

# BGP 技术笔记

## 红茶三杯 CCIE 学习文档

文档版本： 2.0

更新时间： 2013-01-26

文档作者： 红茶三杯

文档地址： <http://ccietea.com>

## 1 基本概念

### 1.1 知识点

1. 每个 AS 都有一个标识号，范围是 1~65535，其中 64512~65535 是保留私用的。
2. BGP 的更新由 TCP 协议承载，使用的端口号是 179 因为 BGP 要求使用 TCP，所以 BGP 对等体之间必须有 IP 层连通性。
3. BGP neighbor：当两台 router 相互之间建立了一条基于 TCP 的 BGP 连接之后，就称他们为邻居或对等体。在邻居刚建立起连接时，它们交换所有的候选 BGP 路由，但在该初始路由交换之后，通常只在网络信息发生变化时才发送增量路由更新，而不会周期性更新。
4. BGP 是设计为在 AS 之间传递路由，因此，它的一跳实际上是一个 AS。
5. BGP 是无类路由选择协议、距离矢量路由协议，自动汇总默认关闭（这点要看 IOS）。
6. BGP 有三个管理距离，从 IBGP 学过来的 200，从 EBGP 学过来的 20，这是因为 BGP 的设计理念的工作于 AS 之间，而 AS 内部，BGP 希望 IGP 协议自己能搞定，因此 IBGP 路由的管理距离设置为一个大 AD 值 200，EBGP 路由设置为一个小 AD 值 20。
7. BGP 选举 routerID 法则和 OSPF 一样。

### 1.2 Tables

#### 1. Neighbor table

```
Router# sh ip bgp summary
```

BGP router identifier 10.1.13.3, local AS number 345

BGP table version is 1, main routing table version 1

Neighbor	V	AS	MsgRcvd	MsgSent	TblVer	InQ	OutQ	Up/Down	State/PfxRcd
10.1.13.1	4	100	3	3	1	0	0	00:00:54	0

关于 BGP 邻居表字段的详细含义，请见本文档 BGP 配置章节的 show 小节

## 2. BGP table

存放所有路由条目，以及其属性。

关于 BGP 表相关字段的详细含义，请见本文档 BGP 配置章节的 show 小节

## 3. Routing table

IP 路由表

## 1.3 TIMER

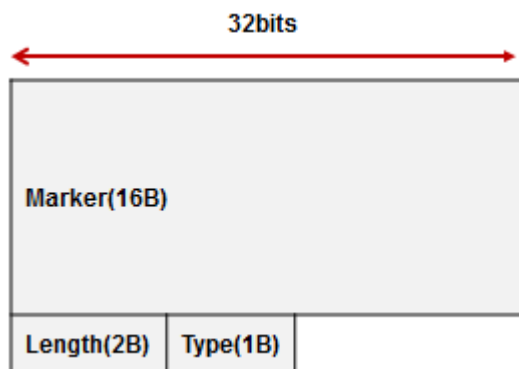
- **KEEPALIVE** 默认 60s，keepalive 计时器不会在 Open 消息中交互，那么两个 BGP 邻居间该计时器如何定？  
如果当前手工配置的 keepalive timer 小于  $\min(\text{holdtime})/3$ ，则取配置值  
如果当前手工配置的 keepalive timer 大于  $\min(\text{holdtime})/3$ ，则取  $\text{int}[\min(\text{holdtime})/3]$   
其中  $\min(\text{holdtime})$  为两台 BGP 邻居间 holdtime 的最小值
- **HOLDTIME** CISCO 默认 180s（3 倍 KEEPLIVE timer），该计时器包含在 open 报文中  
必须收到一个 KEEPLIVE 或更新消息前所允许经过的最大时间。如果两端 Holdtime 不一致，双方接受较小的的时间。

**timer bgp 0 0 邻居永远不 down**

BGP 不会周期性更新路由，仅在需要的时候更新，由于公网的路由可能的动荡的，因此触发更新也会有一定的等待时间，IBGP peer 为 5S；EBGP peer 为 30S，而在这段时间内，BGP 仍可以进行路由信息的搜集，所以 BGP 收敛会比较慢。

## 1.4 消息类型

每种消息都包含 BGP 消息报头，BGP 报文的头部如下：



**Marker :** 用于检测 BGP 对等体之间同步丢失情况，并且在支持验证功能的情况下进行消息验证  
如果消息类型为 open 或 open 消息中没有包含验证消息，标志字段被置为全 1，  
否则标志字段通过某些计算得到（作为验证进程的一部分）

**Length :** 表示 BGP 报文的全部长度，包括头部

**TYPE :** 1-open ; 2-update ; 3-Notification ; 4-keepalive

以下是 BGP 的五种报文及报文解析：

### 1. OPEN 消息

TCP 会话建立起来以后，两个邻居都要发送一个 OPEN 消息，双方使用 OPEN 消息标识自己，并且规定自己的 BGP 运行参数。如果 open 消息被接受，则回送一条 keepalive 消息进行确认，确认后就能发送 update 消息了。OPEN 消息包含以下内容：

- Version 8bit , V4 目前使用较多的版本
- AS 号 16bit 本地 AS 号
- HOLDDTime 路由器必须收到一个 keepalive 或者更新消息之前所允许经过的最大秒数
- BGP Identifier ROUTER-ID 和 OSPF 选取 routerID 的方式一致
- 可选参数长度 用来表示后面可选参数字段的长度
- 可选参数 包含了一个可选参数列表，每个参数都由一个长为 1 个 8 位组 的类型字段、一个长 8 位组的长度字段及一个可变长的包含参数数值的字段组成。用来宣告支持验证、多协议支持和路由刷新等可选功能（常被称为“能力值”，意思就是这家伙具备什么能力）

### 2. KEEPALIVE 消息

如果路由器接受了邻居在 OPEN 消息中的参数，就会应答一个 keepalive 消息，并且在此后 1/3 的 holdtime( 但不小于 1S ) 为周期发送该消息，CISCO 默认 60S。如果协商后保持时间为 0，则不发送 keepalive 保活消息。KEEPALIVE 消息实际上弥补了 TCP 无法确认对端存活情况的缺陷。

KEEPALIVE 消息仅包含 19bytes 的 BGP 头部，除此之外不包含任何其他数据。

### 3. UPDATE 消息

用来公布可用的路由、撤销的路由或者两者兼顾，

每条 update 消息只描述单条 BGP 路由，这是因为 BGP 路径属性只能描述单条路由

消息中包含：

- 网络层可达信息（NLRI） 一个或多个（长度、前缀）二元组，用来公布 IP 地址前缀和前缀长度
- 路径属性
- 被撤销路由

#### 4. Notification 消息

当检测到差错的时候发送，通常会导致 BGP 连接的终止

#### 5. Route-refresh

当路由策略发生变化时，去请求邻居重新通告路由（BGP 不会周期性发送更新）

## 1.5 BGP 状态机

更加详细的 BGP 状态机及邻居关系建立过程，请见笔记文档：《BGP 状态机及邻居关系建立过程》

Peer 状态名称	发什么包	在做什么
Idle	尝试建立 TCP 连接	本地寻找一条到邻居的路由，并开始准备 TCP 的连接及监视远程 peer 启动 TCP 连接。启用 BGP 时，要准备足够的资源
Connect	发 TCP 包	本地找到一条到邻居的路由，并尝试 TCP 三次握手，等待完成中，认证都是在 TCP 建立期间完成的。如果 TCP 连接不上则进入 Active 状态，反复尝试连接。 如果 TCP 建立成功，BGP 进程会向邻居发送 Open 消息并进入 Opensent 状态
Active	发 TCP 包	TCP 连接没建立成功，反复尝试 TCP 连接。
OpenSent	发 Open 包	TCP 连接建立已经成功，开始发送 Open 包，Open 包携带参数协商对等体的建立。 如果接收到 open 消息后，存在差错，则发送 Notification 消息。如果没有差错，则进入 OpenConfirm 状态
OpenConfirm	发 Keepalive 包	参数、能力特性协商成功，自己开始发送 Keepalive 包，等待对方的 Keepalive 包。 如果收到对方的 keepalive 消息则迁移到 Established 状态
Established	发 Update 包	已经收到对方的 Keepalive 包，双方能力特性一致，开始使用 Update 通告路由信息。

## 1.6 BGP neighbor

必须在 BGP 进程中使用 neighbor 来指定 BGP 对等体。他们交换的是路由信息，以及相关属性，而不是链路状态。

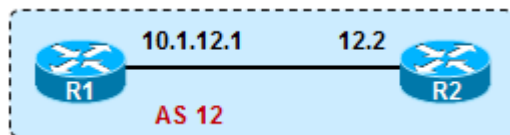
### 1. EBGp 邻居

一般是直连，因为它会去查找直连路由。EBGP 默认 TTL 为 1

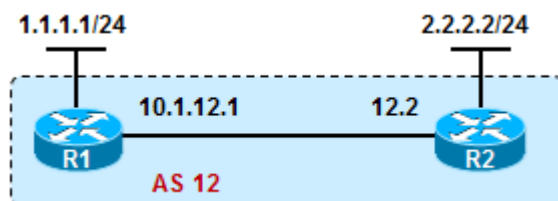
## 2. IBGP 邻居

无需直连。只要求有 TCP 联通性。

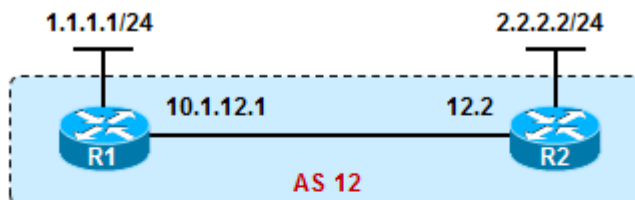
### 1.7 update-source



BGP 无法像 IGP 那样自动发现邻居,而需手工指定,邻居的 IP 由本地的 BGP neighbor 命令指定,而该 BGP 连接的源 IP (更新源) 默认情况下为流量的出接口 IP。注意只有当本地配置的邻居 IP 与邻居用于 BGP 连接建立的源 IP 相同时, BGP 连接才能被正常建立,同时, 仅需保证一方满足条件即可。



IBGP 邻居之间建立邻居, 为了保证邻居关系的稳定, 一般使用 loopback 接口建立, 这是因为如果使用物理接口, 那么物理接口故障, 邻居关系就 DOWN 了, 并且在 AS 内部, 路径可能是冗余的, 邻居之间的 LOOPBACK 路由可通过 IGP 获取并提供一定的路由冗余性(当物理线路也存在冗余的情况下)。在使用 loopback 接口建立 BGP 邻居关系时, 务必注意还需要指定更新源 IP。如下图, 是一个规范配置:



#### R1的配置:

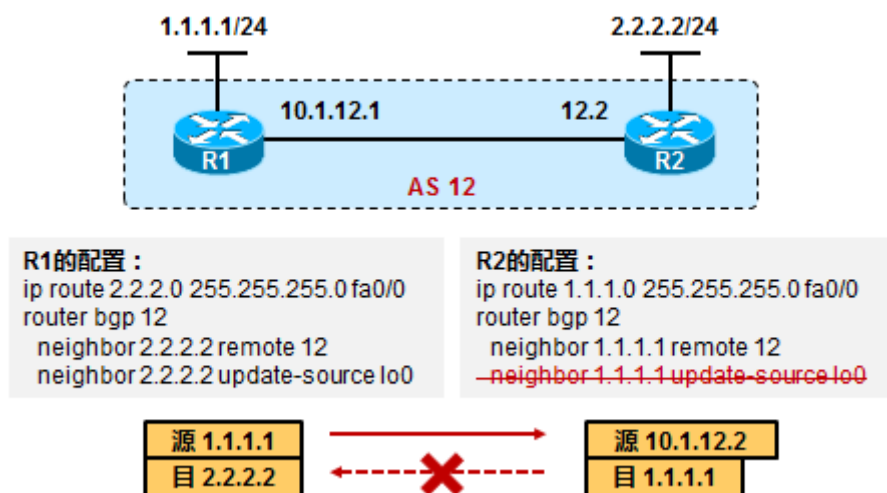
```
ip route 2.2.2.0 255.255.255.0 fa0/0
router bgp 12
neighbor 2.2.2.2 remote 12
neighbor 2.2.2.2 update-source lo0
```

#### R2的配置:

```
ip route 1.1.1.0 255.255.255.0 fa0/0
router bgp 12
neighbor 1.1.1.1 remote 12
neighbor 1.1.1.1 update-source lo0
```



其实, BGP 邻居关系的建立, 仅需一条连接即可, 如下图的配置也是能建立起 BGP 连接的:



R1 使用 1.1.1.1 作为更新源，试图与 2.2.2.2 建立 BGP 连接，而 R2 本地配置的 neighbor IP 为 1.1.1.1，与 R1 的更新源 IP 匹配，尽管 R2 配置的更新源为默认（数据包出接口 IP 10.1.12.2），但是两者之间的 BGP 连接建立是没有问题的，因为 R1 向 R2 的连接是没问题的。同时这里还可以顺便钻研下路由的问题，例如，如果将 R2 本地到 1.1.1.0 的静态路由，改配为默认路由，我们会发现 BGP 连接依然能建立，因为此例中 R2 为 BGP 连接的被动者，R1 有明细路由即可主动发起连接。但是如果将 R1 上到 2.2.2.0 的明细静态路由，改为默认，那就不行了，R1 会认为 “no route to peer”。

- 如果 R1 R2 之间是建立 EBGP 邻居关系，因为 EBGP 邻居关系的建立会检查直连路由，并且默认 TTL=1，而这个时候实际上 R1 的 loopback 口到 R2 是需要至少 2 跳，那么这个时候 还需要两者配置 neighbor xxxx ebgp-multihop 2
- 经试验验证，（在 BGP 的基本配置无误的情况下）R1 和 R2 到对方的 looback 口的网段用静态路由或动态路由互相学习，BGP 邻居关系建立都没问题，但是如果两端都用默认路由互指，则 BGP 邻居关系无法建立，这是为了防环的目的，一边用静态，一边用默认是可以的。

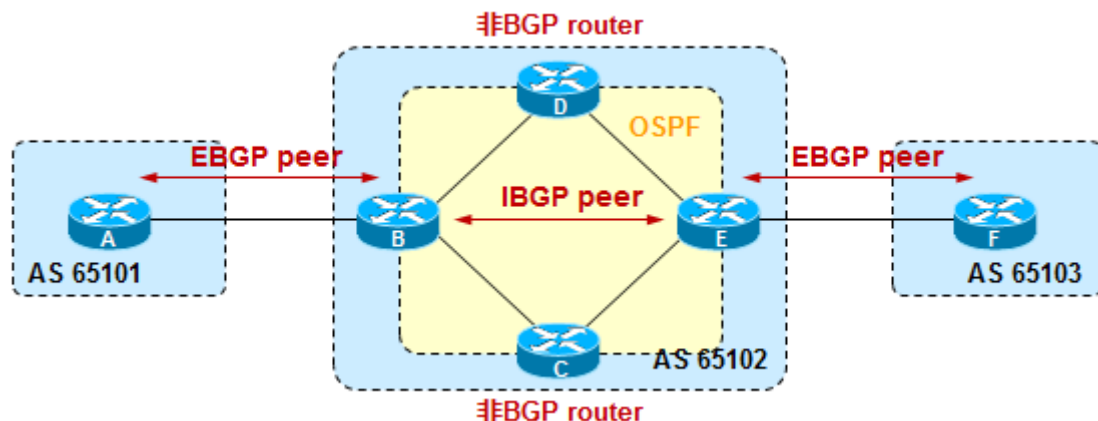
## 1.8 IBGP 水平分割

BGP 防环是通过 AS\_PATH 实现的，而 AS\_PATH 仅仅在路由离开 AS 才被更改，因此在 AS 内，IBGP 就没有 EBGP 的防环能力，为了防止环路出现，BGP 路由器不会将从 IBGP 邻居学习过来的路由再通告给自己其他 IBGP 邻居。

**BGP 规定不将通过一个 IBGP 获悉的路由传播给其他所有 IBGP 邻居。这个是 BGP 的水平分割规则。**

由于水平分割原则存在，BGP 要求 AS 内，须保证 IBGP 全互联（这里是指 neighbor 命令指定）。（根本原因是在 AS 内部，AS-PATH 不会改变，无法使用 AS\_PATH 防环，因此很容易出现环路）

## 1.9 IBGP 与 IGP 同步



AS65102 作为中转区域 ( transit AS ), 在 AS 内部 BCDE 运行 OSPF , 使得 AS 内路由可达 ; D 和 C 都没运行 BGP 协议 , BE 之间建立 IBGP 邻居关系。若 A 上有个网段 1.1.1.0 , A 将其注入 BGP , 并且传递给 EBGP 邻居 B , B 又会传递给 IBGP 邻居 E ( 这个传递过程其实是将 BGP 报文放置于 IP 包内经过 C 或 D 最终传递给 E , 对于 CD 而言 , 这些数据包都是普通 IP 包 , 直接转发不查看 )。最终 E 成功的将 1.1.1.0 的路由传递到了 F。那么 F 即使能学习到 BGP 过来的这条路由 , 当有数据前往 1.1.1.0 网络时 , 将数据包丢给下一跳 E , 而 E 上关于 1.1.1.0 的下一跳是 B , 非直连 , 因此需递归得到其前往 B 的下一跳 , 到 B 的下一跳是 D 或 C , 于是将数据包丢给 D 或 C , 而 C 和 D 是并不知道 1.1.1.0 的 ( 他们只运行了 OSPF , 没有运行 BGP ) , 至此成路由黑洞。注意虽然 CD 没运行 BGP , 但是 BE 之间的 BGP 路由 ( BGP 报文 ) 可以通过 CD 进行转发 , 并且对于 CD 来说 , 这些 BGP 的消息他们自己视为普通的 IP 包 , 直接转发 , 并不查看。

这就是路由黑洞问题 , 为了避免这个问题 , 可以考虑在 BE 上 , 将 BGP 路由重发布进 OSPF 来解决 , 但这么做后果是不可预估的 , 毕竟 BGP 承载的路由条目是相当巨大的。另一个解决方案是 , 要求 AS 内路由器都运行 IBGP 并实现 IBGP 全互联 , 那么该 Transit AS 内的路由器就都能知晓 BGP 路由 , 如此即可解决路由黑洞的问题。

**【同步的概念】** BGP 路由器从 IBGP 邻居学到一条路由后 , 是不启用的 ( 不优化的 ) , 除非它再次从 IGP 学习到相同的路由 , 才会启用。这就是为了防止上面所描述的路由黑洞的问题。

而如果这个网络实现了 IBGP 全互联 ( 彼此之间建立了 IBGP 邻居关系 ) , 那么同步就没有意义了 , 便可关闭同步。因此现今的 CISCO IOS 默认关闭同步规则。

**综上所述 , 要使 IBGP 能够正常工作 , 就必须实施以下配置选项之一 :**

1. 将外部路由重发布进 IGP 中 , 以确保 IGP 与 BGP 同步。但该方法的缺陷在于如果从 BGP 获取大量的路由 , 对于 IGP 来说是个相当大的负担。
2. 建立 IBGP 全互联 , 且关闭同步机制。目前基本上都是使用该方法。但是这个方法有个缺陷 , 即如果 IBGP

邻居太多，管理这些 IBGP 的邻接关系将会是个挑战，并且对设备的负担也较大。好在我们有两种措施可以辅助解决，1 是路由反射器 2 是联邦。

## 1.10 BGP 路由通告

当存在多条路径时，BGP Router 只选取最优的路由（BEST）来使用（没有负载均衡的情况下）

BGP 只把自己使用的路由，也就是自己认为 Best 的路由传递给 BGP peer

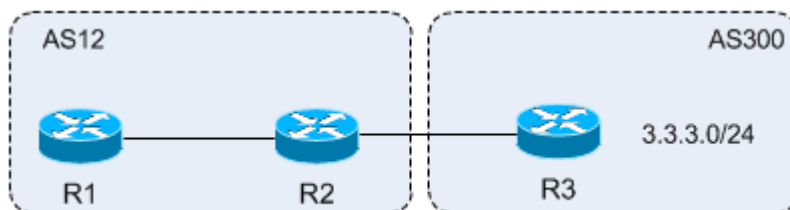
BGP Speaker 从 EBGP 获得的路由会向它所有 BGP 相邻体通告（包括 EBGP 和 IBGP）

BGP Speaker 从 IBGP 获得的路由不向它的 IBGP 相邻体通告（避免环路，水平分割；存在路由 RR 的情况除外）

**BGP Speaker 从 IBGP 获得的路由是否通告给它的 EBGP peer 要视 IGP 和 BGP 同步的情况来决定**

## 2 Advance

### 2.1 关于路由的递归



R1、R2 运行 OSPF，宣告直连网络及各自的 LOOPBACK 接口，R1 LO1 为 1.1.1.1，R2 LO1 为 2.2.2.2

R1、R2 运行 IBGP，使用 LOOPBACK 为更新源，并互相指 neighbor，R2、R3 之间为 EBGP 关系，用直连接口建邻居。R3 宣告 3.3.3.0/24 进 BGP，在 R2 上对 R1 配置 next-hop-self

于是在 R1 上能学习到 3.3.3.0 的路由，R1 的路由表如下：

```

1.0.0.0/24 is subnetted, 1 subnets
C      1.1.1.0 is directly connected, Loopback0
2.0.0.0/32 is subnetted, 1 subnets
O      2.2.2.2 [110/65] via 10.1.12.2, 00:05:16, Serial0/0
3.0.0.0/24 is subnetted, 1 subnets
B      3.3.3.0 [200/0] via 2.2.2.2, 00:03:01
10.0.0.0/24 is subnetted, 1 subnets
C      10.1.12.0 is directly connected, Serial0/0
  
```

这就是 BGP 典型的递归路由，R1 上 3.3.3.0 路由的下一跳为 2.2.2.2，进一步递归 2.2.2.2，关联出接口 S0/0

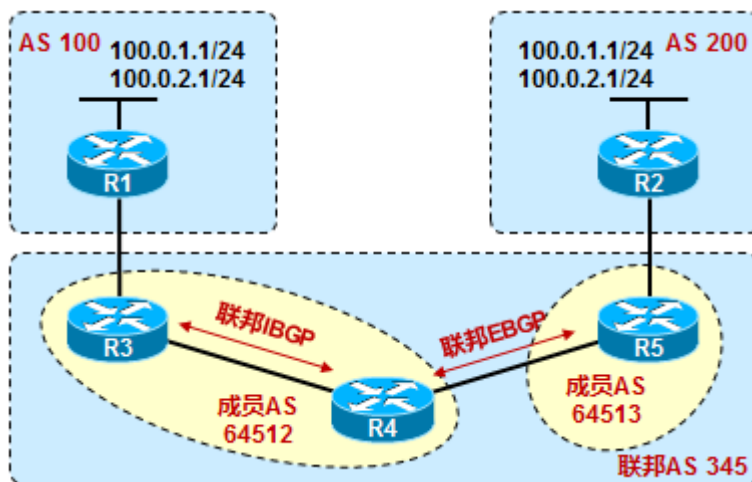


## 2.2 联邦

### 联邦的相关特性：

- 在联邦内部保留联邦外部的 NEXT\_HOP 属性
- 公布给联邦的路由的 MED 属性在整个联邦范围内予以保留
- 路由的 LP 属性在整个联邦范围内予以保留
- 在联邦范围内，将成员 AS 号压入 AS\_PATH，但不公布到联邦外，并且使用 TYPE3、4 的 AS\_PATH
- AS\_PATH 中的联邦 AS 号用于在联邦内部避免环路

### 1. 联邦的配置及实现



为了解决路由传递的问题，除了路由反射器外，还有一个不错的解决方案，就是联邦。

通过将 AS345 定义为联邦 AS (大 AS)，同时在联邦 AS 中创建成员 AS (小 AS)，则可解决 IBGP 路由传递的问题。

那么 R3、R4 之间就是联邦的 IBGP 关系，R4 与 R5 之间是联邦的 EBGP 关系，

在联邦内部，R3 与 R4 都是属于 AS64512，对于 R4 (AS64512)，R5 属于 AS64513，但是对于联邦外部而言，R3R4R5 都是 AS345，外部压根不知道有 AS64512 和 64513 存在。

R3、R4、R5 用 OSPF 保证 AS 内路由互通，同时使用 loopback 建立 BGP 邻居

#### R3 上的配置如下

```
router bgp 64512                                     //使用联邦成员 AS 号建立 BGP
  bgp confederation identifier 345                   //这条命令用来对联邦外的 AS 通告自己的真实 AS 号
  neighbor 4.4.4.4 remote-as 64512
  neighbor 4.4.4.4 update-source Loopback0
  neighbor 10.1.13.1 remote-as 100
```

**bgp confederation iden 345** 配置后，对于联邦 AS 外来说，这个 AS 不是什么 64512 了，而是 345

**R4 的配置：**

```
router bgp 64512
bgp confederation identifier 345
bgp confederation peers 64513
neighbor 3.3.3.3 remote-as 64512
neighbor 3.3.3.3 update-source Loopback0
neighbor 5.5.5.5 remote-as 64513
neighbor 5.5.5.5 ebgp-multihop 3
neighbor 5.5.5.5 update-source Loopback0
```

由于 R4 与 R5 是联邦的 EBGP，同样有 TTL 的问题，因此他们使用 LOOPBACK 建立邻居关系的话，要注意设置 ebgp-multihop。

另外，对于 R4 而言，R5 此刻是一个普通的 EBGP 邻居，并且是另一个 AS64513，而且跟我一点关系没有，联邦的建立就会有问题，因此，还需在 R4 上增加 **bgp confederation peers 64513 命令**，那么 R4 将对 AS64513 视为它的联邦 EBGP peer，而对除了 AS64513 外的 AS 视为普通的 AS。如果联邦内有成员 AS，那么若本地需指多个 confederation peers，则可 **bgp confederation peers xx yy zz**，写多个 AS 号。

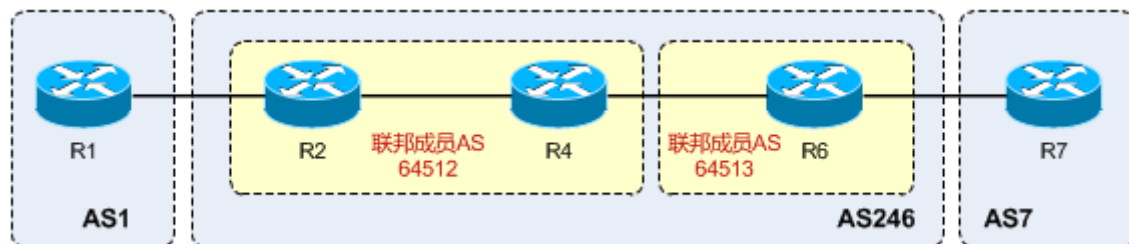
**R5 的配置如下：**

```
router bgp 64513
bgp confederation identifier 345
bgp confederation peers 64512
neighbor 4.4.4.4 remote-as 64512
neighbor 4.4.4.4 ebgp-multihop 4
neighbor 4.4.4.4 update-source Loopback0
neighbor 10.1.25.2 remote-as 200
```

## 2. AS\_CONFED\_SEQUENCE 及 AS\_CONFED\_SET

这两个属性用于在联邦内、成员 AS 间防环

- **AS\_CONFED\_SEQUENCE** 一个去往特定目的地所经路径上的有序 AS 号列表，其用法与 AS\_SEQUENCE 完全一样，区别在于该列表中的 AS 号属于本地联邦中的 AS
- **AS\_CONFED\_SET** 一个去往特定目的地所经路径上的无序 AS 号列表，去用方法与 AS\_SET 完全一样，区别在于列表中的 AS 号属于本地联邦中的 AS



R1 上引入路由 11.11.11.0

**R2 上 BGP 表：**

Network	Next Hop	Metric	LocPrf	Weight	Path
*> 11.11.11.0/24	192.168.12.1	0		0	1 i

**R4 上 BGP 表：**

Network	Next Hop	Metric	LocPrf	Weight	Path
*>i11.11.11.0/24	2.2.2.2	0	100	0	1 i

**R6 上 BGP 表：**

Network	Next Hop	Metric	LocPrf	Weight	Path
*> 11.11.11.0/24	2.2.2.2	0	100	0	(64512) 1 i

**R7 上 BGP 表：**

Network	Next Hop	Metric	LocPrf	Weight	Path
*> 11.11.11.0/24	192.168.67.6			0	246 1 i

在 R4 上连接 R2 的接口抓包，发现更新包中 11.11.11.0 前缀的属性中携带的是 AS\_PATH，内容为 AS 1

在 R6 上连接 R4 的接口抓包，发现更新包中 11.11.11.0 前缀的属性中携带的 AS\_PATH 中包含 2 个内容：

```

Path attributes
├── ORIGIN: IGP (4 bytes)
│   ├── Flags: 0x40 (well-known, Transitive, Complete)
│   ├── Type code: ORIGIN (1)
│   ├── Length: 1 byte
│   └── origin: IGP (0)
├── AS_PATH: (64512) 1 (11 bytes)
│   ├── Flags: 0x40 (well-known, Transitive, Complete)
│   ├── Type code: AS_PATH (2)
│   ├── Length: 8 bytes
│   ├── AS path: (64512) 1  AS_PATH属性包含两个Segment
│   │   ├── AS path segment: (64512)
│   │   │   ├── Path segment type: AS_CONFED_SEQUENCE (3)
│   │   │   ├── Path segment length: 1 AS
│   │   │   └── Path segment value: 64512
│   │   └── AS path segment: 1
│   │       ├── Path segment type: AS_SEQUENCE (2)
│   │       ├── Path segment length: 1 AS
│   │       └── Path segment value: 1
│   └── NEXT_HOP: 2.2.2.2 (7 bytes)

```

可以看出 AS\_CONFED\_SEQUENCE 属性用于在联邦内防环（该属性不会出联邦），而上面的 AS\_SEQUENCE 属性，则将会被随着 11.11.11.0 的路由被传递给 R7，因此在 R7 上抓包，11.11.11.0 的路由中看不到 AS\_CONFED\_SEQUENCE 的属性，因此外部 AS 将联邦视为单个 AS（外界并不知道联邦内部的情况）

## 2.3 路由反射器 RR

### 1. 基本概念

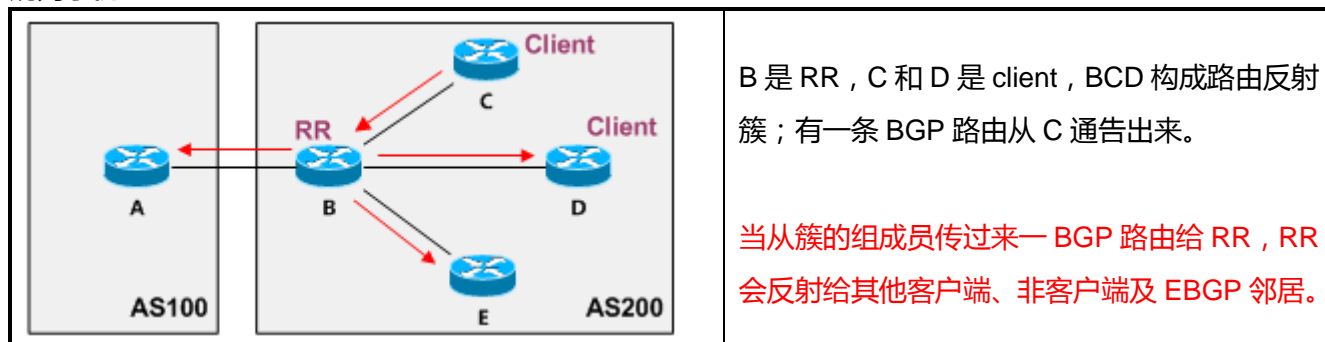
在 AS 内部，由于存在 IBGP 水平分割原则，使得 BGP 路由器之间不得不两两建立 IBGP 连接，以求获得完整的 BGP 路由更新，然而这是个扩展性非常低的做法，同时也给网络设备带来了负担，解决 IBGP 扩展性问题的两种有效的办法是路由反射器及联邦。路由反射器相比于联邦，优势在于，联邦中所有路由器都需要支持并理解联邦机制，而路由反射器只需要 RR 理解反射器机制即可，另外，路由反射器的实现机制也相对简单一些。当然如果希望用各种 EBG 机制来管理大规模 AS，那么联邦将是一个更优的解决方案。

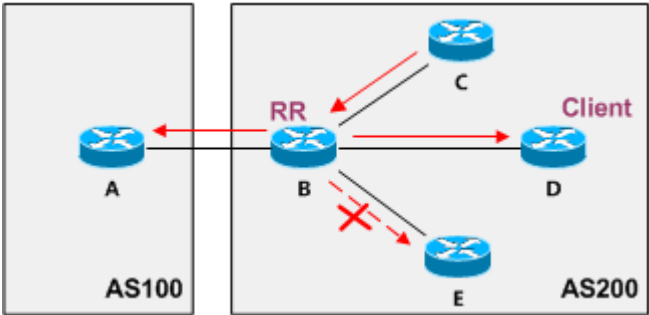
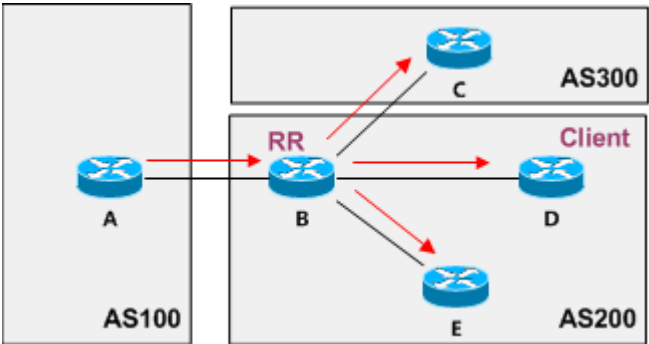
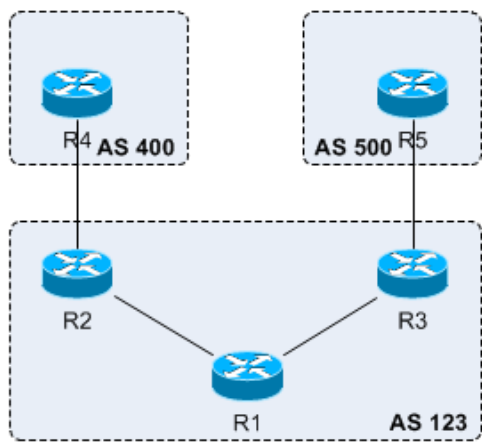
思考路由反射器时，将簇当作一个逻辑的整体去考虑即可，RR 和 client 共同构成反射簇，但是只有 RR 知道（配置只是在 RR 上完成）。注意 RR 只通告或反射它所知道的最佳路径。

为了维护一致的 BGP 拓扑，RR 在反射路由的时候不修改某些 BGP 路径属性，这些属性包括 NH、AS\_PATH、LOCAL\_PREF 和 MED，并且增加了 ORIGINATOR 和 CLUSTER\_LIST 用于防环。

- 如果路由学习至非 client IBGP 对等体，则反射给所有 client 及 EBG 邻居
- 如果路由学习至一 client，则反射给所有非 client IBGP 邻居和除了该 client 以外的所有 client
- 如果路由学习至 EBG 邻居，则反射给所有 client 和非 client IBGP 邻居

### 2. 规则示例



	<p>将路由反射簇看做一个整体 那么 C 传给 RR 后，RR 传给它的 client ( D )，则不传给非 client ( E )</p>
	<p>另一种情况，思路还是一样，将 RR 和 Client 看做一个整体，也就是将簇看做一个整体。</p>
	<p>考虑 R2 是 RR 的情况，R4 和 R5 分别宣告 loop 口 对方能收到路由吗？ 如果 R1 是 RR 呢？</p> <p><b>注意 CLIENT 是不知道自己的身份的</b></p>

### 3. 关于路由反射簇

- 路由反射簇包括反射器及其 Client
- 每一个簇都有唯一的簇 ID
- 每当一条路由被反射器反射后，该反射器（该簇）的 Cluster-ID 就会被添加至路由的 Cluster-list 属性中
- 每当反射器收到一条 Cluster-list 属性已经包含该簇的 ClusterID 的路由时，该路由基于防环的目的将不被反射

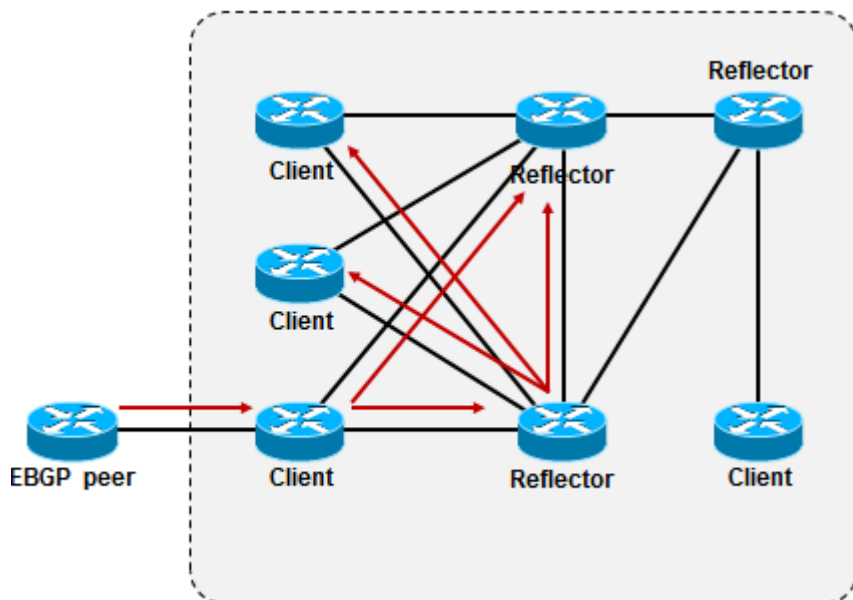
引入 Cluster 是要在 RR 的环境中提供冗余性。在传统的分簇设计中，多个 RR 用来为一个或多个 client 服务，这些 RR 都配置相同的 CLUSTER\_ID，这个 ID 是 4 字节的 IP 形式的标示符，默认情况下就是 RR 自己的 BGP routerID，如果两台 RR 拥有相同的 CLUSTER\_ID，那么他们就属于同一个簇。CLUSTER\_ID 的另一个非常重要的作用是防环，当一台 RR 收到的 BGP 路由更新中携带了与自己 ROUTERID 相同的 CLUSTER\_ID，那么就该 RR 将忽略这条路由更新。

#### 4. ORIGINATOR\_ID 与 CLUSTER\_LIST

本节内容请见 本文档路径属性的相关章节内容。

#### 5. 冗余 RR 环境

单 RR 可能会存在单点故障的问题，因此从冗余性的角度，一个簇中可以拥有多台 RR，Client 与每一台 RR 都有物理连接并建立 BGP 对等体关系，在其中一台 RR 出现故障的情况下，Client 仍然有替代连接。Client 都不知道自己 Client，因此 RR 本身也可以成为别人的 Client



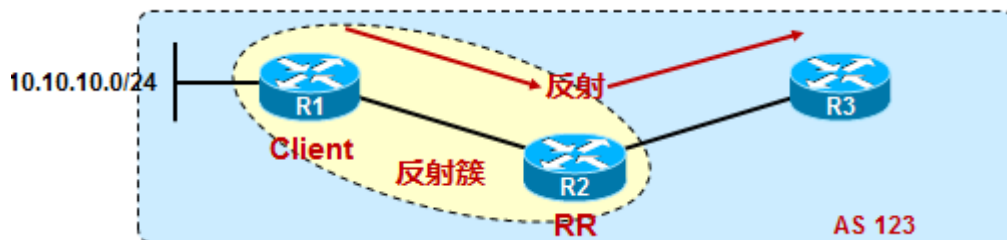
- 冗余RR增加了网络的健壮性
- 使用Originator、Cluster list属性来在冗余RR环境中避免路由环路。
  - 例如将两个RR的ClusterID配置为一样，那么可以起到进一步的防环作用
  - 所有的RR之间建议采用全互联模型
- Client会收到来自两个RR反射的路由，如何决策？

由于 AS\_PATH 属性在 AS 内部不会发生变化（仅当路由离开本 AS 才会被更新），因此 AS 内防环才有水平分割的机制，而路由反射器实际上是放宽了水平分割原则，这个就会给环路带来一定的隐患，因此路由反射器需使用以下两个属性防止环路：

ORIGINATOR\_ID 和 CLUSTER\_LIST 是路由反射器使用的可选非传递属性，用来防止环路。

详见本文档“ORIGINATOR\_ID 和 CLUSTER\_LIST” BGP 属性部分。

#### 6. 配置命令

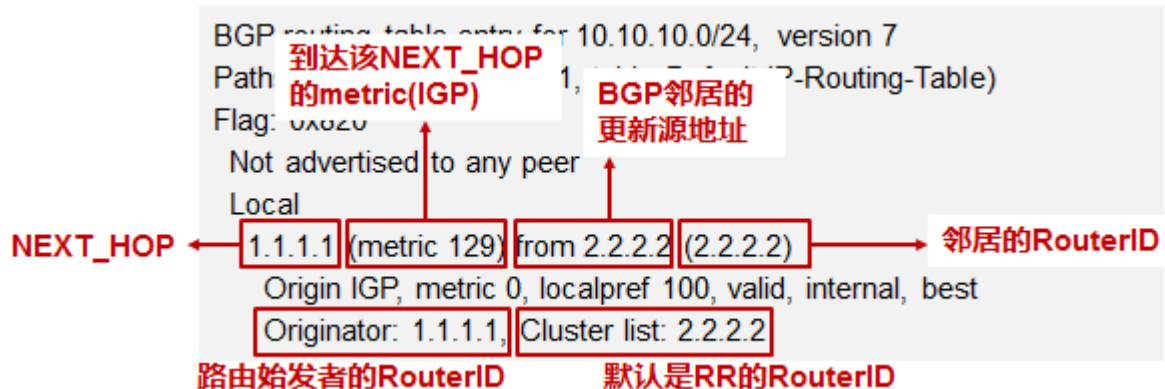


R2 的 BGP 配置如下：

```
router bgp 123
  neighbor 1.1.1.1 remote-as 123
  neighbor 1.1.1.1 update-source Loopback0
```

```
neighbor 1.1.1.1 route-reflector-client
```

R3 上 show ip bgp 10.10.10.0



RR 可修改 cluster-id

```
router bgp 123
  bgp cluster-id 222.222.222.222
```

其他配置命令：

```
bgp client-to-client reflection
```

在配置反射器时，client 到 client 间的反射是默认开启的，但如果客户间是全互联的，此命令加 no，则关闭客户间的反射

## 7. 规划原则

### ● 路由反射器规划原则

- 路由反射器将传输 AS (transit AS) 分割成小单元，也就是反射簇
- 每个簇包含反射器及其 client
- 不支持路由反射器功能的路由器可以充当单路由器簇或充当 client

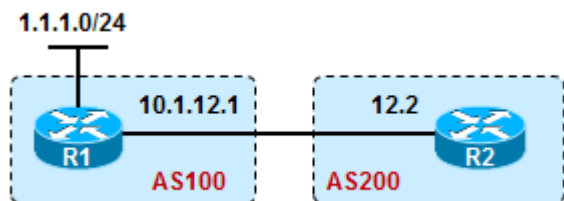
### ● IBGP session 原则

- 路由反射簇中的所有 client 都应该与并且只与簇中所有的 RR 建立 IBGP 连接
- AS 内的路由反射器之间要求全 IBGP 互联
- 非反射器的路由器即可参与 IBGP 全互联也可配置为反射器的 client

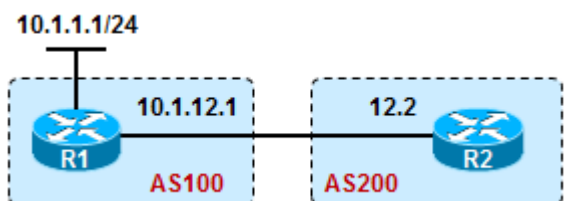
## 2.4 自动汇总

BGP 什么情况下会自动汇总？





- 若 R1 开启 auto-summary，并用重发布直连的方式引入 1.1.1.0/24，则该子网会被自动汇总给 R2
- 若 R1 开启 auto-summary，且 network 1.1.1.0 mask 255.255.255.0，则仍以明细更新给 R2
- 若 R1 开启 auto-summary，且 network 1.0.0.0 mask 255.0.0.0，则通告给 R2 汇总路由 1.0.0.0/8
- 上面这条 network 等同于 network 1.0.0.0 (network 的有类宣告)



这个实验主要验证自动汇总如果不是发生在主类网络边界的情况：

- 若 R1 开启 auto-summary，并用重发布直连的方式引入 10.1.1.0/24，则该子网会被自动汇总给 R2
- 若 R1 开启 auto-summary，并 network 10.0.0.0，则该子网会被自动汇总给 R2

从这个实验分析得出 BGP 的自动汇总，不要求主类网络边界，这与 IGP 要区别开。

因此 BGP 自动汇总 (auto-summary) 只汇总重发布引入的路由，以及使用 network 命令有类宣告方式引入的路由。目前 CISCO IOS 默认关闭自动汇总。

## 2.5 手工汇总

可以不 network 明细路由，而是在路由起源，配置一条汇总静态，然后 network 该汇总路由，这种方法不建议。

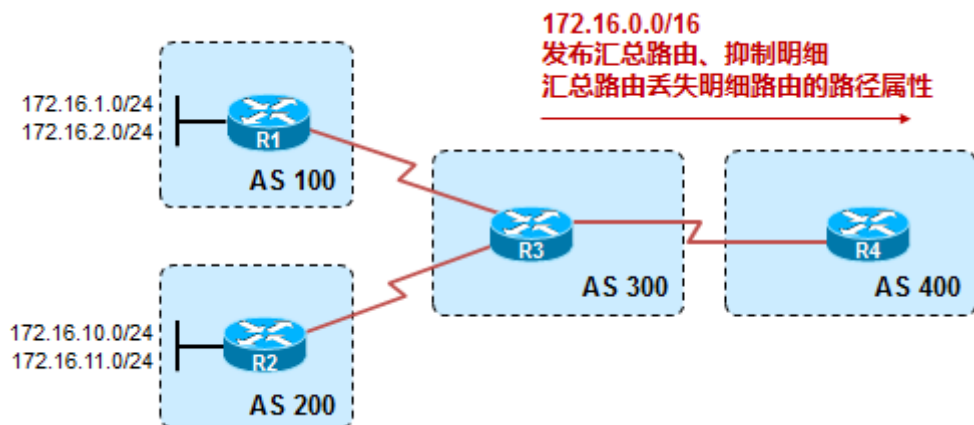
aggregate-address 命令是用于 BGP 手工汇总，以下是该命令所有子命令的详解

### 1. aggregate-address 汇总地址 summary-only

如果 aggregate-address 不加任何关键字，则明细也传递，汇总路由也传递。

加上参数 summary-only 则只传递汇总路由，明细路由被抑制。这种情况下产生的汇总路由，将会丢失底下明细路由的 AS\_PATH 属性，因此可能存在一定的隐患。





### R3 上 show ip bgp

Network	Next Hop	Metric	LocPrf	Weight	Path
*> 172.16.0.0	0.0.0.0			32768	i
s> 172.16.1.0/24	10.1.13.1		0	0	100 i
s> 172.16.2.0/24	10.1.13.1		0	0	100 i
s> 172.16.10.0/24	10.1.23.2		0	0	200 i
s> 172.16.11.0/24	10.1.23.2		0	0	200 i

本地产生的这条汇总路由 172.16.0.0/16，NH=0.0.0.0，weight=默认值 32768，origin=i。

明细路由标记为“s”，因此都被抑制了。

### R4#show ip bgp 172.16.0.0

BGP routing table entry for 172.16.0.0/16, version 4

Paths: (1 available, best #1, table Default-IP-Routing-Table)

Flag: 0x820

Not advertised to any peer

300, (**aggregated by 300 3.3.3.3**)

10.1.34.3 from 10.1.34.3 (3.3.3.3)

Origin IGP, metric 0, localpref 100, valid, external, **atomic-aggregate**, best

可以看到，路由带上了 atomic-aggregate 属性，用来告知下游邻居这是汇总路由且丢失了明细的路径属性。同时 aggregator 属性标识了汇总的地点（AS 及汇总路由器的 RouterID）。

报文抓取如下（R3 发给 R4 的 BGP update 包）：

```

UPDATE Message
  Marker: 16 bytes
  Length: 63 bytes
  Type: UPDATE Message (2)
  Unfeasible routes length: 0 bytes
  Total path attribute length: 37 bytes
  Path attributes
    + ORIGIN: IGP (4 bytes)
    + AS_PATH: 300 (7 bytes)
    + NEXT_HOP: 10.1.34.3 (7 bytes)
    + MULTI_EXIT_DISC: 0 (7 bytes)
    + ATOMIC_AGGREGATE (3 bytes)
      + Flags: 0x40 (well-known, Transitive, Complete)
      Type code: ATOMIC_AGGREGATE (6)
      Length: 0 bytes
    + AGGREGATOR: AS: 300 origin: 3.3.3.3 (9 bytes)
      + Flags: 0xc0 (Optional, Transitive, Complete)
      Type code: AGGREGATOR (7)
      Length: 6 bytes
      Aggregator AS: 300
      Aggregator origin: 3.3.3.3 (3.3.3.3)
    + Network layer reachability information: 3 bytes
      + 172.16.0.0/16
  
```

## 2. aggregate-address 汇总地址 summary-only as-set

汇总命令加上 as-set 关键字之后，产生的这条汇总路由就可以继承明细路由的某些路径属性，从而规避一些问题。as-set 继承明细属性的规则如下：

As-path: 将收到的所有明细路由的 as 号都放置在 {} 中，**计算 AS\_Path 长度时这些 AS 只被算为 1 个 AS**

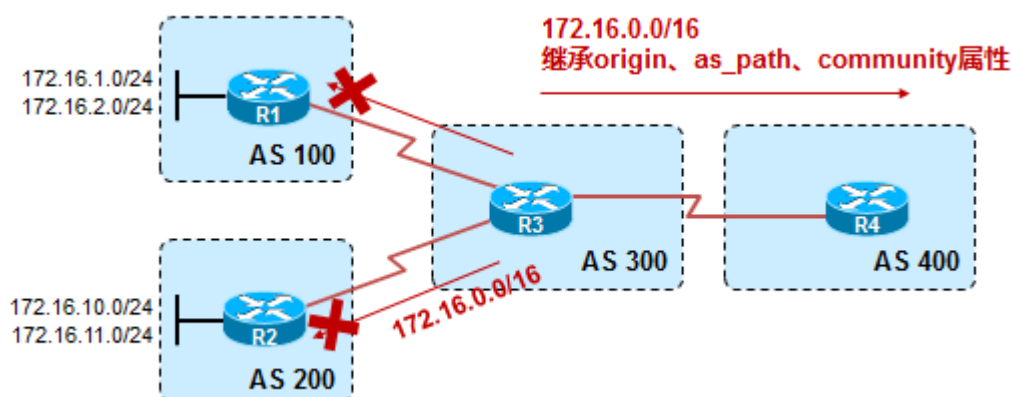
Origin: 继承最差的 origin 属性

Community: 继承所有明细路由的 community，形成一个列表

MED 不继承

LP 取明细路由中 LP 的最大值

NEXT\_HOP 汇总路由为 0.0.0.0（因为汇总路由为本地产生）



在 R3 上做汇总，使用 aggregate-address summary-only as-set，则 R3 上的 BGP 表：

Network	Next Hop	Metric	LocPrf	Weight	Path
*> 172.16.0.0	0.0.0.0		100	32768	<b>{100,200}</b> i
s> 172.16.1.0/24	10.1.13.1	0		0	100 i
s> 172.16.2.0/24	10.1.13.1	0		0	100 i
s> 172.16.10.0/24	10.1.23.2	0		0	200 i
s> 172.16.11.0/24	10.1.23.2	0		0	200 i

看到 R3 上产生的汇总路由，AS\_PATH 继承了明细的 AS\_PATH，以{100, 200}的形式呈现，这样就可以起到防止环路的作用，而不会由于丢失明细路由的 AS\_PATH 而带来隐患。注意这里{}内的 AS\_PATH 类型为 AS\_SET，是无序的 AS 列表。

#### R4 上 show ip bgp 172.16.0.0

BGP routing table entry for 172.16.0.0/16, version 3

Paths: (1 available, best #1, table Default-IP-Routing-Table)

Not advertised to any peer

**300 {100,200}, (aggregated by 300 3.3.3.3)**

10.1.34.3 from 10.1.34.3 (3.3.3.3)

Origin IGP, metric 0, localpref 100, valid, external, best

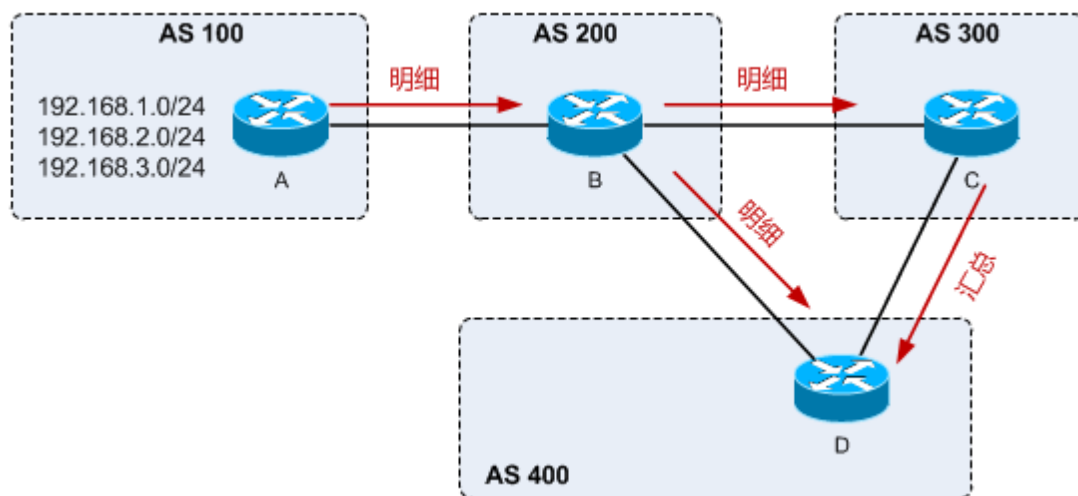
可以看到，由于我们配置汇总的时候，使用了 as-set 关键字使得汇总路由得以继承明细的部分路径属性，因此产生的汇总路由仍保留有 **aggregator** 属性，但是没有 atomic-agg 属性，显然这个属性在 as-set 关键字使用的情况下已经没有必要了。

**继续看看 R4 上抓包的结果 ( R3 发给 R4 的 BGP update 包 ):**

```

Path attributes
+ ORIGIN: IGP (4 bytes)
+ AS_PATH: 300 {100, 200} (13 bytes)
  + Flags: 0x40 (well-known, Transitive, complete)
  Type code: AS_PATH (2)
  Length: 10 bytes
  + AS path: 300 {100, 200}
    + AS path segment: 300
      Path segment type: AS_SEQUENCE (2)
      Path segment length: 1 AS
      Path segment value: 300
    + AS path segment: {100, 200}
      Path segment type: AS_SET (1)
      Path segment length: 2 ASs
      Path segment value: 100 200
+ NEXT_HOP: 10.1.34.3 (7 bytes)
+ MULTI_EXIT_DISC: 0 (7 bytes)
+ AGGREGATOR: AS: 300 origin: 3.3.3.3 (9 bytes)
  + Flags: 0xc0 (Optional, Transitive, complete)
  Type code: AGGREGATOR (7)
  Length: 6 bytes
  Aggregator AS: 300
  Aggregator origin: 3.3.3.3 (3.3.3.3)
  
```

### 补充说明 as\_set 关键字的作用：



A 将明细路由通告给 B，B 将明细路由通告给 C 及 D，C 通告汇总路由，这个时候汇总路由除了会通告给 D 外，也会被传递给 B，而这样一来可能会出问题，因此需要在 C 上 neighbor B 的 IP distribute-list x 来过滤掉汇总路由，否则会出现路由环路。

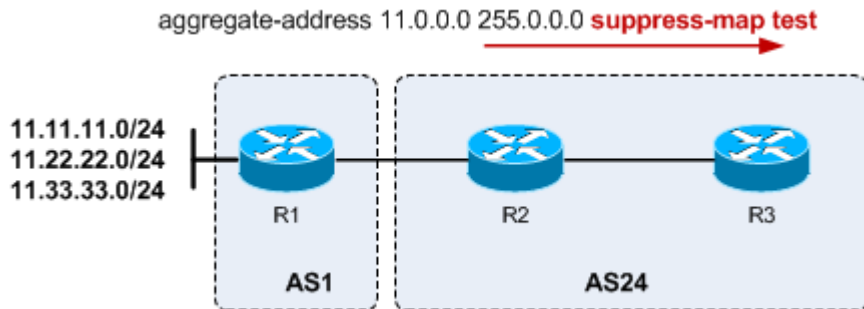
另外，汇总路由被传递给 D 后，D 会继续向 B 来传递，而由于汇总路由源于 C，因此到达 D 的 AS400 时，AS\_PATH 仅有 AS300、400，于是乎 B 路由器接受了该条汇总（因为没有看到自己的 AS 号出现在 AS\_PATH 中）。这样就可能出现环路。解决的办法是在 C 发布汇总路由的时候，设置 AS\_PATH 关键字，以还原 AS 属性，那么这时候，这条汇总路由就会同时宣告 AS\_SET，而 AS\_SET 是一个无序的 AS 列表，当中就有 AS200 的信息，这样 B 路由器收到这条汇总路由，发现 AS\_SET 中有自己的 AS 因此忽略该路由。

### 3. aggregate-address 汇总地址 **suppress-map xxx as-set**

用于宣告聚合及选定的明细路由（抑制特定的明细路由），后面跟上 route-map xxx，被 route-map 匹配（permit）的路由将被过滤，其他放行。

抑制列表虽然调用 route-map，但是 route-map 只能用于匹配，不能用于设置属性（不能用 set 命令）

#### 【实验 1】以下是几种情况



```
ip prefix-list 1 permit 11.11.11.0/24
route-map test per 10
    match ip add pre 1
// 干掉 11.0 放行除了 11.0 外的所有明细
```

```
ip prefix-list 1 permit 11.11.11.0/24
route-map test deny 10
    match ip add pre 1
// 等同于 route-map 不匹配 (permit) 任何条目，因此所有明细都放行
```

```
ip prefix-list 1 deny 11.11.11.0/24
route-map test permit 10
    match ip add pre 1
// 效果同上
```

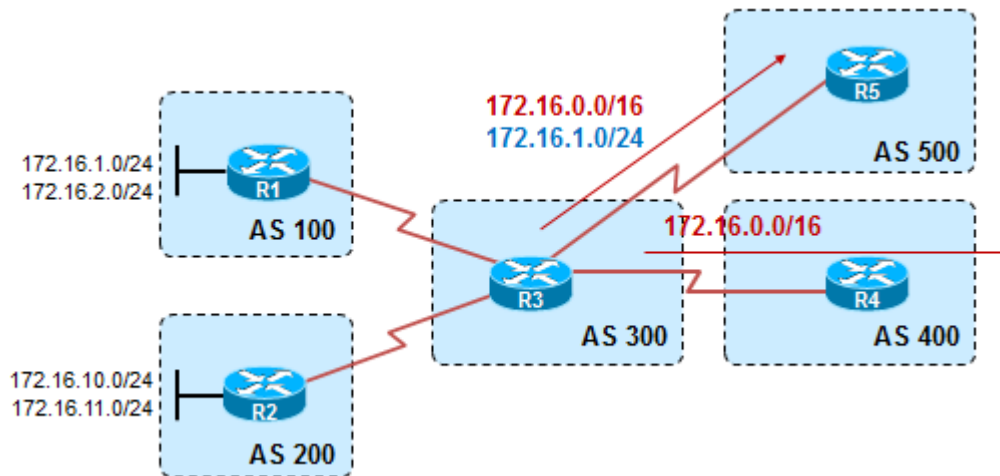
```
route-map test permit 10
// 明细全都不放行
```

```
route-map test deny 10
// 明细全放行
```

#### 【实验 2】邻居+Route-map 实现相同功能

注意用 route-map 实现的话，最后是隐含干掉 any 的，而 **suppress-map 则不同**

### 【实验 3】针对特定邻居取消明细路由抑制



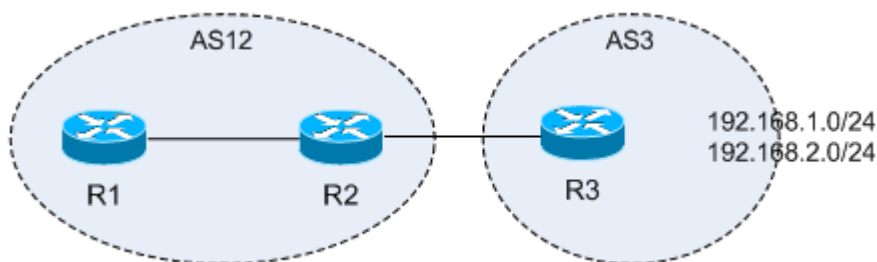
R3 的配置如下：

```
access-list 1 permit 172.16.1.0
route-map unsupp permit 10
  match ip address 11
router bgp 300
  neighbor 10.1.35.5 unsuppress-map unsupp
  aggregate-address 172.16.0.0 255.255.0.0 as-set summary-only
```

R3 上做汇总，抑制掉了所有明细，但是如果只想对特定的邻居放行部分明细，那么可以用 unsuppress-map

#### 4. aggregate-address 汇总地址 **attribute-map abc**

该命令可以更改汇聚路由的属性（注，仅仅对汇总路由产生作用，对明细不起效）。



R2 上对 192.168.0.0/16 进行汇总

```
router bgp 12
aggregate-address 192.168.0.0 255.255.0.0 attribute-map test // 关联 route-map test 对汇聚路由起效
Route-map test permit 10
  Set origin incom
  Set metric xx
```

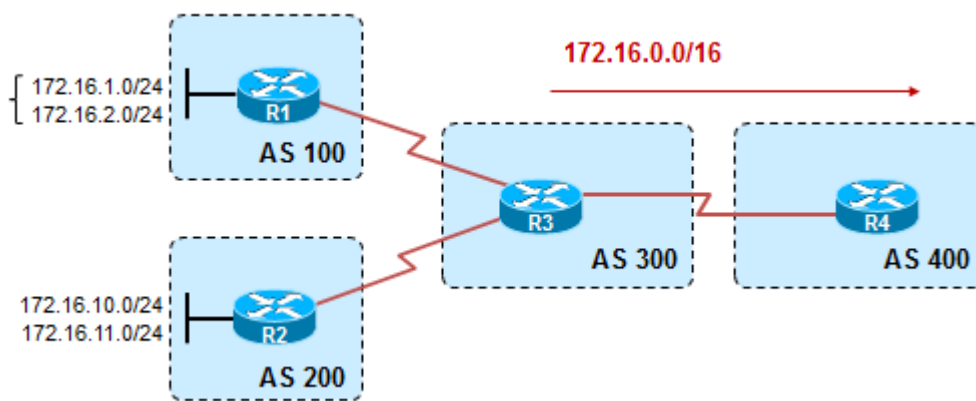
然而这个时候，R3 也会收到来自 R2 发布的汇总路由，这样就有可能形成路由环路，因此需过滤掉。

在 R2 上 BGP 进程增加配置：

```
neighbor 1.1.23.3 distribute-list 1 out
access-list 1 deny 192.168.0.0 0.0.255.255
```

## 5. aggregate-address 汇总地址 as-set advertise-map

advertise-map 与 summary-only 合用时，aggregate-address 的汇总地址下所有明细均被抑制，同时 advertise-map 匹配的条目中明细如果全都挂了，则汇总路由也消失，( 只要 advertise-map 匹配的明细有一条在，汇总就在 )；并且汇总路由仅继承 advertise-map 匹配明细路由的 BGP 路径属性



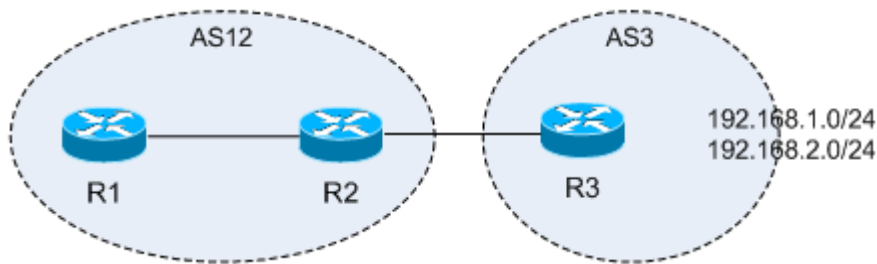
### 在 R3 上

```
ip prefix-list 1 permit 172.16.1.0/24
ip prefix-list 1 permit 172.16.2.0/24
route-map test permit 10
  match ip address prefix-list 1
router bgp 300
  aggregate-address 172.16.0.0 255.255.0.0 as-set summary-only advertise-map test
```

则 R1、R2 上所有 172 网段均被抑制；同时由于 route-map test 只匹配了 1.0 及 2.0 网段( AS100 中的 )，因此这条汇总路由，AS\_PATH 只继承 AS100，也就是 300 100；

同时，如果此时 172.16.1.0 及 172.16.2.0 全都 DOWN 掉了，则汇总路由也消失

### 另一个例子



R3 宣告 192.168.1.0、2.0，同时设置 1.0 的 community 属性为 no-adv

```
access-list 1 permit 192.168.1.0
```

```
route-map test permit 10
```

```
    match ip address 1
```

```
    set community no-advertise
```

```
route-map test permit 20
```

```
    set community none
```

```
router bgp 3
```

```
    neighbor 1.1.23.2 send-community
```

```
    neighbor 1.1.23.2 route-map test out
```

这时默认情况下，R2 能学习到 1.0、2.0，而 R1 只能学习到 2.0

当 R2

```
aggregate-address 192.168.0.0 255.255.0.0
```

之后，R1 能学习到汇总路由及 2.0

当 R2

```
aggregate-address 192.168.0.0 255.255.0.0 as-set
```

之后，R1 只能学习到 2.0，汇总路由被抑制掉了，原因是 1.0 携带了 no-adver 的 community，当汇总路由加了 as-set 关键字后，会继承它的 community，因此汇总路由也携带了 no-adver，这个时候

```
access-list 11 deny 192.168.1.0
```

```
access-list 11 permit any
```

```
route-map adv permit 10
```

```
    match ip address 11
```

```
router bgp 12
```

```
    aggregate-address 192.168.0.0 255.255.0.0 as-set advertise-map adv
```

// 即可忽略 1.0 的 community

注意，这里是在构建汇总路由时不考虑 advertise-map 中拒绝的路由，而单纯通过 adv-map 是无法过滤明细路由的



## 2.6 BGP Deaggregation

BGP Deaggregation，也称为 BGP 拆分。路由汇总我们都知道是什么概念，路由汇总的优势是非常明显的，可以减少路由表的条目从而优化网络，但是同时却也丢失了邻居传递来的路由的精确性，对于汇总前的明细路由我们就一无所知了。那么拆分可以理解为汇总的逆向动作，当我收到一条汇总路由的时候，可能基于某种目的，我希望从汇总路由中抽出特定的明细路由，以此来加强路由的颗粒度，当然这条明细是依赖汇总路由而存在的。

拆分可以通过使用条件注入（conditional injection）来完成，所谓的 conditional injection 指的就是，当特定的汇总路由存在时，我可以生成其下属的特定明细，这些明细路由将被注入到本地 BGP RIB（本地路由表也会加载明细路由信息），以便在本地 AS 中提供比汇总路由更详细的路由选择信息（更长的前缀）。

**Conditional inject 的配置如下（BGP 路由选择进程模式下）：**

```
bgp inject-map map1 exist-map map2 [copy attributes]
```

上述命令的意思是当 map2 所匹配的汇总路由正常时，在本地 BGP RIB 中注入 map1 中定义的明细路由。当汇总路由挂掉，这条明细也就跟着消失，这就是所谓的条件注入—conditional injection。下面我们在看来一下这两个 route-map 的详细内容，这些是需要格外注意的。

- **exist-map 使用的 route-map 最少具有以下两个 match 语句：**

```
match ip address prefix-list
```

上面这条 match 语句用来匹配汇总路由

```
match ip route-source
```

上面这条 match 语句用来匹配发送该汇总路由的邻居 IP。如果指定了 copy attributes 选项，那么被 inject 的明细路由会继承汇总路由的路径属性，否则明细将被当成本地生成的路由。

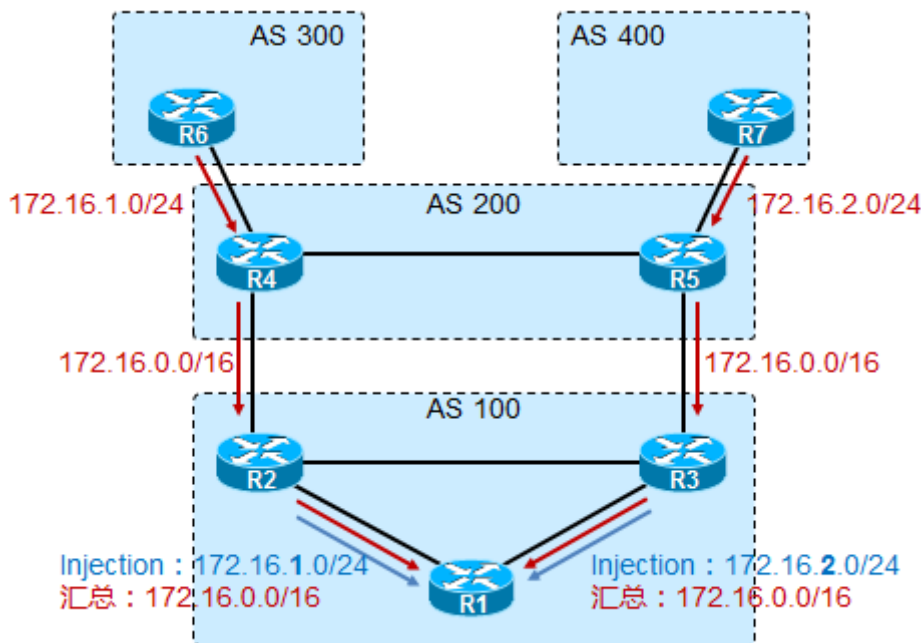
- **Inject-map 使用的 route-map 中**

```
Set ip address prefix-list
```

上面的这条 set 命令用来定义将被注入到本地 BGP RIB 的明细路由。被注入的前缀可以使用

```
Show ip bgp injected-path 来显示
```

下面，我们来看一个示例：



看看上面的拓扑，AS300 中有子网路由 172.16.1.0/24，AS400 中有路由 172.16.2.0/24。

这些子网路由在传递到 AS200 后，由 R4 及 R5 做路由汇总，汇总路由被传递给 AS100。这时候 R1 如果要去往 172.16.1.0 及 2.0 子网，可能就是走一侧，要么走 R2，要么走 R3，这当然不是最优的实现方式，我们希望看到 R1 去往 1.0 子网走 R2，去往 2.0 子网走 R3，那么实现的思路就是在 R2 及 R3 上部署 BGP deaggregation，由 R2 向 AS100 注入条件明细 172.16.1.0/24，R3 注入 172.16.2.0/24，同时，为了防止这两个 conditional 子网路由回流造成不可预估的影响，我们同时为这两条路由分配两个 community 值，1 个是 no-export，另一个是 100:200，其中 100 就不说了，AS200 表示，这条 conditional 路由是针对 AS200 的。

**R2 的配置如下：**

```
ip prefix-list huizong permit 172.16.0.0/16           //用来匹配汇总路由
ip prefix-list mingxi permit 172.16.1.0/24           //用来定义准备注入的条件前缀
ip prefix-list xiayitiao permit 10.1.24.4/32         //用来匹配传递给我汇总路由的 BGP 邻居，这里是 R4 的 IP
route-map RP_mingxi permit 10
  set community 100:200 no-export                    //100:200 表示这是针对 AS200 的
  set ip address prefix-list mingxi
route-map RP_huizong permit 10
  match ip address prefix-list huizong
  match ip route-source xiayitiao
router bgp 300
  bgp inject-map RP_mingxi exist-map RP_huizong copy-attributes
  neighbor 10.1.23.2 remote-as 200
```

R3 的配置大同小异。

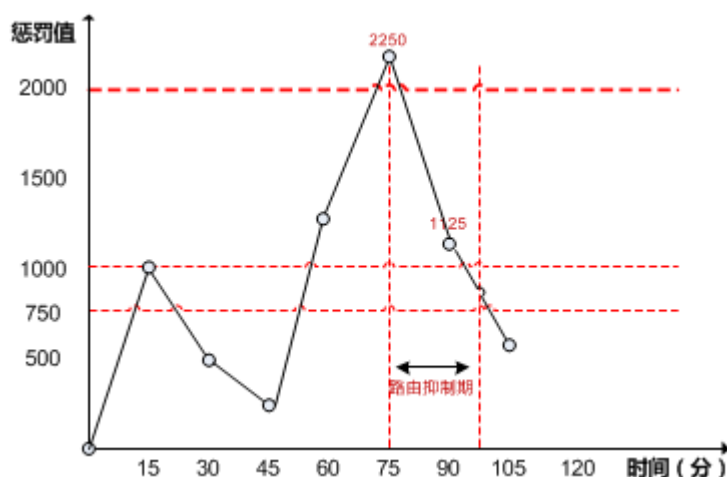
## 2.7 Route Dampening

当路由出现摆动（不断失效、恢复），就给它分配一个惩罚值，摆动越多，惩罚值越大并且不停地积累。

而同时惩罚值又以一定的速率降低，每一个半衰期结束，惩罚值变成原来的一半（如果路由不再翻动的话）

如果惩罚值超出了预先设置的门限--抑制界限：惩罚值达到这个门限，路由被抑制，即不发布，直到 N 个半衰期以后，惩罚值降低到另一个门限：重新使用门限（解除抑制的门限）时，才解除对路由的抑制。

**惩罚值：** 每次摆动增加 1000  
**抑制界限：** 2000  
**重新使用界限：** 750  
**半衰期：** 15 分钟  
**最大抑制时间：** 60 分钟（半衰期的 4 倍）



正常所有路由的惩罚值都是 0，惩罚值是瞬间加的

例如某条路由，出现翻动，则瞬间惩罚值到 1000，这时有个半衰期（15 分钟），15 分钟后（如果路由没有再翻动），惩罚值降低为一半 500，再经过 15 分钟变成 250，这时如果路由又发生一次抖动，则惩罚值再加 1000，变成 1250，在此之前，这条路由正常使用。如果又抖动了一次，则变成 2250，超过了阈值 2000，则这条路由将被抑制，不更新也不使用。15 分钟后，（如果不再翻动了）降低为一半 1125（但仍然被抑制），如此经历数个半衰期，直到降低为 750（重新使用界限），路由再次被启用。

```
bgp dampening           // 启动 bgp dampening，默认对全部 EBGp 路由有效，也可加 route-map
bgp dampening ?         // 修改默认参数
bgp dampening 半衰期 重新使用界限 抑制界限 最大抑制时间
```

show ip bgp

如果路由标记 d，则表示该路由被抑制，如果是 h，则表示路由有翻动的迹象

show ip bgp flap // 查看路由翻动情况

Show ip bgp dampening // 查看哪些路由被抑制了

Dampening 只对 EBGp 路由生效，对 IBGP 路由无效

### 3 BGP 路径属性

路径属性的分类：

公认属性 ( Well-known )	公认必遵 ( Well-known mandatory )	BGP 必须都能识别 ,且在更新消息必须包含	Origin AS-Path Next hop
	公认自决 ( Well-known discretionary )	BGP 必须都能识别 ,更新消息可包含可不包含	Local-Preference ATOMIC_Aggregate
可选属性 ( Optional )	可选传递 ( Optional transitive )	可以不支持该属性，但即使不支持，也应当接受包含该属性的路由并传递给其他邻居	Community Aggregator
	可选非传递 ( Optional non-transitive )	可以不支持该属性，不识别的 BGP 进程忽略包含这个属性的更新消息，并且不传递给其他 BGP 邻居	MED Originator_ID Cluster_list Weight

#### 3.1 Origin

公认必遵属性，明确了路由更新的来源，三种途径

- IGP i 通过 BGP network，也就是起源于 IGP，因为 BGP network 必须保证该网络在路由表中
- EGP e 是由 EGP 这种早期的协议重发布而来
- Incomplete ? 从其他渠道学习到的（确认该路由来源的信息不完全），重发布的路由 origin 都是这个标记

**优选原则是 IGP > EGP > Incomplete**

R2#sh ip bgp

BGP table version is 10, local router ID is 2.2.2.2

Status codes: s suppressed, d damped, h history, \* valid, > best, i - internal, r RIB-failure, S Stale

Origin codes: i - IGP, e - EGP, ? - incomplete

Network	Next Hop	Metric	LocPrf	Weight	Path
---------	----------	--------	--------	--------	------

r > i1.1.1.0/24	1.1.1.1	0	100	0	i
* > i11.11.11.0/24	1.1.1.1	0	100	0	?

## 3.2 AS\_Path

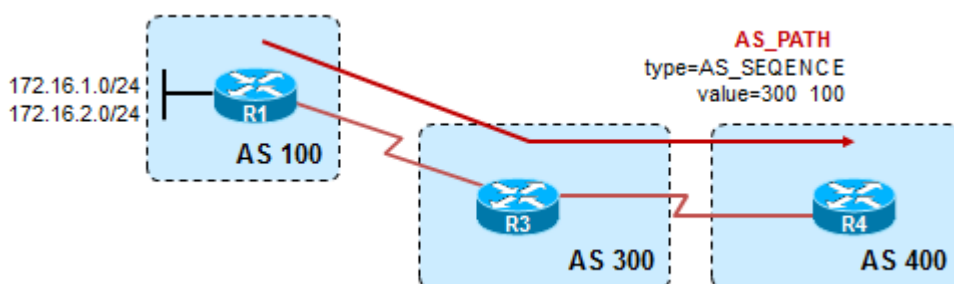
公认必遵属性，描述到达目标网络所要经过的 AS 号序列。最重要的作用是防环，如果 BGP speaker 发现自己的 AS 号出现在接收到的 BGP 路由更新的 AS\_PATH 中，那么说明可能有路由环路，则忽略该路由更新。另一个重要的作用，是在 BGP 选路过则中，作为 AS 跳数的丈量，当然，AS\_PATH 中包含的 AS 个数越少，表示距离目的地更近。

仅当 update 消息被发送给其他的 AS (EBGP peer) 时，BGP 路由器才会将其 AS 号追加在 AS\_PATH 中。这句话也隐含了另一个意思，那就是如果要修改 AS\_PATH 属性，则必须在 AS 边界路由器上执行策略，对 IBGP 邻居去执行修改 AS\_PATH 的策略是无效的。

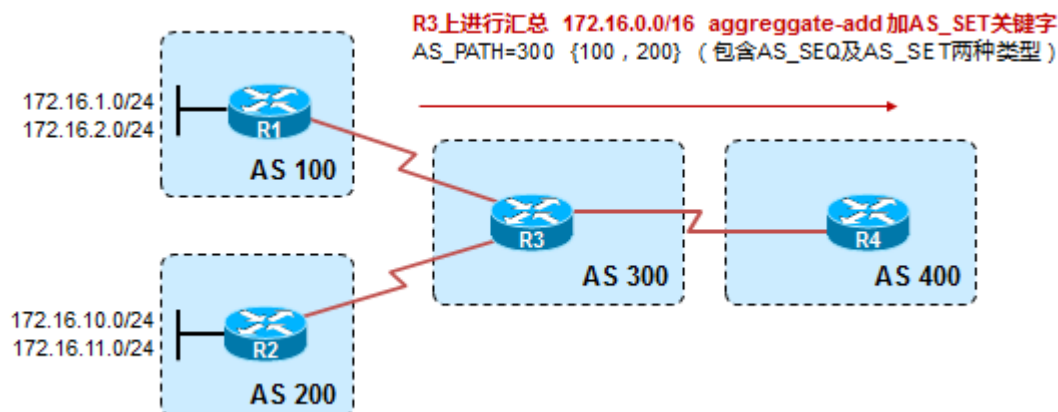
### 1. AS\_PATH 路径属性的四种类型

- **AS\_SET** : 一个去往特定目的地所经路径上的无序 AS 号列表
- **AS\_SEQUENCE** : 一个有序的 AS 号列表
- **AS\_CONFED\_SEQUENCE** 一个去往特定目的地所经路径上的有序 AS 号列表，其用法与 AS\_SEQUENCE 完全一样，区别在于该列表中的 AS 号属于本地联邦中的 AS
- **AS\_CONFED\_SET** 一个去往特定目的地所经路径上的无序 AS 号列表，去用方法与 AS\_SET 完全一样，区别在于列表中的 AS 号属于本地联邦中的 AS

以上四种类型是通过 AS\_PATH 属性中的类型代码进行区分。关于这两个联邦特有的 AS\_PATH 类型，详细内容请见本文档“联邦”一小节，对于前两种 AS\_PATH 怎么理解呢？



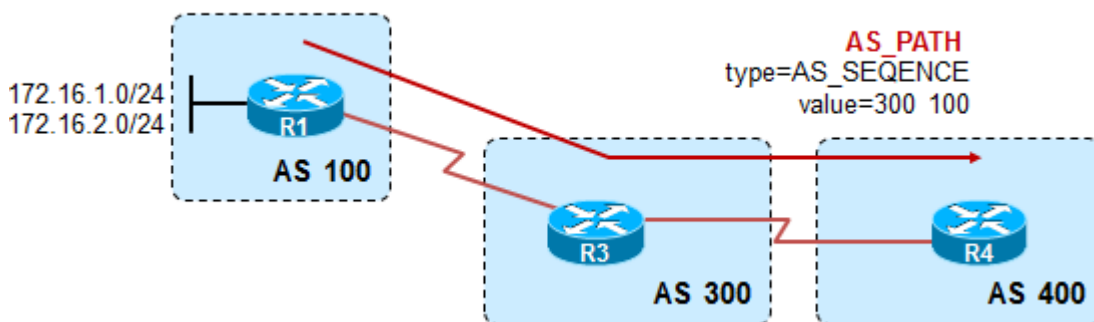
AS\_Sequence 很好理解，上图在不做任何策略的情况下，传递到 R4 的 BGP 路由携带的 AS\_PATH 的类型即为 AS\_Sequence，这是个有序的 AS 号列表，那么当 R4 收到路由更新的时候，AS\_PATH 为 300,100，R4 就知道要去目的地，需经 AS300 再到 AS100，才可达目的。



再看上面这张图，在 R3 上我们做了手工路由汇总，这条汇总路由由 R3 产生并且传递给了 R4，那么这时候这条汇总路由的 AS\_PATH 为 300，丢失了其下的明细路由的 AS\_PATH 属性，那么如果 R4 与 R1 或 R2 之间如果有 EBGP 的连接，那么就出现了路由环路，R1、R2 都会收到这条汇总路由并且接受路由更新。这是因为这条汇总路由并未携带任何关于 AS100 及 AS200 的信息。

所以我们其实还是需要在汇总路由上保留其明细的 AS\_PATH，而至于保留下来的明细路由的 AS\_PATH 中 AS 号是什么顺序已经不重要了，因为我们出于防环的目的的话，只要 AS 号就够了。所以 R3 产生的这条汇总路由其实携带了两个 AS\_PATH 类型，一是 AS\_Seq，值为 300，另一个类型为 AS\_set，包含{100,200}两个 AS 号，并且是无序的。于是乎 R4 上收到的 AS\_PATH 综合起来就是 300 {100,200}。值得注意的是，这个括号里的 AS 号，在进行 AS\_PATH 计算的时候，只当一跳来算，关于这点的验证，请看选路规则章节。

## 2. AS\_PATH 类型 2 : AS\_SEQUENCE 详解



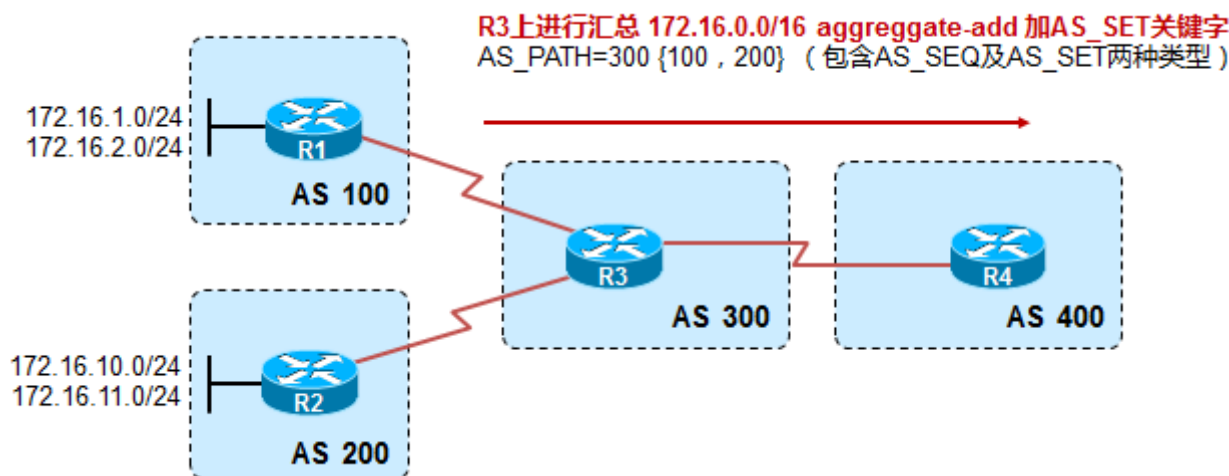
上图仅做基本的 BGP 配置，在 R1 上发布 1.0 及 2.0，不做汇总，R4 收到的 1.0 及 2.0 BGP 路由，BGP 更新包如下，可以留意到 AS 的长度为 2，即“300, 100”，注意，这是个有序的 AS 列表。

```

Border Gateway Protocol
├─ UPDATE Message BGP update包
│   Marker: 16 bytes
│   Length: 51 bytes
│   Type: UPDATE Message (2)
│   Unfeasible routes length: 0 bytes
│   Total path attribute length: 20 bytes
│   └─ Path attributes BGP路径属性
│       └─ ORIGIN: IGP (4 bytes)
│       └─ AS_PATH: 300 100 (9 bytes) AS_PATH属性开始
│           └─ Flags: 0x40 (well-known, Transitive, Complete)
│               Type code: AS_PATH (2) BGP路径属性类型字段
│               Length: 6 bytes
│               └─ AS path: 300 100
│                   └─ AS path segment: 300 100
│                       Path segment type: AS_SEQUENCE (2) AS_PATH类型字段
│                       Path segment length: 2 ASs 该AS_PATH含2个AS
│                       Path segment value: 300 100 值为300 100
│       └─ NEXT_HOP: 10.1.34.3 (7 bytes)
│       └─ Network layer reachability information: 8 bytes
│           └─ 172.16.2.0/24 路由前缀
│           └─ 172.16.1.0/24

```

### 3. AS\_PATH 类型 1：AS\_SET 详解



R3 上对 R1、R2 过来的明细进行汇总 (汇总命令加上 AS\_SET 关键字), 这样一来 R3 产生的这条 BGP 汇总路由就继承了明细的 AS\_PATH 属性, 在 R4 上抓包会发现汇总路由的 AS\_PATH=300 {100,200}: 再看看抓包的结果, 发现该条汇总路由的 AS\_PATH 属性包含两部分 (segment), 值为 300 的这个 segment 为 AS\_SEQ 类型, 值为{100, 200}的这个 segment 为 AS\_SET 类型。所以 AS\_SEQ 这个 AS\_PATH 用来标识这条汇总路由的起源地, 它是有序的 AS 列表, 另外 AS\_SET 这个 AS\_PATH 类型用来标识汇总前的明细的 AS 列表, 它是无序的, 亦可用来防环。当使用了 as-set 关键字的手工汇总命令, 由于继承了明细的 AS\_PATH 等属性, 汇总路由的路径属性已经很丰富和完备了, 因此这种情况下就不需要 atomic-aggregate 这个属性了, 我们看下面的抓包结果, 有 aggregator, 但是没有 atomic-agg。



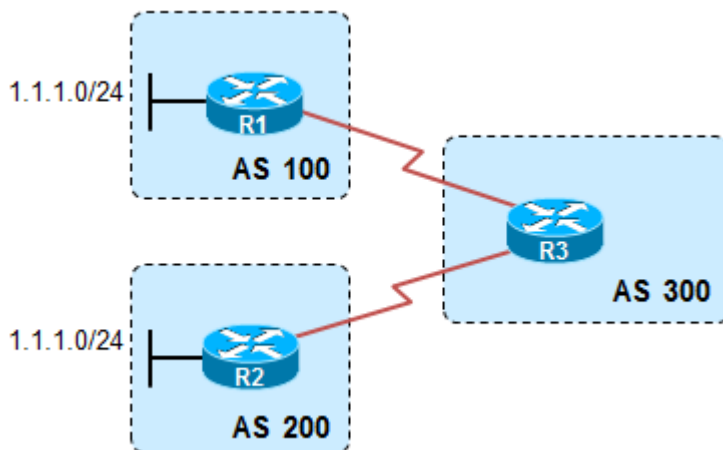
```

Border Gateway Protocol
├─ UPDATE Message
│   Marker: 16 bytes
│   Length: 66 bytes
│   Type: UPDATE Message (2)
│   Unfeasible routes length: 0 bytes
│   Total path attribute length: 40 bytes
├─ Path attributes
│   └─ ORIGIN: IGP (4 bytes)
│   └─ AS_PATH: 300 {100, 200} (13 bytes)
│       └─ Flags: 0x40 (well-known, Transitive, Complete)
│           Type code: AS_PATH (2)
│           Length: 10 bytes
│       └─ AS path: 300 {100, 200}
│           └─ AS path segment: 300
│               Path segment type: AS_SEQUENCE (2)
│               Path segment length: 1 AS
│               Path segment value: 300
│           └─ AS path segment: {100, 200}
│               Path segment type: AS_SET (1)
│               Path segment length: 2 ASs
│               Path segment value: 100 200
│       └─ NEXT_HOP: 10.1.34.3 (7 bytes)
│       └─ MULTI_EXIT_DISC: 0 (7 bytes)
│       └─ AGGREGATOR: AS: 300 origin: 10.1.34.3 (9 bytes)
└─ Network layer reachability information: 3 bytes
    └─ 172.16.0.0/16
    
```

AS\_SEQ

AS\_SET, 是无序的

#### 4. 使用 route-map 修改 AS\_PATH



可以通过策略控制 AS\_PATH 从而来影响路由选路, 例如上图, 如果希望 R3 到达 1.1.1.0 网络优先走 R1, 那么可以在 R2 上做策略, 如下:

```

ip prefix-list 1 permit 1.1.1.0/24
route-map test
  match ip add pre 1
  
```



```
set as-path prepend 6666
neighbor 10.1.13.3 route-map test out
```

与此一来，R3 上从 R2 学来的 1.1.1.0 BGP 路由，AS\_PATH 就由原先的 “200 i” 变成 “200 6666 i”，当然这不太建议，因为 6666 这个 AS 并不存在，虽然加长了 AS\_PATH 但是带入了一个压根不存在 AS 这不是扯淡么，我们可以 set as-path prepend 200 200 200（重复本 AS），如此来增长路由的 AS\_PATH 从而实现路由策略并且规避不必要的麻烦。另外，如果同样的配置，在 R3 上对 R2 neighbor 10.1.23.2 route-map test，in 那么 R3 上从 R2 学到的关于 1.1.1.0 的路由 AS\_PATH 就会变成 “6666 200 i”，这是因为 R3 收到 1.1.1.0 时，已经是 “200 i” 了，因此做 prepend 只能在 200 i 前面插入，请注意与前面的区别。

**注意：使用策略修改 AS\_PATH 只能在 BGP 路由器上对其 EBGP 邻居执行，那是因为 AS\_PATH 只有在 AS 边界才会发生改变，因此如果对 IBGP 邻居做 AS\_PATH 的修改策略，那就扯淡了。**

## 5. AS\_PATH 几个命令

```
Neighbor 路由来源邻居 allowas-in 几个我的 as 号
```

一般 BGP 路由器是不收包含自己 AS 号的路由信息的，但是可以 neighbor 路由来源邻居 allowas-in 几个我的 as 号，来破例。

```
bgp bestpath as-path ignore
```

让 CISCO 路由器在其决策过程中忽略 AS-path 属性

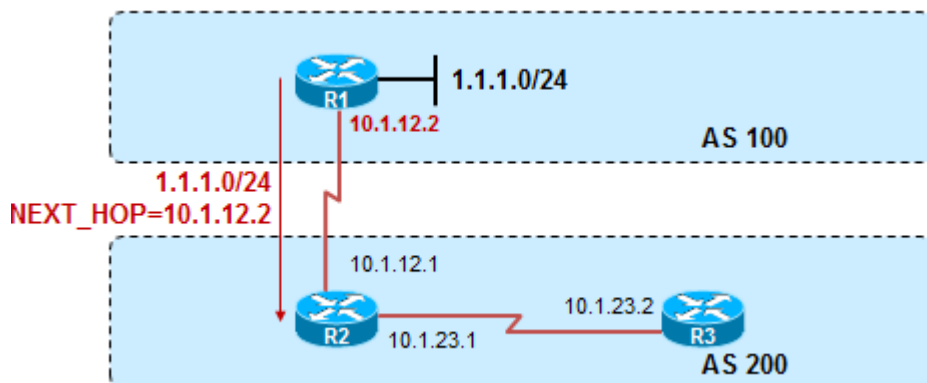
## 3.3 NEXT-HOP

**公认必遵属性，描述了到目的地的下一跳路由器。**

**BGP next-hop 的规则如下：**

### 1. 如果 BGP 路由传递自 EBGP peer

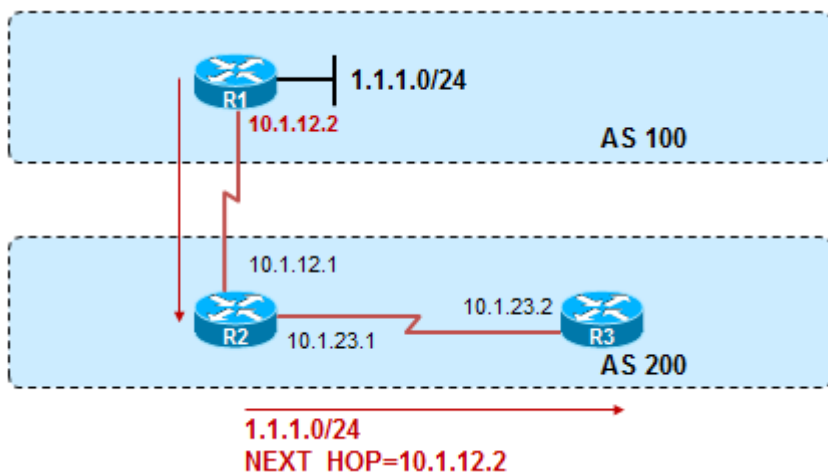
那么这条 BGP 路由的 NH 就是通告者的接口 IP。EBGP 邻居一般使用对方的直连接口 IP 互相指 neighbor，但如果是采用 LOOPBACK 接口建 EBGP 邻居，则 NH 就是 EBGP 邻居的更新源地址（当然一般 EBGP 邻居不使用 LOOPBACK 口建邻居）。



【补充】 若同 AS 内的 BGP 路由器 A 引入了某路由，A 路由器将该路由传递给了其 IBGP 邻居 B，又被 B 传递给了其 IBGP 邻居 C（做了反射器），则在 C 上，该条路由 NEXT\_HOP 仍然为 A 路由器（的更新源 IP）

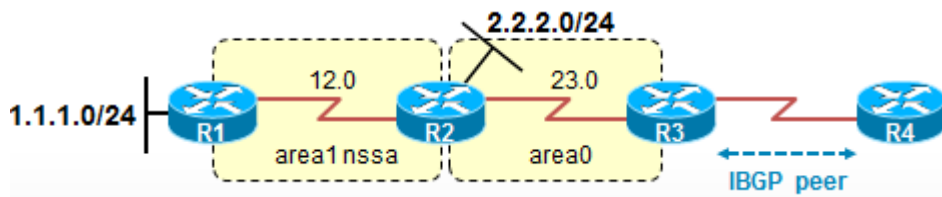
## 2. 如果路由传递自 IBGP 邻居，并描述的是 AS 外的目的地

从外部 AS 传递进来的路由，NH 为通告该路由的 EBGP peer，并且这个 NH 会跟随着路由在 AS 内传递而不发生改变（除非做了策略），始终指向的是下一个 AS（通告该路由的 EBGP peer 接口 IP）。



## 3. 如果路由传递自 IBGP 邻居，并由 AS 内 BGP 路由器引入

- 如果是通过 aggregate-address 命令被注入的，那么 next-hop 等于执行汇总路由器（的更新源 IP）
- 如果是通过 network 或重发布注入的，那么在注入前该前缀的 IGP 下一跳将成为 BGP 的 next-hop



R1、R2、R3 跑 OSPF，R3 上的路由表如下：

```
O E2 1.1.1.0 [110/20] via 10.1.23.2, 00:06:40, Serial0/0
```

```
O    2.2.2.2 [110/65] via 10.1.23.2, 00:06:40, Serial0/0
O IA  10.1.12.0 [110/128] via 10.1.23.2, 00:06:40, Serial0/0
C    10.1.23.0 is directly connected, Serial0/0
C    10.1.34.0 is directly connected, Serial0/1
```

R3 与 R4 建立 IBGP 邻居关系，并且 R3 上，将 OSPF 路由重发布进 BGP

```
redistribute ospf 1 match internal external 1 external 2
```

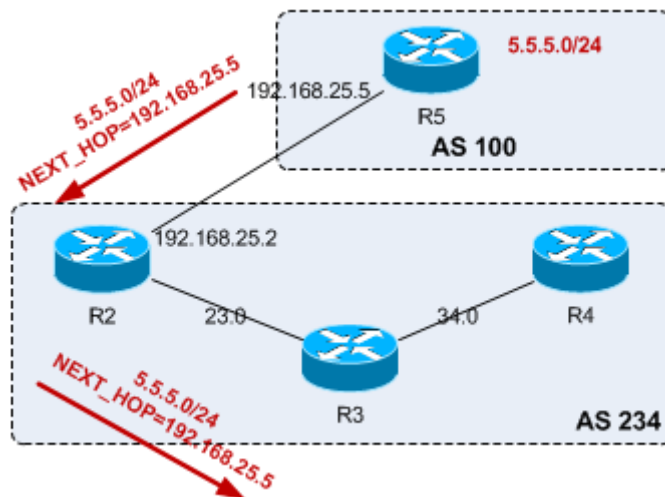
然后在 R4 上观察 BGP 表：

Network	Next Hop	Metric	LocPrf	Weight	Path
*>i1.1.1.0/24	10.1.23.2	20	100	0 ?	
*>i2.2.2.2/32	10.1.23.2	65	100	0 ?	
*>i10.1.12.0/24	10.1.23.2	128	100	0 ?	
*>i10.1.23.0/24	10.1.34.3	0	100	0 ?	

我们发现，无论是 O、O IA 还是带有 FA 的 OE2 路由，被重发布进 BGP 后，这些 BGP 路由的 Next-hop 均为它们在 OSPF 中的 IGP 下一跳。同时本地激活了 OSPF 了直连接口所在网段也会被注入 BGP，注入后 Next\_Hop 为本路由器(BGP 更新源 IP)。

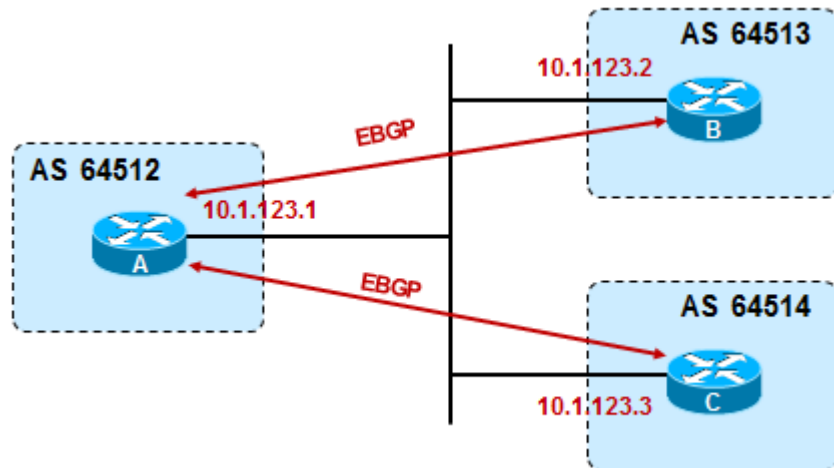
- 如果本地的 BGP 宣告者成了下一跳地址，那么在本地 BGP RIB 中的下一跳字段就是 0.0.0.0

#### 4. 通过 next-hop-self 可以变更 next-hop 属性



上图 R5 更新给 EBGP 邻居 R2 BGP 路由 5.5.5.0 时，NEXT\_HOP=192.168.25.5，该属性将一直跟随本条 BGP 路由在 AS234 中传递，这时对于 R4、R3 来说，并不知道如何前往 192.168.25.5，因此 5.5.5.0 的路由无法正常装入路由表。解决方法之一是 R2 在 IGP 路由中注入 25.0 这条外部路由，另一个方法是 R2 对其 IBGP 邻居使用 next-hop-self，修改这条前缀的 NEXT\_HOP 属性。

#### NEXT\_HOP on shared Media (在共享介质上的运作)



RouterB 将路由 100.100.100.0/24 传递给 A，NEXT\_HOP 为 10.1.123.2；

RouterA 将路由 100.100.100.0/24 传递给 C，此时 NEXT\_HOP 保持不变；

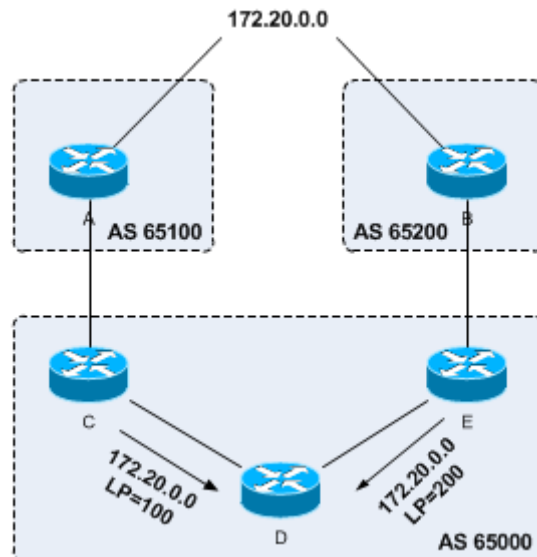
如果路由器收到某条 BGP 路由，该路由的 NEXT\_HOP 地址值与该路由器的接口 IP（更新源）同属一个网段，那么该条路由的 NEXT\_HOP 地址将保持不变并传递给它的（这个相同网段上的）BGP 邻居，这个其实是一种优化机制，但是这种机制在 NBMA 环境中是否有问题呢？

### NEXT\_HOP on NBMA network

仍然看上图，中间的网络如果不是广播多路访问网络，而是一个帧中继网络，那么就要注意，C 收到的路由，NEXT\_HOP 为 10.1.123.2，那么如果 C 路由器上没有到该 IP 的 PVC，就会出问题，所以这点要考虑进去。

## 3.4 LOCAL-PREFERENCE

**公认自决属性。** LP 就是本地优先级，只能在 AS 内部，在 IBGP 对等体之间传递，而不会传递给其他 EBGP 邻居



当一个 AS 收到一个去往同一目的地的、但经过两个 AS 的路由，则根据两条路由的 local-pref 值来决定

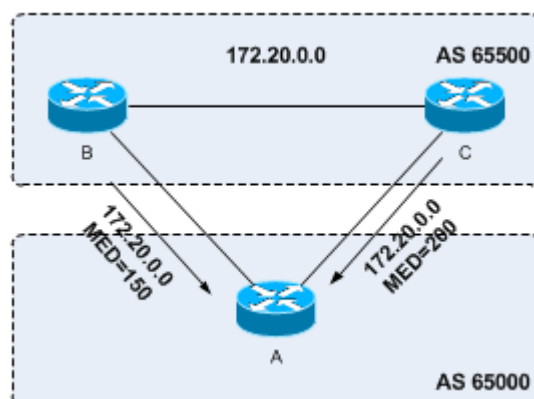
Local-pref 只影响离开 AS 的业务量（**选大的 LP 值**）

一般做在 AS 边界，告诉自己 AS 内部从自己走的 local-pref 值

1. 只能在 IBGP Peer 之间传递（除非做了策略否则 LP 值在 AS 内的 IBGP 邻居间传递不会丢失），不能在 EBGP Peer 之间传递，如果在 EBGP Peer 之间收到的路由的路径属性中携带了 Local Preference，则会触发 Notification 报文，造成会话中断；但是可以再 AS 边界路由器上使用 IN 方向的策略。
2. `bgp default local-preference 500` //修改默认 LP 值
3. BGP 路由器在向其 EBGP 邻居发送路由更新时，不能携带 LP 属性，对方收到该 EBGP 路由的 LP 值为空（连 LP 这个字段都没有），但是它会在本地为这条路由赋一个默认值，也就是 100，然后再传递给自己的 IBGP
4. 本地 network 及重发布的路由，LP 默认 100，并能在 AS 内向其他 IBGP 邻居传输，传输过程中除非部署策略，否则 LP 不变

## 3.5 MED

**可选非传递，一般用于 AS 之间来影响路由。**



在 B 和 C 上对 A 做策略，即在 AS 边界对外部 AS 做 MED，使得 A 选择**更低**的 MED 值。

CISCO 默认 MED 为 0

默认情况下，只比较来自同一邻居 AS 的 BGP 路由的 MED 值，就是说如果同一个目的地的两条路由来自不同的 AS，则不进行 MED 值的比较。换句话说，默认对于多条路径，只有在 AS\_SEQUENCE 中的第一个 AS 相同的情况下，才比较 MED；任何打头的 AS\_CONFED\_SEQUENCE 都将被忽略。MED 只是在直接相连的自治系统间影响业务量，而不会跨 AS 传递

### 1. MED 设置方法:

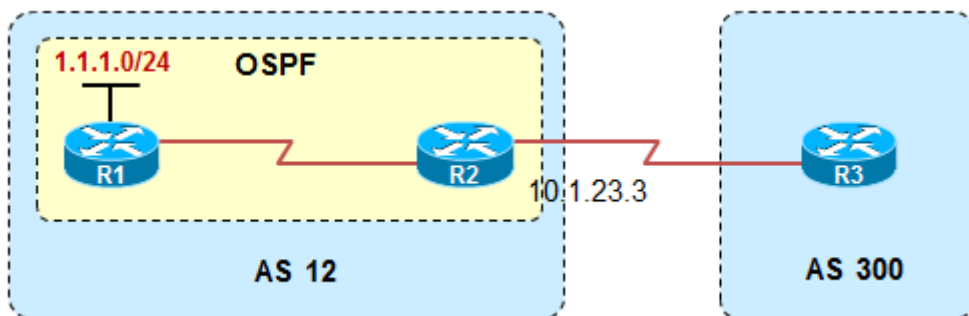
- 1) 将 IGP 路由引入 BGP 时关联 Route-map 进行设置
- 2) 对 BGP Peer 应用 IN/OUT 方向的 Route-map 进行设置
- 3) 非 Route-map(自动)方式:
  - 使用 network 或 redistribute 方式将 IGP 路由引入 BGP 时,MED 将继承 IGP 路由的 Metric(直联路由及静态路由的 Metric 为 0)
  - 使用 aggregate-address 方式引入路由，则 MED 为空

### 2. 比较原则及配置注意事项

- 1) 本地在将一条 BGP 路由通告给 EBGW Peer 时，是否携带 MED 值,需要根据以下条件进行判断(不对 EBGW Peer 使用 Route-map 的情况下)：
  - 如果该 BGP 路由是本地始发(network 或 redistribute)的,则携带 MED 值发送给 EBGW Peer (如果 MED 为空,则设置为 0)
  - 如果该 BGP 路由是从其他 BGP Peer 学习过来的,那么将该路由通告给 EBGW Peer 时不携带 MED (为空), 换句话说，就是 MED 不会被传出本 AS
- 2) 本地在将一条 BGP 路由通告给 IBGP Peer 时，一定会携带 MED 值
  - 如果接收或产生的路由的 MED 为空,那么在向 IBGP Peer 通告时,将 MED 设置为 0

**总结 1) 2) 两点就是 MED 在 IBGP 之间传递没有问题（不会丢失），但是在 EBGW 之间传递要看路由是否起源于自己。**

### 3. MED 继承 IGP 的 Metric



R1、R2 运行 OSPF，R2 学习到 1.1.1.0/24 的 OSPF 路由（metric 为 65）

在 R2 上 BGP 路由选择进程里 network 1.1.1.0 mask 255.255.255.0

则 R2 本地引入 1.1.1.0 的 BGP 路由，并继承从 OSPF 学习到的 1.1.1.0 的 metric 2，那么在 R3 上

Network	Next Hop	Metric	LocPrf	Weight	Path
*> 1.1.1.1/32	10.1.23.2	<b>65</b>		0	12 i

所以在将 IGP 路由 network 进 BGP 时，MED 值继承 IGP 协议中的 metric 值

### 总结如下：

network 本地从 IGP 路由协议学习到的路由进 BGP，MED 值继承 IGP 协议中的 metric

network 本地直连接口的网段进 BGP，MED 值为 0；network 本地静态路由进 BGP，MED 值为 0

redistribute 本地从 IGP 路由协议学习到的路由进 BGP，MED 值继承 IGP 协议中的 metric

redistribute 本地直连接口网段进 BGP，MED 值为 0；redistribute 本地静态路由进 BGP，MED 值为 0

## 4. 其他配置命令

### 1) bgp always-compare-med

默认情况下只比较来自同一 AS 的路由的 med，如果想对于所有路径都比较 MED，则要配置此命令。  
同时如果使用了这个选项，那么建议在整个 AS 中都这么做，以避免路由选择环路。

### 2) bgp bestpath med missing-as-worst

如果某路由 MED 属性丢失一般的做法是给 MED 默认设置为 0，但是如果配置了这条命令，则如果收到没有 MED 值的路由，将该路由的 MED 设置为最大值 4294,9672,95

### 3) set metric-type internal

用在 route-map 中，当匹配某些条目，且使用该 set 命令后，将 route-map 应用在某个邻居（如 out 方向），则向给邻居更新这些路由时，BGP 的 MED 属性将会继承这些路由在本地 IGP 的 metric 值

### 4) bgp bestpath med confed

配置这条命令，在选路时，路由器只比较所有带有 AS\_CONFED\_SEQ 属性的路由条目，此命令用于联邦路由器，同时 weight 和 LP 比 MED 具有更高的优先级

### 5) bgp deterministic-med

用于确保来自相同的 AS 的不同对等体通告的路由**先**进行 MED 比较

此命令和 bgp always-compare-med 的关系如下：

如有相同前缀的路由如下（默认按路由接受顺序的逆序排序）

路由 1：AS_PATH：500	MED：150	external	RouterID：172.16.13.1
路由 2：AS_PATH：100	MED：200	external	RouterID：1.1.1.1
路由 3：AS_PATH：500	MED：100	Internal	RouterID：172.16.8.1

- 如果两条命令都关闭：



则按缺省的从上往下的顺序进行比较

则 1、2 先比较，2 优选（因为 2 具有更小的 RID）；然后 2 与 3 比较，2 为 external，所以 2 优选

- 如果打开 always-compare-med

则 1、2 先比较，1 优选（因为 1 具有更小的 MED）；由于 1、3 在一个相同的 AS 中，所以再次比 MED，选 3

- 如果 deterministic-med 打开，路由前缀的信息按 AS 重新分组

路由 1：AS\_PATH：100      MED：200      external      RouterID：1.1.1.1

路由 2：AS\_PATH：500      MED：100      Internal      RouterID：172.16.8.1

路由 3：AS\_PATH：500      MED：150      external      RouterID：172.16.13.1

则 2、3 同 AS 内先比较，2 的 MED 小所以优选；2 与 1 再比较，1 为 external 路由所以被优选（MED 不比较）

- 如果都打开

则 2、3 同 AS 内先比较，2 的 MED 小所以优选；2 与 1 再比较，由于开了 always-compare-med 所以比 MED，2 的 MED 小，所以 2 为最佳路径

## 6) Default-metric x

修改 MED 值的缺省值

对从其他动态路由选择域重发布进来的路由生效，对直连路由不生效（仍然为 0）

## 3.6 COMMUNITY

### 1. 基本概念

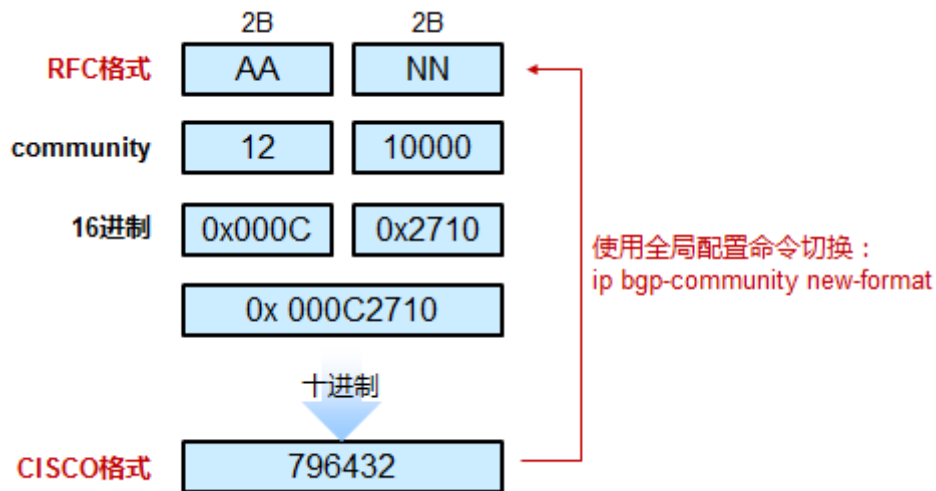
可选传递，用于简化路由策略的执行

可以将某些路由分配一个特定的 COMMUNITY 属性，之后就可以基于 COMMUNITY 值而不是每条路由进行 BGP 属性的设置了。COMMUNITY 属性对邻居起作用，在设置后，同时需要向邻居发送（SEND）

COMMUNITY 属性是一组 4 个 8 位组的数值，RFC1997 规定前 2B 表示 AS 号，后 2B 表示基于管理目的设置的标示符，格式为 AA:NN，而 CISCO 默认显示格式为 NN:AA，可使用全局配置命令 ip bgpcommunity new-format 将 CISCO 默认格式改为 RFC 格式。

例如将 AS12 的某条路由由 COMM 值改为 10000，RFC 采用十六进制表示 COMMUNITY 属性，而 CISCO 采用十进制。RFC 格式为 12:10000，十六进制为 0x000C2710，再转换为十进制 796432





```

COMMUNITIES: 12:10000 (7 bytes)
  Flags: 0xc0 (Optional, Transitive, Complete)
  Type code: COMMUNITIES (8)
  Length: 4 bytes
  Communities: 12:10000
    Community: 12:10000
      Community AS: 12
      Community value: 10000
  
```

0-65535 ( 0x00000000-0x0000FFFF ) 及 4294901760-4294967295 02710046  
( 0xFFFF0000-0xFFFFFFFF ) 是保留的团体值在这些值之外还定义了几个众所周知的值如 no-export、local-as 等，具体值请见 RFC。

## 2. 在 route-map 中 set community

route-map test permit 10

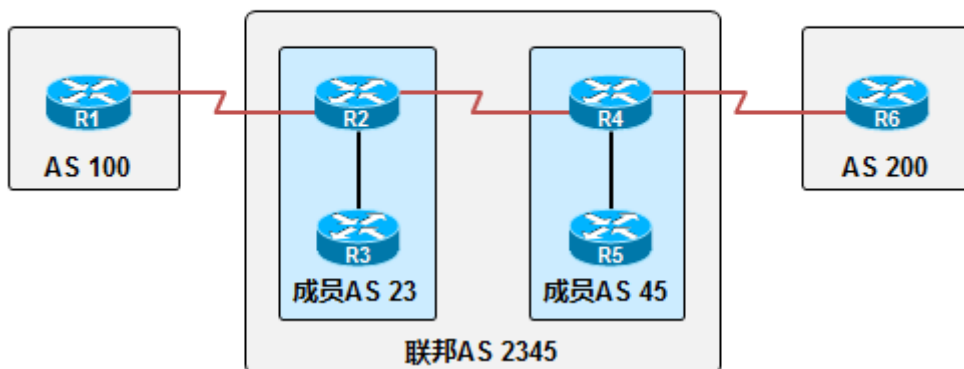
set community ?

<1-4294967295>	community number	
aa:nn	community number in aa:nn format	
additive	Add to the existing community	设置 commu 值为附加，否则为覆盖
internet	Internet (well-known community)	默认所有路由都属于该团体
local-AS	Do not send outside local AS (well-known community)	
no-advertise	Do not advertise to any peer (well-known community)	
no-export	Do not export to next AS (well-known community)	
none	No community attribute	

下边我们来看一下 community 的这几个众所周知值的作用：

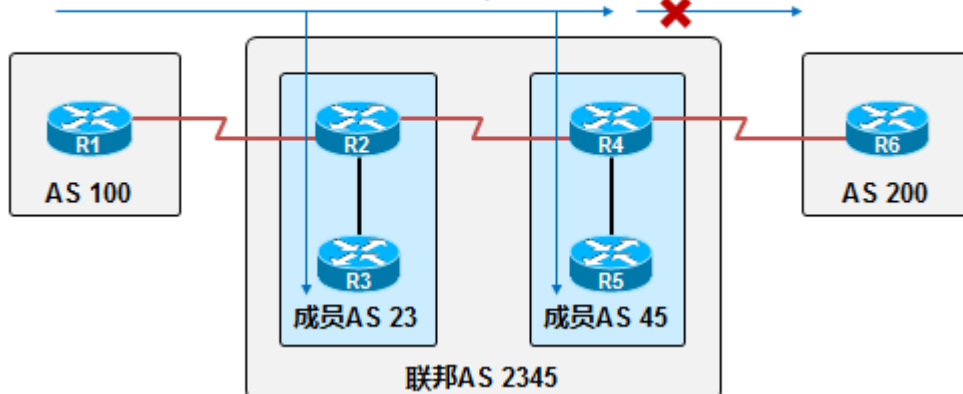
- no-advertise

Route  
Community=no-adv  
R2不会将该路由再通告给任何BGP peer



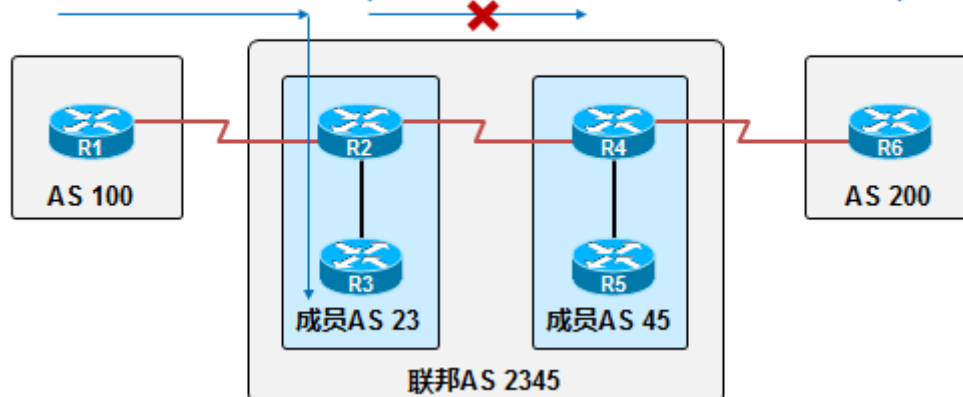
- no-export

Route  
Community=no-export  
R2不会将该路由再通告给任何EBGP peer ( 联邦EBGP仍会传递 )

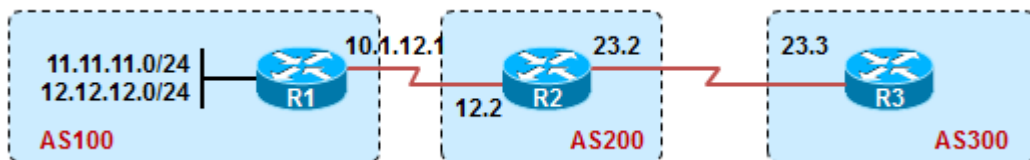


- local-as

Route  
Community=local-as  
该路由只能在本AS内传递 ( 如果定义了联邦, 则为只在联邦成员AS传递 )



配置示例 ( 为路由分配 community ):



在 R1 上为路由 11.11.11.0/24 分配 community 100:11，并且传递给 R2，那么 R1 上配置如下：

```
ip prefix-list 11 permit 11.11.11.0/24
route-map test permit 10
    match ip address prefix-list 11
    set community 100:11
router bgp 100
    network 11.11.11.0 mask 255.255.255.0
    neighbor 10.1.12.2 remote-as 200
    neighbor 10.1.12.2 send-community // 默认 community 不发送，因此必须配置该命令
    neighbor 10.1.12.2 route-map test out
```

注意 community 默认不发送，必须 send-community。

另外一条路由前缀可以携带多个 community 形成一个列表，如果要针对特定路由在原有的 community 基础之上再增加一个 community，则在 route-map 中 set community 时，增加 additive 关键字。

### 3. 使用 ip community-list 匹配 community 值

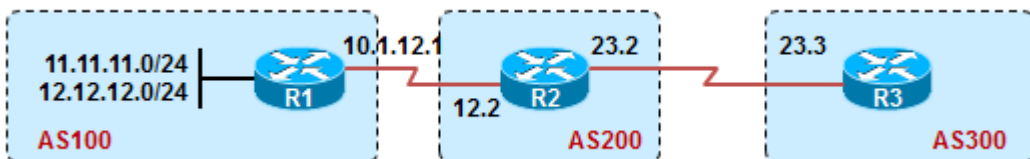
#### ● ip community-list ?

<1-99> Community list number (standard)  
 <100-500> Community list number (expanded)  
 expanded Add an expanded community-list entry  
 standard Add a standard community-list entry

ip community-list 也像 ACL 那样，有标准和扩展之分，1-99 为标准，100-199 为扩展。扩展的 community-list 可以使用正则表达式匹配路由。

这里有一点需要注意，ip bgp-community new-format 用于转换 community 在 CISCO IOS 中的显示格式，当在扩展的 community-list 列表中使用正则表达式的过滤结果会由于格式的选择不同而不同。

#### ● ip community-list 示例 1：



在上面实验的基础上，R1 传递给 R2 的 11.11.11.0/24 路由，携带了 community 值 100:11，这个值可以在 R2 上使用 ip community-list 进行匹配，从而可以进一步在 route-map 中用这个 community-list 去设置策略。我们现在在 R2 上用 community-list 去匹配 100:11，通知添加一个 no-export 的 community 到该路由。

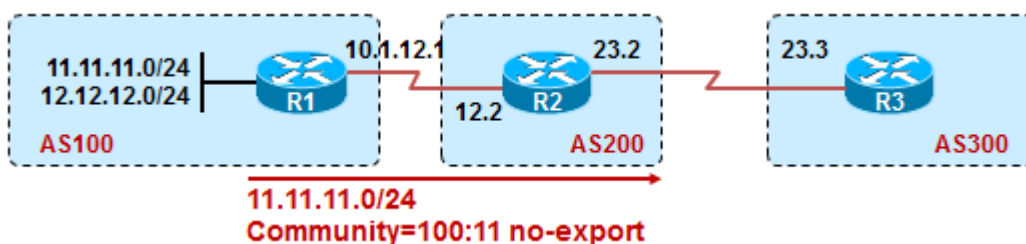
**R2 的配置如下：**

```
ip community-list 11 permit 100:11
route-map test permit 10
  match community 11
  set community no-export additive
router bgp 200
  neighbor 10.1.12.1 remote-as 100
  neighbor 10.1.23.3 remote-as 300
  neighbor 10.1.23.3 send-community
  neighbor 10.1.23.3 route-map test out
```

**R3 上 show ip bgp 11.11.11.0**

```
BGP routing table entry for 11.11.11.0/24, version 5
Paths: (1 available, best #1, table Default-IP-Routing-Table, not advertised to EBGp peer)
Flag: 0x820
Not advertised to any peer
200 100
  10.1.23.2 from 10.1.23.2 (10.1.23.2)
    Origin IGP, localpref 100, valid, external, best
    Community: 100:11 no-export
```

● **ip community-list 的示例 2 (逻辑关系)：**



当 11.11.11.0/24 路由前缀携带的 community 属性为 “100:11 no-export”，我们做如下测试：

```
ip community-list 11 permit 100:11
```

匹配。这种写法将匹配 community 中包含 100:11 的路由。

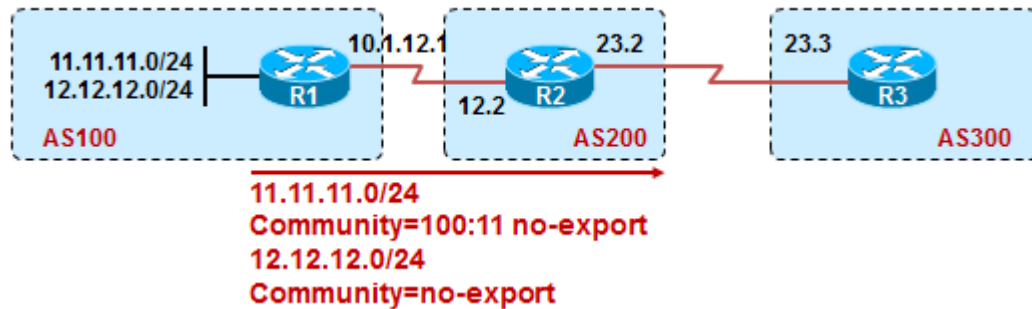
```
ip community-list 11 permit 100:11 no-advertise
```

不匹配。这种写法要求 community 中同时包含 100:11 及 no-advertise 才匹配成立。

```
ip community-list 11 permit 100:11
ip community-list 11 permit no-export ( 或将 no-export 换成 no-adv )
```

匹配。只要 community 中包含 100:11 或 no-export

● ip community-list 的示例 3 ( 严格匹配 community ):



注意我们实验环境的变化, 12.12.12.0 携带的 community 值为 no-export 如果我们只希望匹配 no-export community 值 ( 的路由 ), 那么怎么写呢? 如果是直接去匹配 no-export, 会连同 11.11.11.0 也一并匹配上了, 所以就要使用到 exact-match 关键字了。在 R3 上如果配置如下:

```
ip community-list 11 permit no-export
route-map test permit 10
match community 11 exact-match // 严格匹配
```

如果不加 exact-match 关键字 则该 community-list 将匹配 11 及 12 路由 加了之后 则只匹配 community 为 no-export 的路由, 不能多, 不能少。

#### 4. 在 community 列表中删除特定的 community 值

前面已经说了, 一条路由, 允许携带多个 community 值, 构成一个 community 列表  
那么如果想删除某个或者某几个 community 值, 例如

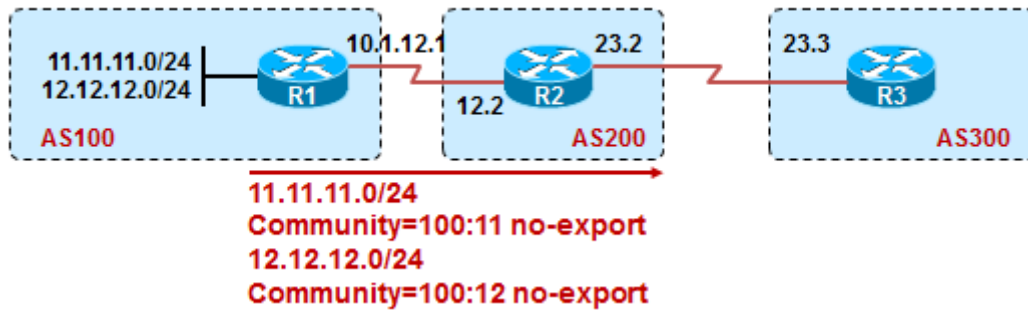
12:11 12:1111 no-export 这个 comm 列表, 要删除其中的 no-export, 则可

```
ip community-list 1 permit no-export // 匹配要删除的 commu 值
route-map test permit 10
set comm-list 1 delete // 用这条命令删除
```

如果要删除多个 community ( 而不是所有 ), 则可在一个 community-list 中写多条, 如

```
ip community-list 1 permit no-export
ip community-list 1 permit 12:1111
```

然后再用 set comm-list 1 delete 去删除, 注意上面 community-list 的写法, 要用多行, 实验验证的结果是在同一行写多个 commu 如 ip community-list 1 permit no-export 12:1111 则不生效, 删除不了这两值



R3 收到两条路由，分别携带的 community 属性如上，现在，我们只想删除 11 路由的 no-export 属性。

```
ip community-list 11 permit 100:11 no-export //这条用来匹配 11 路由
ip community-list standard del permit no-export //这条用来删除 no-export 属性，是个命令列表
route-map test permit 10
  match community 11
  set metric 1111
  set comm-list del delete
route-map test permit 20

router bgp 300
  neighbor 10.1.23.2 remote-as 200
  neighbor 10.1.23.2 route-map test in
```

如果想把 11 路由的 100:11 及 no-export 两个 community 值都删除，则修改 del 这条 community-list 即可：

```
ip community-list standard del permit no-export
ip community-list standard del permit 100:11
```

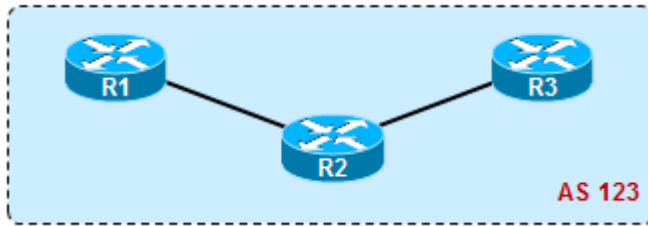
## 5. COST community :

使用 set excommunity cost x 在 route-map 去设置

可以影响 BGP 选路（在等价负载均衡的前一个），有点类似于“tie breaker”在路径选择的时候，就是在纠结的时候打破纠结

可以针对路由设置 cost，cost-ID 相等的时候，优选 cost 小的。如果 cost 相等，则选择 cost-ID 小的  
set excommunity cost pre-bestpath 1 20，则 cost 值的将极大的影响选路原则，将会成为超越 weight 值的第一条规则

注意，这个时候在 send-community 的时候要加上扩展 extend 关键字



R1、R2、R3 使用 LOOPBACK 建立邻居关系，地址分别为 1.1.1.1、2.2.2.2、3.3.3.3，运行 OSPF 使得 AS 内部 LOOPBACK 间路由能各自学习到。R1、R3 均向 R2 更新 100 网段的路由，在不做任何策略的前提下，BGP 的选路规则会一直比到 RID，最终优选 R1 的路由。

```
R2#sh ip b 100.100.100.0
```

```
BGP routing table entry for 100.100.100.0/24, version 2
```

```
Paths: (2 available, best #2, table Default-IP-Routing-Table)
```

```
Flag: 0x820
```

```
Not advertised to any peer
```

```
Local
```

```
3.3.3.3 (metric 65) from 3.3.3.3 (3.3.3.3)
```

```
Origin IGP, metric 0, localpref 100, valid, internal
```

```
Local
```

```
1.1.1.1 (metric 65) from 1.1.1.1 (1.1.1.1)
```

```
Origin IGP, metric 0, localpref 100, valid, internal, best
```

这时候就优选了 R1，如果希望让 R2 优选 R3 呢？R2 增加配置如下：

```
ip prefix-list 100 permit 100.100.100.0/24
```

```
route-map COST1 permit 10
```

```
match ip address prefix-list 100
```

```
set extcommunity cost 1 10
```

```
route-map COST2 permit 10
```

```
match ip address prefix-list 100
```

```
set extcommunity cost 1 5
```

```
router bgp 123
```

```
neighbor 1.1.1.1 route-map COST1 in
```

```
neighbor 3.3.3.3 route-map COST2 in
```

如果按以下配置，那么 COST 的比较，将会超越 weight，成为最优选比较的规则

```
route-map COST1 permit 10
```

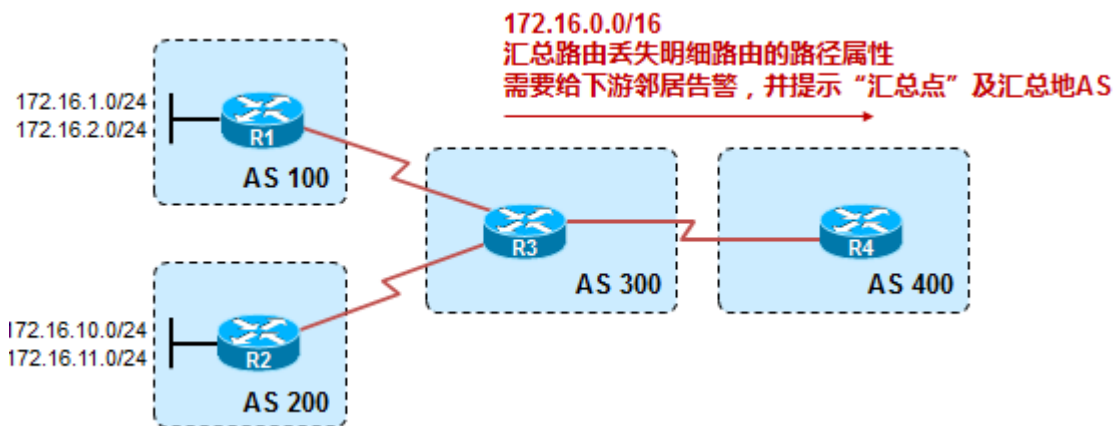
```
match ip address prefix-list 100
```



set extcommunity cost pre-bestpath 1 10

### 3.7 Atomic\_Aggregate 及 aggregator

Atomic\_Aggregate 是公认自决属性；Aggregator 是可选可传递。



我们看上图，R1、R2 均发布了路由，R3 上对这些明细路由进行汇总，汇总成 172.16.0.0/16，汇总路由被 R3 传递给了其他 BGP 邻居，然而，这条汇总路由丢失了底下明细路由的所有属性，其中属 AS\_PATH 最关键，因为一旦丢失了明细路由的 AS\_PATH 属性，这条汇总路由就极有可能产生路由环路。因此有必要让 R3 警告下游 BGP peer，告诉他们 1、这是条汇总路由，2、汇总路由发生的地方，也就是汇总路由的始发 AS 和始发路由器。

R4#show ip bgp 172.16.0.0

BGP routing table entry for 172.16.0.0/16, version 4

Paths: (1 available, best #1, table Default-IP-Routing-Table)

Flag: 0x820

Not advertised to any peer

300, (**aggregated by 300 3.3.3.3**) // 汇总路由产生自 AS300，由 BGP router 3.3.3.3 产生

10.1.34.3 from 10.1.34.3 (3.3.3.3)

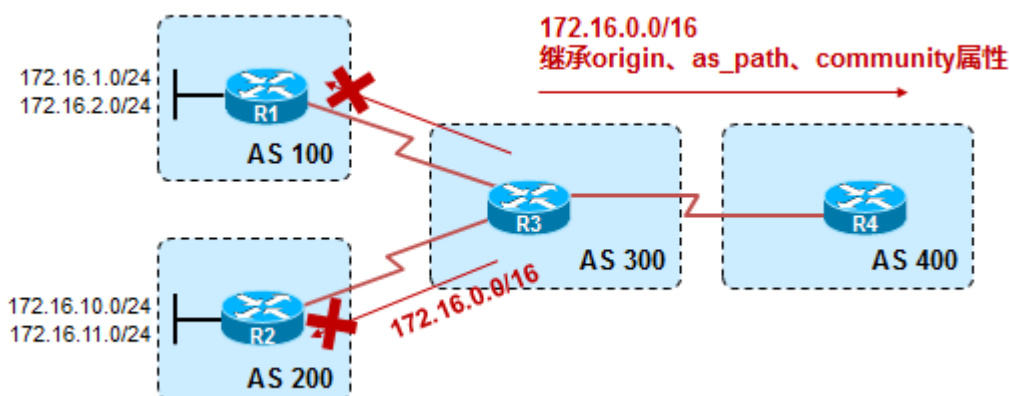
Origin IGP, metric 0, localpref 100, valid, external, **atomic-aggregate**, best

抓包如下：

```

UPDATE Message
  Marker: 16 bytes
  Length: 63 bytes
  Type: UPDATE Message (2)
  Unfeasible routes length: 0 bytes
  Total path attribute length: 37 bytes
  Path attributes
    + ORIGIN: IGP (4 bytes)
    + AS_PATH: 300 (7 bytes)
    + NEXT_HOP: 10.1.34.3 (7 bytes)
    + MULTI_EXIT_DISC: 0 (7 bytes)
    + ATOMIC_AGGREGATE (3 bytes)
      + Flags: 0x40 (Well-known, Transitive, Complete)
      Type code: ATOMIC_AGGREGATE (6)
      Length: 0 bytes
    + AGGREGATOR: AS: 300 origin: 3.3.3.3 (9 bytes)
      + Flags: 0xc0 (Optional, Transitive, Complete)
      Type code: AGGREGATOR (7)
      Length: 6 bytes
      Aggregator AS: 300
      Aggregator origin: 3.3.3.3 (3.3.3.3)
  Network layer reachability information: 3 bytes
    + 172.16.0.0/16
  
```

注意这里 atomic\_aggregate 为公认自由决定属性，我们来理解一下：



在 R3 上做汇总，使用 aggregate-address summary-only as-set，注意加上了 as-set 关键字

这样一来，汇总路由就继承了明细的 AS\_PATH 属性，从而 atomic\_aggregate 就显得多余了，因此，上面这种情况下，产生的汇总路由就不携带 atomic\_aggregate 了。我们看看：

R4 上 show ip bgp 172.16.0.0

BGP routing table entry for 172.16.0.0/16, version 3

Paths: (1 available, best #1, table Default-IP-Routing-Table)

Not advertised to any peer

300 {100,200}, (aggregated by 300 3.3.3.3)

10.1.34.3 from 10.1.34.3 (3.3.3.3)

Origin IGP, metric 0, localpref 100, valid, external, best

的确没有 atomic\_aggregate。

### 3.8 ORIGINATOR\_ID 和 CLUSTER\_LIST

由于 AS\_PATH 属性在 AS 内部不会发生变化（仅当路由离开本 AS 才会被更新），因此 AS 内防环才有水平分割的机制，而路由反射器实际上是**放宽了水平分割原则**，这个就会给环路带来一定的隐患，因此**路由反射器需使用以下两个属性防止环路**：

**ORIGINATOR\_ID** 和 **CLUSTER\_LIST** 是路由反射器使用的可选非传递属性，用来防止环路。

**ORIGINATOR\_ID** 是一个路由反射器创建的 32bit 值，该数值是**本地 AS 中路由发起方的 IBGP routerID**，注意，这个发起方未必是这条路由的引入者（下面有实验验证），如果发起方发现其 RID 在所接收到的路由的 ORIGINATOR\_ID 中，那么就知道已经出现了路由环路，因此忽略该路由。

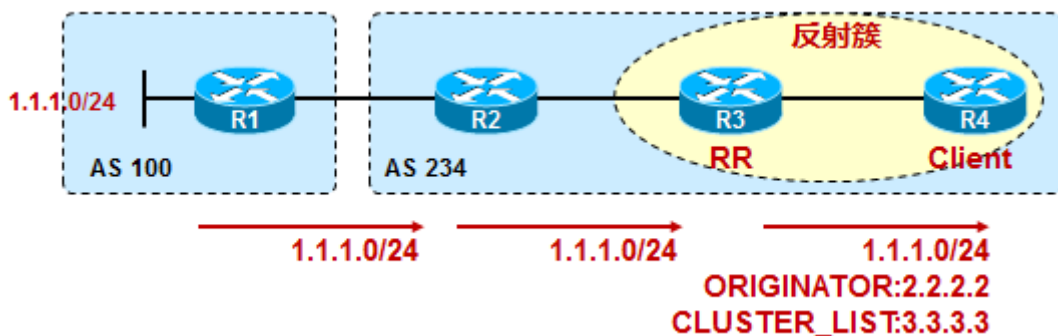
**CLUSTER\_LIST** 是一串路由传递所经过的路由反射簇（cluster）的 ID，AS 内的每个路由反射簇都有一个 32bit 的簇 ID，如果簇中包含了多个 RR，则需手工为每个 RR 配置簇 ID。

当 RR 将来自客户的路由反射给非客户时，同时将其簇 ID 附加到 CLUSTER\_LIST 中，如果 CLUSTER\_LIST 为空，则创建一个。如果路由反射器发现其本地簇 ID 出现在了其收到的路由的 CLUSTER\_LIST 中，那么就知道出现了环路，则忽略该路由。CLUSTER\_LIST 属性只用于 RR 防环。

注意：RR 只在反射路由的时候才创建或更新 CLUSTER\_LIST，而下面几种情况，RR 不会创建该属性：

- ◆ RR 自己始发的路由
- ◆ 当 RR 向 EBGP 邻居发送路由更新时，将会清除已有的 CLUSTER\_LIST 属性
- ◆ 当 RR 从 EBGP 邻居收到路由，传递给 client 或非 client，不会创建 CLUSTER\_LIST。

#### ● 验证实验 1 ORIGINATOR\_ID 的取值

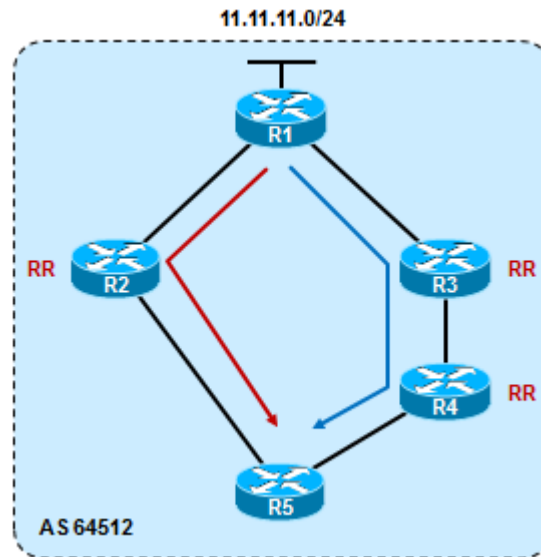


R1 处于 AS100，将 1.1.1.0/24 的路由传递给 EBGP 邻居 R2。

IBGP 邻居关系如图所示，使用各自的 loopback 口建邻居。R2 收到路由后，继续传递给 IBGP 邻居 R3，这时候 R3 由于是 RR，因此将非 client IBGP 邻居发来的路由，传递给自己的 client R4，此时，R3 在路由中添加 ORIGINATOR 及 CLUSTER\_LIST 属性，其中 ORIGINATOR 属性为 R2 的 routerID 也就是 2.2.2.2（而

不是 R1 的 routerID，因为根本没必要）。

- 验证实验 2 CLUSTER\_LIST 在 BGP 选路中的影响



```
R5#show ip bgp 11.11.11.0
BGP routing table entry for 11.11.11.0/24, version 2
Paths: (2 available, best #2, table Default-IP-Routing-Table)
Flag: 0x820
Not advertised to any peer
Local
  1.1.1.1 (metric 129) from 4.4.4.4 (4.4.4.4)
    Origin IGP, metric 0, localpref 100, valid, internal
    Originator: 1.1.1.1, Cluster list: 4.4.4.4, 3.3.3.3
Local
  1.1.1.1 (metric 129) from 2.2.2.2 (2.2.2.2)
    Origin IGP, metric 0, localpref 100, valid, internal, best
    Originator: 1.1.1.1, Cluster list: 2.2.2.2
```

### 3.9 Weight

CISCO 私有，作用范围是本路由器（不传递），该值不既不会被包含在 update 消息中，也不会传递给任何 BGP 邻居，只在路由器本地产生影响，可以理解为权重值，越大越优先。

取值是范围 0-65535。

- 如果路由是从其他邻居学过来的，则默认值（在本地该路由）是 0
- 本地 network 产生的路由 weight 是 32768

本地重发布的直连接口路由、静态路由的 weight 为 32768

本地汇总产生的 BGP 路由 weight 值也为 32768

## 4 BGP 配置

### 4.1 建立 BGP 邻居

#### 1. router bgp asnumber

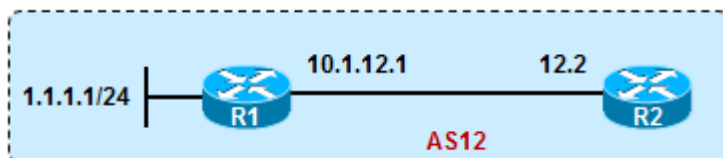
AS 号为 2B，但是随着网络发展，2B 不够用了，所以现在扩充到了 4 个 B router bgp ？可以查看（注：仅在特定 IOS 版本下支持）

#### 2. network xxx mask yyy route-map zzz

在 IGP 中 network 命令用于确定要发送和接收路由更新的接口，以及通告哪些直连的网络。

而在 BGP 中，network 命令与 BGP 在哪些接口上运行无关。仅仅是将本地某个网路注入到 BGP 里。该路由我们称为可靠的。只有该路由存在于路由器的路由表中（非 B），在 BGP 下 network 该网段才有可能使得该路由条目传递给 BGP 邻居。务必注意，network 只能有效注入那些在路由表中存在的非 B 的路由。如果我本地有个 loopback，4.4.4.0/24 而我 BGP 中 network 4.0.0.0，那么宣告不成功（BGP 默认不自动汇总，当然注意 IOS 版本）；

network 不加掩码就是有类宣告，加上掩码就是无类宣告



- 若 R1 关闭 auto-summary，且 network 1.1.1.0 mask 255.255.255.0 则 R2 能学到 BGP 路由 1.1.1.0/24
- 若 R1 关闭 auto-summary，且 network 1.1.1.0，则宣告不成功，因为不加 mask 为有类宣告
- 若 R1 开启 auto-summary，且 network 1.1.1.0，则宣告不成功
- 若 R1 开启 auto-summary，且 network 1.0.0.0，则 R2 能学习到汇总路由 1.0.0.0/8

#### 3. neighbor ip remote-as as-number

neighbor 的这个地址，必须是 IP 可达的

#### 4. 多跳 EBGp

EBGP 邻居通常必须是直连的才能建立起 EBGp 会话。但

RA(as200)-----RB(as100)-----RC(as100)

RC 和 RA 之间隔着 RB ,在这种情况下 ,EBGP 通过使用可以配置的 “多跳 EBGP”的特性穿过一台非 BGP 路由器运行。也就是说 RB 可以不支持 BGP , 配置如下 ( RA )

```
router bgp 200
neighbor RC-ip remote-as 100
neighbor RC-ip ebgp-multihop 2    //他们之间跳数为 2
```

另, 如果两个 EBGP 之间有多条链路, 并使用各自的 Loopback 端口作为 source , 那么在配置的时候也要注意。ebgp-multihop 2 ( 因为 loopback 跨越了两台路由器, 其中包括自己这台), 并且还必须指静态路由到对方的 loopback 接口。)

## 4.2 路由重发布

### 1. BGP 重发布到 IGP

将 BGP 路由重发布到 IGP 时, 默认不重发布 iBGP 路由, 只重发布 EBGP 路由, 需使用 bgp redistribute-internal 来让 iBGP 路由顺利被重发布  
注意, 在 MPLS VPN 环境中的 PE 没有这个限制

### 2. OSPF 重发布进 BGP

```
router bgp x
 redistribute ospf 1
```

默认情况下, 只会将 OSPF 中 O 及 O IA 的路由重发布进 BGP

```
redistribute ospf 1 match external 1 external 2
```

以上命令只重发布 OSPF 外部路由 E1、E2 进 BGP

```
redistribute ospf 1 match internal external 1
```

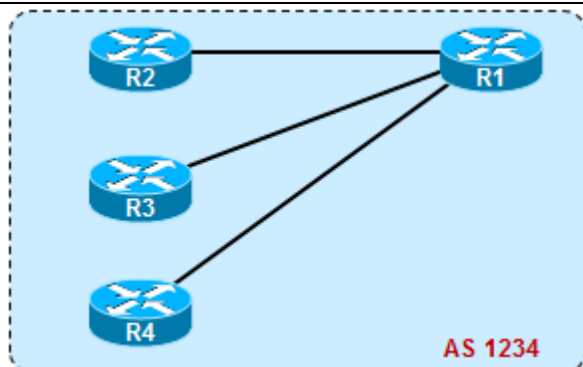
以上命令只重发布 OSPF 内部路由及 E1 路由进 BGP

```
redistribute ospf 1 match nssa-external 1 nssa-external 2
```

以上命令只重发布 OSPF NSSA 路由进 BGP

## 4.3 peer group

### 基本配置



```

router bgp 1234
  neighbor IBGP_peers peer-group
  neighbor IBGP_peers remote-as 1234
  neighbor IBGP_peers update-source loopback0
  neighbor IBGP_peers password cisco
  neighbor IBGP_peers send-community
!
  neighbor 2.2.2.2 peer-group IBGP_peers
  neighbor 3.3.3.3 peer-group IBGP_peers
  neighbor 4.4.4.4 peer-group IBGP_peers
  
```

无法对 peer-group 的 member 单独使用 out 方向的策略，只能对整个 group，in 方向是没有这个限制的。  
同一个 Peer Group 中的所有邻居，必须全部为 iBGP 邻居，或者全部为 eBGP 邻居

## 4.4 配置查看及验证

### 1. show ip bgp

显示 BGP 表，例如：

```

R1#sh ip b
BGP table version is 1, local router ID is 12.12.12.12
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
               r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network        Next Hop        Metric   LocPrf   Weight   Path
*   11.11.11.0/24   0.0.0.0          0             32768    i
  
```

第一栏的可能取值如下：

\* 可用的路由(next-hop 可达)



s 被抑制的路由条目，例如做了路由汇总，抑制了明细  
d 被惩罚（dampening）的路由，路由受到了惩罚，虽该路由当前可能正常，但惩罚期结束前不会被通告  
h 被惩罚（dampening）的路由，路由可能出现了故障（down），有历史信息，但没有最佳路由  
r 路由没有被装载进 RIB 表，例如由于 AD 值等原因导致  
S 大写的 S，stale，表示过期的路由

**第二栏 > BGP 算法选出的最优路径**

**第三栏 为空，或为 i。为空表示该路由从 EBGp 邻居获取，为 i 表示这是从 IBGP 学习到的路由**

**最后一栏 指出该 BGP 路由信息的起源**

## 2. show ip bgp summary

BGP router identifier 12.12.12.12, local AS number 64512

**BGP table version** is 2, **main routing table version** 2

1 network entries using 117 bytes of memory

1 path entries using 52 bytes of memory

2/1 BGP path/bestpath attribute entries using 248 bytes of memory

0 BGP route-map cache entries using 0 bytes of memory

0 BGP filter-list cache entries using 0 bytes of memory

BGP using 417 total bytes of memory

BGP activity 1/0 prefixes, 1/0 paths, scan interval 60 secs

Neighbor	V	AS	MsgRcvd	MsgSent	TblVer	InQ	OutQ	Up/Down	State/PfxRcd
10.1.12.2	4	200	1083	1082	2	0	0	17:57:27	0

**BGP table version** 每当 BGP 表发生变化，这个值+1

**main routing table version** 被注入到主路由表中的最后一个 table version

AS 邻居的 AS 号

MsgRcvd 从邻居那收到的消息数

MsgSent 发送给该邻居那的消息数

TblVer 发送给该邻居的最后一个 bgp table 的 version

inQ 来自该邻居的等待处理的消息数

outQ 队列中等待被发送到该邻居的消息数

State BGP 当前的回话状态基本的如 opensent 等，admin 表示邻居被管理型 shutdown，当处于 establish 状态时，不会显示状态提示，而显示一个表示 PfxRcd 的数字，也就是从该邻居处收到的前缀条目

### 3. show ip b rib-failure

显示没有被加载进 RIB 中的 BGP 路由，以及没被加载的原因

### 4. show ip bgp 路由前缀

```
R5#sh ip b 30.30.30.0
BGP routing table entry for 30.30.30.0/24, version 2
Paths: (2 available, best #2, table Default-IP-Routing-Table)
Flag: 0x820
Not advertised to any peer
Local
  3.3.3.3 (metric 65) from 4.4.4.4 (4.4.4.4)
    Origin IGP, metric 0, localpref 100, valid, internal
    Originator: 3.3.3.3, Cluster list: 4.4.4.4
Local
  3.3.3.3 (metric 65) from 3.3.3.3 (3.3.3.3)
    Origin IGP, metric 0, localpref 100, valid, internal, best
```

NEXT\_HOP  
到达该NEXT\_HOP的metric(IGP)

BGP邻居的更新源地址  
邻居的RouterID

### 5. show tcp bri

### 6. sh ip bgp neighbors {address} received-routes

显示从指定邻居收到的所有 BGP 路由（本地入站策略执行前的原始路由）

### 7. sh ip bgp neighbors {address} routes

显示从指定邻居那里收到的所有路由（上一条命令的子集，这里显示的是执行入站策略后剩余的路由）

### 8. sh ip bgp neighbors {address} advertised-routes

显示通告给特定邻居的所有 BGP 路由

### 9.

## 4.5 邻居身份验证

BGP 支持 MD5 身份验证，要在 BGP 对等体之间的 TCP 连接上启用 MD5 身份验证，使用命令 如下：

```
neighbor xxx password xxx
```

同一个 BGP 连接，密码必须一致；不同的邻居可设置不同的密码。配置认证后，将通过对等体之间的 TCP 连接传输的所有数据段进行验证。

以下是配置了 MD5 身份验证后报文的变化（所有 TCP 包增加了 MD5 HASH 字段）：

注意这只是基本的身份验证功能，并不是加密

```

Transmission Control Protocol, Src Port: 49864 (49864), Dst Port: bgp (179)
  Source port: 49864 (49864)
  Destination port: bgp (179)
  [Stream index: 0]
  Sequence number: 1      (relative sequence number)
  [Next sequence number: 46      (relative sequence number)]
  Acknowledgement number: 1      (relative ack number)
  Header length: 40 bytes
  Flags: 0x18 (PSH, ACK)
  Window size: 16384
  Checksum: 0x0f72 [validation disabled]
  Options: (20 bytes)
    TCP MD5 signature ← MD5的hash值
    EOL
  [SEQ/ACK analysis]
  Border Gateway Protocol BGP报文
  
```

## 4.6 邻居管理

### 1. neighbor maximum-prefix xx

限制从邻居接受的前缀最大数 如果超出了这个数 路由器就会关闭与该邻居的 BGP 连接 在应用 clear ip bgp xxx 之前都不会再次建立 BGP 会话

### 2. neighbor maximum-prefix xx restart 2

超过 xx 这个数，断开与邻居的 BGP 连接，2 分钟后才能重新连接

### 3. Neighbor maximum-prefix 300 90% warning-only

当从邻居接受的前缀数量超过了最大数 300 的 90% 时（默认 75%），生成一条日志消息

### 4. 重置 BGP 连接

#### ● 硬重置

所有的 BGP 邻居的 TCP session 及 BGP 邻居状态都重新复位

**clear ip bgp \***

**clear ip bgp {neighbor-address}**

#### ● 软重置

不拆除并重建 TCP 或 BGP 连接，而是仅触发更新操作以便让新的路由策略生效

软重置可以仅由于出站或入站策略，也可同时用于出入站策略

**clear ip bgp soft out** // 出站软重置，重新发送 update 消息给所有邻居

**clear ip bgp soft in** // 入站软重置，本地发送 route-refresh 给所有 BGP 邻居

**clear ip bgp {neighbor-address} soft in/out** 指定特定的邻居，建议

执行入站软重置的方法有两种：1、使用存储的路由更新信息的软重置 2、动态软重置

### 1> 使用存储的路由更新信息的软重置

```
neighbor x.x.x.x soft-reconfiguration inbound
```

首先在 BGP 进程中使用上述命令，这条命令会将 x.x.x.x 邻居发送过来的最原始的 BGP 路由存储在本地内存中，当配置入站软重置后（clear ip bgp soft in），路由器不再向邻居发送更新请求，而是直接在内存中存储的那些 BGP 路由中执行新配置的入站策略，以此来防止触发大批量的路由更新而造成资源的浪费，但是这种操作仍会耗费内存，因此在使用的时候要非常慎重。

### 2> 动态的入站软重置

CISCO IOS 12.1 开始全面支持入站路由的动态软重置配置，这种软重置方法直接在本地向特定邻居发送 route-refresh 消息。

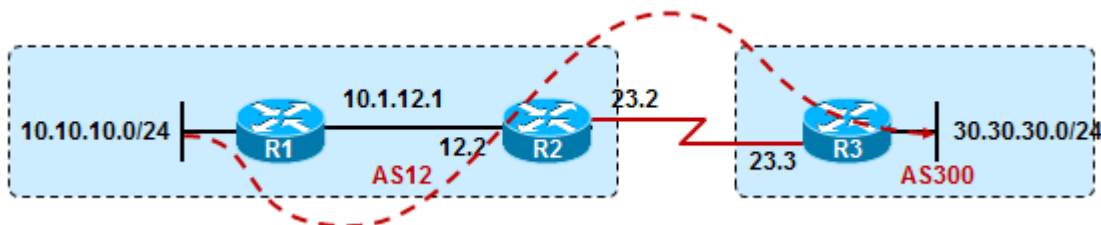
## 4.7 BGP Policy Accounting

### 1. 基本概念

BGP PA 能够对 BGP peer 发送来的 IP 流量进行度量或流量统计

Policy accounting is enabled on an input interface, and counters based on parameters such as community list, autonomous system number, or autonomous system path are assigned to identify the IP traffic.

### 2. 配置案例



配置的思路是，在 R2 上，想要抓取 fast0/0 口（也就是 R1 发过来的），去往 30.30.30.0/24 的流量

那么先在 R2 上对 30.30.30.0/24 的路由前缀进行标记，使用 community 标记，同时配置一个 route-map PA，将特定 community 标记的路由设置 traffic-index。配置如下：

```
ip prefix-list 30 permit 30.30.30.0/24
route-map setComm permit 10
  match ip address prefix-list 30
  set community 300:30
ip community-list 30 permit 300:30
route-map PA permit 10
  match community 30
  set traffic-index 30
```

```
router bgp 12
  table-map PA
  neighbor 10.1.23.3 route-map setComm in

interface fast0/0
  ip address 10.1.12.2 255.255.255.0
  bgp-policy accounting input
```

接下去在 R1 上向 30.30.30.0/24 去 ping 10 个 ICMP 包，那么就可以在 R2 上看到相应的流量：

```
R2#sh cef interface s0/0 policy-statistics input
Serial0/0 is up (if_number 4)
  Corresponding hwidb fast_if_number 4
  Corresponding hwidb firstsw->if_number 4
  BGP policy accounting on input is enabled.

Index          Packets          Bytes
...
30             10             1000
```

如果嫌输出太多，可以用管道符：sh cef interface s0/0 policy-statistics input | in 30

## 5 BGP 决策

### 5.1 决策概述

**BGP 路由信息数据库 RIB，包含三个部分：**

ADJ-RIBs-IN 存储来自对等体的、未经处理的消息。所含的路由都是可用路由

Loc-RIB BGP 路由器通过对 adj-RIBs-In 中的路由使用它的本地路由策略而选择的路由

ADJ-RIBs-OUT 公布给对等体的路由

BGP 的决策过程，是对 adj-RIBs-IN 中的路由使用本地路由策略，同时将选定的或修改过的路由放到 Loc-RIB 和 Adj-RIBs-Out 中

1. Weight 越大越优先
2. Local\_Pref 越大越优先
3. 起源于本地的路由优先（如本地 network 的，或 aggregate 的），即下一跳是 0.0.0.0(在 BGP 表中,当前路

由器通告的路由的下一跳为 0.0.0.0)

- 规则详解

以下优先级依次降低：default-originate（针对每个邻居配置）、default-information-originate（针对每种地址簇配置）、network、redistribute、aggregate-address

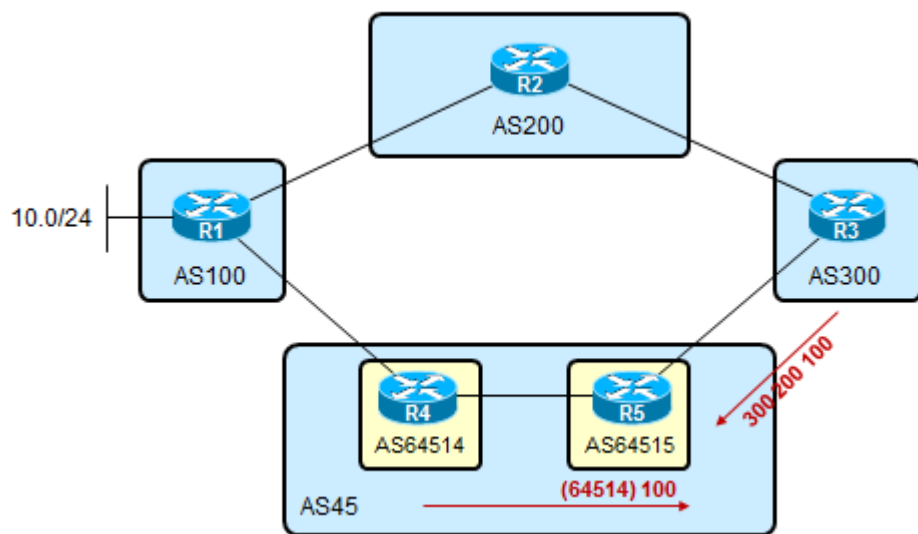
#### 4. AS-Path 越短越优先

- 规则详解

在做聚合路由时，使用 as-set 后产生的 AS-Path 列表中的{}里的 AS 号长度只算一个 AS 号的长度  
在联盟内的 AS-Path 列表中的()的 AS 号长度不做计算依据

bgp bestpath as-path ignore 这条命令如果配置了，则跳过此规则

- 规则验证（联邦内成员 AS 号不参与 AS\_PATH 长度计算）



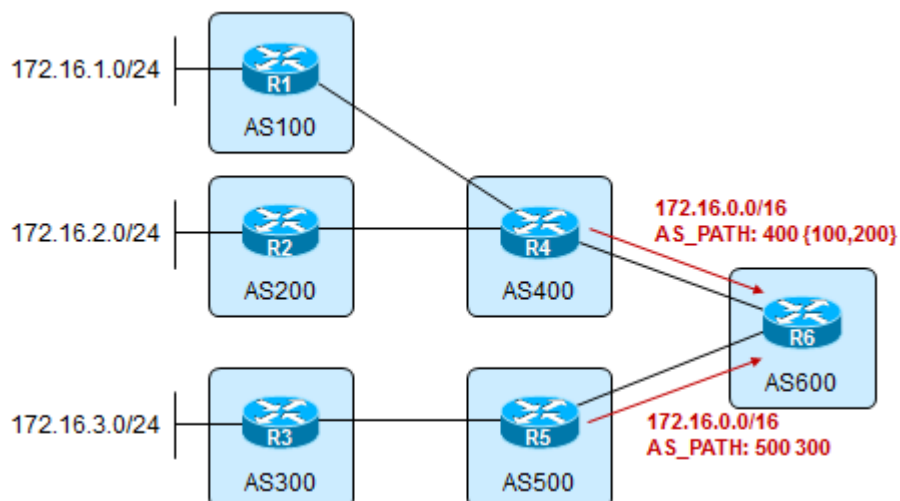
所有设备完成基本配置，建立 BGP 邻居关系。黄色区域为联邦成员 AS。

R5 将会同时从 R4 及 R3 收到关于 10.0 网络的 BGP 路由，R5 首先会优选 R3，因为 R4 传递给 R5 的路由保持了 NEXT\_HOP，仍为 R1，因此 R5 上 R4 过来的这条路由下一跳不可达，通过在 R4 修改修改下一跳为自己后，R5 优选 R4。

此时在 R1 上对 R4 使用策略，将 10.0 的路由 AS\_PATH 插入 100 的 AS 号，结果一边为 R4 过来的 (64514) 100 100，另一边为 R3 过来的 300 200 100，R5 仍优选 R4，因为(64512)不参与 AS\_PATH 长度计算。

此时在 R1 上对 R4 使用策略，将 10.0 的路由 AS\_PATH 插入 100 100 的 AS 号，R5 优选 R3。关于为什么 R5 会选择 R3，经过实验得出，这里 R5 收到的两条路由，一条来自联邦 EBGp 邻居，一条来自 EBGp 邻居，最终优选 EBGp 邻居的路由。

- 规则验证（AS\_SET 类型的 AS\_PATH，也就是{}中的 AS 号，在 AS\_PATH 计算时作为一跳）



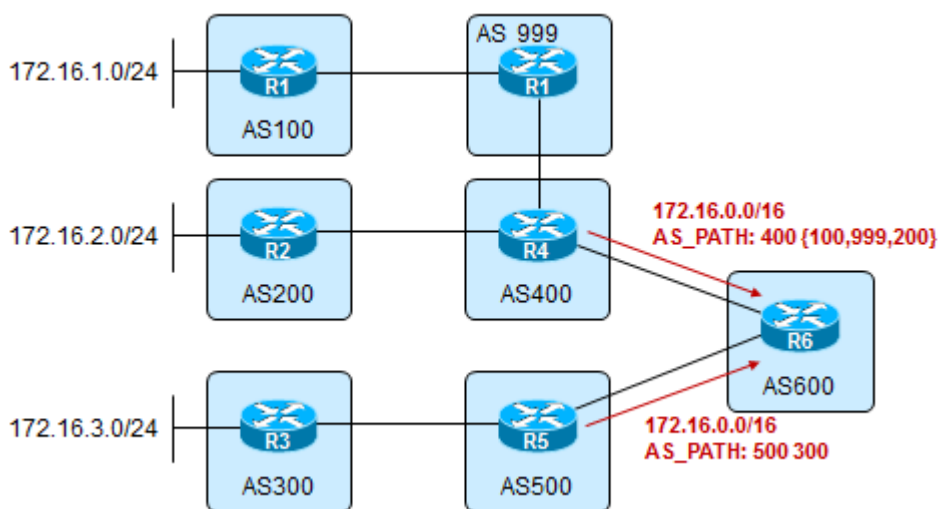
BGP AS 规划如上，R1、R2、R3 各自在 BGP 进程中 network loopback 路由（不做任何策略）；  
在 R4 及 R5 上分别进行路由汇总：

```
aggregate-address 172.16.0.0 255.255.0.0 as-set summary-only
```

如此一来，R6 将分别从 R4 及 R5 上收到汇总路由：172.16.0.0/16

注意到虽然 R4 传递过来的路由 400 {100,200}，大括号中的 AS，在 AS\_PATH 计算中仅作为一个 AS 计算，  
因此 R6 上收到的这两条 EBGP 路由，AS\_PATH 长度相等，最终，R6 将优选最老的 EBGP 邻居的路由。

再爽一把，下面的拓扑，现象和上面是一样的，要注意使用 as-set 关键字的汇总路由，底下的明细路由的 AS\_PATH 都会被放入 {} 中，也就是 AS\_SET 中。



## 5. Origin 属性（优先：IGP > EGP > Incomplete）

## 6. MED 越小越优先

### ● 规则详解

默认情况下仅有当所有的备选路由来自同一个 AS 的不同 EBGP 邻居时，才会比较 MED；

bgp always-compare-med 这条命令，可以使得即使路由来源于不同的 AS 也进行 MED 的比较



7. 优选 EBGP 邻居发来的路由（相对于 IBGP 邻居），在联邦 EBGP 和 IBGP 中优选联盟 EBGP 路由
8. 优选到 BGP NEXT\_HOP 最近的路由，该路由是去往下一跳路由器 IGP 度量值最小的路由
9. 如果有多条来自相同相邻 AS 的路由并通过 Maximum-paths 使多条路径可用，则将所有开销相同的路由加入 Loc-RIB

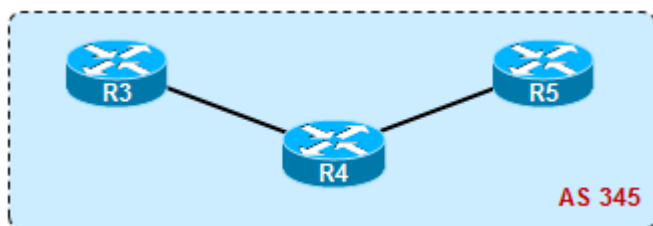
- 规则详解

当前面的 8 条选路原则都无法优选出最优路由时，并且在 BGP 进程下面配置了 maximum-paths [ibgp] n,n 的取值为 2-6,那么将执行等价负载均衡

如果不关联 ibgp 关键字，那么只会对 EBGP 路由执行等价负载均衡（默认仅对 EBGP 路由）

如果不配置 maximum-paths，那么将进行到下一条选路原则

- 实验验证（IBGP 等价负载均衡）



R3、R5 都向 R4 发布同一条网段的路由，在前 8 条规则都无法决策的情况下，如果 R4 配置了：

Maximum-paths ibgp 2，则在 R4 上，ip 路由表里，将出现 BGP 路由的负载均衡，但是要注意，R4 在 BGP 表中，只会优选一条路由，应是 R3（因为 R3 的 RouterID 3.3.3.3 小于 R5 的 5.5.5.5）

- 实验验证（EBGP 等价负载均衡）

EBGP 负载均衡，经过验证，只有当路由来源于同一个 AS 内的不同 EBGP 邻居时并且符合等价负载均衡的条件下，配置 maximum-paths x 才能实现负载均衡，如来源于不同的 AS，则配置了该命令也无效

- 非等价负载均衡请见 5.3 小节

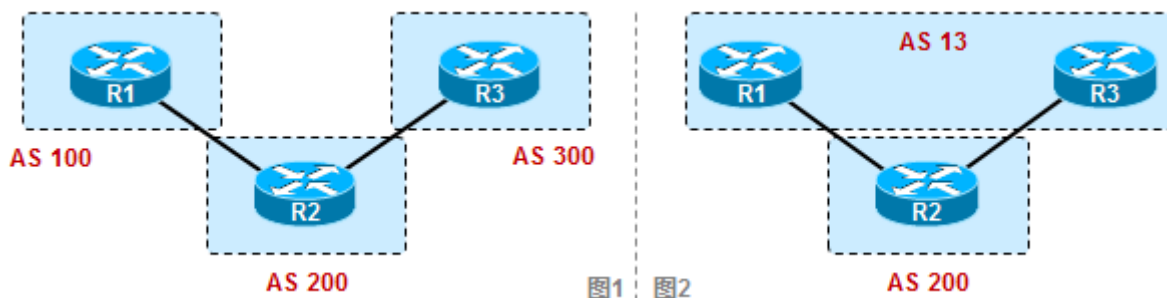
10. 如果路由都来自 EBGP 邻居，则优选最老的 EBGP 邻居传来的路由，降低滚翻的影响（此条主要对 EBGP 路由起效，但是现在基本不用该条，因不确定性太强）

- 规则详解

最老的 EBGP 邻居，意味着有可能是最稳定的 BGP 邻居，因此这里做了优选，当然，这条规则可控性差，一般不作为选路策略使用。以下情况发生，将会跳过该条规则：

- 1) 配置了 bgp bestpath compare-routerid 命令后，将跳过该条原则，拥有最小 RID 的路由被选为最优
- 2) 多条路径具有相同的 RouterID，因为这些路由都是从同一台路由器接收过来的。

- 实验验证



该实验是在 IOS C3640-IK9O3S-M Version 12.4(25)环境下测试，

图 1 R1、R3 同时向 R2 发布 13.13.13.0 的 BGP 路由，没有做任何的策略

由于实验中先配置的 R1，因此 R2 去往 13 网段，优选的是 R1，这时候在 R2 上 clear 一下 BGP 邻居 R1，则与 R1 BGP 连接 DOWN，R2 去往 13 的路由 又优选了 R3，待 R2 与 R1 恢复 BGP 连接并再次收到 R1 的 13 路由更新后，R2 仍优选 R3

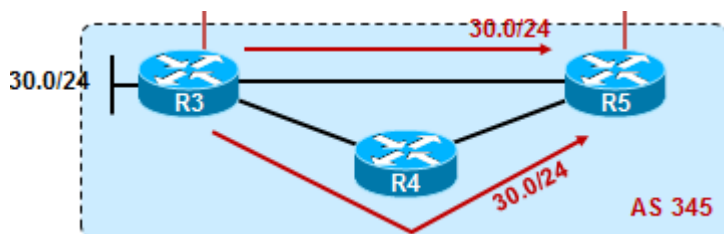
图 2 R1、R3 为同一个 AS，也是同时向 R2 更新 13 网段的路由，实验效果也符合规则描述  
如果是在 IBGP 环境下，也就是 R1、R2、R3 都在一个 AS 内，那么该条规则不适用

## 11. BGP 邻居的 RID 越小越优先

### ● 规则详解

RID 是路由器上的最大 IP 地址，倾向于赋给回环地址。也可以通过 bgp router-id 命令手工设置。如果一条路径包含 RR 属性，路由产生者 ID ( originator ) 将在最优路径选择过程中代替 RID

### ● 实验验证



R3-R4 ; R4-R5 ; R3-R5 维护 IBGP 邻居关系并且都是用各自的 LOOPBACK 口作为更新源及建立 BGP 邻居关系，地址分别为 3.3.3.3、4.4.4.4、5.5.5.5。AS 内运行 OSPF 协议，使得所有路由器都能获取到其他路由器的 LOOPBACK 网段。在 R3 上发布 30.0 网段的 BGP 路由，同时配置 R4 为 RR，R3 为 RR client。R5 会同时从 R3 及 R4 收到 30 网段的路由更新，它将如何选路？那一条选路规则生效？

首先规则 8 不适用，因为两条 BGP 路由 NEXT\_HOP 相等，都是 3.3.3.3 不具有可比性（到达 3.3.3.3 的 metric 就一个值）

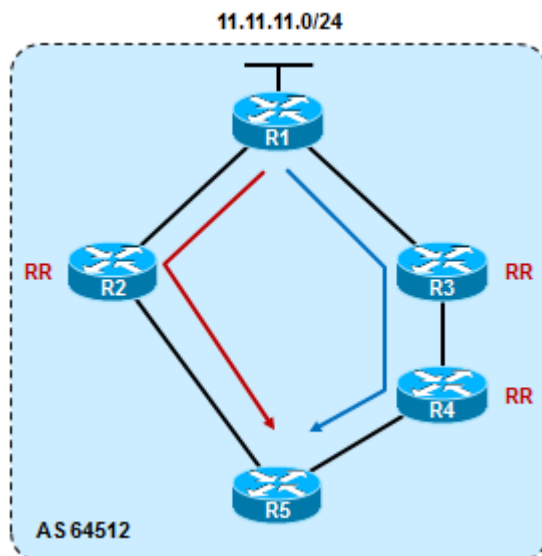
其次规则 9、10 略过

再次规则 11，似乎生效了，但是此时将 R3 的 BGP routerID 改的比 R4 大，发现 R5 仍然优选 R4，这是因为“如果一条路径包含 RR 属性，产生者 ID 将在最优路径选择过程中代替 RID”

因此到规则 12：如果多条路径始发路由器 ID 或路由器 ID 相同，那么优选 Cluster-List 最短的路径

## 12. 如果多条路径始发路由器 ID 或路由器 ID 相同，那么优选 Cluster-List 最短的路径

- 实验验证



```
R5#show ip bgp 11.11.11.0
```

BGP routing table entry for 11.11.11.0/24, version 2

Paths: (2 available, best #2, table Default-IP-Routing-Table)

Flag: 0x820

Not advertised to any peer

Local

1.1.1.1 (metric 129) from 4.4.4.4 (4.4.4.4)

Origin IGP, metric 0, localpref 100, valid, internal

Originator: 1.1.1.1, Cluster list: 4.4.4.4, 3.3.3.3

Local

1.1.1.1 (metric 129) from 2.2.2.2 (2.2.2.2)

Origin IGP, metric 0, localpref 100, valid, internal, best

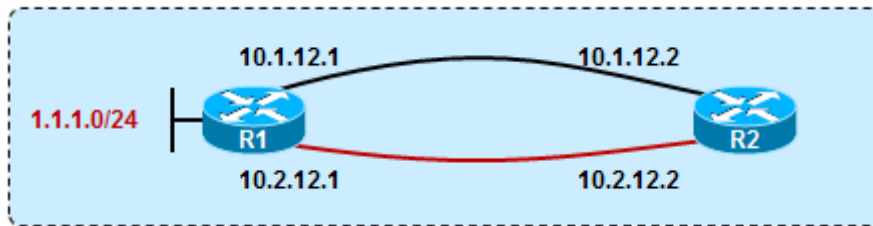
Originator: 1.1.1.1, Cluster list: 2.2.2.2

## 13. 选择邻居 ip 地址最小的路由 ( BGP 的 neighbor 配置中的那个邻居的地址，也就是邻居的更新源 IP )

- 规则详解

注意这个地址，是邻居用来建立 BGP 邻居关系的地址。

- 实验验证



R1R2 建立 IBGP 邻居关系，维护两条 BGP 连接，分别用本地的接口与对端的直连接口建邻居；  
R1 宣告 1.1.1.0 进 BGP，R2 会分别从 10.1.12.1 及 10.2.12.1 学习到这两条路由

BGP routing table entry for 1.1.1.0/24, version 2

Paths: (2 available, best #2, table Default-IP-Routing-Table)

Not advertised to any peer

Local

10.2.12.1 from **10.2.12.1** (1.1.1.1)

Origin IGP, metric 0, localpref 100, valid, internal

Local

**10.1.12.1** from **10.1.12.1** (1.1.1.1)

Origin IGP, metric 0, localpref 100, valid, internal, best

很明显 R2 优选了 10.1.12.1，这是因为前 12 条规则都无法决策，因此最后比较 neighbor IP，所以优选 10.1.12.1，当然，如果这时将 R1 的 10.1.12.1 地址改成 10.11.12.1，并且在 R2 上做 neighbor ip 的相应修改，那么 R2 去往 1.1.1.0 会切换到 10.2.12.1，因为这个 IP 小

### 补充说明：

1. 其中 3 意思是我自己 network 的

该比较原则主要是指本地在进入一条 IGP 路由进去 BGP 表时，使用不同的方式比如 network 或 redistribute 等，那么这些方式之间是存在优先顺序的：network>redistribute>aggregate

注意，该原则是不会作为 BGP 路由选路策略的

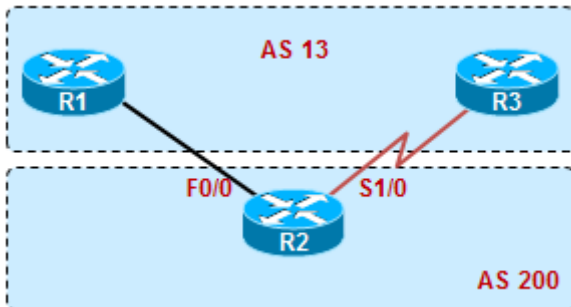
## 5.2 决策补充

如果一条路径满足下列任何一个条件，那么它在最佳路径选择过程中就不是有效的候选者

- 路径的下一跳不可达
- 路径未同步，但同步功能已开启
- 路径被入方向的 BGP 策略所拒绝，并且路由器配置了 soft reset
- 路由被惩罚 dampend

## 5.3 非等价负载均衡

### 1. EBGP 非等价负载均衡



在 R2 上，能分别从 R1 及 R3 上学习到 100.0 的 EBGP 路由，如果配置上 maximum-paths 2 则 R2 上，去往 100 网络就会同时将 R1、R3 作为下一跳进行等价负载均衡（虽然 R3 的上联口带宽更小，但是是直连）

**R2 的配置如下：**

```
router bgp 200
neighbor 10.1.12.1 remote-as 13
neighbor 10.1.23.3 remote-as 13
maximum-paths 2
```

**R2#sh ip b 100.100.100.0**

```
BGP routing table entry for 100.100.100.0/24, version 3
Paths: (2 available, best #1, table Default-IP-Routing-Table)
Multipath: eBGP
Flag: 0x860
  Advertised to update-groups:
    1
  13
    10.1.23.3 from 10.1.23.3 (3.3.3.3)
      Origin IGP, metric 0, localpref 100, valid, external, multipath, best
  13
    10.1.12.1 from 10.1.12.1 (1.1.1.1)
      Origin IGP, metric 0, localpref 100, valid, external, multipath
```

**R2#sh ip route**

```
B      100.100.100.0 [20/0] via 10.1.23.3, 00:02:57
      [20/0] via 10.1.12.1, 00:02:57
```

如何进行非等价负载均衡呢？这个图中，R2 的上联接口带宽其实是不等的。

**R2 的配置增加如下：**

```
router bgp 200
  bgp dmzlink-bw
  neighbor 10.1.12.1 dmzlink-bw
  neighbor 10.1.23.3 dmzlink-bw
  maximum-paths 2
```

**R2#sh ip b 100.100.100.0**

```
BGP routing table entry for 100.100.100.0/24, version 5
Paths: (2 available, best #1, table Default-IP-Routing-Table)
Multipath: eBGP
Flag: 0x860
  Advertised to update-groups:
    1
  13
    10.1.23.3 from 10.1.23.3 (3.3.3.3)
      Origin IGP, metric 0, localpref 100, valid, external, multipath, best
      DMZ-Link Bw 193 kbytes
  13
    10.1.12.1 from 10.1.12.1 (1.1.1.1)
      Origin IGP, metric 0, localpref 100, valid, external, multipath
      DMZ-Link Bw 12500 kbytes
```

**R2# sh ip ro 100.100.100.0**

```
Routing entry for 100.100.100.0/24
  Known via "bgp 200", distance 20, metric 0
  Tag 13, type external
  Last update from 10.1.12.1 00:07:30 ago
  Routing Descriptor Blocks:
    10.1.23.3, from 10.1.23.3, 00:07:30 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
      Route tag 13
  * 10.1.12.1, from 10.1.12.1, 00:07:30 ago
```

Route metric is 0, **traffic share count is 60**

AS Hops 1

Route tag 13

F0/0 口带宽为 100M，S0/0 口带宽为 1.544M，除一下，刚好比为将近 1:60

## 2. IBGP 非等价负载均衡

DMZ-Link Bw 193 kbytes 是一个扩展的 community 属性，因此，如果希望将该值传递给 AS 内的恰 IBGP 邻居，需加上 send-community，并且加扩展关键字；

DMZ-Link Bw 只能对本路由器的单跳 EBGP 邻居配置

# 6 BGP 策略

## 6.1 Weight

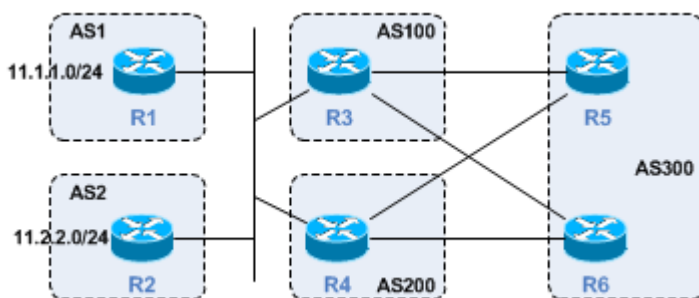
**neighbor 1.1.1.1 weight 2000**

将邻居发送的路由均设置为 2000 weight (本地)

**neighbor 1.1.1.1 filter-list 1 weight xxx**

将邻居发送来的、且被 filter-list 1 匹配的条目，weight 设置为 xxx

- 注意 neighbor 1.1.1.1 filter-list x weight yyy 这条命令可以对一个邻居使用多次，也就是对不同的路由设置不同的 weight。但是 neighbor 1.1.1.1 filter-list 1 这条命令，在同一个邻居 in 或 out 方向，只能用一次，这两条命令是有很大区别
- 如果同时使用 neighbor 1.1.1.1 weight 和 neighbor 1.1.1.1 filter-list 1 weight xxx，则后者优先



R3、R4 同时向 R5 及 R6 更新 AS1 及 AS2 内的路由，如果要求 R5 去往 AS1 的路由走 R3，去往 AS2 的路由走 R4

那么在 R3 上的配置：

```
ip as-path access-list 1 permit _1$ //匹配 AS ( 100,1 ) 及 ( 200,1 )
ip as-path access-list 2 permit _2$ //匹配 AS (100,2 ) 及 ( 200,2 )
```



```
router b 300
```

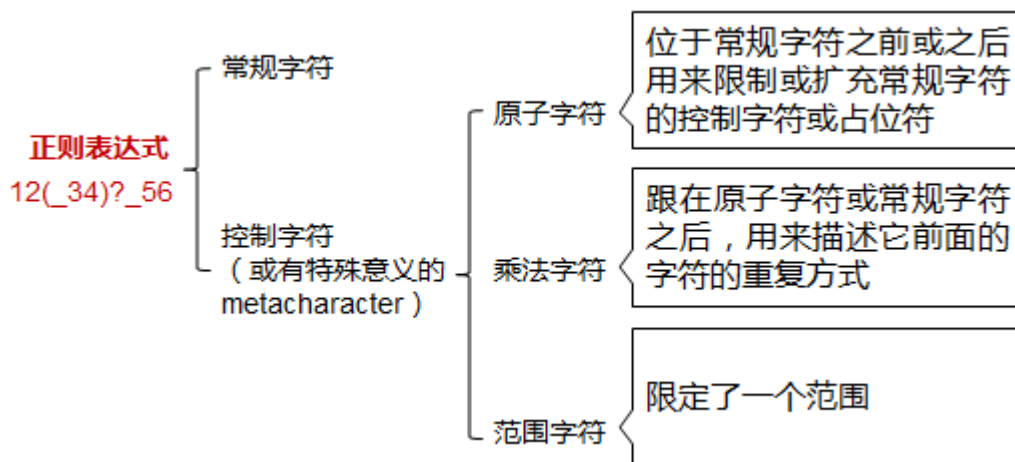
```
neighbor R3 filter-list 1 weight 4000
```

```
neighbor R3 filter-list 2 weight 5000
```

某些 IOS 不支持 neighbor R3 filter-list 1 weight 命令

## 6.2 正则表达式

### 1. 正则表达式的组成



#### ● 原子字符

.	匹配任何单个的字符，包括空格
^	一个字符串的开始
\$	一个字符串的结束
_	下划线，匹配任意的一个分隔符如 ^、\$、空格、tab、逗号、{、}
	管道符，逻辑或
\	转义符，用来将紧跟其后的控制字符转变为普通字符

#### 原子字符示例：

^a.\$	匹配一个以 a 开始，任意单一字符结束的字符串，如 a0，a!等
^100_	匹配 100、100 200、100 300 400 等
^100\$	匹配 100
100\$ 400\$	匹配 100、1400、300 400 等
^(65000\)\$	仅仅匹配(65000)
[123].[7-9]	匹配 123 中的一个加上任意字符再加上 7 到 9 中的一个，如 167、3 空格 7 等

## ● 乘法字符

*	匹配前面字符 0 次或多次出现
+	匹配前面字符 1 次或多次出现
?	匹配前面字符的 0 次或 1 次出现

一个乘法字符可以应用于一个单字符或多个字符，如果应用于多字符，需将字符串放入 ( ) 中。

乘法字符示例：

abc*d	匹配 abd、abcd、abccd、abcccd 等
abc+d	匹配 abcd、abccd、abcccd 等
abc?d	匹配 abd、abcd、abcdefg 等
a(bc)?d	匹配 ad、abcd、aaabcd 等

## ● 范围字符

[ ]	表示一个范围。只匹配包含在范围内的字符之一。 可以在一个范围的开始使用 ^ 来排除范围内的所有字符，也可以使用下划线 _ 来指定一个区间。
-----	--------------------------------------------------------------------------

范围字符示例：

[abcd]	匹配只要出现了 a、b、c、d 的内容
[a-c 1-2]\$	匹配 a、a1、62、1b、xv2 等
[^act]\$	匹配不以 a 或 c 或 t 结尾的内容
[123].[7-9]	159 220、91 70

## 2. 使用 as-path access-list 匹配路由

注意 as-path access-list 也是默认隐含拒绝所有

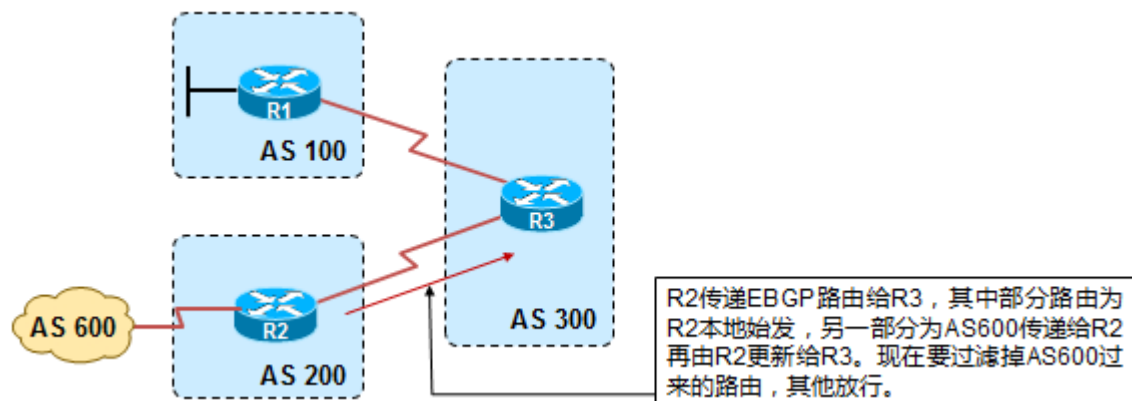
可使用 show ip bgp regexp 来检查所配置的正则表达式的结果。

正则表达式示例：

^\$	匹配不包含任何 AS 号的 AS_PATH，也就是本 AS 内的路由
.*	一个点和一个星号，匹配所有，任何
^100\$	就匹配 100 的这个 AS_PATH
_100\$	以 100 结束的 AS_PATH，也就是路由起源于 100AS 的路由
^10[012349]\$	匹配 100、101、102、103、104、109 这些 AS_PATH

<code>^10[^\0-6]\$</code>	匹配除了 100~106 外的 AS_PATH
<code>^10.</code>	匹配 100~109，以及 10，因为 “.” 也包含空格
<code>^(100 200)\$</code>	匹配包含 100 及 200 的 AS_PATH
<code>12(_34)?_56</code>	匹配 12 56 及 12 34 56

● 配置示例 1：as-path access-list 搭配 filter-list



在 R2 上开启 loopback，IP 分别为 172.16.10.0/24、172.16.11.0/24

R2 的配置如下：

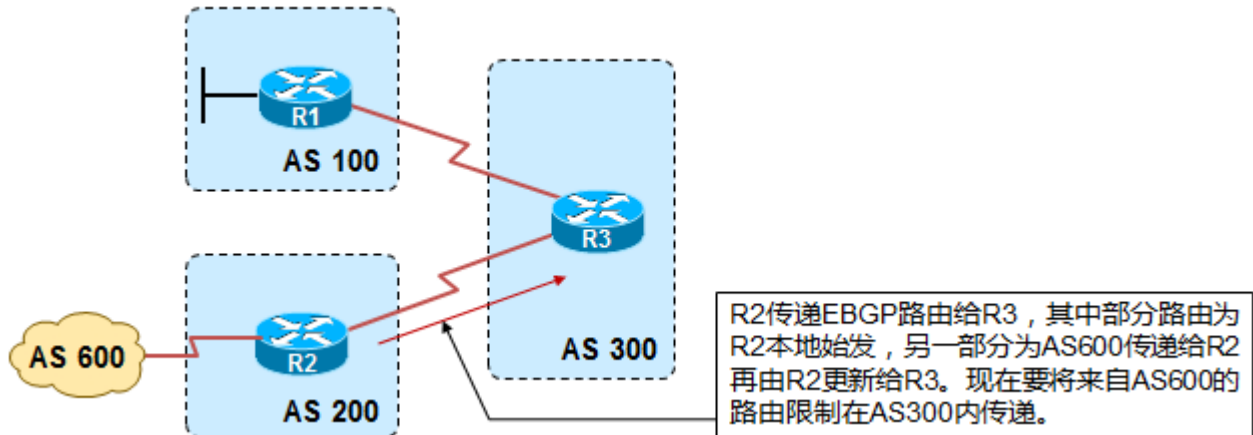
```
ip prefix-list 10 seq 5 permit 172.16.10.0/24
ip prefix-list 11 seq 5 permit 172.16.11.0/24
route-map test permit 10
  match ip address prefix-list 11
  set as-path prepend 600
route-map test permit 20
router bgp 200
  network 172.16.10.0 mask 255.255.255.0
  network 172.16.11.0 mask 255.255.255.0
  neighbor 10.1.23.3 remote-as 300
  neighbor 10.1.23.3 route-map test out
```

通过上述配置，模拟 R3 收到来自 AS200 及 AS600 的路由，此时在 R3 上要过滤掉来自 AS600 的路由，由于 AS600 的路由，AS\_PATH 的尾端均为 600，因此正则表达式可用：`_600$`，下划线用于匹配空格。但是注意，as-path access-list 也是默认隐含 deny any 的，因此要注意 deny 掉了来自 600AS 的路由，其他的路由要放行。R3 配置如下：

```
ip as-path access-list 1 deny _600$
ip as-path access-list 1 permit .*
router bgp 300
```

```
neighbor 10.1.13.1 remote-as 100
neighbor 10.1.23.2 remote-as 200
neighbor 10.1.23.2 filter-list 1 in
```

● 配置示例 2 : as-path access-list 搭配 route-map



R2 的配置与上面大同小异。关键看 R3 的配置：

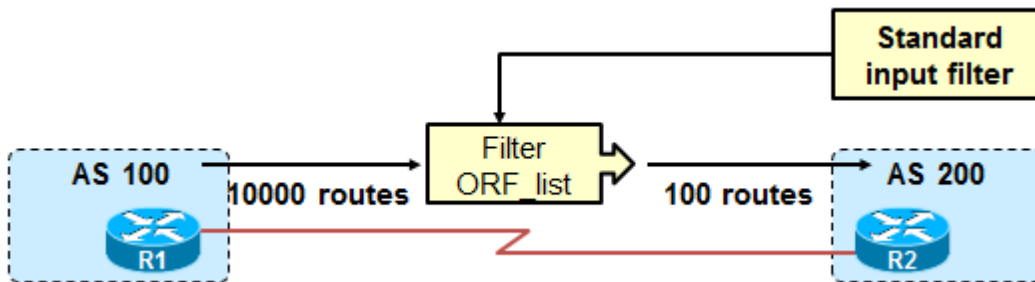
```
ip as-path access-list 1 permit _600$
route-map setCommu permit 10
match as-path 1
set community no-advertise
route-map setCommu permit 10
router bgp 300
neighbor 10.1.23.2 route-map setCommu in
```

### 3. 正则表达式在 CISCO IOS 中的其他应用

Show 和 more 的输出信息中，可使用管道符 | 搭配正则表达式来过滤输出结果。

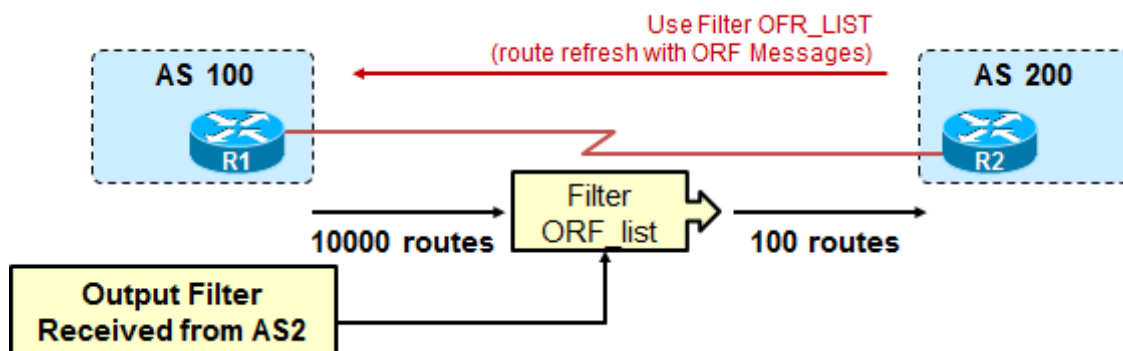
```
show run | in route
```

## 6.3 ORF

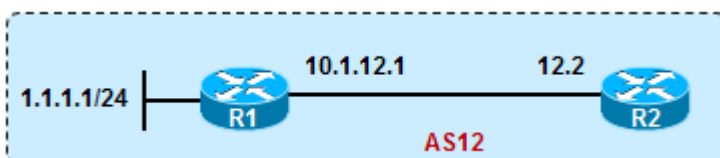


AS100 发了 10000 条路由给 R2，如果 R2 只希望收取其中 100 条路由，其余过滤，那么一般的做法是在 R2 上执行 in 方向的 filter，因为 AS100 对于 AS200 而言，是不可控的。但是这种方法，实际上对于链路带宽及网络资源的角度来说，是存在一定的浪费的。

ORF 的特性很简单，就是 R2 将这个 list 推送给 R1，让这个策略在 R1 这端就执行过滤，如下图。



配置如下：



**Sender ( R2 ) 省去基本配置，如 BGP 邻居关系等**

```
router bgp 12
  address-family ipv4 unicast
  neighbor 10.1.12.1 capability orf prefix-list send
  neighbor 10.1.12.1 prefix-list FILTER in
```

```
ip prefix-list FILTER deny 1.1.1.0/24
ip prefix-list FILTER permit
```

**receiver ( R1 )**

```
router bgp 100
  address-family ipv4 unicast
  neighbor 10.1.12.2 capability orf prefix-list receive
```

所以实际上 ORF 是借助 prefix-list 去实现路由过滤的一个特性，在本地将 prefix-list 推送给对端，让对端来执行路由前缀的过滤。

一旦部署了 ORF，那么 BGP peer 之间在建立 BGP 邻居关系的时候，在 open 消息里就会去进行 ORF 的能力协商，如果协商成功了，则使用 route-refresh 报文推送 ORF 内容。

下图是协商能力参数的 open 消息内容（注意 ORF 部分）

```

OPEN Message
  Marker: 16 bytes
  Length: 56 bytes
  Type: OPEN Message (1)
  Version: 4
  My AS: 12
  Hold time: 180
  BGP identifier: 2.2.2.2
  Optional parameters length: 27 bytes
  Optional parameters
    Capabilities Advertisement (8 bytes)
    Capabilities Advertisement (4 bytes)
    Capabilities Advertisement (4 bytes)
    Capabilities Advertisement (11 bytes)
      Parameter type: Capabilities (2)
      Parameter length: 9 bytes
      Cooperative route filtering capability (9 bytes)
        capability code: Cooperative route filtering capability (130)
        capability length: 7 bytes
      Capability value
        Address family identifier: IPv4 (1)
        Reserved: 1 byte
        Subsequent address family identifier: Unicast (1)
        Number of ORFs: 1
        ORF Type: Cisco PrefixList ORF-Type (128)
        Send/Receive: Send (2)

```

下图是 sender 使用 route-fresh 报文推送 prefix-list

## Border Gateway Protocol

### ROUTE-REFRESH Message

Marker: 16 bytes

Length: 44 bytes

Type: ROUTE-REFRESH Message (5)

Address family identifier: IPv4 (1)

Reserved: 1 byte

Subsequent address family identifier: Unicast (1)

#### ORF information (21 bytes)

ORF flag: Immediate

ORF type: Cisco PrefixList ORF-Type

ORF len: 17 bytes

#### ORFEntry-PrefixList (10 bytes)

ACTION: Add MATCH: deny

Entry Sequence No: 5

PrefixMask length lower bound: 0

PrefixMask length upper bound: 0

1.0.0.0/8

prefix-list  
第一个条目

#### ORFEntry-PrefixList (9 bytes)

ACTION: Add MATCH: Permit

Entry Sequence No: 10

PrefixMask length lower bound: 0

PrefixMask length upper bound: 32

0.0.0.0/0

ORF prefix length: 0

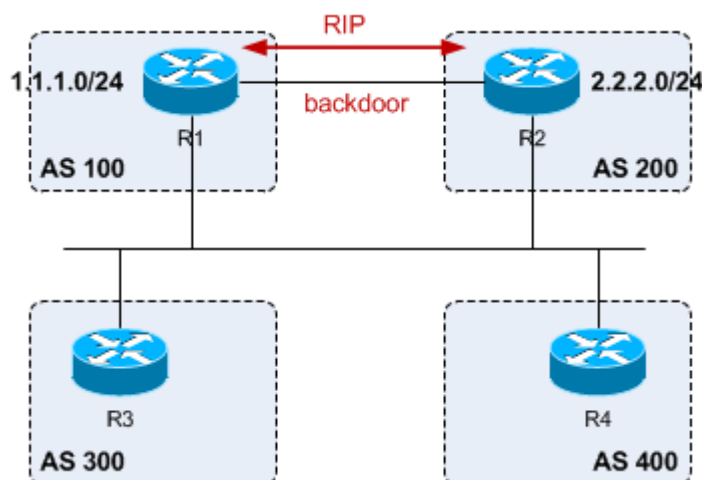
ORF prefix: 0.0.0.0

prefix-list  
第二个条目

## 6.4 AD 值及后门

修改 BGP 的默认 AD 值，命令如下：

distance bgp x y z //x 为 EBGp 路由，y 为 IBGP 路由，z 为本地 network 宣告的 BGP 路由



R1，R2 之间，原先与 R3 和 R4 是通过一个多路访问网络更新路由，跑的是 BGP。



R1 及 R2 由于是私密合作伙伴，有些特殊的流量可能需要受保护，因此增加了 R1、R2 之间的专线，这时如果需保证 1.1.1.0 及 2.2.2.0 间的流量互访通过专线(跑了 RIP)走，当专线 DOWN 了，才走下面的多路访问网络。

虽然 R1、R2 之间运行了 RIP 协议，互相更新 1.1.1.0 及 2.2.2.0 的路由，但是，由于 R1 同时从 RIP 及他的 EBGp 邻居学习到 2.2.2.0，RIP 的管理距离为 120，EBGP 为 20，所以优选 EBGp 的路由。如何使 R1 优选 RIP 路由呢？

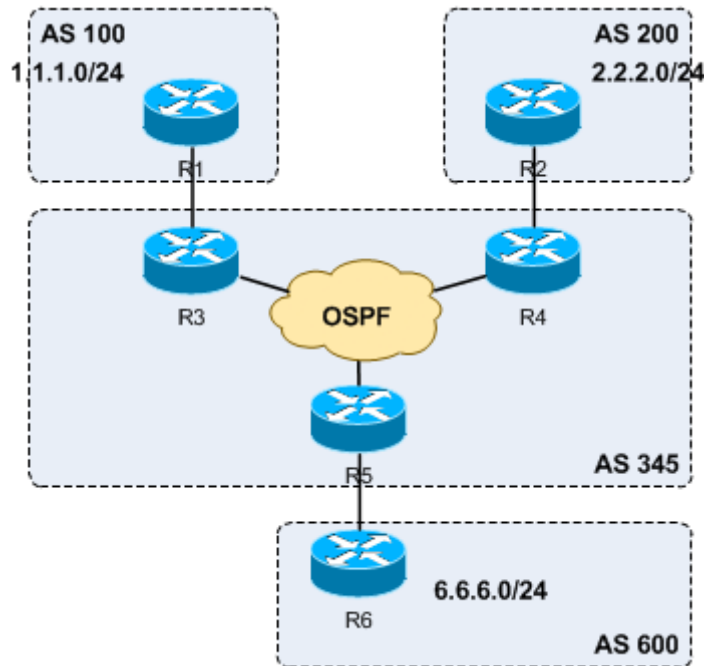
- **解决方案 1**，修改 BGP 的默认 AD 值，使得其 EBGp 路由 AD 值大于 RIP 从而达到优选的目的，但是这个方案未必合适，因为他会对所有的 BGP 路由都产生印象，如果仅仅希望对 1.1.1.0 及 2.2.2.0 产生影响呢？
- **解决方案 2**，由于 BGP 本地路由（本地 network 引入的）默认的 AD 值为 200，比 RIP 大，因此如果在 R1 上，将其从 EBGp 邻居学习到的 2.2.2.0 路由（此时为 EBGp 路由，AD 为 20），同时在本地的 network 一下（变成 BGP 本地路由），那么这条路由的 AD 将变为 200，比 RIP 大，所以此时优选 RIP 路由。R2 同理，在 BGP 进程中 network 1.1.1.0

这个方案貌似可以解决问题，但是却带来了新的问题，首先以 R1 为例，R1 采用重发布的方式引入 1.1.1.0，这条路由通过 BGP 的方式被更新给了 R2、R3、R4（通过多路访问链路），这时候在这些路由器上，1.1.1.0 路由的 original 为 incomplete；而为了实现上面的需求，R2 在其本地 network 了 1.1.1.0，这一来，1.1.1.0 在 R2 当做本地路由引入 BGP，AD 值变为 200 的同时，R2 向它的 EBGp 邻居通告 1.1.1.0，而其通告的 1.1.1.0，original 属性为 i，如此 R3、R4 同时从 R1 及 R2 学习到 BGP 路由 1.1.1.0，经过决策，去往 1.1.1.0 的流量会从 R2 发往 R1，这样就次优路径了。

- **解决方案 3**，所以上面的问题在于，R2 不应该向其 EBGp 邻居通告 1.1.1.0，因此 CISCO 支持使用命令 network 1.1.1.0 backdoor 来解决该问题，一来该路由变成了本地路由，AD 值成了 200，而来路由器不会向其 EBGp 邻居通告该路由。

## 6.5 路由标记 TAG (待修订)

支持 TAG 的路由协议有：RIPv2、EIGRP、OSPF、ISIS、BGP



R1、R2、R6 都通过 EBGP 通告了各自 AS 的路由，在 AS345 中，这些路由被重发布进 OSPF，接着又在其他两台路由器重发布回了 BGP，然后被发布给他们的 EBGP 对等体。

这里由于中间隔了个 OSPF，而 OSPF 又不了解 BGP，因此在重发布过程中，BGP 的属性就会被丢失。

BGP 可以利用 OSPF 包中的路由标记在穿越 OSPF 时携带 AS\_PATH 信息。实际上 CISCO 的 BGP 自动完成了该过程（注意，自动进入 OSPF TAG 字段的只有 AS\_PATH 属性一值）。

例如在 R3 上查看 R5 发过来的 6.6.6.0 这条 OSPF 外部路由，可以看到 tag 字段为 600（自动继承了 AS\_PATH）

而这个时候，R3 将 OSPF 重发布进 BGP 时，默认是不会自动恢复 AS\_PATH 属性的，因此需要：

使用 set as-path tag，来恢复

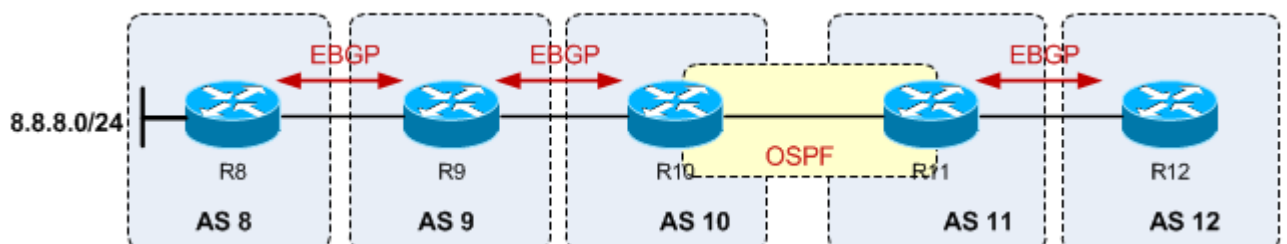
例如本例在 R3 上，

```
Route-map test per 10
```

```
Set as-path tag
```

```
router bgp 345
```

```
redis ospf 1 route-map test
```



R8 重发布 8.8.8.0 进 BGP，一直传到 R10，在 R10 上，该路由的 AS\_PATH 为 **9 8 ?**，

在 R10 上，将该路由重发布进 OSPF，R11 通过 OSPF 学习到了 8.8.8.0，  
在 R11 上，该路由为：

```
R11#sh ip ro 8.8.8.0
```

```
Routing entry for 8.8.8.0/24
```

```
Known via "ospf 1", distance 110, metric 1
```

```
Tag 9, type extern 2, forward metric 64
```

```
Last update from 192.168.101.10 on Serial0/0, 00:12:46 ago
```

```
Routing Descriptor Blocks:
```

```
* 192.168.101.10, from 10.10.10.10, 00:12:46 ago, via Serial0/0
```

```
Route metric is 1, traffic share count is 1
```

```
Route tag 9
```

**因此**：该路由被重发布进 OSPF 时，自动将这条外部路由继承 BGP 的 A\_PATH 属性，注意，只继承 AS\_PATH 列表中，第一个 AS 号（也就是去往目的地离本 AS 最近的那个）

这时候继续在 R11 上将 OSPF 路由重发布进 BGP，以便 R12 能学习到，R12 上学到路由后，发现 BGP 属性（包括 AS\_PATH）都丢失了。因为在 OSPF 重发布进 BGP 时，默认不会自动将 tag 值复制进 AS\_PATH，需在 R11 上

```
route-map test permit 10
```

```
set as-path tag
```

```
router bgp 11
```

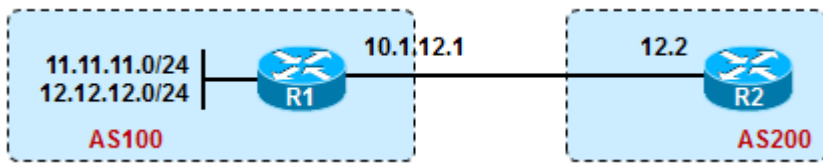
```
redistribute ospf 1 route-map test
```

配置后，发现软清路由没用，没办法，只能用 clear ip b \*重置 bgp 连接了。之后再 R12 上看到的 8.8.8.0 路由 AS\_PATH 为：11 9？

关于进一步的问题，请见 RFC 1403 - BGP OSPF Interaction

## 6.6 BGP 过滤器

## 6.7 Prefix-list

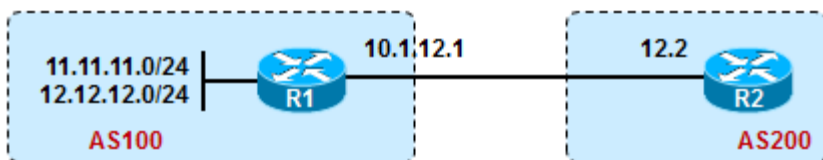


R1 与 R2 建立 EBGP，R1 上 network 宣告 11 及 12 网段，欲过滤掉 12 网段，不传其传递给 R2，可：

```
ip prefix-list 11 deny 12.12.12.0/24
ip prefix-list 11 permit 0.0.0.0/0 le 32
router bgp 100
 network 11.11.11.0 mask 255.255.255.0
 network 12.12.12.0 mask 255.255.255.0
 neighbor 10.1.12.2 remote-as 200
 neighbor 10.1.12.2 prefix-list 11 out
```

当然，在 R2 上做 in 方向的前缀列表进行过滤也是可以的

## 6.8 Distribute-list



R1 上，过滤掉 11.11.11.0/24 路由，其他放行

```
R1(config)# ip prefix-list 11 seq 5 deny 11.11.11.0/24
R1(config)# ip prefix-list 11 seq 10 permit 0.0.0.0/0 le 32
R1(config)# router bgp 100
R1(config-router)# network 11.11.11.0 mask 255.255.255.0
R1(config-router)# network 12.12.12.0 mask 255.255.255.0
R1(config-router)# distribute-list prefix 11 out
```

## 6.9 Route-map

- 可以在以下的 BGP 命令中使用 route-map  
neighbor

bgp dampening

network

redistribute

- 可以为特定的目的在不同的命令中使用 route-map

suppress-map

unsuppress-map

advertise-map

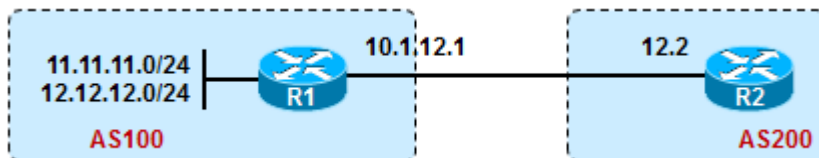
inject-map

exist-map

non-exist-map

tabel-map

- 示例 1：使用 route-map 关联 neighbor 过滤路由



```
R1(config)# access-list 1 deny 11.11.11.0
R1(config)# access-list 1 permit any
R1(config)# route-map rp permit 10
R1(config-route-map) match ip address 1
R1(config)# router bgp 100
R1(config-router)# neighbor 10.1.12.2 route-map rp out
```

## 6.10 Policy-list

### 1. policy-list 的概念

- 可预先将包含一组 match 语句的 route-map 定义成一个命令列表，这个列表称为 policy-list
- 这些 policy-list 可以在 route-map 中被调用
- 一个 policy-list 就像个只包含 match 语句的 route-map
- 当 route-map 被执行，被其调用的 policy-list 中所包含的 match 语句将一并被遍历且执行 match 动作
- 这是一种在大中型网络中运用、使得配置“模块化”的特性

### 2. policy-list 的特性

- Ipv6 不支持
- 12.0(22)S 和 12.2(15)T 之前的 CISCO IOS 版本不支持该特性，另外更老的 IOS 版本的路由器重启存在

路由策略配置丢失的风险

- Policy-list 中不能包含 set 语句，但是它被 route-map 调用后，该 route-map 中可以包含 set 语句
- Policy-list 只在 BGP 中支持，其他的 IP 路由协议并不支持这个特性

### 3. policy-list 的配置

```
ip policy-list as100 permit
  match as-path 1
  match community 1
```

上述命令创建一个 policy-list；Policy-list 只能包含 match 语句

```
route-map RP permit 10
  match policy-list as100
  match ip address prefix-list 100
  set local-preference 300
```

上述命令在 route-map 中调用定义好的 policy-list；一个 route-map 中可调用多个 policy-list

### 4. 配置示例

```
ip prefix-list 1 permit 10.0.0.0/8
ip as-path access-list 1 permit ^100_
ip as-path access-list 2 permit ^200_
ip community-list 1 permit 300:105
```

```
ip policy-list as100 permit
  match as-path 1
  match community 1
```

```
ip policy-list as200 permit
  match as-path 2
  match community 1
```

逻辑的或

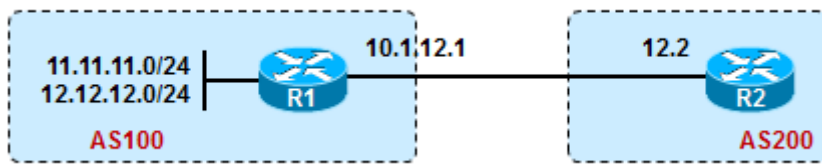
```
route-map Test permit 10
  match ip address prefix-list 1
  match policy-list as100 as200
  set local-preference 110
route-map Test permit 20
```

## 6.11 advertise-map

### 1. neighbor x.x.x.x advertise-map xx exist-map xx

跟下面其实是一条命令，这么敲进去，show 一下又变成下面的东东了。

### 2. neighbor xxx advertise-map yyy non-exist-map zzz

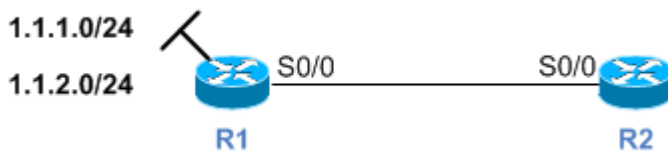


R1 上如若做如下配置：

```
ip prefix-list 1 permit 11.11.11.0/24
ip prefix-list 2 permit 12.12.12.0/24
route-map RP1 permit 10
  match ip address prefix-list 1
route-map RP2 permit 10
  match ip address prefix-list 2
router bgp 100
  neighbor 10.1.12.2 advertise-map RP1 non-exist-map RP2
```

如果 RP2 匹配的路由在 BGP 表中存在，则通告 RP2；如果 RP2 挂了，则通告 RP1  
亲测这个特性相当不稳定，不建议使用。

## 6.12 Unsuppress-map

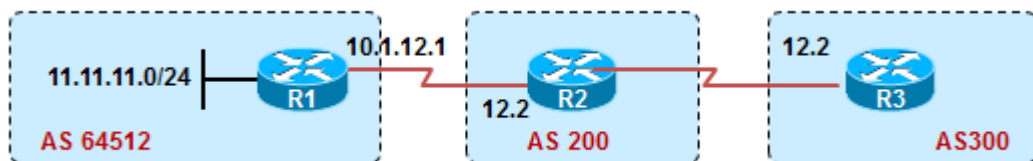


在 R1 上对 1.1.1.0 及 1.1.2.0 做汇总，成 1.0.0.0/8，如果加上 summary-only 关键字，则两个明细会被抑制掉  
如果希望对某个邻居取消抑制某条明细，如 1.1.1.0，则

```
ip prefix-list 1 seq 5 permit 1.1.1.0/24
route-map test permit 1
  match ip address prefix-list 1
router bgp 1
  neighbor R2 unsuppress-map test
```



## 6.13 私有 AS 号

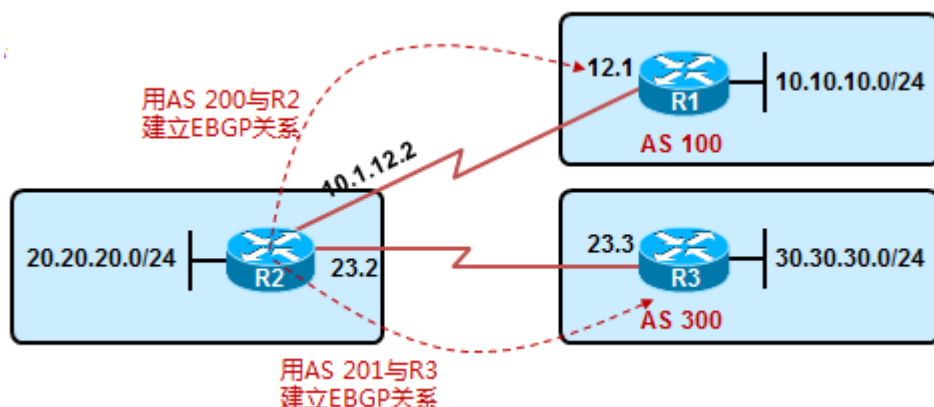


AS64512 为私有 AS 号，是不允许宣告到 Internet 的，如果不加限制，R3 收到 R1 的路由，AS\_PATH 肯定包含 64512 这个私有 AS 号，假设 R3 为 Internet 路由器，如何在 R2 上，向 R3 传递路由时，屏蔽掉私有 AS 呢？只需一条非常简单的命令：neighbor 10.1.23.3 remove-private-as

## 6.14 DUAL AS

### 1. 基本概念

- 默认情况下，在单台 Router 上只能启动一个 BGP 进程，并且只能属于一个 AS。DUAL AS 允许我们在不终端现有 BGP 连接的情况下，在 primary AS 下同时运行一个 secondary AS，从而提供一种网络迁移的机制。
- 在迁移期间，运行 DUAL AS 的路由器可以同时使用 primary AS 及 secondary AS 与外部 AS 建立 EBGP 连接，并且都能进行 BGP 路由的更新和传递。
- 在不用断开现有连接的情况下，可缩短网络迁移的时间，且不用大批量的变更设备 BGP 配置

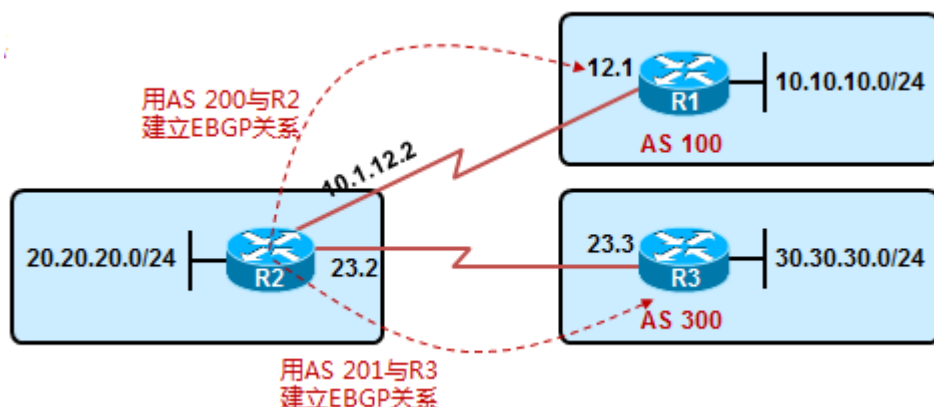


举个简单的例子：假设 R2 (AS200) 为一个 Customer Network，R1 (AS100) 是 R2 的 ISP，此时 BGP 邻居关系已经建立起来了，并且路由都已经收敛完毕，突然出现另一个 ISP-R3 (AS300)，并且它给 R2 分配的 AS 号为 AS201，这时候 R2 就很尴尬了，只能重新配置 BGP，断开 R1 的连接并与 R3 建立 BGP 连接。

DUAL AS 提供我们一种迁移的解决方案：允许 R2 的 BGP 进程同时兼顾两个 AS 号，200 及 201，其中 200 为 Primary AS，201 为 Secondary AS，R2 可以在保持原有的与 R1 的 BGP 连接及路由更新不受影响的情况下，

新增与 R3 的 BGP 连接，并且是使用 AS201 与 R3 ( AS300 ) 建立的 EBGP 邻居关系。

## 2. 配置命令



R1 的配置如下：

```
router bgp 100
 network 10.10.10.0 mask 255.255.255.0
 neighbor 10.1.12.2 remote-as 200
```

R3 的配置如下：

```
router bgp 300
 network 30.30.30.0 mask 255.255.255.0
 neighbor 10.1.23.2 remote-as 201 // R3 指 R2 的 remote-as 为 201，也即是 secondary AS
```

R2 的配置如下：

```
router bgp 200 // AS200 为 Primary AS，用于配置 BGP 进程
 network 20.20.20.0 mask 255.255.255.0
 neighbor 10.1.12.1 remote-as 100
 neighbor 10.1.23.3 remote-as 300
 neighbor 10.1.23.3 local-as 201 [no-prepend] [replace-as] [dual-as]
```

## 3. 命令详解

R1 发布 10.0、R2 发布 20.0、R3 发布 30.0 的路由，接下去看看 R2 上分别配置如下不同的命令后（关键字），这几条 BGP 路由被特定的路由器学习到之后的 AS\_PATH 的变化：

- neighbor 10.1.23.3 local-as 201

学习者	10.10.10.0/24	20.20.20.0/24	30.30.30.0/24
R1	本地始发	200	<b>200 201 300</b>
R2	100	本地始发	300
R3	<b>201 200 100</b>	<b>201 200</b>	本地始发

- neighbor 10.1.23.3 local-as 201 **no-prepend**

学习者	10.10.10.0/24	20.20.20.0/24	30.30.30.0/24
R1	本地始发	200	<b>200 300</b>
R2	100	本地始发	300
R3	201 200 100	201 200	本地始发

注意这里的变化：R2 将 30.0 的路由传递给 R1 时，不再插入 secondary AS 号

所以，no-prepend 关键字的作用是：Do not prepend local-as to updates from ebgp peers，翻过一下：向 Primary AS 的 EBGp 邻居通告路由时，不附加 secondary AS 号

- neighbor 10.1.23.3 local-as 201 no-prepend **replace-as**

学习者	10.10.10.0/24	20.20.20.0/24	30.30.30.0/24
R1	本地始发	200	200 300
R2	100	本地始发	300
R3	<b>201 100</b>	<b>201</b>	本地始发

注意变化，replace-as 关键字让 R2 向 secAS 的 EBGp 邻居发送路由时，用 local-as201 替代真实 AS200  
所以 replace-as 关键字的作用是：Replace real AS with local AS in the EBGp updates，翻译一下：当路由器向 secondaryAS 的 EBGp 邻居发送更新时，用 secondaryAS 号替代 Pri AS 号。

- neighbor 10.1.23.3 local-as 201 no-prepend replace-as **dual-as**

Accept either real AS or local AS from the ebgp peer。翻译一下：EBGP 对等体既可以使用 PriAS 也可以使用 SecAS 对本地指 remote-as，例如在 R2 上配置上述命令后，R3 也就是 10.1.23.3，在其 BGP 进程中，即可使用 neighbor 10.1.23.2 remote-as 200，亦可使用 remote-as 201 去指 BGP 邻居 R2

## 6.15 默认路由

### 1. 方法一：静态默认路由 network

Ip route 0.0.0.0 0.0.0.0 null0

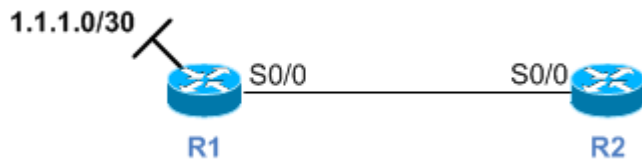
然后在 BGP 进程中 network 0.0.0.0 即可将此默认路由注入 BGP 进程

使用该方法在 BGP 中引入的默认路由会被传递给所有 BGP 邻居

### 2. 方法二：neighbor xxx default-originate

BGP 进程中：neighbor xxx default-originate 向特定的邻居传递默认路由

此配置无需路由表中存在默认路由。有点类似 OSPF 的 default-originate always



可以有条件的下发默认路由

```

access-list 1 permit 1.1.1.0 0.0.0.3
route-map test permit 10
  match ip address 1
router bgp 100
  neighbor R2 default-originate route-map test
  
```

如此只要 1.1.1.0 网络不 DOWN，R1 就始终会向 R2 下发默认路由

注意 access-list 1 的写法，实验的结果是，acl 必须匹配接口的 IP 和掩码，例如 如果写成 permit 1.1.1.0 0.0.0.255 就不行了

### 3. 方法三：default-information originate

在本地配置一条静态的默认路由，如 ip route 0.0.0.0 0.0.0.0 null0

默认情况下 BGP 在重发布 static 的时候，不允许注入静态默认路由；

除非在同时在 BGP 进程中 default-information originate

```

ip route 0.0.0.0 0.0.0.0 null0
router bgp x
  default-information originate
  redistribute static
  
```

## 6.16 其他配置

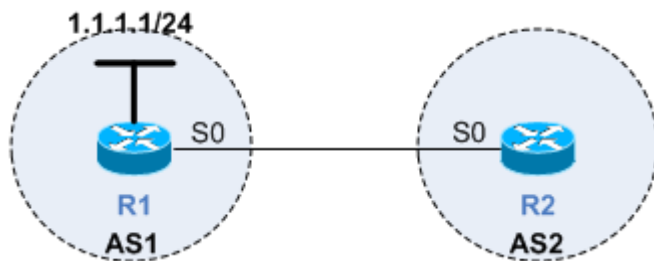
### 1. bgp fast-external-fallover

默认开启，如果一个接口 DOWN 了，BGP 会话立即中止。

如果一个接口处于 flapping 状态，那么将会给网络带来大量的 update 和 withdrawn 消息。因此对于一个翻动的接口，建议 no bgp fast-external-fallover，如此接口 shutdown 后，BGP 会话和邻居关系将仍在，直到 Holddown 计时器超时

## 7 故障案例分析

### ● 【案例】路由翻动



如图，R1、R2 开设 LOOPBACK 接口，地址分别为 1.1.1.1、2.2.2.2/24

R1、R2 之间运行 EIGRP，与此同时，R1、R2 在 EIGRP 进程中宣告各自的 LOOPBACK 接口

R1、R2 之间形成 EBGP 邻居关系( 使用各自 LOOPBACK 口 )与此同时，R1 在 BGP 进程中 network 1.1.1.0 mask 255.255.255.0 引入 1.1.1.0 网络

观察一段时间，R2 会出现什么故障？为什么会出现这样的故障？

## 8 参考书目

<b>TCP/IP 卷二</b>	本笔记包含 TCPIP 路由技术卷二 BGP 部分的几乎所有内容，并做了详细扩展
<b>CCNP ROUTE</b>	包含书中所有内容
<b>BGP 设计与实现</b>	推荐阅读
<b>RFC1771</b>	A Border Gateway Protocol 4
<b>RFC1403</b>	BGP OSPF Interaction