist” 现在为空; 触发 向RP方向发送(\*,G) 修剪



- ④ “rtr-a” 收到修剪; 从 (\*,G) “oilist”中移除E0.  
(在多路访问网络中, 有3秒的修剪延迟.)
- ⑤ “rtr-a” (\*,G) “oilist” 现为空; 触发向RP方向发送 (\*,G) 修剪.



⑥ 修剪向RP方向持续进行

6. 其他

→ (\*, G) 状态规则

- **(\*,G)** 建立
  - 接收1个(\*,G)加入信息, 或者IGMP Report
  - 收到组播源的信息, 必需建立(S,G)条目
- **(\*,G)** 指明组的默认转发
  - IIF = 循RP方向的RPF接口
  - OIL = 以下接口
    - 接收(\*,G)加入信息, 或
    - 有直接连接的成员, 或
    - 手动配置
- **(\*,G)** 删除
  - 当 OIL = 空, 以及
  - 没有子(S,G)状态存在

#### → (S,G) 状态规则

- **(S,G)** 建立
  - 接收(S,G)加入信息, 或修剪信息, 或者
  - 在“注册”过程中(连接源的第1跳路由器触发)
  - 建立对应(\*,G)(如果不存在)
- **(S,G)** 指明如何转发“S”到“G”
  - IIF = 循着源的RPF接口
    - 如果设置“RP-bit”, 则是循着RP方向
  - OIL = 刚开始的时候, 拷贝(\*,G) OIL条目(除去IIF)
- **(S,G)** 删除
  - 一般是(S,G)条目超时

#### → OIL规则

- **OIL** 添加接口
  - 通过接收加入信息
    - 添加进(\*,G)的接口, 也被添加到所有(S,G)条目
- **OIL** 接口移除
  - 通过接收修剪信息
    - 从(\*,G)移除的接口, 也从全部(S,G)条目底下移除
  - 接口超时计时器, 倒计时到0
    - 通过接收周期性的加入信息重设计时器(3分钟.)
    - 或
    - 通过IGMP成员报告

#### → 触发join/prune规则

- 触发 **Join/Prune** 信息
  - **(\*,G)** 加入信息触发时机:
    - (\*,G) OIL从空到非空
  - **(\*,G)** 修剪信息触发时机:
    - (\*,G) OIL 从非空到空
  - **(S,G)** 加入信息触发时机:
    - (S,G) OIL 从空到非空
  - **(S,G)** 修剪信息触发时机:
    - (S,G) OIL从非空到空
  - **(S,G)RP-bit** 修剪触发时机:
    - (S,G)RPF 信息 **不等于** (\*,G) RPF 信息

#### → PIM状态标识

Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C - Connected,  
 L - Local, P - Pruned, R - RP-bit set, F - Register flag,  
 T - SPT-bit set, J - Join SPT, M - MSDP created entry,  
 X - Proxy Join Timer Running, A - Candidate for MSDP Advertisement,  
 U - URD, I - Received Source Specific Host Report, Z - Multicast Tunnel  
 Y - Joined MDT-data group, y - Sending to MDT-data group

## 六、RP

### 1、Auto-RP

#### →C-RP选举及其作用

C-RP以224.0.1.39发送自己希望成为RP的消息给MA，然后MA**选择其中的IP较大的成为RP**，然后以224.0.1.40发送泛洪，让其他的路由器知道谁是RP，C-RP是候选RP，监听RP的情况。

怎么发现RP失效呢？

RP每60s发送消息给MA告诉自己还存活， $3*60=180s$ 还没有给MA发送消息，也就是MA三倍时间收不到RP的消息，就判定RP失效。重新选举RP。

#### →MA及其作用

Mapping Agent

接收C-RP发送的消息，选举出RP（IP地址大的），并告诉其他设备

#### →多个MA引发的问题

多个MA引发的问题:RP Source来回的切换；dense-mode的回退。

解决DM回退：

**R(config-if)#ip pim sparse-mode** ——将PIM的模式控制在稀疏模式，或者只有下面的两条命令就可以防止回退DM

**R(config)#no ip pim dm-fallback** ——关闭DM回退

**R(config)#ip pim autorp listener** ——让RP对224.0.1.39和224.0.1.40用PIM-DM方式运作，让MA可以接收到C-RP的announce

#### →工作原理和存在的问题

作为C-RP的路由器，周期60s以224.0.1.39这一组播地址向MA发送announce消息，宣告自己是RP（这一过程是以PIM-DM来发送的，因为此时没有RP），发送到MA后，MA选取具有最大C-RP的IP成为全网的RP，然后MA以224.0.1.40的地址向全网所有的路由器泛洪RP是谁（之后有RP出现了，这之后就是以PIM-SM来转发数据了）

Cisco私有的协议

其他注意：**Auto-RP默认优于静态RP**，可以用命令：IP PIM RP-address xxxx override使静态的RP高

### 2、PIMv2 BSR

#### →与Auto-RP的区别

PIMv2 BSR的作用和Auto-RP是一致的，在实现上有以下差别：

- 1、Auto-RP需要独立的组播组 224.0.1.39、224.0.1.40 来通告 RP 信息到整个网络，其他路由器在获得 RP 之前，需依赖密集模式来转发组播通告。而 PIMv2 BSR 使用 PIM 协议来通告 RP，逐跳逐跳通告，不需要依赖密集模式
- 2、Auto-RP 是 Cisco 专有协议，而 BSR 是和 PIMv2 一起。
- 3、BSR 不能限制通告范围，但可设置边界。
- 4、BSR 可以设置优先级，选择出最优的 C-RP 和 C-BSR。

此处的不同就是BSR完全是以PIM-SM来工作的；

而且是在多个C-BSR中选择一个BSR，作为对C-RP信息的汇聚，然后发送去让C-RP自己决定谁是RP；

**同时BSR是支持RP负载分担的**，具体的方法是通过hash值来控制（比如我此时的hash配置为32，那么对于32bits就没有剩余，也就是0， $2^0=1$ ，也就是每一个组播组就会分一个RP；若是配置的hash为24，那么还有8位，剩下的八位一共可以有 $2^8=256$ 个组播组，也就是每256个组播组分配一个RP）；

还有一点就是没有像Auto-RP那样会有224.0.1.39和224.0.1.40来完成RP的选举，因为在BSR中知道BSR的地址（单播路由获知），所以就会以Unicast的方式发送给BSR自己的C-RP信息，也是60s一次，然后BSR集合所有收集的C-RP信息，然后以224.0.0.13泛洪给所有的PIM路由器，有所有的PIM路由器自己来选举RP。

#### →BSR的选举和作用

BSR选择优先级高的，

若优先级相同，比较IP地址大的成为BSR

#### →C-RP和RP选举

先比较优先级，大的成为RP（Cisco的IOS是小的优先）

若优先级一致，然后比较hash值，大的优先

上面的都一致，就比较IP地址，大的优先

#### →工作原理

没有像Auto-RP那样会有224.0.1.39和224.0.1.40来完成RP的选举，因为在BSR中知道BSR的地址（单播路由获知），所以就会以Unicast的方式发送给BSR自己的C-RP信息，然后BSR集合所有收集的C-RP信息，然后以224.0.0.13泛洪给所有的PIM路由器，有所有的PIM路由器自己来选举RP。

其他注意：BSR的lo0不需要运行PIM，然后Auto-RP需要，因为它需要发组播消息。

相应的BSR信息不回泛洪至其他网络（接口下）：IP PIM BSR-border

BSR中的hash配置为0表示不参加负载分担

### 3、任播 RP和MSDP

#### →理解任播的概念

任播可以理解一种特殊的单播，所有的IP都是一样，那么目的想与路由器通信，就与最近的那一路由器通信

#### →理解MSDP的工作原理

**多播源发现协议**从其他PIM域中发现多播源，用SA（Source Active源有效）报文交互消息，建立MSDP peer之后，IP地址较大的路由器将侦听TCP端口639，而IP地址较小的路由器则试图主动连接到端口639。

MSDP消息类型：SA；SA Request；SA Response；Keepalive；Notification

SA消息里面的内容：多播源地址，多播组地址，发信RP的IP地址

RP接收到SA后，通过查看该多播组的（\*，G）出站接口列表中是否有接口来确定其域中是否有该SA的多播组成员。如果没有组成员，RP就不做任何的操作，如果有组成员，那么RP就向该多播源发送一条（S,G）加入信息，从而建立了一个穿越AS边界去往RP的该多播源树的树枝。当多播报到达RP之后，就会沿着自己的共享树被转发到该RP所在域的组成员，之后这些组成员的DR就可以利用标准的PIM-SM进程选择加入去往该多播源的RPT树。

#### 4、RP安全

→抑制Auto-RP/BSR信息

阻止RP通告（224.0.1.39）进入或者是离开网络，使用的是命令

R2(config-if)#ip multicast boundary ?

<1-199> IP access-list number

<1300-2699> IP access-list number (expanded range)

WORD IP access-list name

若要阻止BSR的信息进出网络的话，可以直接在接口上配置命令：ip pim bsr-border

→过滤C-RP通告

这里把R4作为MA

R4(config)#ip pim rp-announce-filter ?

group-list Group address access-list ——定义C-RP通告的哪些组是被接收的<若没有定义，就拒绝所有>

rp-list RP address access-list ——定义接收哪些C-RP的通告的组播组，那么前面又过滤，所以这里是deny 一个组播地址，permit any就可以实现只允许一个组播组的放行。

→控制源注册

在R3上作为C-RP，R5作为MA和组播源，R1的Lo0作为组播信息的接收者，R1要到R3上去加入，R5要到R3上去注册，这里控制R3，在R3上配置以下命令，使R5不能完成注册：

R3(config)#ip pim accept-register ?

list Access list

route-map Route-map

## 七、PIM协议扩展

### 1、SSM

为指定源组播提供了解决方案，使用IGMPv3协议，可以说是通过PIM-SM的一部分机制来实现的（邻居发现，DR选举，加入过程），由于知道S，所以不用选举RP

好处：简化了协议处理；提高了报文转发效率。

→使用组播组地址

232.0.0.0-232.255.255.255

→基本工作原理

主机端DR周期的发送IGMPv3消息，假如有主机要接收来自特定源的组播消息，则终端回复一个IGMPv3 report，包含想要接收的组播源地址S以及组播组地址G。当然，加入有主机想接收，也可以主动发送report消息。

这里特点是：虽然配置的是稀疏模式，但是却不需要RP，在接收者哪儿加入的是静态组232/8并且指定特定源。在SSM里面只有SPT。

### 2、双向PIM

→工作原理

PIM稀疏双向模式是PIM稀疏模式的一种扩展，PIM稀疏模式不能朝上游方向传输数据包，因为这与会所有的组播路由器所执行的RPF检查相冲突。只有当所有其他流量在共享中通过RP向下游流动时，所有加入信息才会包含在注册信息中发送给RP。

具体的说，PIM双向模式适用于many-to-many的环境中，因为这种环境下，假如采用的是PIM-SM，那么会产生大量的（S,G）条目，而采用PIM双向模式最显著的特点就是可以减少路由表的条目。

改用双向模式之后，基本可以这么认为，所有从源发出的组播数据报逐跳向RP方向传递，到达RP之后，逐跳向组播接收客户端方向传递，因此RP最好位于网络的核心位置，否则可能会出现次优路径。

→防环思想

双向PIM虚制定一个DF，即转发路由器，来避免拓扑的环路。每一个网段需要选举一个DF，负责将PIM将适当的组播流量向上游转发，

对RP具有最佳路由的路由器成为DF。

→DF选举规则

对RP具有最佳路由的路由器成为DF。

→幻影RP

配置：IP pim bidir-enable

IP pim rp-address 1.1.1.1 bidir

同样RP地址所在接口需要运行PIM-SM，否则不安装下游组的加入信息。

创建\*组的条件：



- 1、收到\*组的加入信息；
- 2、或者创建S组，但是在路由条目中没有对应的父路由（\*组），那么就创建一个\*组。

**创建S组的条件：**

- 1、收到组播数据流；
- 2、收到明确的S组加入。

先有\*组再有S组，\*组中的outgoing口会自动的copy在S组  
 先有S组后有\*组，这个S组中的outgoing是不会copy到\*组

\*组和S组的RPF接口不一致就会触发RPT剪切，因为我不知道S我才依靠的是\*组，换句话说，我知道了S，就触发了RPT剪切。

```
SM S-----SPT-----RP-----RPT-----R
SSM S-----SPT-----SPT-----R
BIDR S-----RPT-----RP-----RPT-----R
```