Ingeniería de tráfico con BGP

WALC 2012

Programa

- Multihoming
- Ingenieria de tráfico
- Implementando BGP

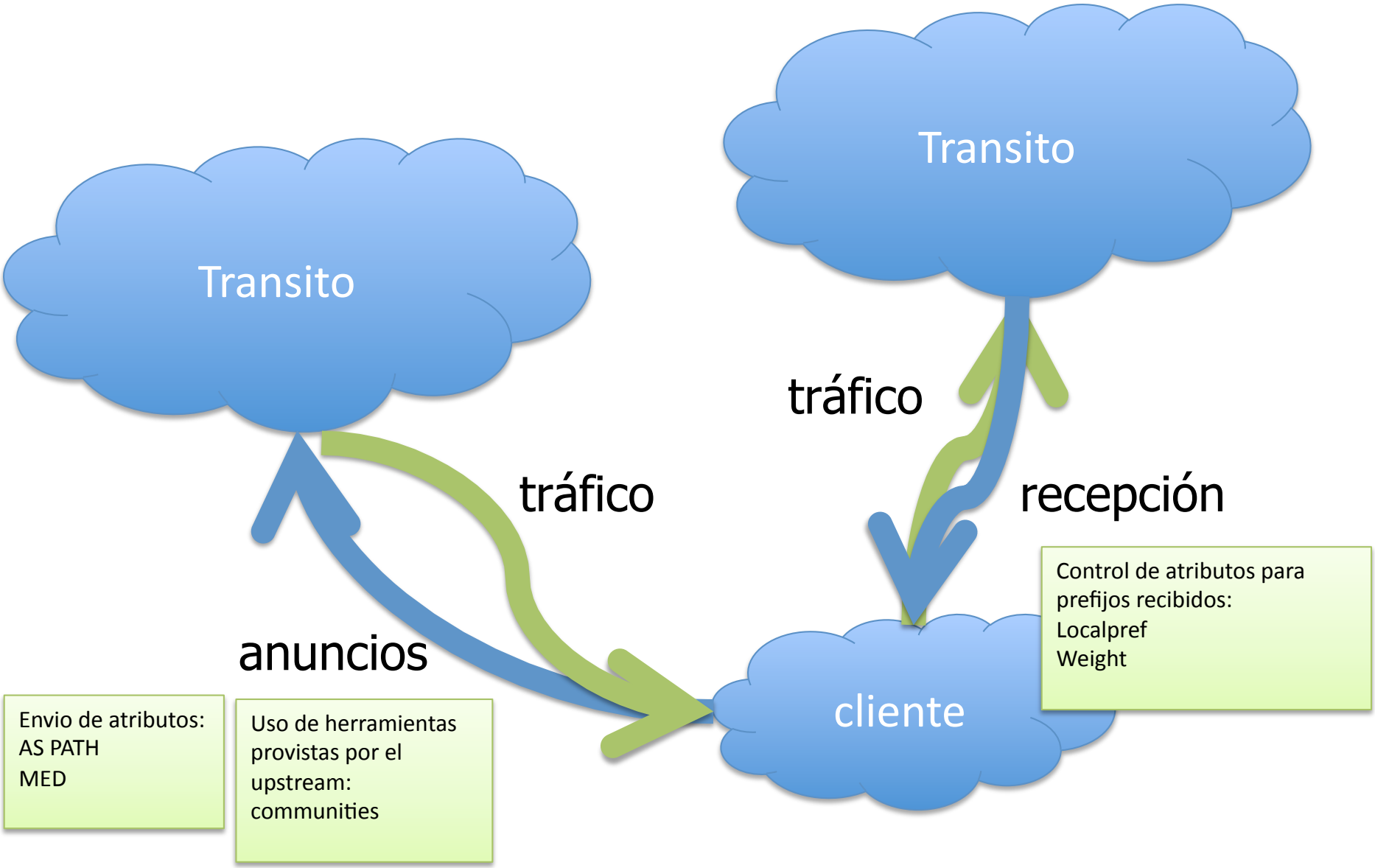
Repaso de BGP

Por qué utilizarlo?

En que casos se necesita BGP?

- Necesitas crecer (El IGP está con muchas rutas)
- Tienes tus propias IPs
- Tienes tu propio AS
- Eres un cliente con dos conexiones a ISPs (Multihoming)
- Necesitas recibir todas las rutas en Internet
- Necesitas implementar una politica de enrutamiento, balanceo, redundancia, etc.
- Quieres conectarte a un IXP (NAP)
- Quieres hacer peering con otro proveedor

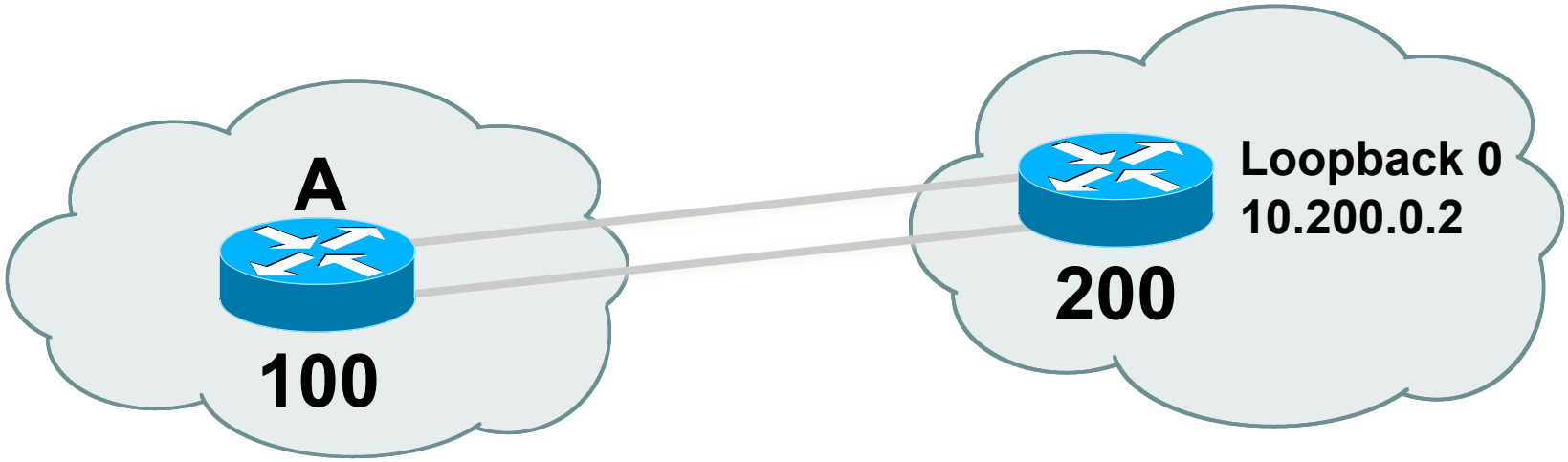
Resumen BGP TE



Conectando a un ISP

Router A:

```
interface loopback 0
ip address 10.60.0.1 255.255.255.255
!
router bgp 100
neighbor 10.200.0.2 remote-as 200
neighbor 10.200.0.2 update-source loopback0
neighbor 10.200.0.2 ebgp-multi-hop 2
```



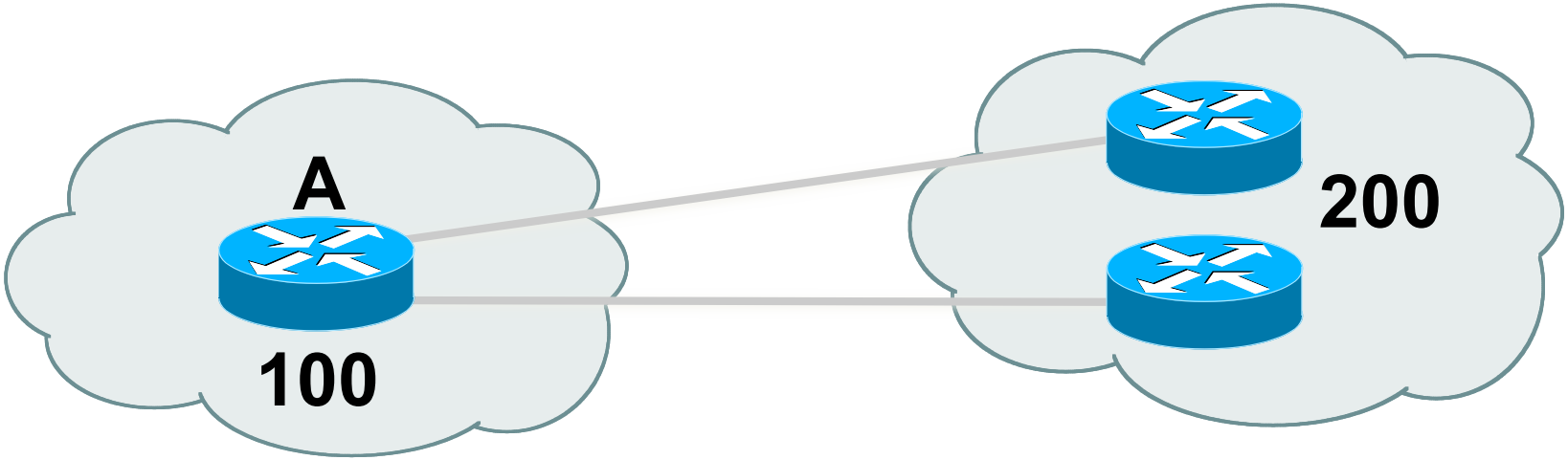
Balanceo de Cargas—Multiples Caminos desde el mismo AS

Router A:

router bgp 100

neighbor 10.200.0.1 remote-as 200

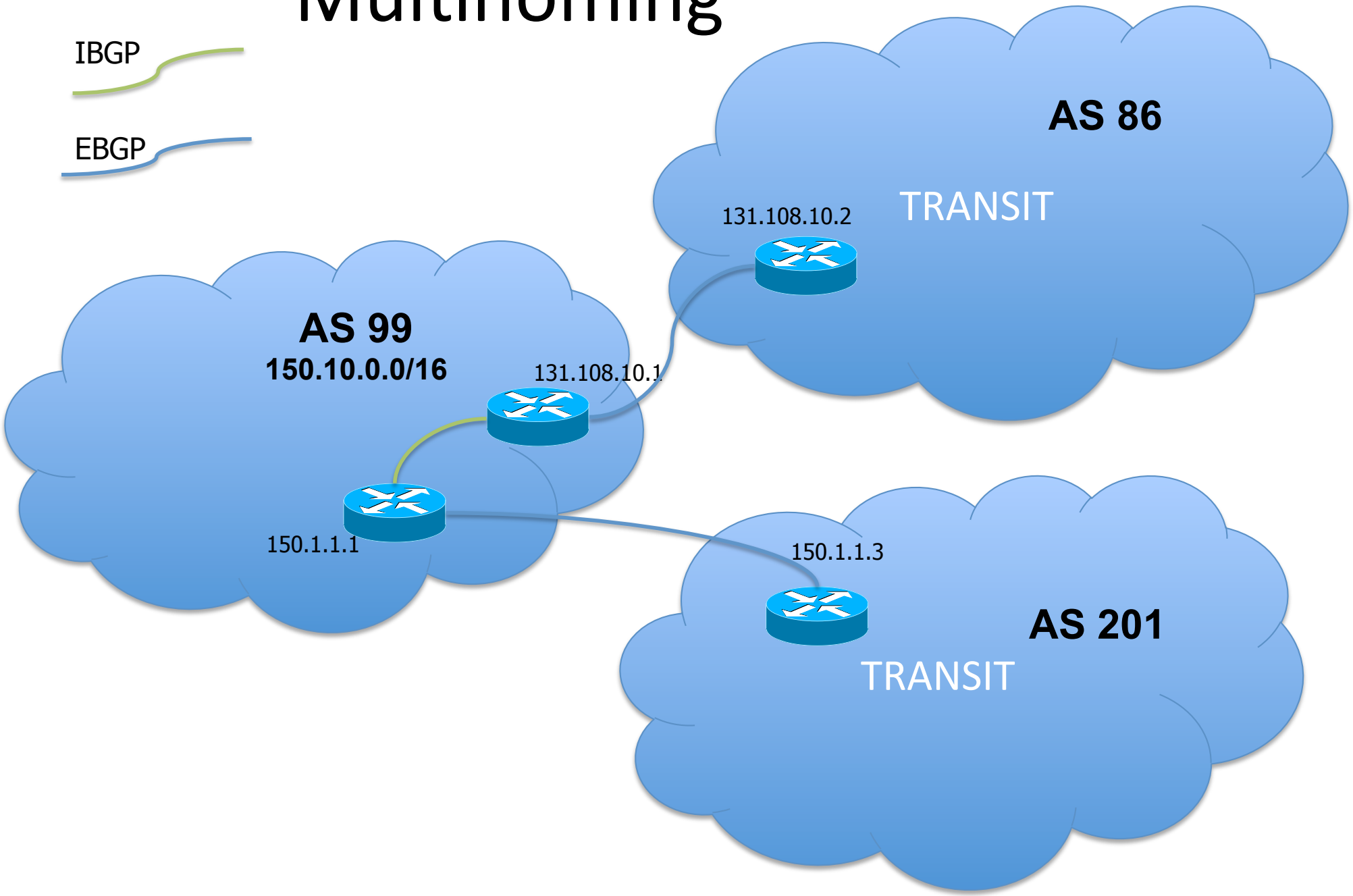
neighbor 10.300.0.1 remote-as 200



Qué es Multihoming?

- Conectarse a dos o más ISPs para incrementar:
  - Confiabilidad—si un ISP falla, todavía funciona
  - Desempeño—mejores caminos a destinos comunes en Internet

Multihoming



Tipos de Multihoming

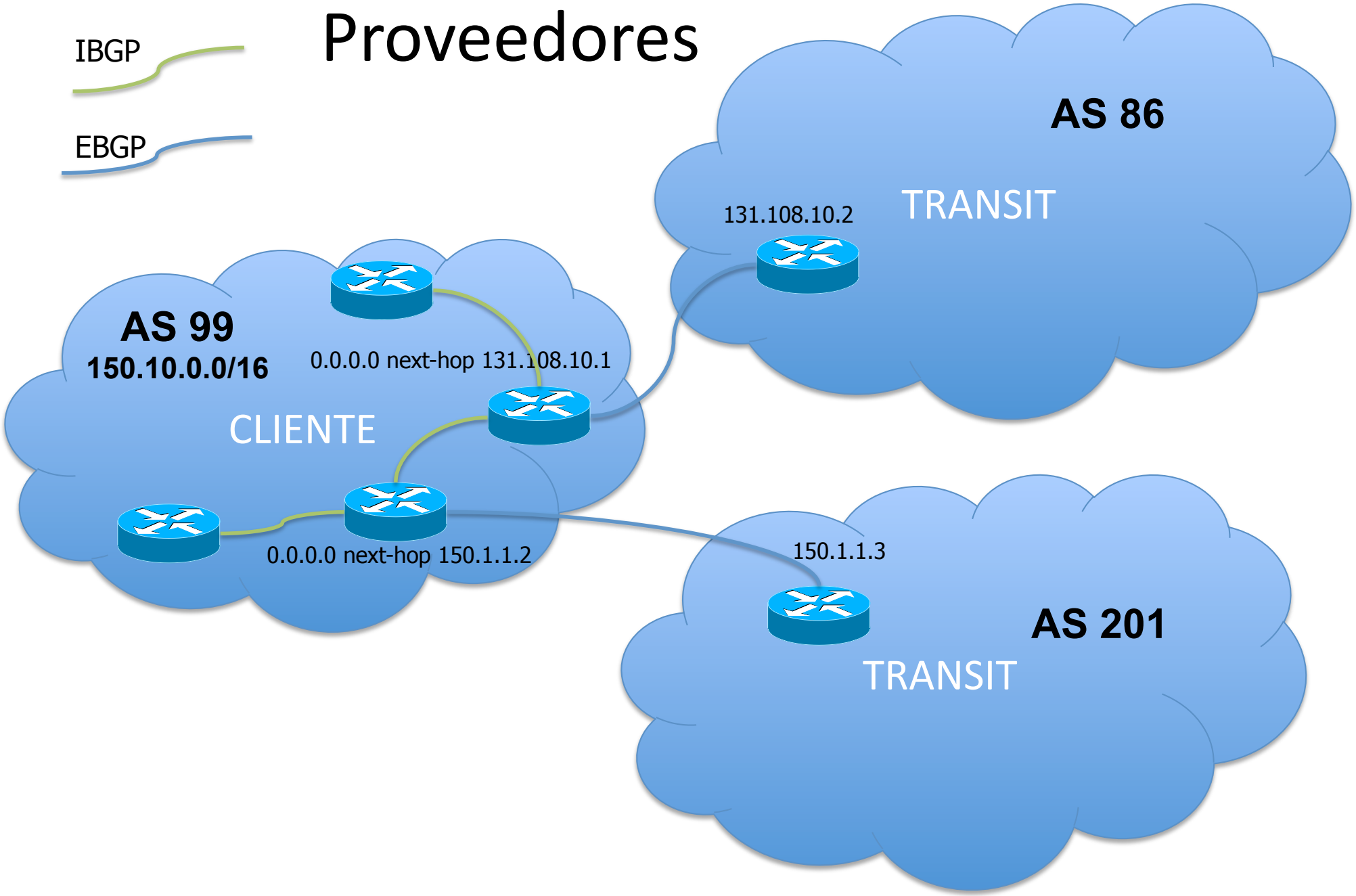
Tres casos comunes:

- Ruta Default de todos los proveedores
- Rutas de Clientes + Default de todos
- Rutas completas de todos

Recibir Default de Todos los Proveedores

- Bajos requerimientos de memoria/CPU
- Decide basado en la métrica de IGP cual es el default
- Camino de entrada decidido por “Internet”

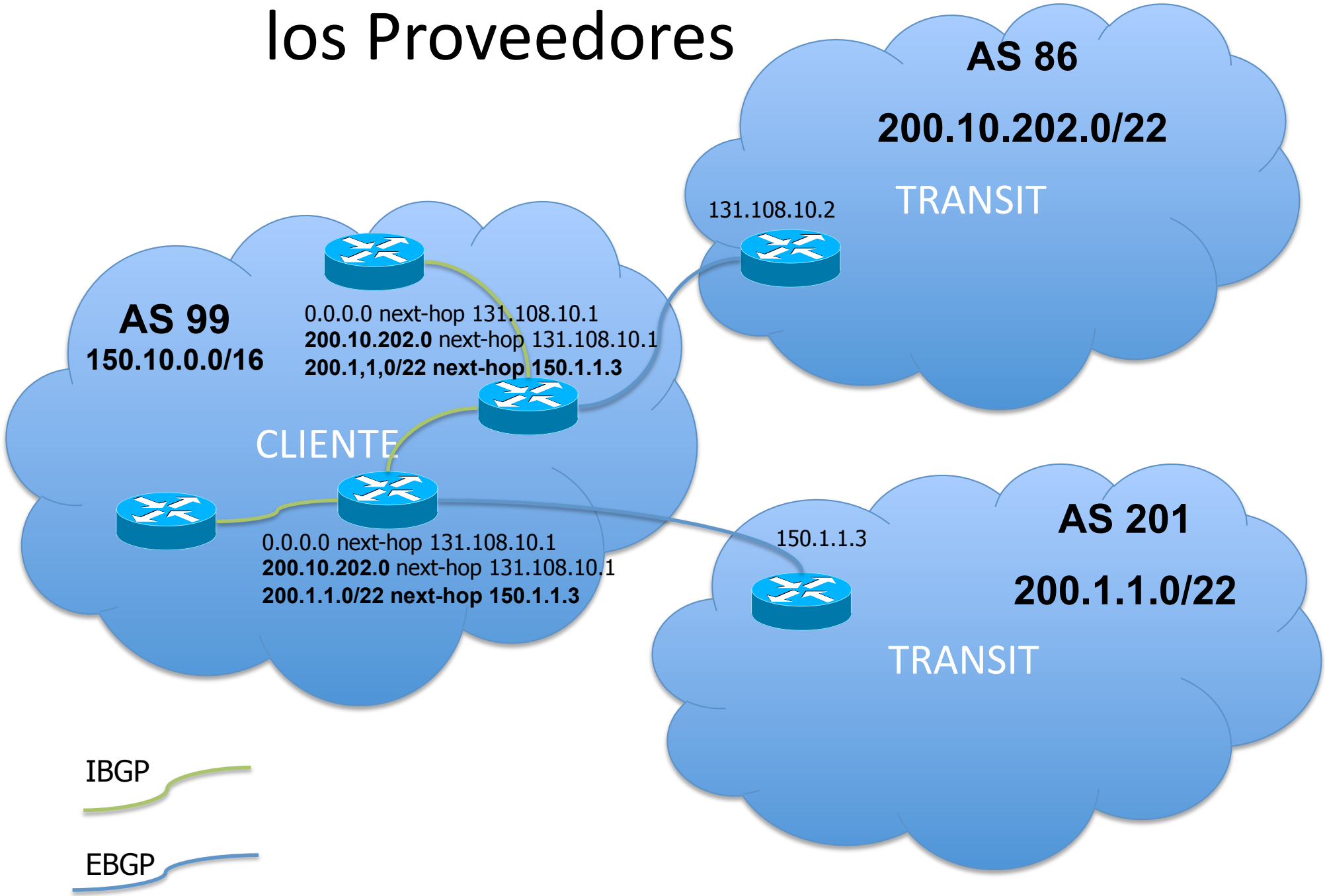
Default de Todos los Proveedores



Clientes + Default de Todos los Proveedores

- Uso bajo de memoria y CPU
- “Mejor” camino—usualmente el camino AS (AS PATH) más corto
- Se puede modificar basado en prefijo, as-path, o comunidad
- Métrica de IGP al default usado para todos los destinos

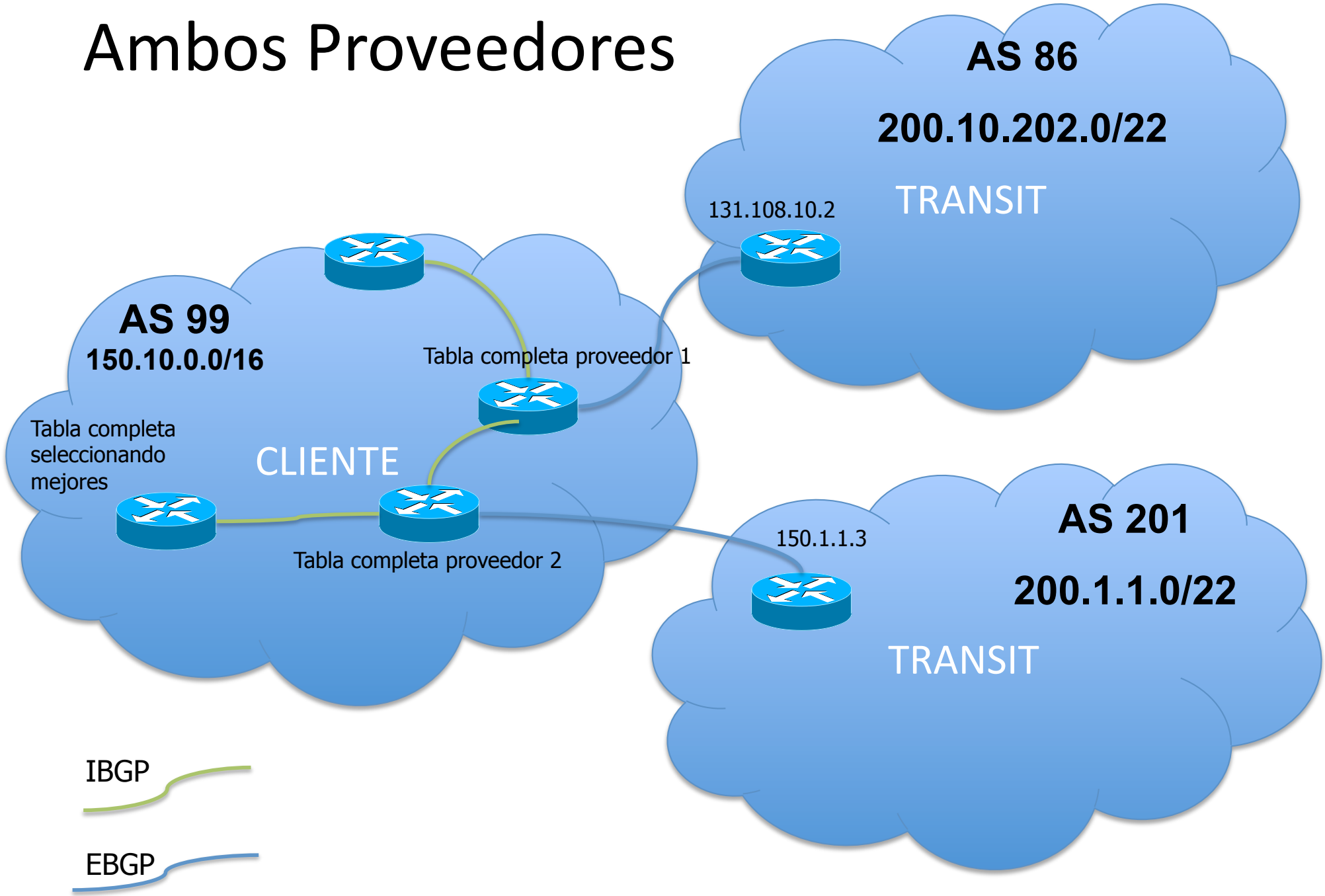
Clientes +Default de Todos los Proveedores



Rutas Completas de Ambos Proveedores

- Solución de mayores requerimientos de memoria/ CPU
- Alcanza todos destinos basado en el “mejor” camino—usualmente el que tiene el camino mas corto
- Todavía puede ajustar manualmente usando local-pref y comparación de as-path/comunidad/prefijo

Rutas Completas de Ambos Proveedores



Controlando la Entrada de Tráfico

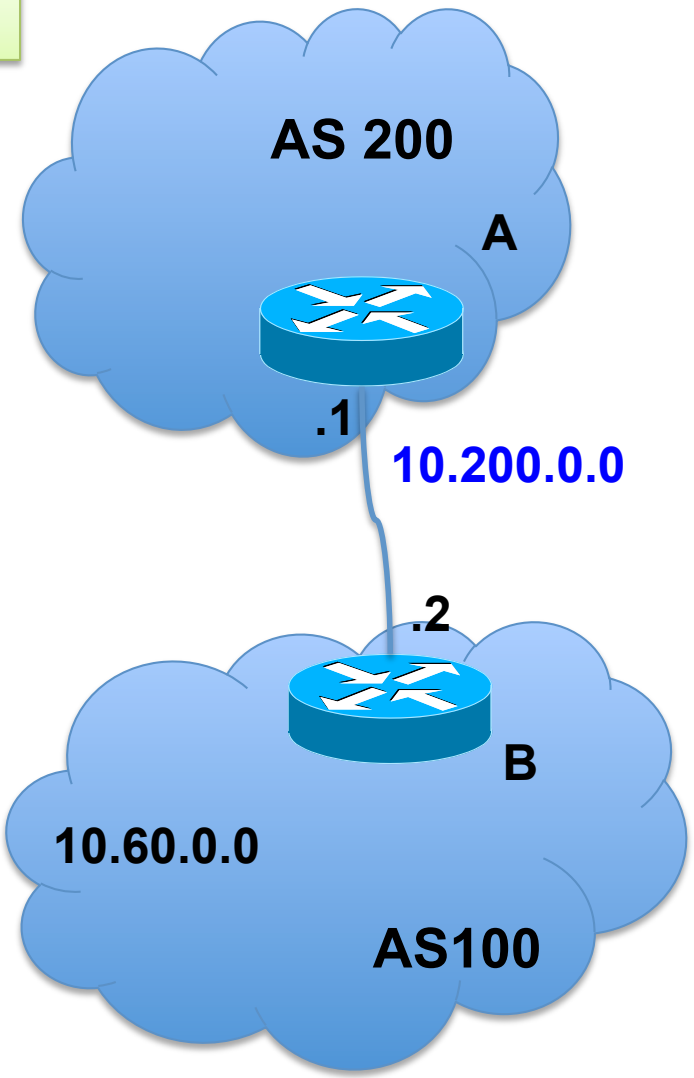
- La entrada es muy dificil de controlar debido a la falta de una métrica transitiva
- Puedes dividir los anuncios de prefijos entre los proveedores, pero no es correcto
- Recomendaciones:
  - Sumariza anuncios
  - Usa comunidades
  - Usa advertise-map

Consideraciones de Clientes

- Procedimiento
  - Configure BGP
  - Genere una ruta agregada estable
  - Configure la política de entrada
  - Configure la política de salida
  - Configure loadsharing/multihoming

Conectandose a un ISP

- **AS 100 es un cliente de AS 200**



Router B:

router bgp 100

aggregate-address 10.60.0.0 255.255.0.0 summary-only

neighbor 10.200.0.1 remote-as 200

neighbor 10.200.0.1 route-map ispout out

neighbor 10.200.0.1 route-map ispin in

Que es agregación?

- Hacer anuncios equivalente pero menor cantidad para reducir la tabla de ruteo global
  - 10.1.1.0 255.255.255.0
  - 10.2.0.0 255.255.0.0
  - => 10.0.0.0 255.0.0.0

Cómo Agregar?

- aggregate-address 10.0.0.0 255.0.0.0 {as-set} {summary-only} {route-map}
- Usar *as-set* para incluir la información de camino y comunidad basado en las rutas específicas
- *summary-only* suprime las rutas específicas
- route-map para configurar otros atributos

Por qué Agregar?

- Reducir el número de prefijos a anunciar
- Incrementar estabilidad — rutas agregadas se mantienen aún si las específicas son inestables
  - router bgp 100
  - aggregate-address 10.0.0.0 255.0.0.0 as-set summary-only
  - network 10.1.0.0 255.255.0.0

  - ip route 10.1.0.0 255.255.0.0 null0

BGP entre vecinos

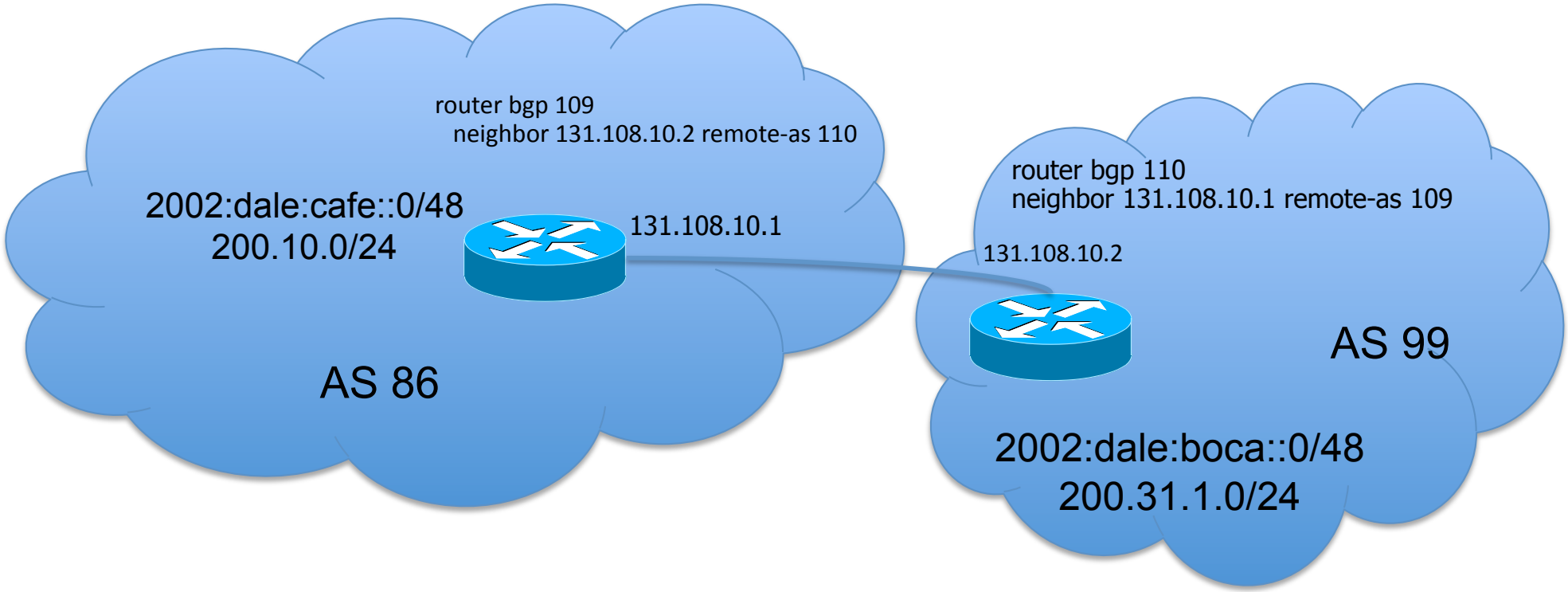
Anuncios de Prefijos	Eliminación de Prefijos	Cambio de Atributos en prefijos

Atributos de BGP Utilizados para Definir la Política de Enrutamiento

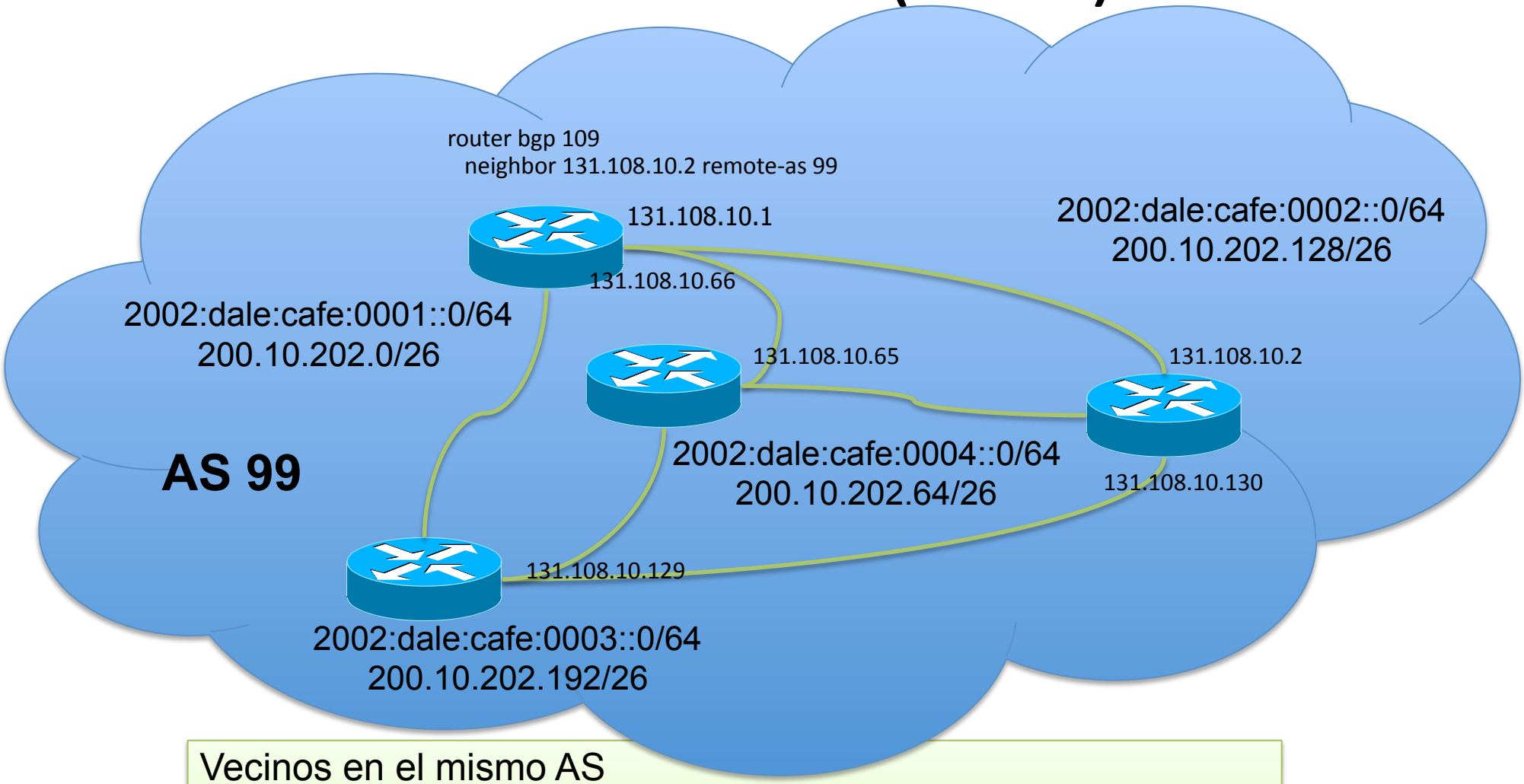
- ORIGIN
- AS-PATH
- NEXT-HOP
- MED
- LOCAL_PREF
- COMMUNITY

BGP Externo (eBGP)

- Entre routers en ASs diferentes
- Multihop
  - Entre interfaces
  - Entre loopbacks

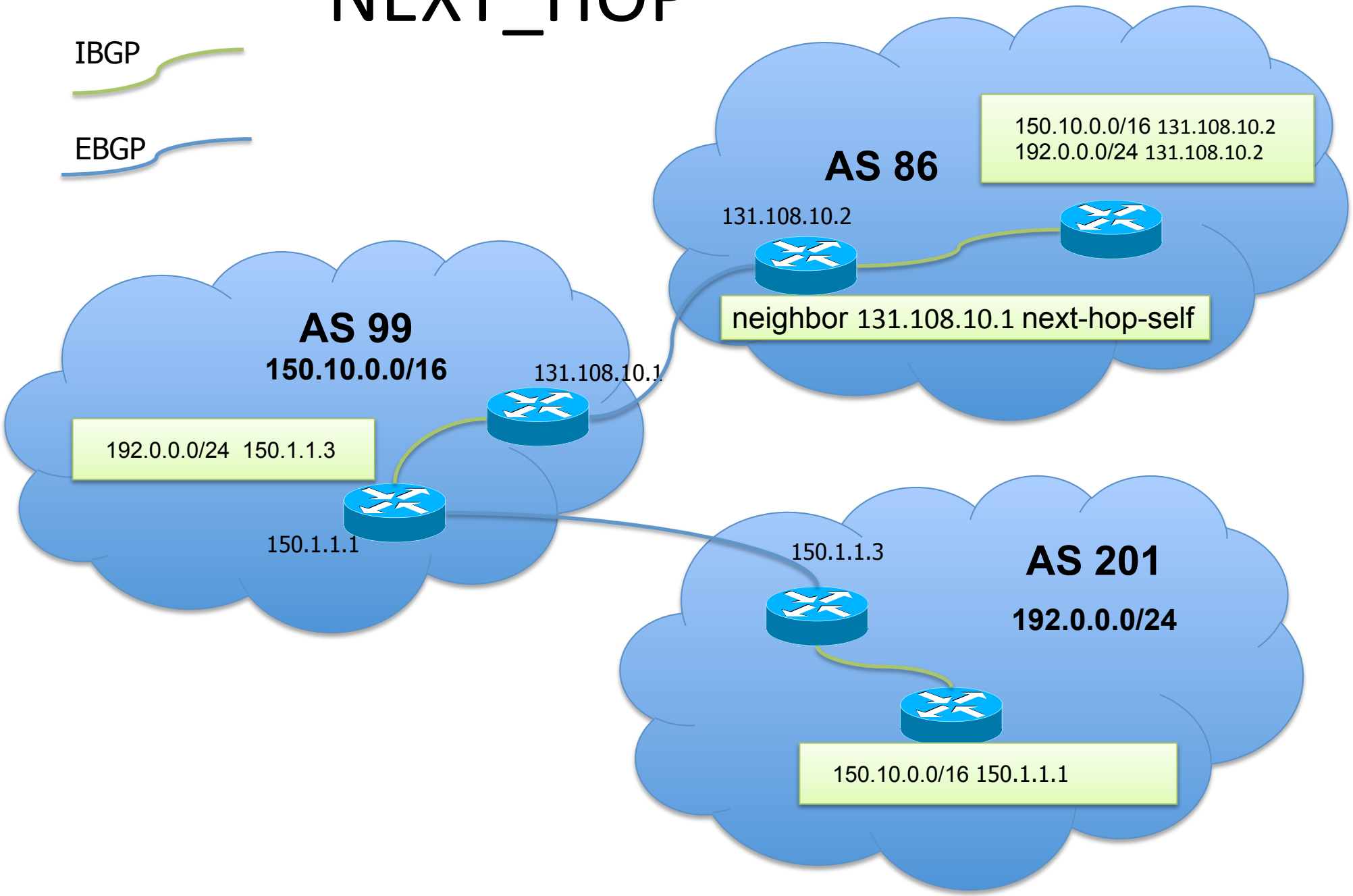


BGP Interno (iBGP)



Vecinos en el mismo AS
Todos tienen la misma tabla (No se modifica el Next-hop)
Todos vecinos BGP entre si (aunque no haya conexión directa)
No anuncia otras rutas aprendidas por iBGP

NEXT_HOP


IBGP
EBGP
AS 86
150.10.0.0/16 131.108.10.2
192.0.0.0/24 131.108.10.2
131.108.10.2
neighbor 131.108.10.1 next-hop-self
AS 99
150.10.0.0/16
131.108.10.1
192.0.0.0/24 150.1.1.3
150.1.1.1
150.1.1.3
AS 201
192.0.0.0/24
150.10.0.0/16 150.1.1.1

AS-PATH

Secuencia de ASs

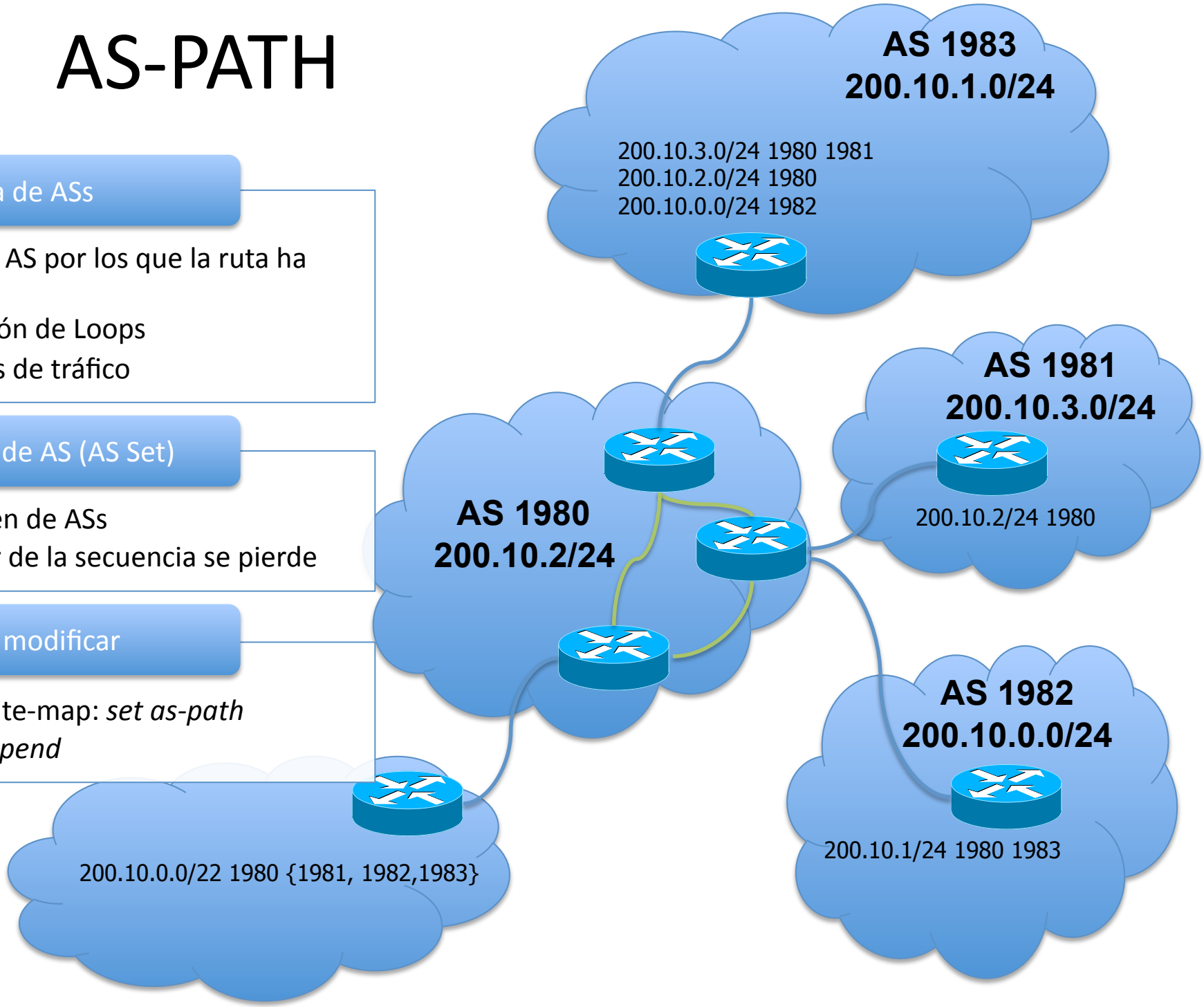
- Lista de AS por los que la ruta ha pasado
- Detección de Loops
- Políticas de tráfico

Conjunto de AS (AS Set)

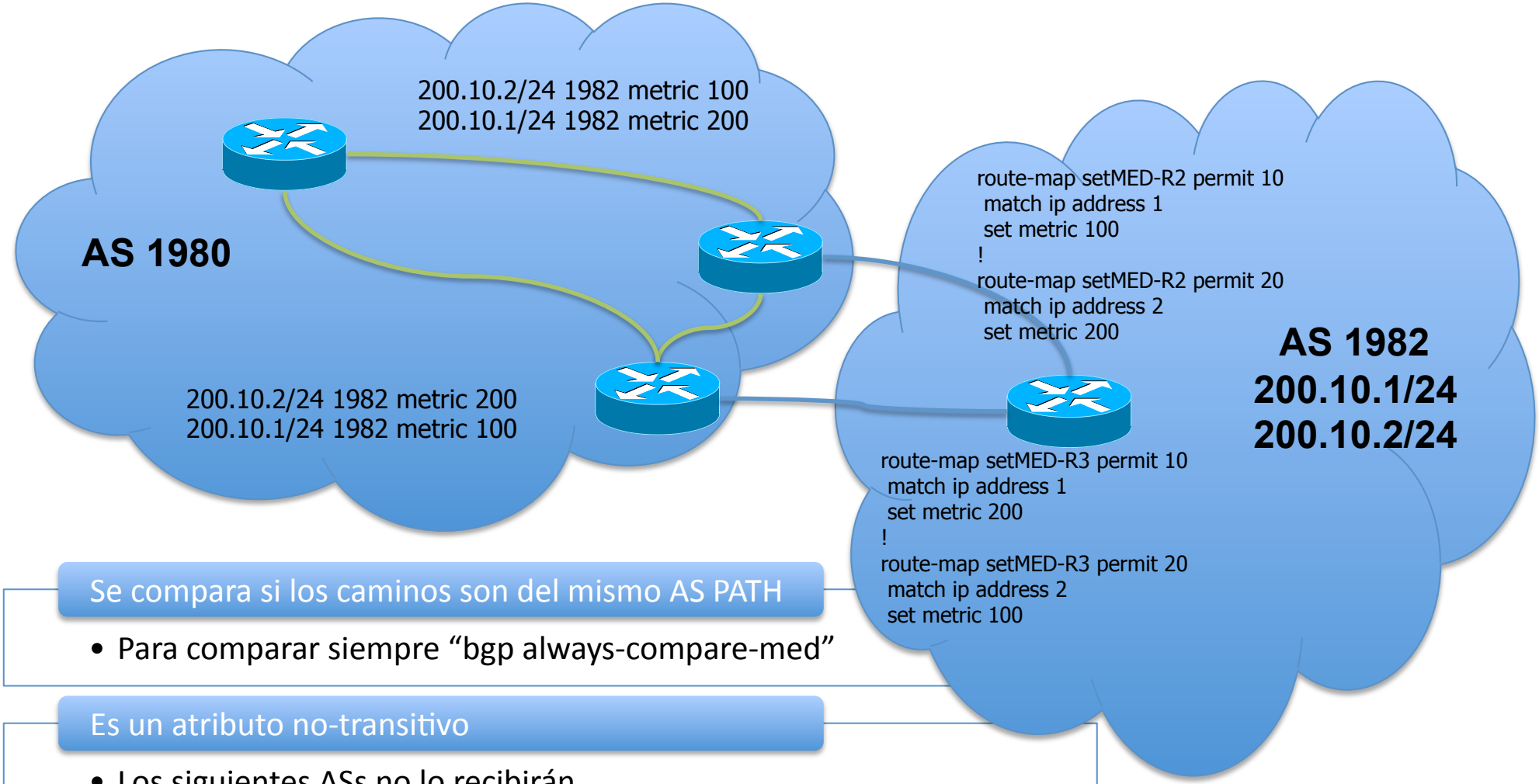
- Resumen de ASs
- El order de la secuencia se pierde

Se puede modificar

- Con route-map: *set as-path*
- *Con prepend*



MED: Multi Exit Discriminator



Se compara si los caminos son del mismo AS PATH

- Para comparar siempre “bgp always-compare-med”

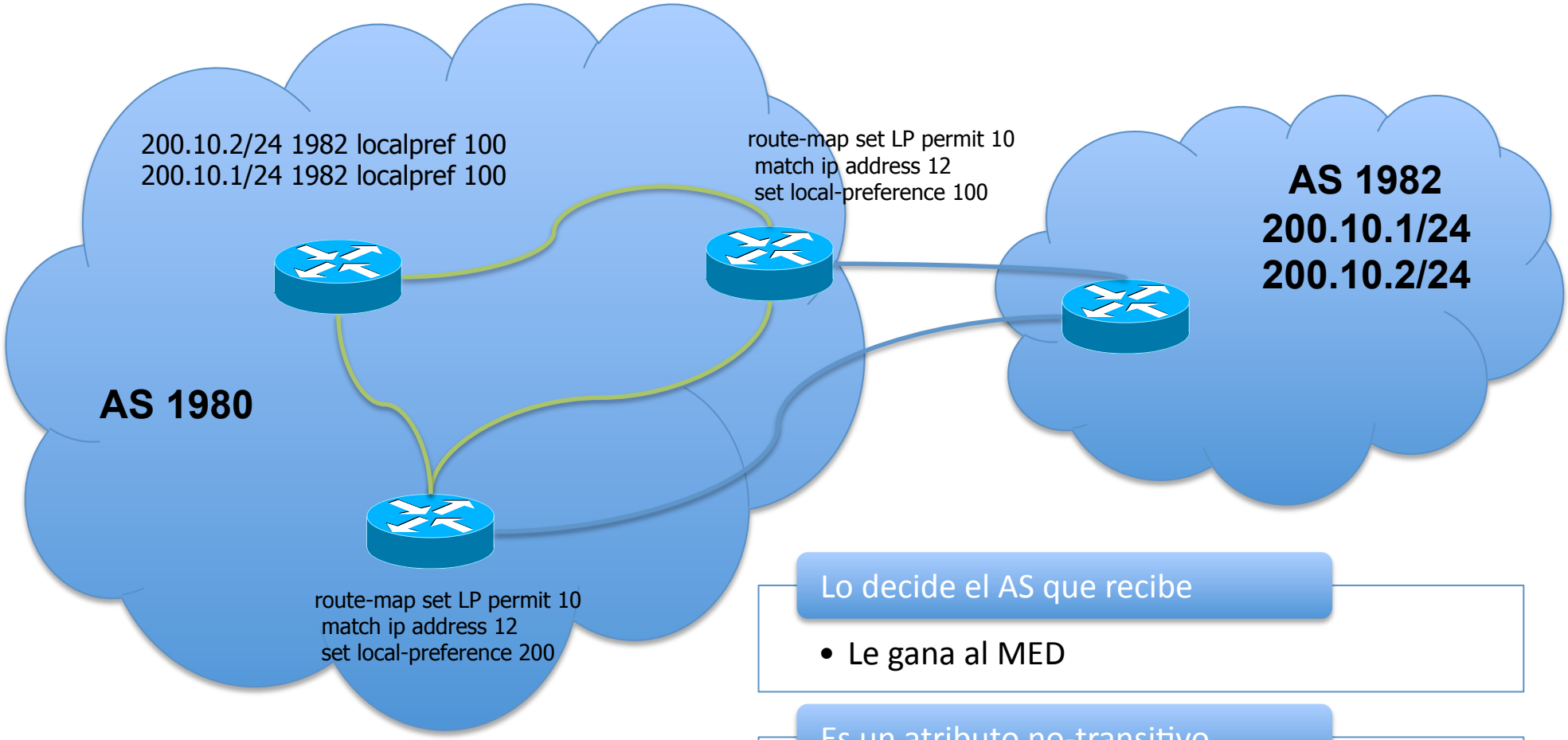
Es un atributo no-transitivo

- Los siguientes ASs no lo recibirán

Se modifica con

- route-map: *set metric*
- *set metric-type internal*

LOCAL PREFERENCE



Lo decide el AS que recibe

- Le gana al MED

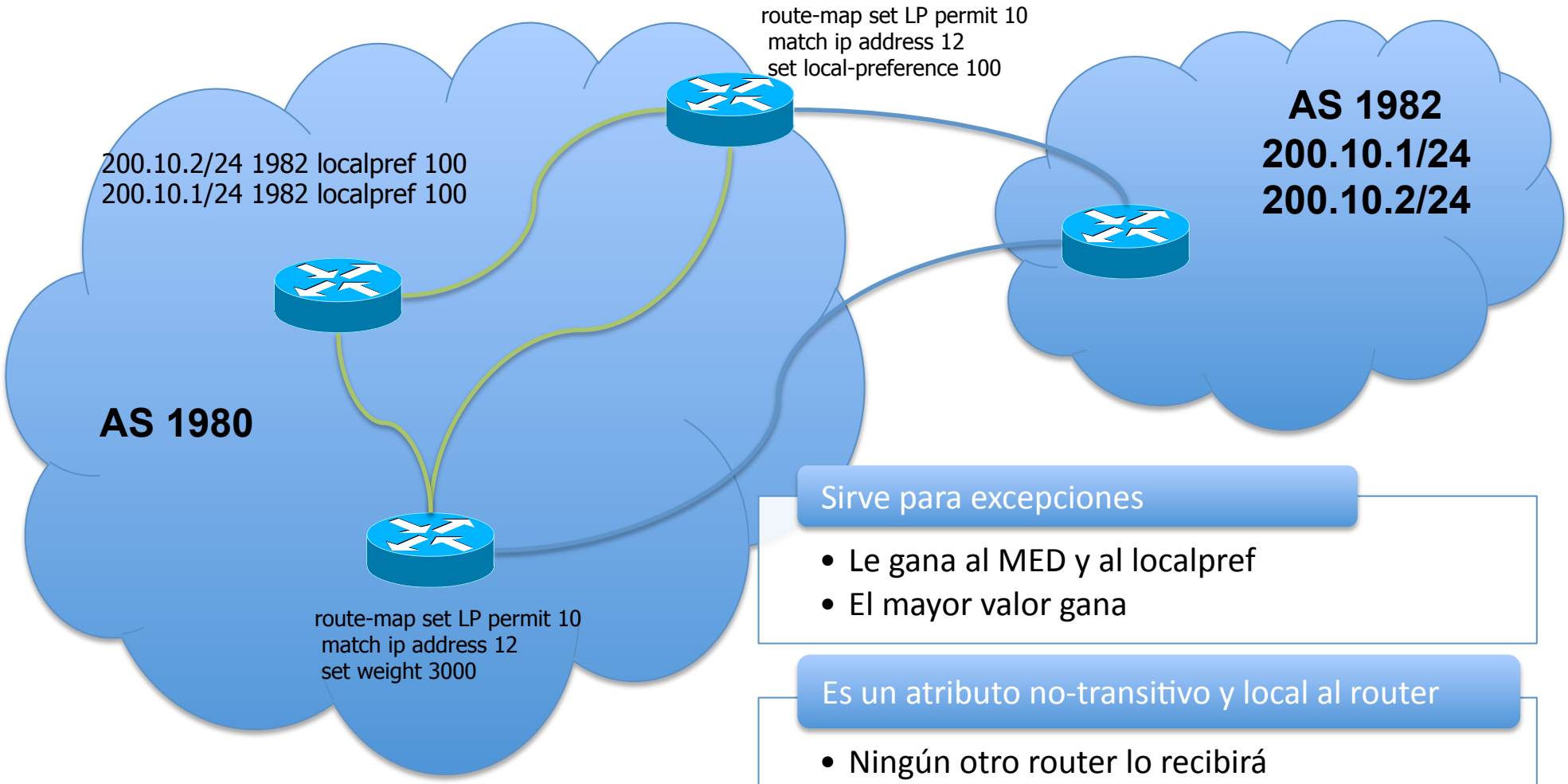
Es un atributo no-transitivo

- Los siguientes ASs no lo recibirán

Se modifica con

- route-map: *set local-preference*

WEIGHT



Sirve para excepciones

- Le gana al MED y al localpref
- El mayor valor gana

Es un atributo no-transitivo y local al router

- Ningún otro router lo recibirá

Se modifica con

- route-map: *set weight*

COMMUNITY

Marcado de prefijos

- Cada prefijo en la tabla BGP puede estar marcado
- Es transitivo (los ASs y los routers vecinos pueden pasarse esta información)
- Los AS intermedios pueden modificar, borrar o agregar comunidades

Permite agrupar los destinos según múltiples criterios

- Características de redes aprendidas (peering, transit, cliente, etc.)
- Carácterísticas propias de la red (país, región, internas, etc.)
- Modificaciones permitidas para los clientes (cambios local-pref, export, etc)

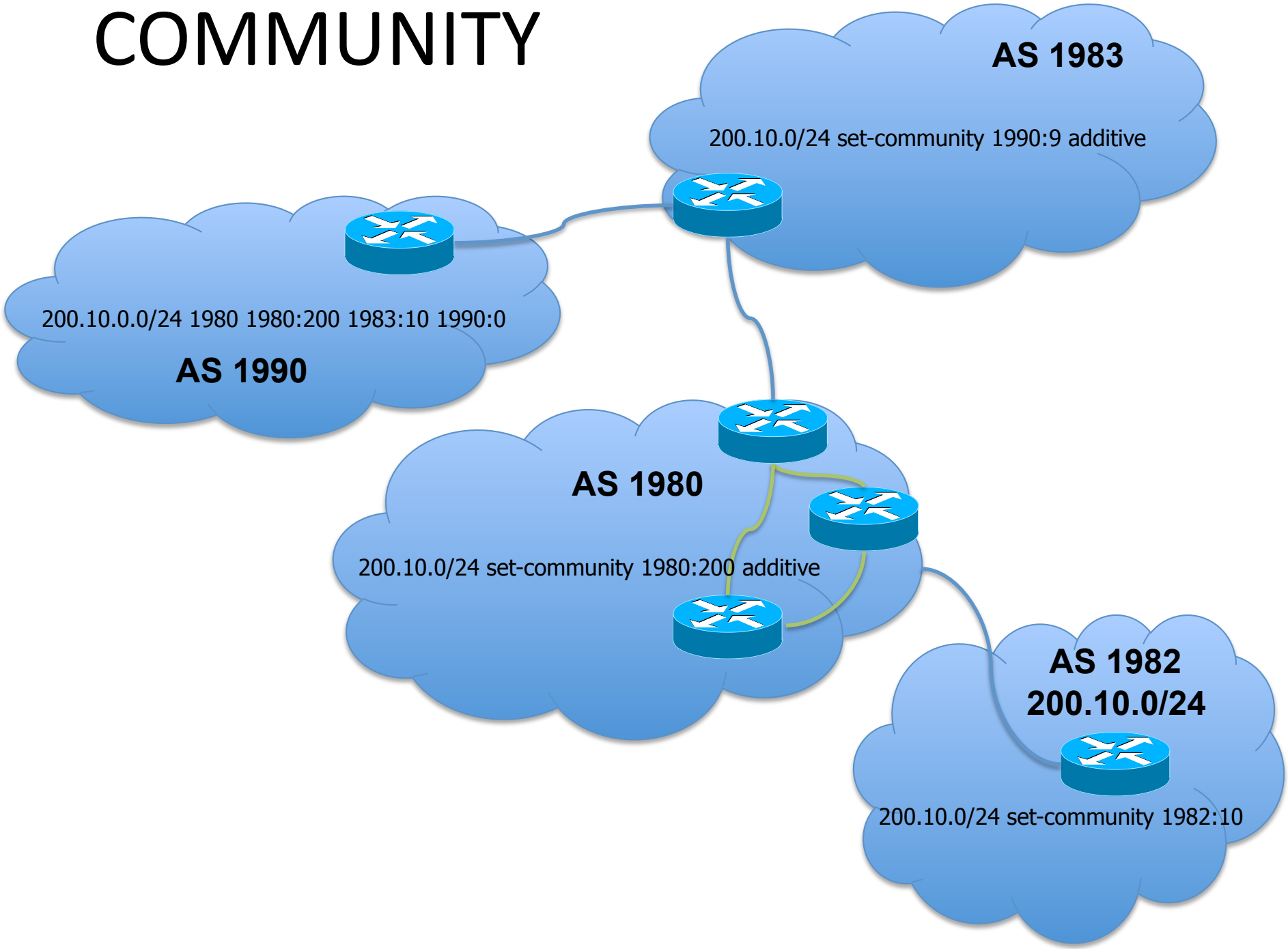
Simplifica la implementación de politicas

- Se aplican reglas comunes a los grupos marcados
- Los prefijos pueden pertenecer a multiples grupos
- Hace escalable la implementación de políticas

Son 32 bits (4 bytes) agrupados en 2 números

- Se representa como #:#
- Generalmente el primer # es el AS que pone o interpreta el valor de esa comunidad

COMMUNITY


AS 1983
200.10.0/24 set-community 1990:9 additive
200.10.0.0/24 1980 1980:200 1983:10 1990:0
AS 1990
AS 1980
200.10.0/24 set-community 1980:200 additive
AS 1982
200.10.0/24
200.10.0/24 set-community 1982:10

COMMUNITIES

Comunidades Tipicas por destino:

- Destinos aprendidos de los clientes
- Destinos aprendidos de los peers
- Destinos aprendidos de transit ISP
- Destinos aprendidos en IXPs

Comunidades útiles para los clientes

- Modificar el localpref
- No-export
- No anunciar a determinados vecinos

Well-known communities

- *local-AS* *No enviar a los peers EBGP*
- *no-advertise* *No enviar a ningún peer*
- *no-export* *No exportar fuera del AS/Confederación*

ORIGIN

- IGP (Interior)—creado con comando *network* en la configuración de BGP
- EGP (EBGP)—Aprendido por EBGP
- Incomplete—Redistribuido el IGP en BGP

Comando *set* en un route-map

- as-path Añade una cadena de AS para el atributo AS-PATH
- comm-list set BGP community list (for deletion)
- community Atributo de Comunidad
- dampening Configura parámetros para dampening
- local-preference Atributo de preferencia local de BGP
- metric Valor Metric para el protocolo de enrutamiento
- origin Codigo de origen BGP
- weight Peso BGP para la tabla de enrutamiento
- ip next-hop { A.B.C.D | peer-address }

Defaults (usados en agregación)

- NEXT_HOP = local (0.0.0.0)
- WEIGHT = 32768
- LOCAL_PREF = 100
- ORIGIN = IGP
- MED = ninguna

Atributos de BGP

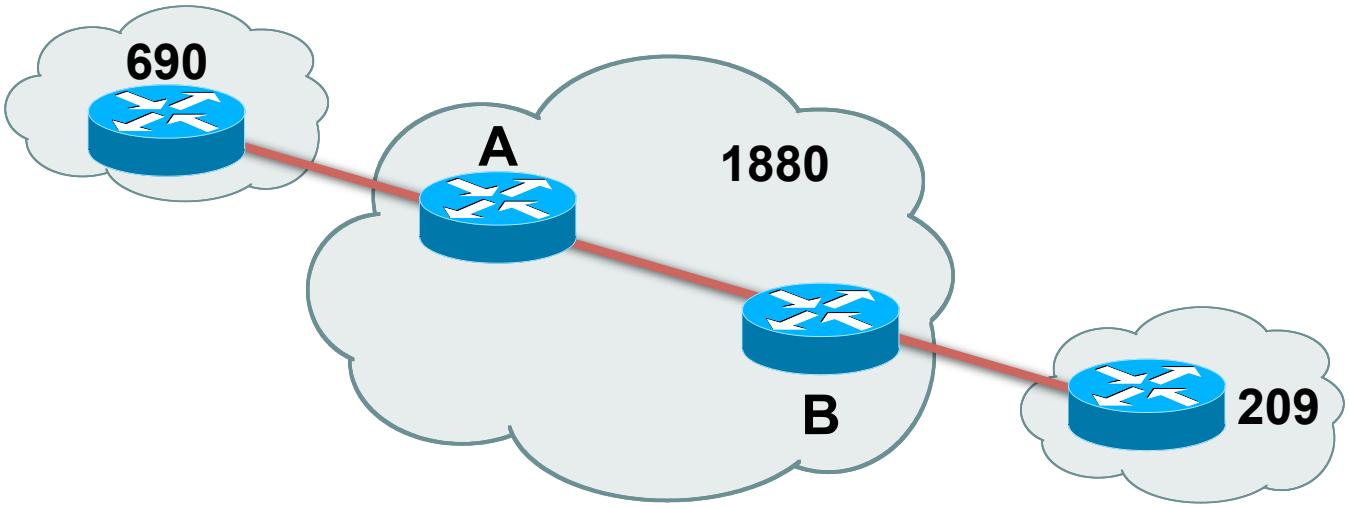
75k1#sh ip bgp 10.0.0.0
BGP routing table entry for 10.0.0.0/24, version 139267814
Paths: (1 available, best #1)
  Not advertised to any peer
  ! AS-PATH **AS** **ID**
  65000 64000 {100 200}, (aggregated by 64000 16.0.0.2)
  ! NEXT-HOP IGP METRIC PEER-IP PEER-ID
   10.0.10.4 (metric 10) from 10.0.0.1 (10.0.0.2)
    Origin IGP, metric 100, localpref 230, valid, aggregated
    internal (or external or local),
    atomic-aggregate, best
    Community: 64000:3 100:0 200:10
    Originator: 10.0.0.1, Cluster list: 16.0.0.4, 16.0.0.14

Algoritmo Básico Para Decidir

Considera solo las rutas sincronizadas sin referencia circular y un next-hop válido:

Mayor WEIGHT
Mayor LOCAL PREFERENCE
ORIGINADA LOCALMENTE (eg network/aggregate)
Más Corto AS-PATH
Menor ORIGIN (IGP < EGP < incomplete)
Menor MED
EBGP
IBGP
Menor IGP METRIC al next-hop

Sincronización



- **Router A no anunciará los prefijos de AS209 hasta que haya convergencia en el IGP.**
- **Asegurarse de que los next-hops del iBGP pueden ser vistos via IGP:**

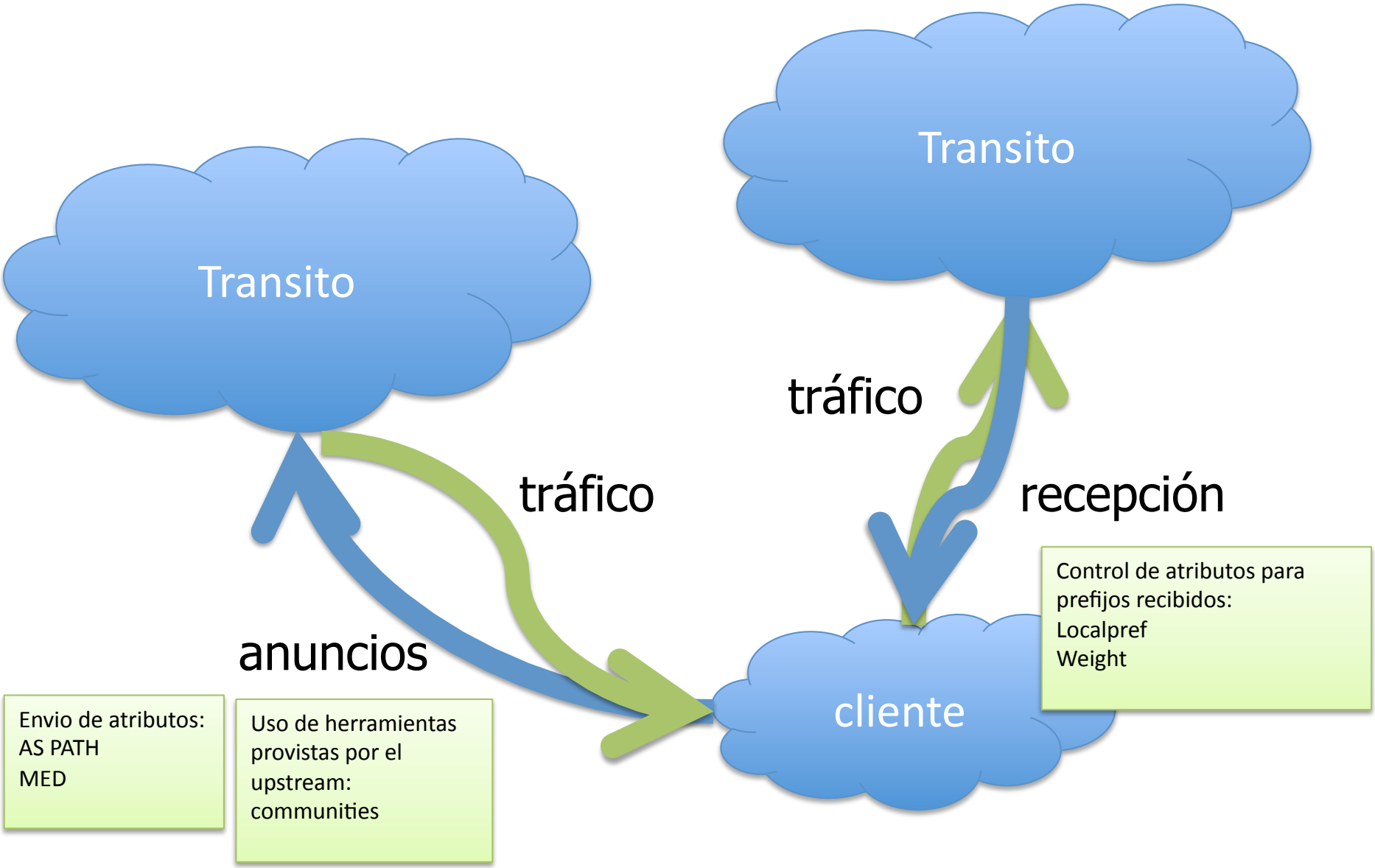
```
router bgp 1880
no synchronization
```

Consideraciones Generales

- Sincronización: no recomendada
- No dejar que BGP tenga prioridad sobre IGP
- auto-summary: no. En su lugar usar comandos de agregación

```
router bgp 100
no synchronization
no auto-summary
distance 200 200 200
```

Resumen BGP TE



Implementando iBGP

Route Reflectors, Peer Groups

Guías para un iBGP Estable

- Establecer la conexión usando direcciones de loopback
  - neighbor { ip address | peer-group}
    update-source loopback0
- Independiente de fallos de la interfase física
- Balanceamiento de la carga es realizado por el IGP

Guía Para Escalar iBGP

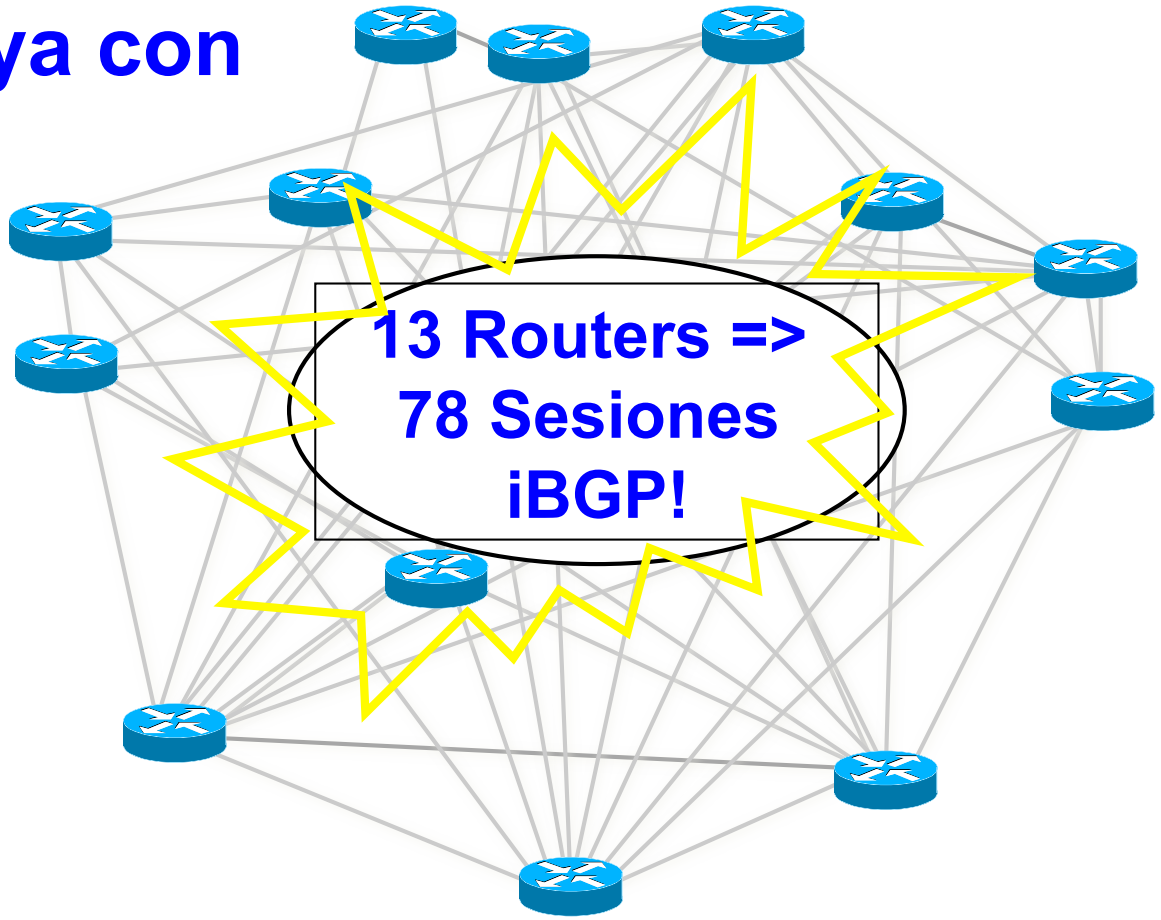
- Usar *peer-group* y *route-reflector*
- Solo llevar *next-hop* en el IGP
- Solo llevar todas las rutas en BGP si es necesario
- No redistribuir BGP en el IGP
- No redistribuir IGP en el BGP

Qué es un *peer-group*?

- Todos los miembros del *peer-group* tienen una política de salida común
- Actualizaciones generadas solo una vez para el *peer-group*
- Simplifica la configuración
- Miembros pueden tener diferentes políticas de entrada

Por qué usar *Route-Reflectors*?

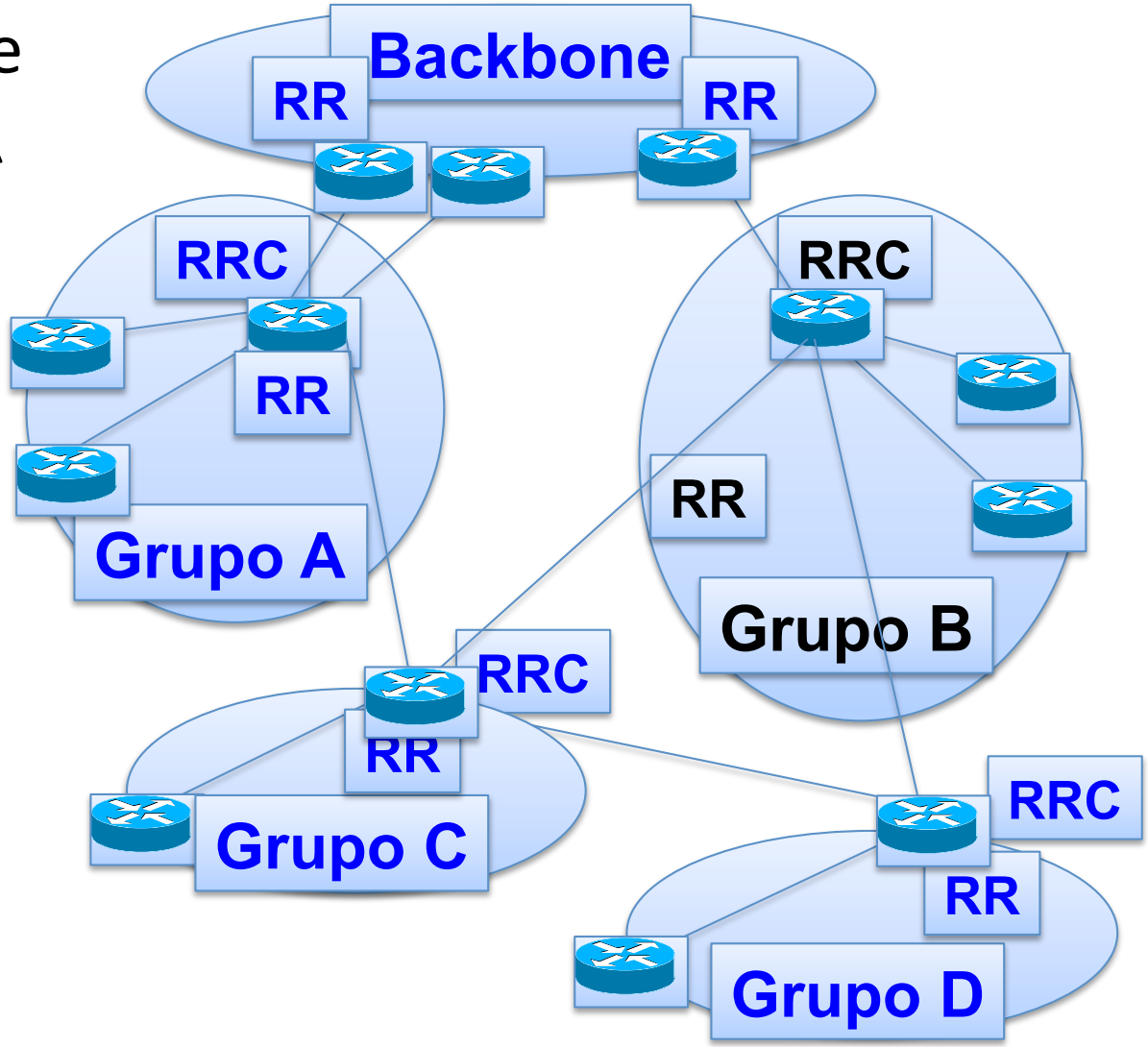
Para evitar una maya con N(n-1)/2 sesiones



n=1000 => casi medio millón de sesiones iBGP!

Usando *Route-Reflectors*

Para evitar un loop de RR la topología de RR debe reflejar la topología física



Qué es un *Route-Reflector*?

- El Reflector recibe información de clientes BGP y sesiones comunes
- Si el mejor camino es de un BGP cliente, reflejarlo a BGP clientes y normales
- Si el mejor camino es de BGP normal, reflejarlo a los clientes del RR

Desplegando *Route-Reflectors*

- Divida el backbone en varios grupos
- Cada grupo contiene al menos 1 RR (multiples para redundancia), y multiples clientes
- Los RRs crean una maya completa de iBGP

Por qué una Política de Entrada?

- Aplicar una comunidad reconocible para usar en los filtros de salida u otras políticas
- Configurar local-preference para modificar el default de 100
- Balanceo de carga en ambientes de conexión dual
  - route-map ISP in permit 10
  - set local-preference 200
  - set community 100:2

Por qué una política de Salida?

- El filtrado de prefijos de salida ayuda a protegernos contra errores (también podemos aplicar filtros de comunidades y as-path)
- Enviar comunidades basado en acuerdos con el ISP
    - route-map ISPout permit 10
    - match ip address prefix-list outgoing
    - set community 100:1 additive

Consideraciones de Proveedores

- Agrupar clientes para aplicar políticas
- Ofrecer una selección del número de rutas a anunciar
- Peerings con otros proveedores
- Protegerse contra los problemas de configuración de los clientes
- Ofrecer herramientas de balanceo (comunidades)

Guías para el Agrupado de Clientes

- Definir por los menos tres *“peer-groups”*:
  - cliente-default — envía solo default
  - cliente-cliente — envía solo las rutas de clientes
  - Cliente-completo — envía todas las rutas
- Identificar las rutas a traves de comunidades
  - 2:100=clientes; 2:80=peers
- Aplicar un “prefix-list” para cada vecino BGP

Ejemplos de configuración: solo ruta-default a clientes

```
neighbor cliente-default peer-group
neighbor cliente-default description Envía Default
neighbor cliente-default default-originate
                                route-map ruta-default
neighbor cliente-default route-map cliente-entrada in
neighbor cliente-default prefix-list niega-todo out

ip prefix-list niega-todo seq 5 deny 0.0.0.0/0 le 32
```

Peer Groups para Peerings y Puntos de Intercambio

- Similar al EBGP para agregado de clientes excepto que no se usa el filtrado de prefijos (porque no hay un registro)
- En su lugar se usa *maximum-prefix* y chequeos de sanidad de prefijos
- Se puede usar filtros por AS (requiere actualizaciones)

Ejemplos de configuración: Peerings e IXPs

```
neighbor nap peer-group
neighbor nap prefix-list chequeo sanidad in
neighbor nap route-map nap-salidas out
neighbor nap maximum prefix 30000

route-map nap-salida permit 10
match community 1       ; solo clientes
set metric-type internal  ; MED = métrica IGP
set ip next-hop peer-address ; la nuestra
```

Ejemplo: Prefix-List chequeo-sanidad

```
# no aceptamos default
ip prefix-list chequeo-sanidad seq 10 deny 0.0.0.0/8 le 32
# no aceptamos 10/8 per RFC1918
ip prefix-list chequeo-sanidad seq 20 deny 10.0.0.0/8 le 32
# reservado por IANA – dirección de loopback
ip prefix-list chequeo-sanidad seq 25 deny 127.0.0.0/8 le 32
# no acepta la red 128.0 – reservado por IANA
ip prefix-list chequeo-sanidad seq 35 deny 128.0.0.0/16 le 32
```

Ejemplo: Prefix-List chequeo-sanidad

```
# no acepta 172.16 por RFC1918
ip prefix-list chequeo-sanidad seq 40 deny 172.16.0.0/12 le 32
# no acepta clase C 192.0.0.0 reservado por IANA
ip prefix-list chequeo-sanidad seq 50 deny 192.0.0.0/24 le 32
# no acepta 192.168/16 por RFC1918
ip prefix-list chequeo-sanidad seq 55 deny 192.168.0.0/16 le 32
# no acepta nada en red 223 – reservado por IANA
ip prefix-list chequeo-sanidad seq 70 deny 223.255.255.0/24 le 32
# no acepta clase D/Experimental
ip prefix-list chequeo-sanidad seq 75 deny 224.0.0.0/3 le 32
```

Resumen

- Escalabilidad:
  - Uso de atributos, especialmente comunidad (community)
  - Uso de peer-groups y route-reflectors
- Estabilidad:
  - Uso de dirección de loopback para el iBGP
  - Generación de agregados
  - Aplicación de claves
  - Siempre filtrar anuncios de entrada y salida

Resumen

- Simplicidad—soluciones estandares:
  - Tres opciones de multihoming
  - Agrupar clientes en comunidades
  - Aplicación de políticas estandares en el borde
  - Evitar “configuraciones especiales”
  - Automatice la generación de la configuracion (RR & RtConfig)