Chapter 7: BGP Implementation

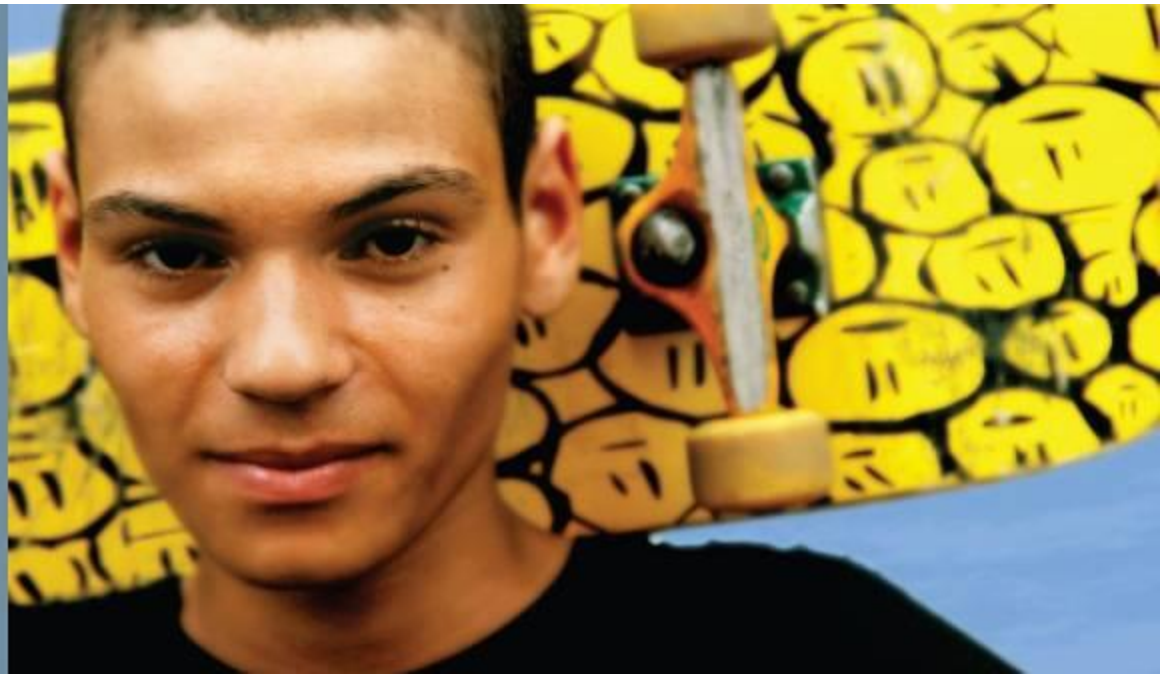
CCNP ROUTE: Implementing IP Routing

Cisco | Networking Academy®
Mind Wide Open™

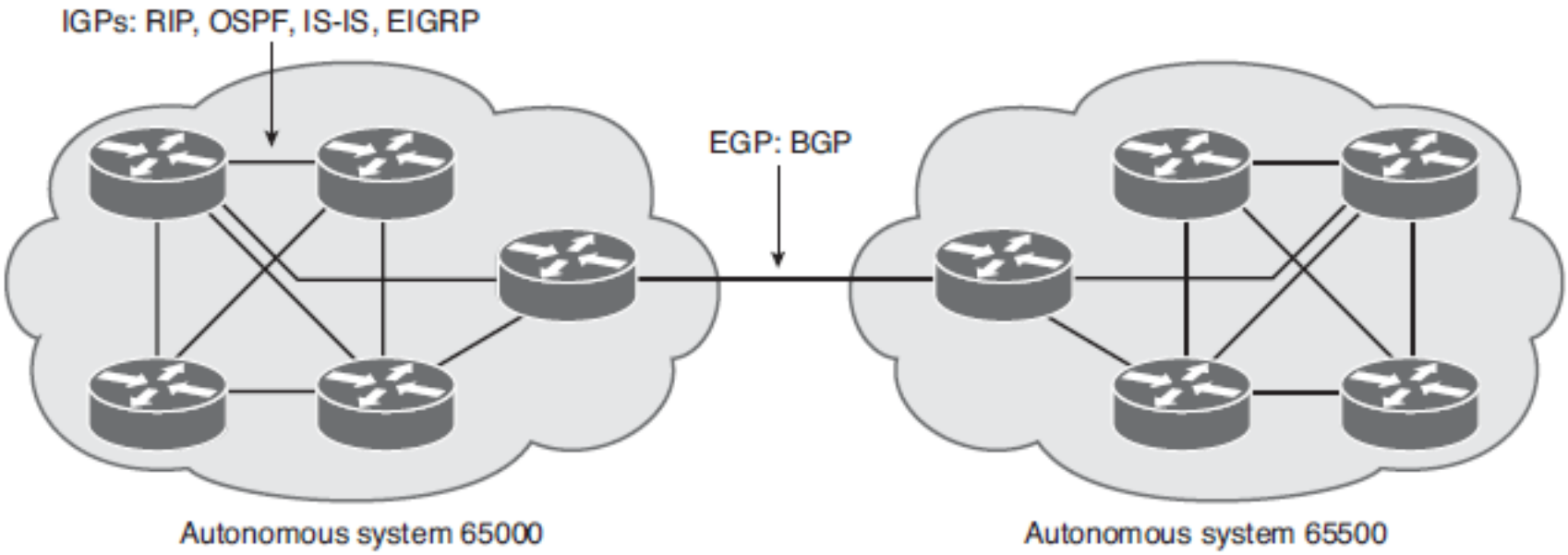
Chapter 7 Objectives

- BGP Terminology, Concepts, and Operation
- Implementing Basic BGP
- BGP Attributes and the Path-Selection Process
- Controlling BGP Routing Updates
- Implementing BGP for IPv6 Internet Connectivity

BGP Terminology, Concepts, and Operation



Autonomous System Numbers



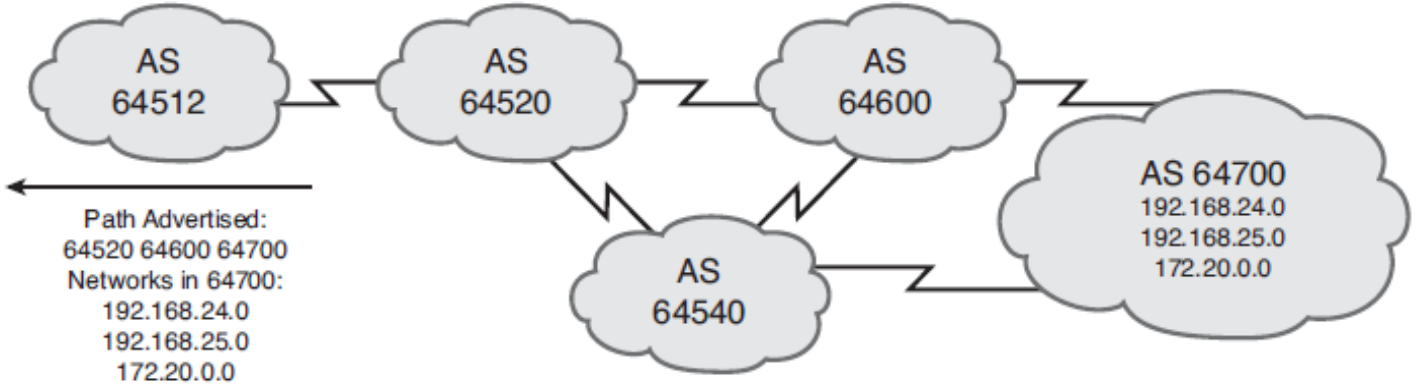
- The ASN is a very important parameter required when configuring BGP.

A complete table of 16-bits and 32-bits ASN available

Number	Bits	Description	Reference
0	16	Reserved for RPKI unallocated space invalidation[19]	RFC 6483, RFC 7607
1 - 23455	16	Public ASNs	
23456	16	Reserved for AS Pool Transition	RFC 6793
23457 - 64495	16	Public ASNs	
64496 - 64511	16	Reserved for use in documentation and sample code	RFC 5398
64512 - 65534	16	Reserved for private use	RFC 1930, RFC 6996
65535	16	Reserved	RFC 7300
65536 - 65551	32	Reserved for use in documentation and sample code	RFC 5398, RFC 6793
65552 - 131071	32	Reserved	
131072 - 4199999999	32	Public 32-bit ASNs	
4200000000 - 4294967294	32	Reserved for private use	RFC 6996
4294967295	32	Reserved	RFC 7300

BGP Path Vector Characteristics

- The path vector information includes a list of the full path of BGP autonomous system numbers (hop by hop) necessary to reach a destination network; this is the *AS-path* attribute.
- The autonomous system path information is used to construct a loop-free graph, since router running BGP does not accept a routing update that already includes its autonomous system number in the path list.

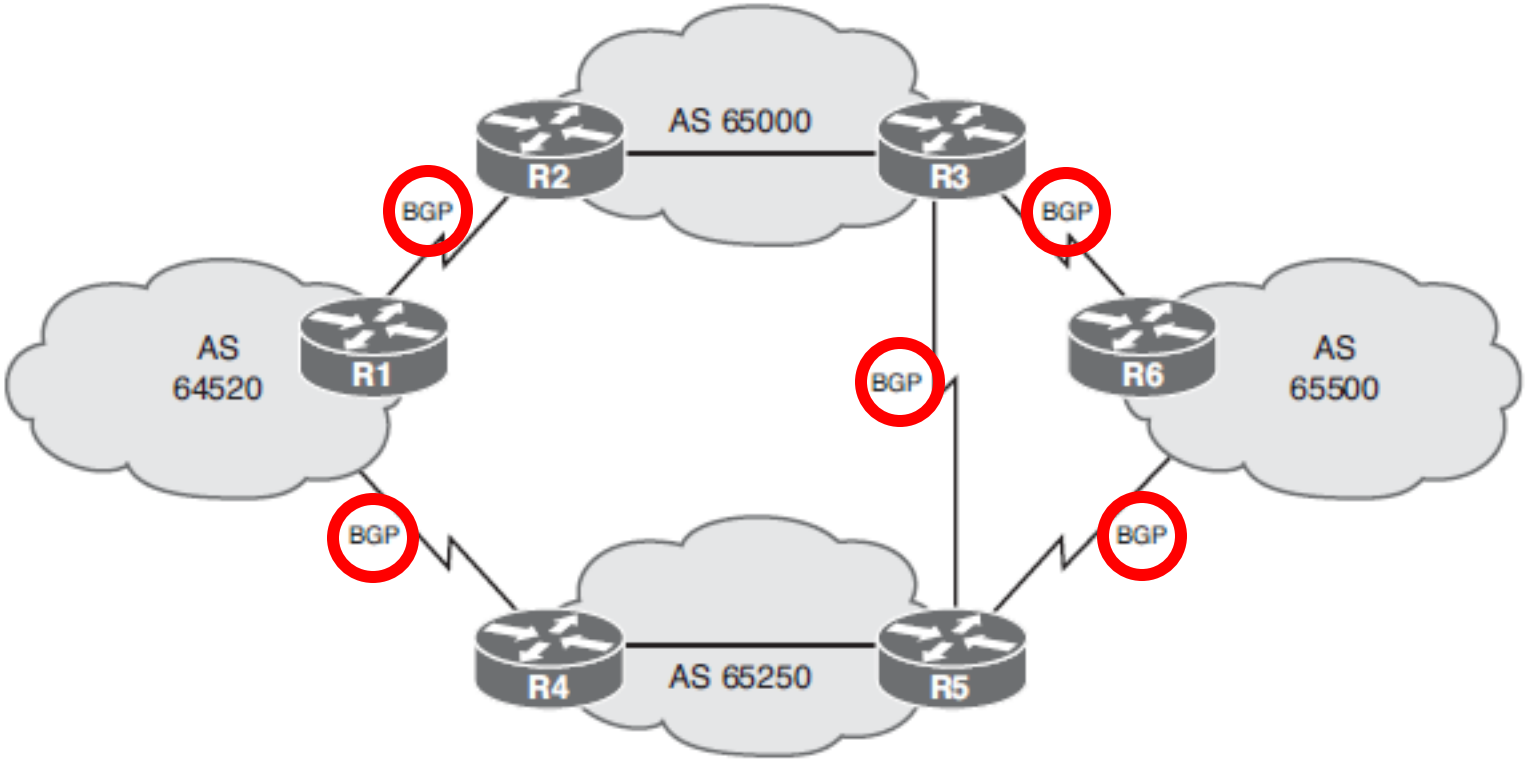


BGP Terminology, Concepts, and Operation

- Using BGP between autonomous systems
- Comparing BGP with other scalable routing protocols
- BGP path vector characteristics
- BGP characteristics
- BGP tables
- BGP message types
- When to use BGP
- When not to use BGP

BGP Use Between Autonomous Systems

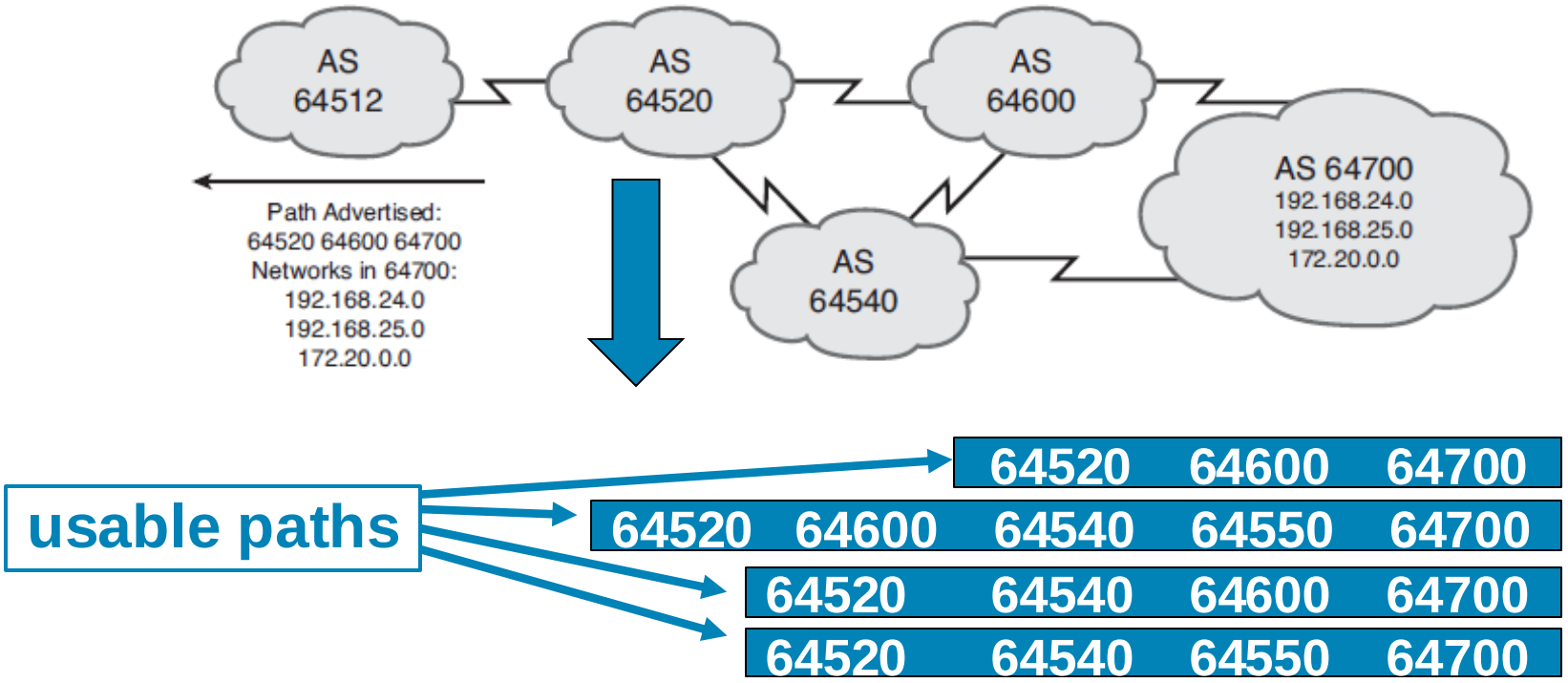
- The main goal of BGP is to provide an interdomain routing system that guarantees the loop-free exchange of routing information between autonomous systems. BGP routers exchange information about paths to destination networks.



BGP Path Vector Characteristics

- BGP allows routing-policy decisions to be applied to the path of BGP autonomous system numbers so that routing behavior can be enforced at the autonomous system level and to determine how data will flow through the autonomous system.
- The policies are based on the **attributes** carried in the routing information and configured on the routers.
- BGP specifies that a BGP router can advertise to its peers (neighbors) in neighboring autonomous systems only those routes that it uses.

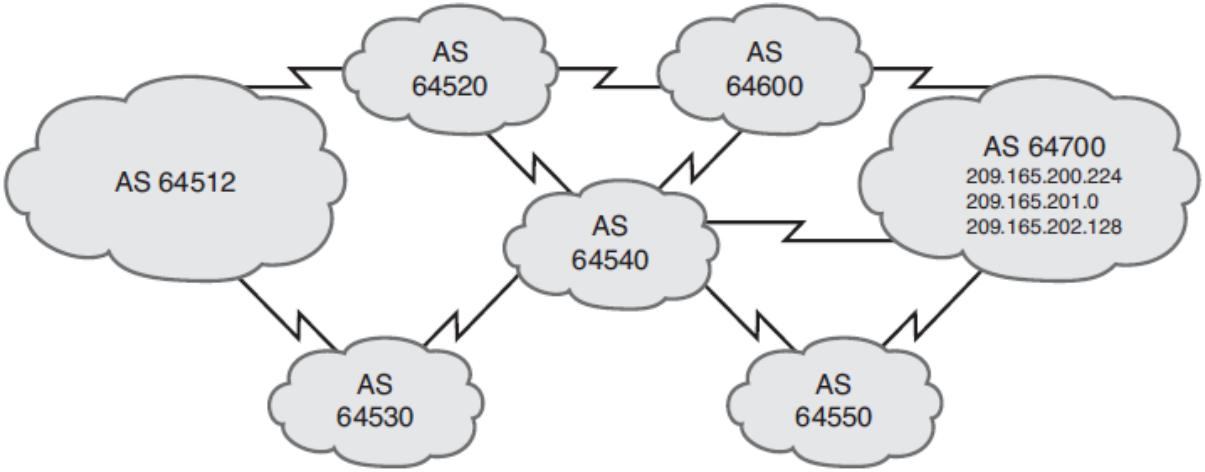
BGP Path Vector Characteristics



- Autonomous system **64512** does not see all these possibilities.
- Autonomous system 64520 advertises to autonomous system 64512 only its best path (in this case, 64520 64600 64700), the same way that IGPs announce only their best least-metric routes. This path is the only path through autonomous system 64520 that autonomous system 64512 sees.

BGP Supports the Internet's Hop-by-Hop Routing Paradigm

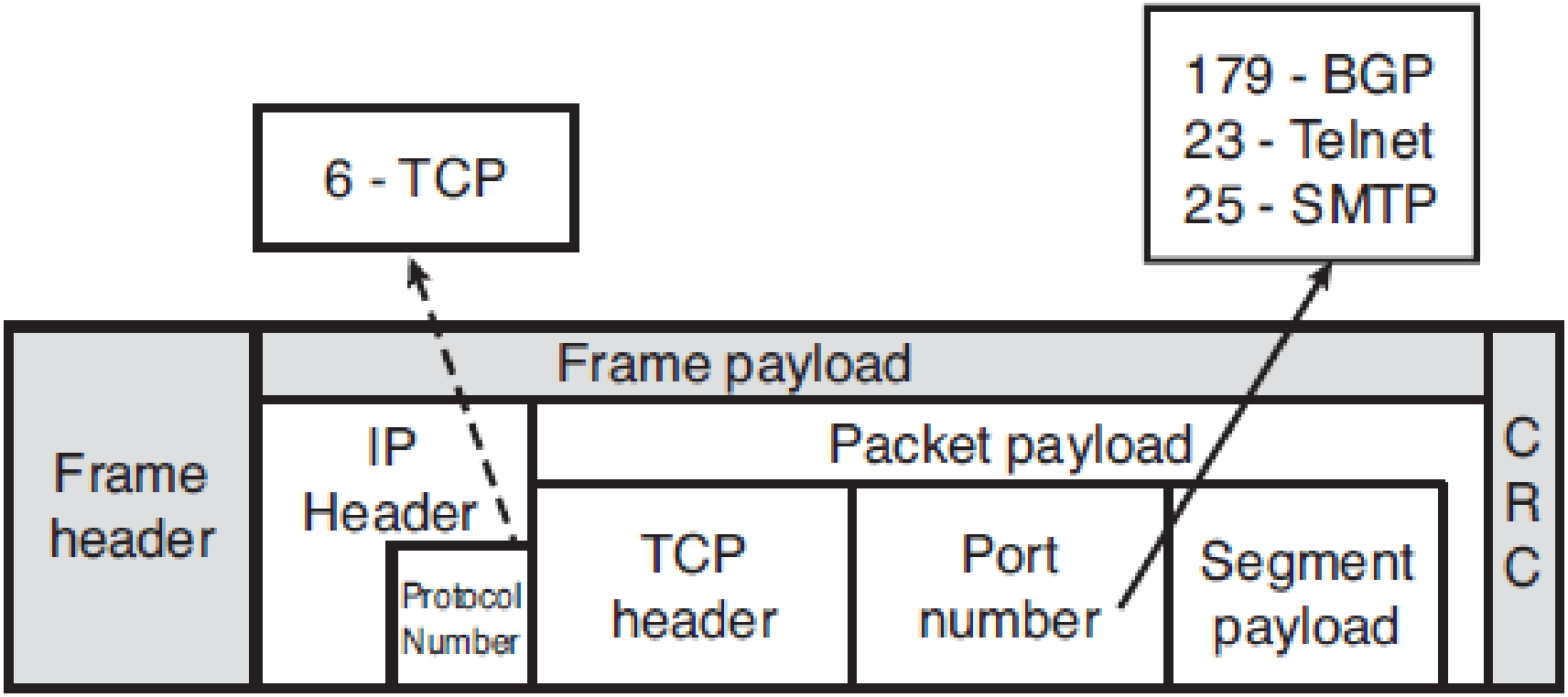
- To reach the networks in autonomous system 64700, autonomous system 64512 can choose to use the path through autonomous system 64520 or it can choose to go through the path that autonomous system 64530 is advertising.
- Autonomous system 64512 selects the best path to take based on its own BGP routing policies.



BGP Characteristics

- BGP is sometimes categorized as an advanced distance vector protocol, but it is actually a **path vector protocol**.
- BGP uses the Transmission Control Protocol (TCP) as its transport protocol, which provides connection-oriented reliable delivery.
- In this way, BGP assumes that its communication is reliable and, therefore, BGP does not have to implement any retransmission or error-recovery mechanisms, like EIGRP does.
- BGP information is carried inside TCP segments using protocol port number 179.

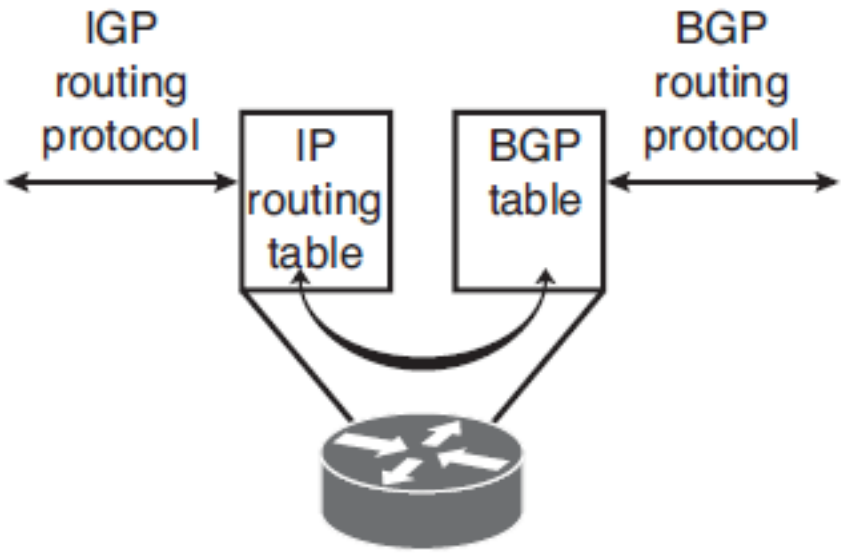
BGP Frame / Packet / Segment



BGP Characteristics

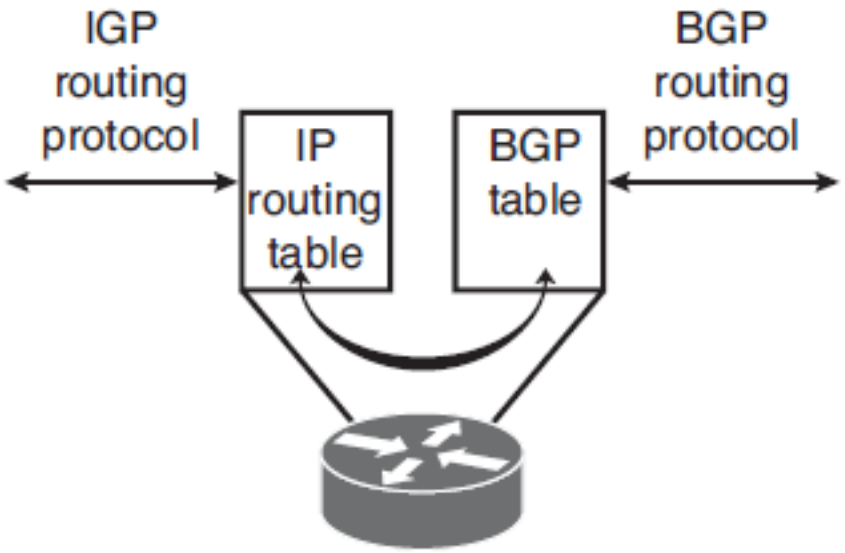
- Two routers speaking BGP (called *BGP speakers*) establish a TCP connection with one another and exchange messages to open and confirm the connection parameters.
- These two routers are called *BGP peer* routers or *BGP neighbors*
- After the TCP connection is made, the routers exchange their full BGP tables
- However, because the connection is reliable, BGP routers need to send only changes (incremental updates) after that.
- Periodic routing updates are not required on a reliable link, so triggered updates are used.
- BGP sends **keepalive messages**, similar to the hello messages sent by OSPF and EIGRP.
- differences EIGRP, OSPF vs BGP – one-for-one window

BGP Tables



- BGP keeps a **neighbor table** – containing a list of neighbors with which it has a BGP connection.
- BGP also keeps its own **BGP table** – for storing BGP information received from and sent to other routers.

BGP Table



- It is important to remember that this **BGP table** is separate from the IP routing table in the router.
- The router offers the best routes from the BGP table to the IP routing table and can be configured to share information between the two tables (by redistribution).

BGP Routing Process

- For BGP to establish an adjacency, you must configure it explicitly for each neighbor.
- BGP forms a TCP relationship with each of the configured neighbors and keeps track of the state of these relationships by periodically sending a **BGP/TCP keepalive message**.
- After establishing an adjacency, the neighbors **exchange** their **best BGP routes**.
- Each router collects these routes from each neighbor with which it successfully established an adjacency and places them in its BGP table; **all routes that have been learned from each neighbor are placed in the BGP table**.
- Each learned path is associated with BGP attributes. The single best route for each network is selected from the BGP table using these attributes in the BGP route-selection process and then offered to the IP routing table.

BGP Routing Process

- Each router compares the offered BGP routes to any other possible paths to those networks in its IP routing table, and the best route, based on administrative distance, is installed in the IP routing table.
- External BGP (eBGP) routes (BGP routes learned from an external autonomous system) have a default administrative distance of 20.
- Internal BGP (iBGP) routes (BGP routes learned from within the autonomous system) have a default administrative distance of 200.
- A router may have a best BGP route to a destination, but that route might not be installed in the IP routing table because it has a higher administrative distance than another route.
- That best BGP route will still be propagated to other BGP routers, though.

BGP Message Types

BGP defines the following message types:

- **Open** (start BGP, Hold time 180sec)
- **Keepalive** (like Hello, every 60sec)
- **Update** (just 1 path, more networks)
- **Notification** (end BGP, when error happens)

Note Keepalive messages have a length of 19 bytes. Other messages may be between 19 and 4096 bytes long.

Open and Keepalive Messages

- After a TCP connection is established, the first message sent by each side is an open message.
- If the open message is acceptable, a keepalive message confirming, the open message is sent back by the side that received the open message.
- When the open is confirmed, the BGP connection is established, the update, keepalive and notification messages can be exchanged.
- BGP peers initially exchange their full BGP routing tables.
- From then on, incremental updates are sent as the table changes.
- Keepalive packets are sent to ensure that the connection is alive between the BGP peers, and notification packets are sent in response to errors or special conditions.

Open Message

An open message includes the following information:

- **Version**
  - 8-bit field. BGP implementations today use the current version, BGP-4.
- **My autonomous system**
  - 16-bit field indicates the sender's autonomous system number. The peer router verifies this information; if it is not the autonomous system number expected, then the BGP session is torn down.
- **Hold time**
  - 16-bit field indicates the maximum number of seconds that can elapse between the successive keepalive or update messages from the sender
- **BGP router identifier (router ID)**
  - This 32-bit field indicates the sender's BGP identifier. The BGP router ID is an IP address assigned to that router and is determined at startup.
- **Optional parameters**
  - A length field indicates the total length of the optional parameters field in octets. These parameters are Type, Length, and Value(TLV) encoded.
  - An example of an optional parameter is session authentication.

Update Messages

An update message has information about **one path only**; multiple paths require multiple **update** messages. All the attributes in the update message refer to that path, and the networks are those that can be reached through that path.

An update message might include the following fields:

- **Network layer reachability information (NLRI)**
  - A list of networks (IP address prefixes and their prefix lengths) that can be reached by this path.
- **Path attributes**
  - The AS-path, origin, local preference, and so forth. Each path attribute includes the attribute type, attribute length, and attribute value (TLV).
- **Withdrawn routes**
  - A list of IP address prefixes for routes that are being withdrawn from service, if any.

Notification Messages

- A BGP router sends a notification message when it detects an error condition.
- The BGP router closes the BGP connection immediately after sending the notification message.
- Notification messages include:
  - an error code,
  - an error subcode, and
  - data related to the error.

BGP Neighbor States

- BGP is a state machine that takes a router through the following states with its neighbors:
  - ***Idle***
  - ***Connect***
  - ***Active***
  - ***Open sent***
  - ***Open confirm***
  - ***Established***
- Only when the connection is in the *Established* state are update, keepalive, and notification messages exchanged.

When to Use BGP

BGP use in an autonomous system is most appropriate when the effects of BGP are well understood and at least one of the following conditions exists:

- The autonomous system has multiple connections to other autonomous systems.
- The autonomous system allows packets to transit through it to reach other autonomous systems (for example, it is a service provider).
- Routing policy and route selection for traffic entering and leaving the autonomous system must be manipulated.

When Not to Use BGP

Do not use BGP if one or more of the following conditions exist:

- A single connection to the Internet or another autonomous system.
- Lack of memory or processor power on edge routers to handle constant BGP updates.
- You have a limited understanding of route filtering and the BGP path-selection process.
- If the routing policy that will be implemented in an autonomous system is consistent with the policy implemented in the ISP autonomous system.

Implementing Basic BGP

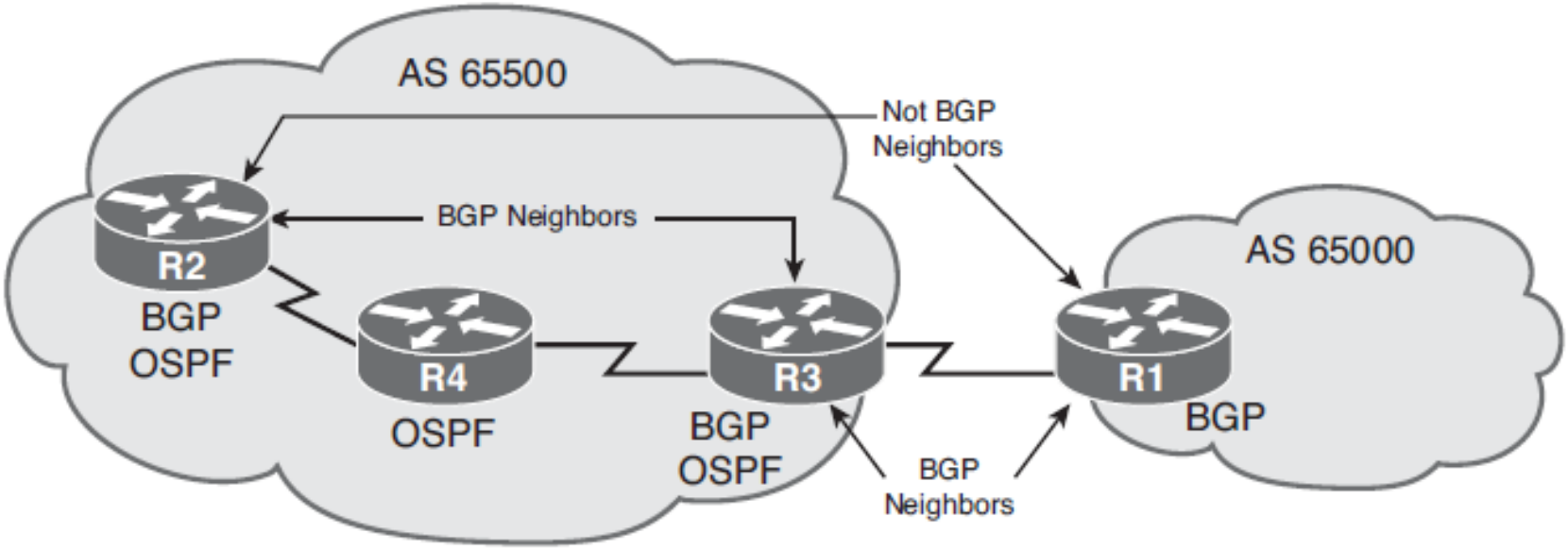
Implementing Basic BGP

- BGP neighbor relationships
- Basic BGP configuration requirements
- Entering BGP configuration mode
- Defining BGP neighbors and activating BGP sessions
- Basic BGP configuration and verification

BGP Neighbor Relationships

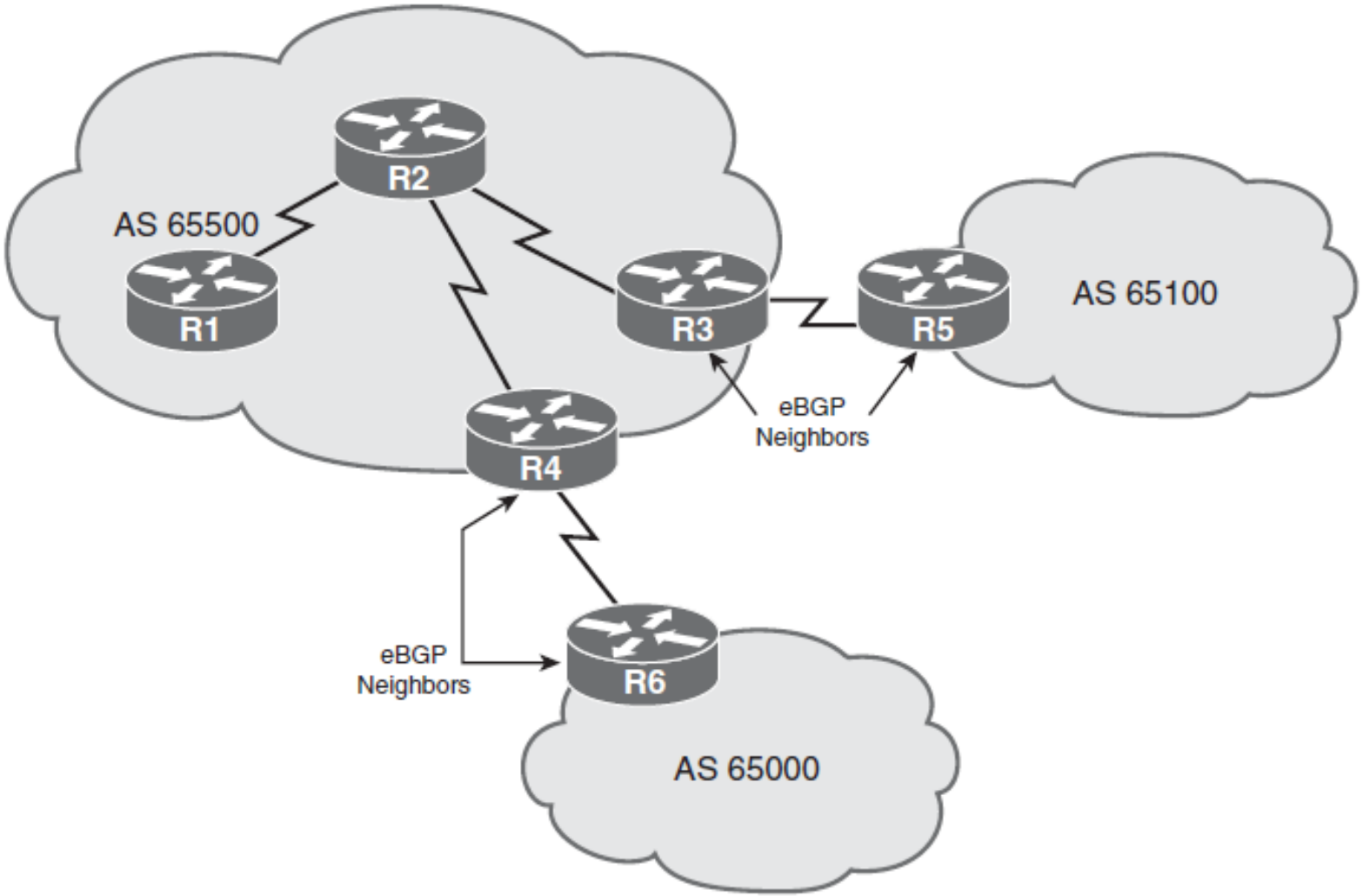
- No single router can handle communications with the tens of thousands of the routers that run BGP and are connected to the Internet, representing more than 48,000 autonomous systems.
- A BGP router forms a direct neighbor relationship with a limited number of other BGP routers.
- Through these BGP neighbors, a BGP router learns of the paths through the Internet to reach any advertised network.
- Recall that any router that runs BGP is called a BGP speaker.
- A BGP peer, also known as a BGP neighbor, is a BGP speaker that is configured to form a neighbor relationship with another BGP speaker for the purpose of directly exchanging BGP routing information with one another.

BGP Neighbor Relationships



- BGP peers can be either internal (iBGP) or external (eBGP) to the autonomous system.

External BGP (eBGP) Neighbors

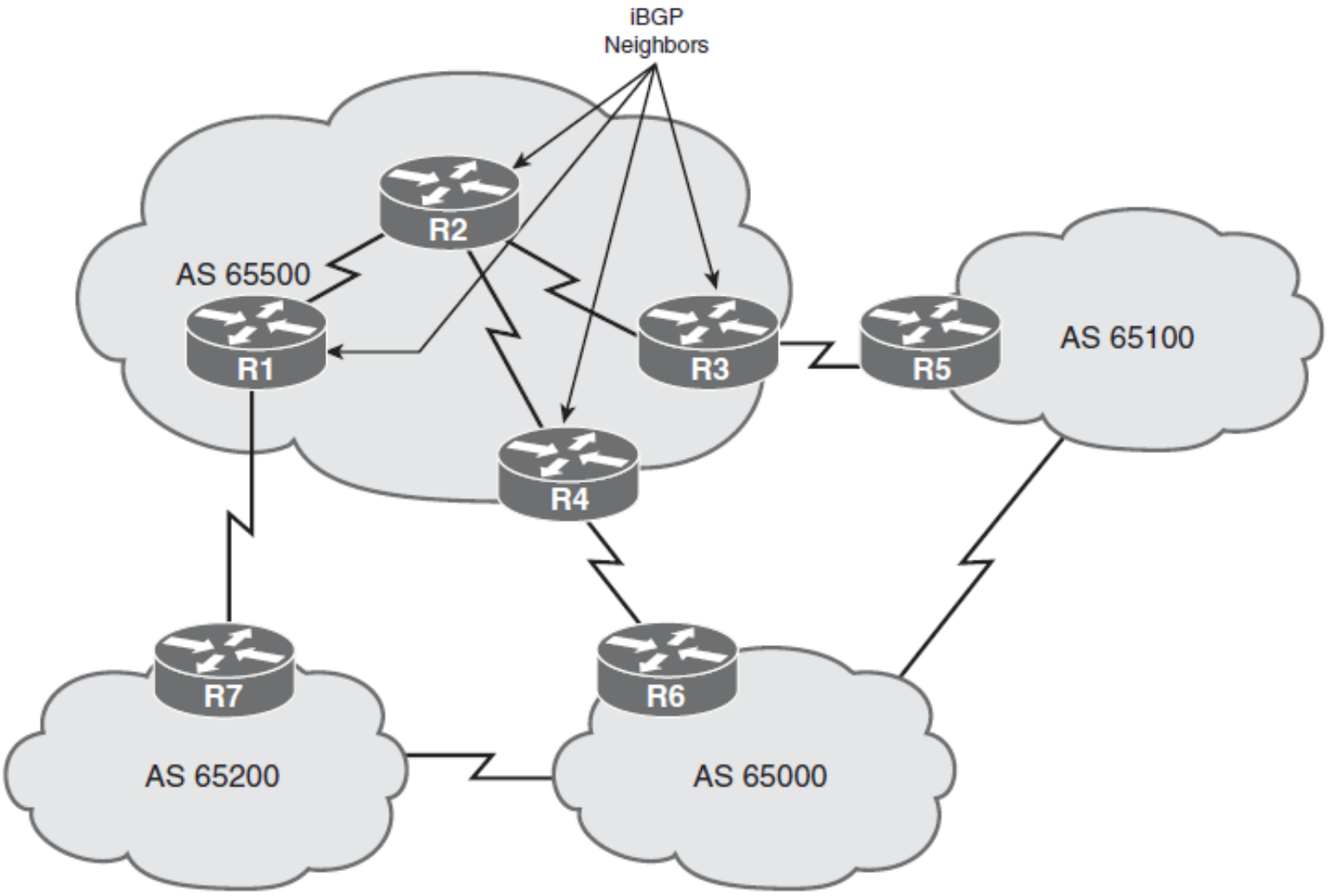


External BGP (eBGP) Neighbors

There are several requirements for an eBGP neighbor relationship (also called an eBGP neighborship):

- **Different autonomous system number**
  - eBGP neighbors must reside in different autonomous systems to be able to form an eBGP relationship.
- **Define neighbors**
  - A TCP session must be established between neighbors before starting BGP routing update exchanges.
- **Reachability**
  - The IP addresses used in the **neighbor** command must be reachable; eBGP neighbors are **usually directly connected**.

Internal BGP (iBGP) Neighbors

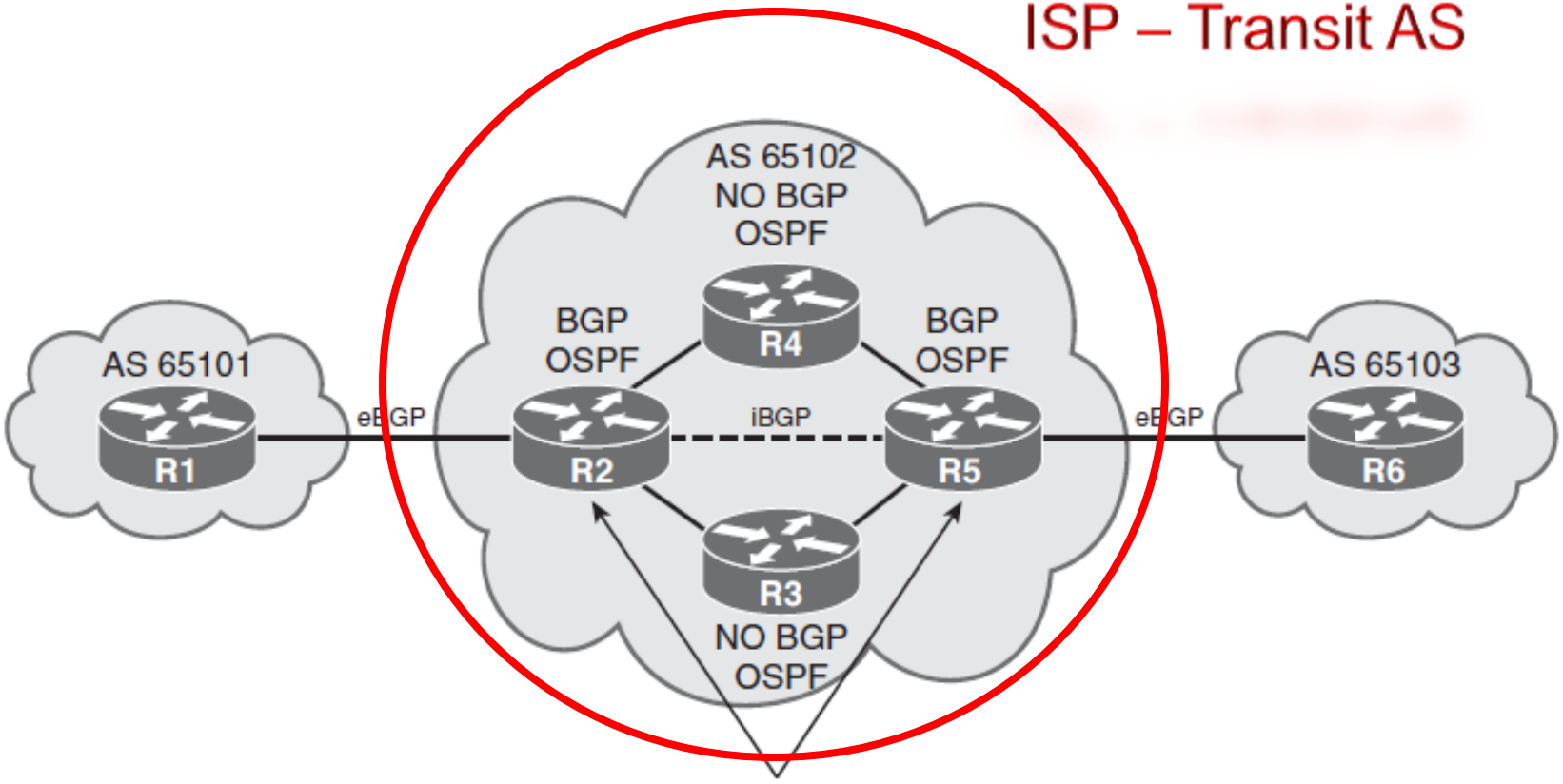


Internal BGP (iBGP) Neighbors

There are several requirements for an iBGP neighbor relationship (also known as an iBGP neighborship):

- **Same autonomous system number**
  - iBGP neighbors must reside in the same autonomous system to be able to form an iBGP relationship.
- **Define neighbors**
  - A TCP session must be established between neighbors before they start exchanging BGP routing updates.
- **Reachability**
  - iBGP neighbors must be reachable. An IGP typically runs inside the autonomous system, and provides this reachability.

iBGP in a Transit Autonomous System



Redistributing BGP into OSPF is not recommended;
instead run iBGP on all routers within the AS.

iBGP in a Transit Autonomous System

- A transit autonomous system, such as autonomous system 65102 , is an autonomous system that routes traffic from one external autonomous system to another external autonomous system.
- Transit autonomous systems are typically ISPs. All routers in a transit autonomous system must have complete knowledge of external routes.
- One way to achieve this goal is to redistribute BGP routes into an IGP at the edge routers; however, this approach has problems.
- Because the current Internet routing table is very large, redistributing all the BGP routes into an IGP is not a scalable way for the interior routers within an autonomous system to learn about the external networks.
- Another method that you can use is to run iBGP on all routers within the autonomous system.

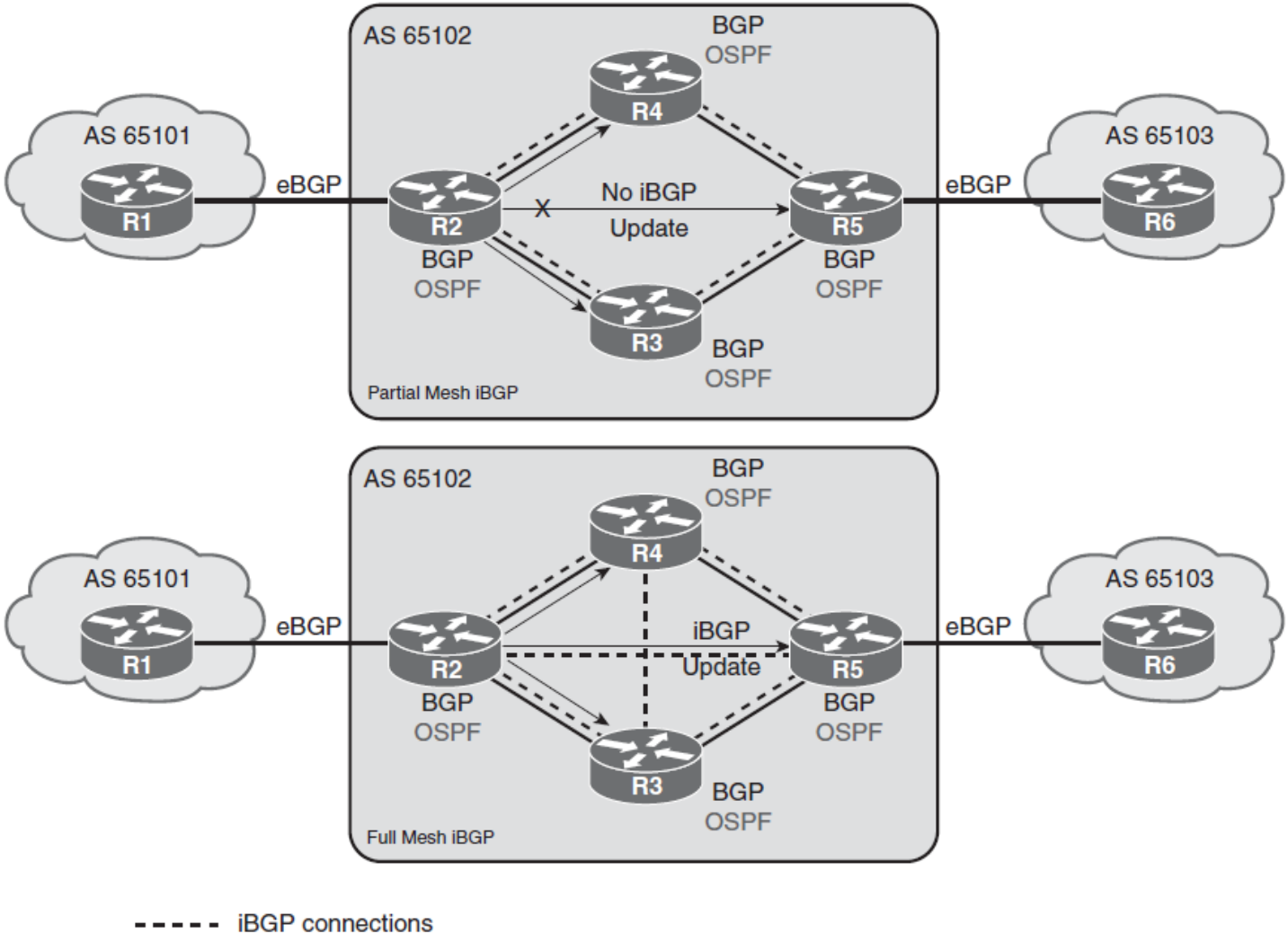
iBGP in a Non-transit Autonomous System

- A non-transit autonomous system, such as an organization that is multihoming with two ISPs, does not pass routes between the ISPs.
- To make proper routing decisions, however, the BGP routers within the autonomous system still require knowledge of all BGP routes passed to the autonomous system.
- BGP does not work in the same manner as IGPs.
- Because the designers of BGP could not guarantee that an autonomous system would run BGP on all routers, a method had to be developed to ensure that iBGP speakers could pass updates to one another while ensuring that no routing loops would exist.

TCP and Full Mesh

- TCP sessions cannot be multicast or broadcast because TCP has to ensure the delivery of packets to each recipient. Because TCP cannot use broadcasting or multicasting, BGP cannot use it either.
- To avoid routing loops within an autonomous system, BGP specifies that routes learned through iBGP are never propagated to other iBGP peers; this is sometimes referred to as the BGP split-horizon rule.
- Thus, each eBGP/iBGP router needs to send routes to all the other iBGP neighbors in the same autonomous system.
- Because they cannot use broadcast or multicast, an iBGP neighbor relationship must be configured between each pair of routers.
- Recall that the **neighbor** command enables BGP updates between BGP speakers.
- By default, each BGP speaker is assumed to have a **neighbor** statement for all other iBGP speakers in the autonomous system; this is known as *full-mesh iBGP* .
- When a change is received from an external autonomous system (eBGP), the receiving BGP router is responsible for informing all of its iBGP neighbors of the change.
- iBGP neighbors that receive this update do not send it to any other iBGP neighbor because they assume that the sending eBGP/iBGP neighbor is fully meshed with all other iBGP speakers and has sent each iBGP neighbor the update.

BGP Partial-Mesh and Full-Mesh Examples



Basic BGP Configuration Requirements

The next step is to gather the parameters needed to provide the BGP configuration details. For basic BGP, these details include the following:

- The autonomous system numbers (of your own network and of all remote autonomous systems)
- The IP addresses of all the neighbors (peers) involved
- The networks that are to be advertised into BGP

Basic BGP configuration requires the following main steps:

- **Step 1.** Define the BGP process.
- **Step 2.** Establish the neighbor relationships.
- **Step 3.** Advertise the networks into BGP.

Entering BGP Configuration Mode

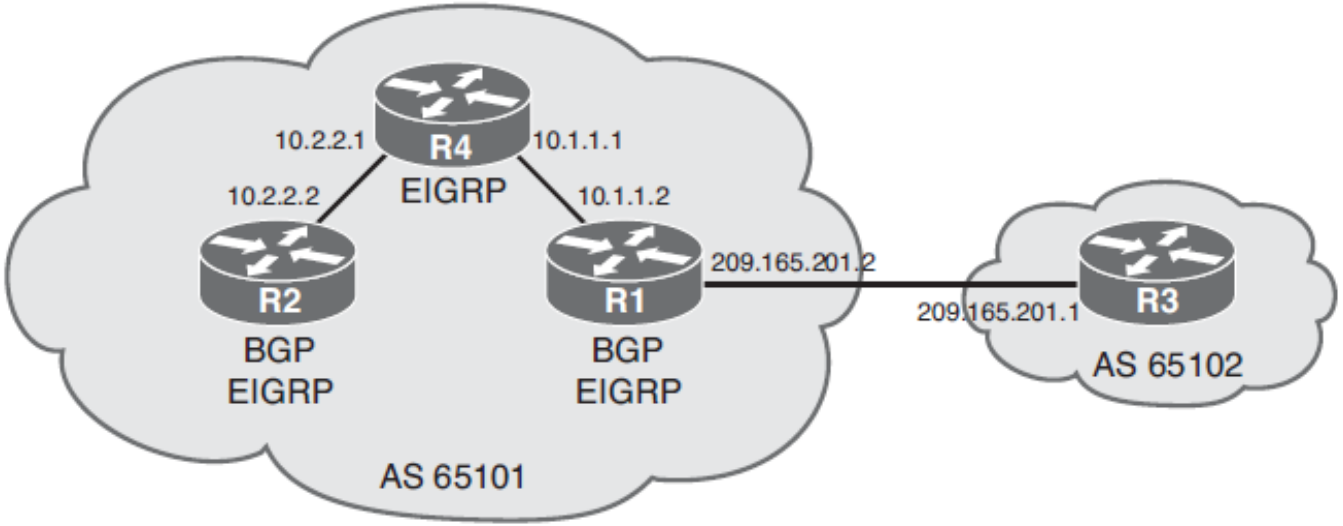
- Use the **`router bgp`** *`autonomous-system`* global configuration command to enter BGP configuration mode and identify the local autonomous system in which this router belongs.
- The BGP process needs to be informed of its autonomous system so that when BGP neighbors are configured it can determine whether they are iBGP or eBGP neighbors.
- Only one instance of BGP can be configured on a router at a time. For example, if you configure your router in autonomous system 65000 and then try to configure the **router bgp 65100** command, the router informs you that you are currently configured for autonomous system 65000.

Defining BGP Neighbors and Activating BGP Sessions

- Use the **`neighbor`** *`ip-address`* **`remote-as`** *`autonomous-system`* router configuration command to activate a BGP session for external and internal neighbors and to identify a peer router with which the local router will establish a session.

Parameter	Description
ip-address	Identifies the peer router
autonomous-system	Identifies the peer router's autonomous system

BGP Configuration Example



R1

```
router bgp 65101
  neighbor 10.2.2.2 remote-as 65101
  neighbor 209.165.201.1 remote-as 65102
```

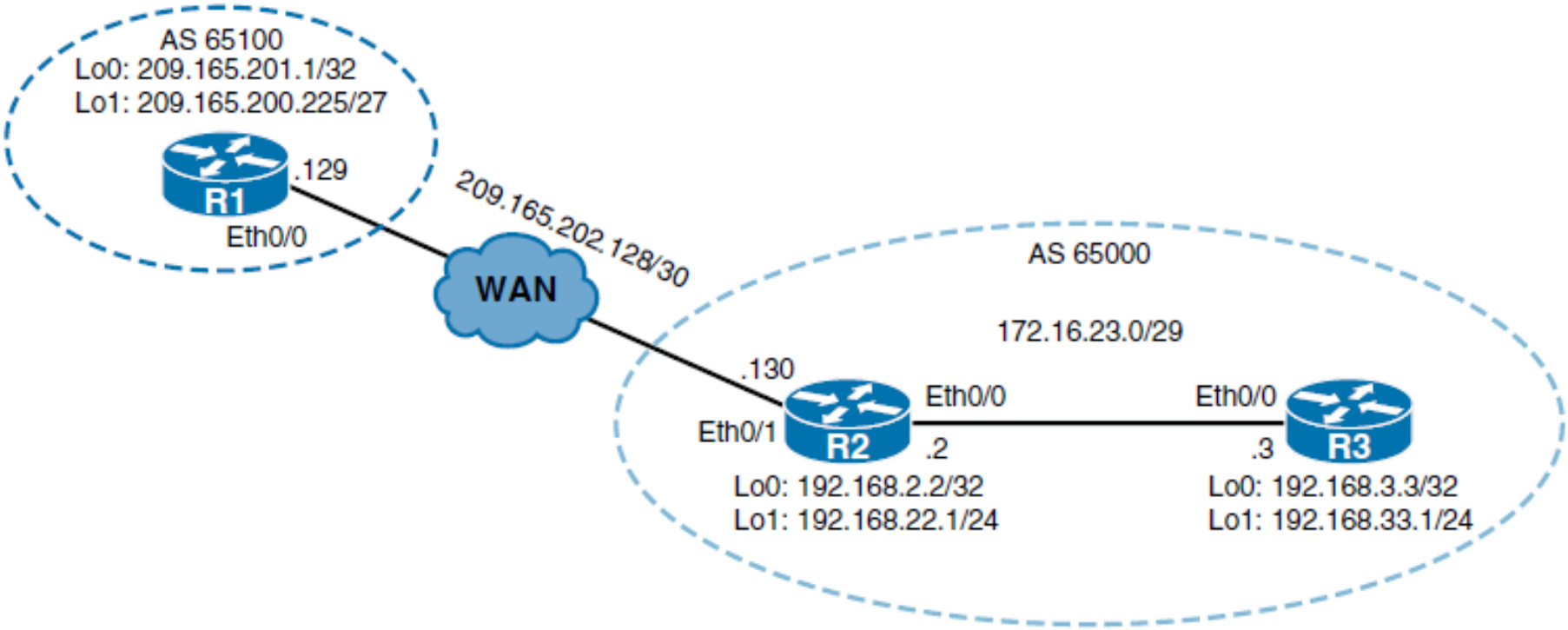
R2

```
router bgp 65101
  neighbor 10.1.1.2 remote-as 65101
```

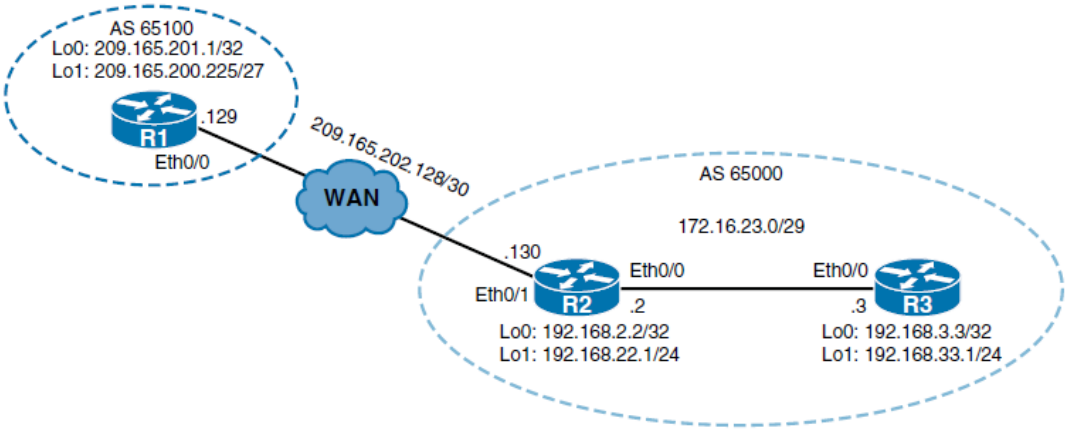
R3

```
router bgp 65102
  neighbor 209.165.201.2 remote-as 65101
```

Basic BGP Configuration and Verification



Configuring and Verifying an eBGP Session




```
R1(config)# router bgp 65100
R1(config-router)# neighbor 209.165.202.130 remote-as 65000
```

```
R2(config)# router bgp 65000
R2(config-router)# neighbor 209.165.202.129 remote-as 65100
```

```
R1# show ip bgp summary
BGP router identifier 209.165.201.1, local AS number 65100
BGP table version is 1, main routing table version 1
Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
209.165.202.130 4 65000      91      93        1    0    0 01:20:28        0
```

Verifying an eBGP Session

```
R1# show ip bgp summary
BGP router identifier 209.165.201.1, local AS number 65100
BGP table version is 1, main routing table version 1
Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
209.165.202.130 4 65000      91      93        1    0    0 01:20:28        0
```

The first part of this command output describes the local router:

- **BGP router identifier:** The IP address that all other BGP speakers recognize as representing this router
- **Local AS number:** The local router's autonomous system number

The next part of this command output describes the BGP table:

- **BGP table version:** This is the version number of the local BGP table; it increases when the BGP table changes.
- **Main routing table version:** This is the last version of BGP database that was injected into the main routing table.

Verifying an eBGP Session

```
R1# show ip bgp summary
BGP router identifier 209.165.201.1, local AS number 65100
BGP table version is 1, main routing table version 1
Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
209.165.202.130 4 65000      91      93        1    0    0 01:20:28        0
```

- **Neighbor:** The IP address, used in the **neighbor** statement
- **Version (V):** The version of BGP this router is running with the listed neighbor.
- **AS:** The listed neighbor's autonomous system number.
- **Messages received (MsgRcvd):** The number of BGP messages received from this neighbor.
- **Messages sent (MsgSent):** The number of BGP messages sent to this neighbor.
- **TblVer:** The last version of the BGP table that was sent to this neighbor.

Verifying an eBGP Session

```
R1# show ip bgp summary
BGP router identifier 209.165.201.1, local AS number 65100
BGP table version is 1, main routing table version 1
Neighbor        V     AS MsgRcvd MsgSent   TblVer  InQ  OutQ  Up/Down  State/PfxRcd
209.165.202.130 4  65000      91      93        1    0     0  01:20:28        0
```

- **In queue (InQ):** The number of messages from this neighbor that are waiting to be processed.
- **Out queue (OutQ):** The number of messages queued and waiting to be sent to this neighbor. TCP flow control prevents this router from overwhelming a neighbor with a large update.
- **Up/down:** The length of time this neighbor has been in the current BGP state (established, active, or idle).
- **State:** The current state of the BGP session **:** active, idle, open sent, open confirm, or idle (admin).
- **Prefix received (PfxRcd):** When the session is in the established state, this value represents the number of BGP network entries received from this neighbor.

Verifying an eBGP Session

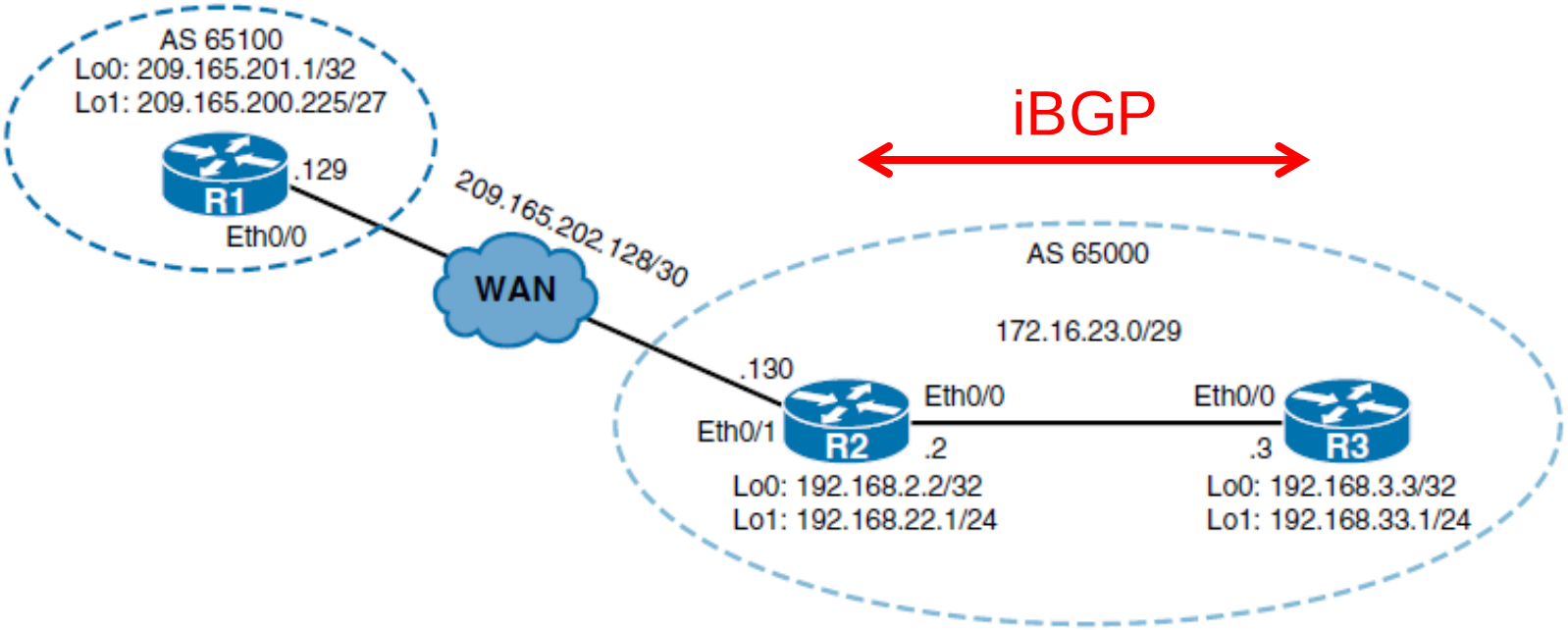
- The **`show ip bgp neighbors`** command supplies additional information, such as the negotiated capabilities, supported address families, and others.

```
R1# show ip bgp neighbors
BGP neighbor is 209.165.202.130,  remote AS 65000, external link
  BGP version 4, remote router ID 192.168.22.1
  BGP state = Established, up for 01:21:17
  Last read 00:00:25, last write 00:00:00, hold time is 180, keepalive interval is
60 seconds
  Neighbor sessions:
    1 active, is not multisession capable (disabled)
  Neighbor capabilities:
    Route refresh: advertised and received(new)
    Four-octets ASN Capability: advertised and received
    Address family IPv4 Unicast: advertised and received
    Enhanced Refresh Capability: advertised and received
    Multisession Capability:
    Stateful switchover support enabled: NO for session 1
  Message statistics:
    InQ depth is 0
    OutQ depth is 0

                         Sent       Rcvd
    Opens:                  1          1
    Notifications:          0          0
    Updates:                1          1
    Keepalives:            92         90
    Route Refresh:          0          0
    Total:                 94         92
  Default minimum time between advertisement runs is 30 seconds

 For address family: IPv4 Unicast
  Session: 209.165.202.130
<Output omitted>
```

show ip bgp neighbors Command Options

```
R1# show ip bgp neighbors 209.165.202.130 ?
  advertised-routes  Display the routes advertised to a BGP neighbor
  dampened-routes    Display the dampened routes received from neighbor (eBGP
                     peers only)
  flap-statistics    Display flap statistics of the routes learned from
                     neighbor (eBGP peers only)
  paths              Display AS paths learned from neighbor
  policy             Display neighbor polices per address-family
  received           Display information received from a BGP neighbor
  received-routes    Display the received routes from neighbor
  routes             Display routes learned from neighbor
  |                  Output modifiers
  <cr>
```

Configuring and Verifying an iBGP Session




```
R2(config)# router bgp 65000
R2(config-router)# neighbor 172.16.23.3 remote-as 65000

R3(config)# router bgp 65000
R3(config-router)# neighbor 172.16.23.2 remote-as 65000
```

Configuring and Verifying an iBGP Session

```
R2(config)# router bgp 65000
R2(config-router)# neighbor 172.16.23.3 remote-as 65000

R3(config)# router bgp 65000
R3(config-router)# neighbor 172.16.23.2 remote-as 65000
```

```
R2# show ip bgp summary
BGP router identifier 192.168.22.1, local AS number 65000
BGP table version is 1, main routing table version 1
Neighbor        V    AS  MsgRcvd  MsgSent  TblVer  InQ  OutQ  Up/Down  State/PfxRcd
172.16.23.3     4  65000    13       13       1     0     0  00:08:23        0
209.165.202.129 4  65100   287      284       1     0     0  04:16:06        0

R2# show ip bgp neighbors
BGP neighbor is 172.16.23.3,  remote AS 65000, internal link
  BGP version 4, remote router ID 192.168.33.1
  BGP state = Established, up for 00:08:38
<Output omitted>
```

Advertising Networks in BGP

- Use the **network** *network-number* [**mask** *network-mask*] router configuration command to inject routes that are present in the IPv4 routing table into the BGP table so that they can be advertised in BGP.

Parameter	Description
network-number	Identifies an IPv4 network to be advertised by BGP.
mask *network-mask*	(Optional) Identifies the subnet mask to be advertised by BGP. If the network mask is not specified, the default mask is the classful mask.

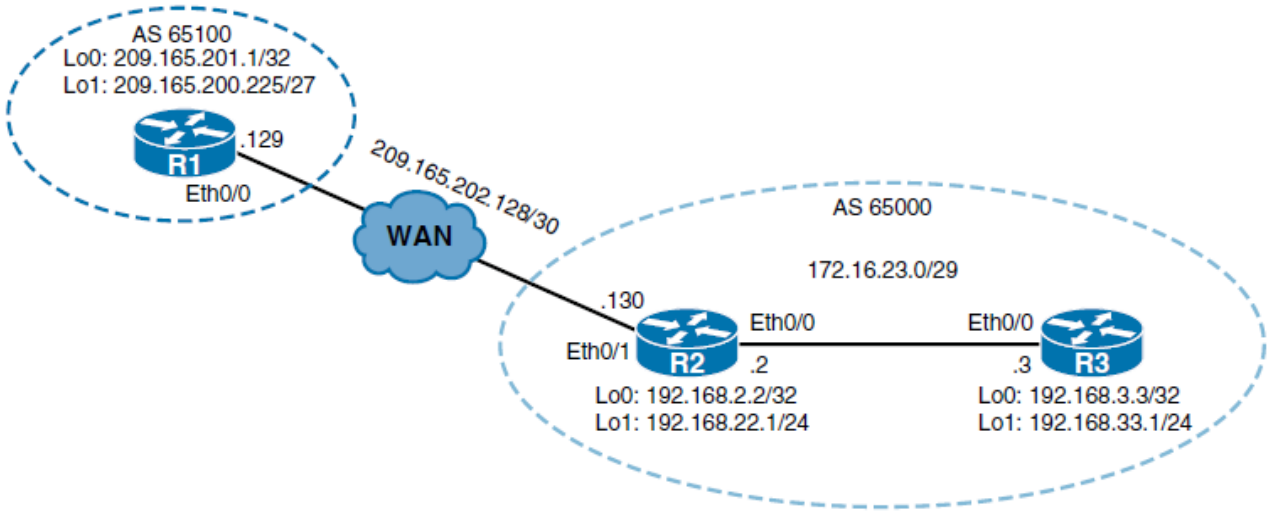
Advertising Networks in BGP

- It is important to note that the BGP **network** command determines which networks this router advertises.
- Unlike for IGPs, the **network** command does not start BGP on specific interfaces.
- Rather, it indicates to BGP which networks it should originate from this router.
- The list of **network** commands must include all networks in your autonomous system that you want to advertise, not just those locally connected to your router.
- The **mask** parameter indicates that BGP-4 allows classless prefixes; it can advertise subnets and supernets.
- Notice the difference between the **neighbor** command and the **network** command: The **neighbor** command tells BGP where to advertise; the **network** command tells BGP what to advertise.
- If the **mask** parameter is not specified, this command announces only the classful network number; at least one subnet of the specified major network must be present in the IP routing table to allow BGP to start announcing the classful network as a BGP route.

Advertising Networks in BGP

- If the **mask** *network-mask* is specified, an exact match to the network (both address and mask) must exist in the routing table for the network to be advertised.
- Before BGP announces a route, it checks to see whether it can reach it.
- For example, if you want to advertise the 192.168.0.0/24 route, and by mistake you configure network **192.168.0.0 mask 255.255.0.0** instead of **network 192.168.0.0 mask 255.255.255.0**, BGP looks for 192.168.0.0/16 in the routing table. In this case, it would find 192.168.0.0/24 but will not find 192.168.0.0/16. Because the routing table does not contain a specific match to the network, BGP does not announce the 192.168.0.0/24 network to any neighbors.
- If you want to advertise the CIDR block 192.168.0.0/16, you might try configuring network 192.168.0.0 mask 255.255.0.0. Again, BGP looks for 192.168.0.0/16 in the routing table, and if it never finds 192.168.0.0/16, BGP does not announce the 192.168.0.0/16 network to any neighbors. In this case, you can configure a static route to the CIDR block toward the null interface, with the **ip route 192.168.0.0 255.255.0.0 null0** command, so that BGP can find an exact match in the routing table.
- After finding an exact match in the routing table, BGP announces the 192.168.0.0/16 network to its neighbors.

Advertising Networks in BGP - Example




```
R3(config)#router bgp 65000
R3(config-router)#network 192.168.33.0 mask 255.255.255.0
```

```
R3# show ip bgp
BGP table version is 2, local router ID is 192.168.33.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
*>   192.168.33.0     0.0.0.0                  0         32768 i
```

Examining the BGP Table

```
R3# show ip bgp
BGP table version is 2, local router ID is 192.168.33.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
*>   192.168.33.0     0.0.0.0                  0         32768 i
```

- The *status codes* are shown at the beginning of each line of output, and the *origin codes* are shown at the end of each line.
- A row with an asterisk (*) in the first column means that the next-hop address is valid.
- A greater-than sign (>) in the second column indicates the best path for a route selected by BGP.
- This route is offered to the IP routing table.

Examining the BGP Table

```
R3# show ip bgp
BGP table version is 2, local router ID is 192.168.33.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
*>   192.168.33.0     0.0.0.0                  0         32768 i
```

- The third column is either blank or has an *i* in it. If it is blank, BGP learned that route from an external peer (eBGP). If it has an *i*, an iBGP neighbor advertised this route to this router.
- The fourth column lists the networks that the router learned.
- The fifth is the Next Hop - If this column contains 0.0.0.0, this router originated the route.

Examining the BGP Table

```
R3# show ip bgp
BGP table version is 2, local router ID is 192.168.33.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
*>   192.168.33.0     0.0.0.0                  0         32768 i
```

- The next three columns list three BGP path attributes associated with the path: **Metric**, which is also called the multi-exit discriminator (MED); **Local preference** (LocPrf); and **Weight**.
- The column with the Path header may contain a sequence of autonomous systems in the path – where from left to right, the first autonomous system listed is the adjacent autonomous system from which this network was learned.

Examining the BGP Table

```
R3# show ip bgp
BGP table version is 2, local router ID is 192.168.33.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
*>   192.168.33.0     0.0.0.0                  0         32768 i
```

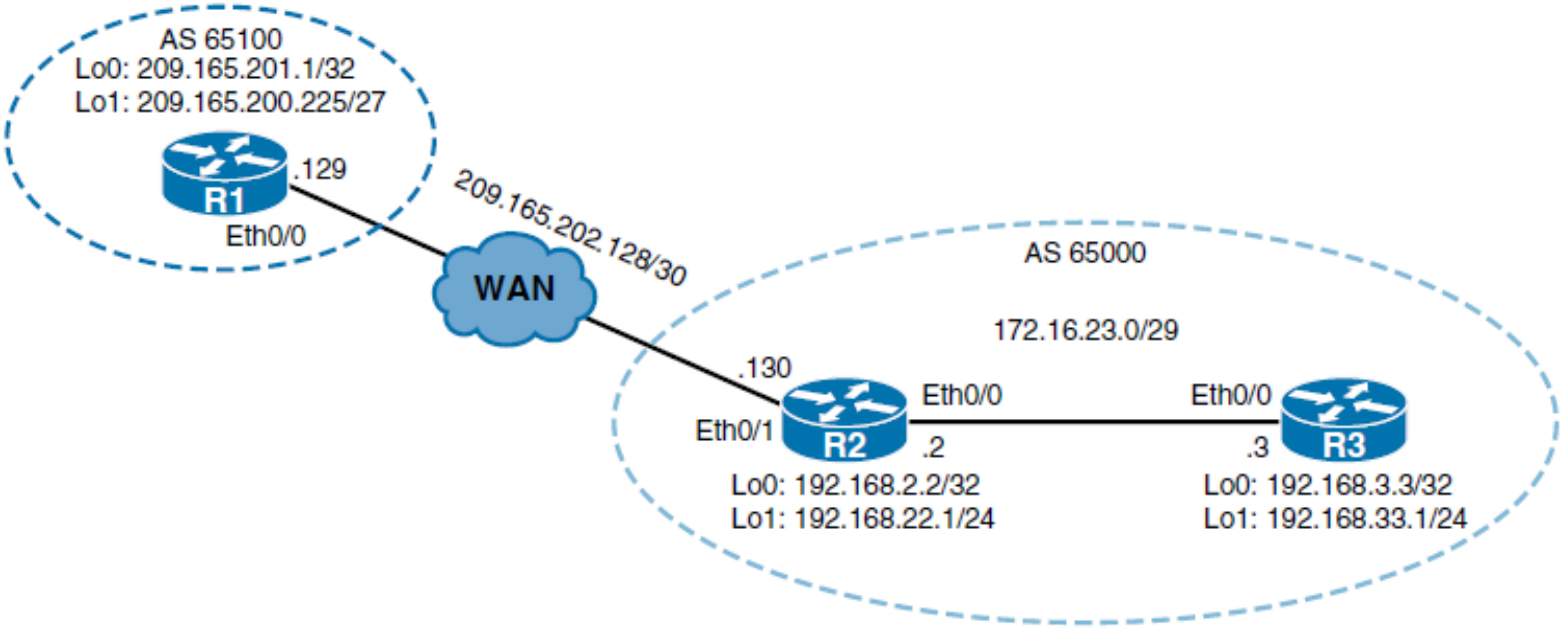
- If the last column has an ***i*** in it, the original router probably used a **network** command to introduce this network into BGP.
- The character ***e*** signifies that the original router learned this network from EGP, which is the historic predecessor to BGP.
- A question mark (***?***) signifies that the original BGP process cannot absolutely verify this network's availability because it is redistributed from an IGP into the BGP process.

Examining the BGP Table

```
R2# show ip bgp
BGP table version is 4, local router ID is 192.168.22.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
<Output omitted>

     Network          Next Hop            Metric LocPrf Weight Path
 *>i 192.168.33.0     172.16.23.3              0    100      0 i

R2# show ip route bgp
<Output omitted>
B     192.168.33.0/24 [200/0] via 172.16.23.3, 01:20:57
```

```
R1# show ip bgp
BGP table version is 4, local router ID is 209.165.201.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
<Output omitted>
     Network          Next Hop          Metric LocPrf Weight Path
 *>  192.168.33.0     209.165.202.130                      0 65000 i

R1# show ip route bgp
<Output omitted>
B     192.168.33.0/24 [20/0] via 209.165.202.130, 01:16:42
```

Next-Hop-Self Feature




```
R1(config)# router bgp 65100
R1(config-router)# network 209.165.200.224 mask 255.255.255.224
```

Next-Hop-Self Feature

```
R2# show ip bgp
BGP table version is 6, local router ID is 192.168.22.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
<Output omitted>
     Network          Next Hop            Metric  LocPrf  Weight  Path
 *>  192.168.22.0    0.0.0.0                  0           32768  i
 *>i 192.168.33.0    172.16.23.3              0     100        0  i
 *>  209.165.200.224/27
                      209.165.202.129          0                0  65100 i

R2# show ip route bgp
<Output omitted>

B      192.168.33.0/24 [200/0] via 172.16.23.3, 20:15:50
       209.165.200.0/27 is subnetted, 1 subnets
B         209.165.200.224 [20/0] via 209.165.202.129, 00:00:56
```

Next-Hop-Self Feature

```
R3# show ip bgp
BGP table version is 3, local router ID is 192.168.33.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
 *>i 192.168.22.0     172.16.23.2              0    100      0 i
 *>  192.168.33.0     0.0.0.0                  0         32768 i
 * i 209.165.200.224/27
                      209.165.202.129          0    100      0 65100 i

R3# show ip route bgp
<Output omitted>
B     192.168.22.0/24 [200/0] via 172.16.23.2, 05:56:53

R3# show ip route 209.165.202.129
% Network not in table
```

- Notice, though, that the entry is not designated as a best route; the > character is missing.
- This is because R3 does not have a route to the next-hop address (209.165.202.129), and therefore the route is not installed in the routing table.

Configuring Next-Hop

```
R2(config)# router bgp 65000
R2(config-router)# neighbor 172.16.23.3 next-hop-self
```

```
R3# show ip bgp
<Output omitted>

     Network          Next Hop            Metric  LocPrf  Weight  Path
 *>i 192.168.22.0     172.16.23.2              0     100       0  i
 *>  192.168.33.0     0.0.0.0                  0           32768  i
 *>i 209.165.200.224/27
            172.16.23.2                        0     100       0  65100 i
R3#show ip route bgp
<Output omitted>

B     192.168.22.0/24 [200/0] via 172.16.23.2, 06:57:03
      209.165.200.0/27 is subnetted, 1 subnets
B        209.165.200.224 [200/0] via 172.16.23.2, 00:02:51
```

Understanding and Troubleshooting BGP Neighbor States

After the neighbor command executed, the BGP application tries to set up a session with the neighbor. BGP is a state machine that takes a router through the following states with its neighbors:

- **Idle**
  - The router is searching the routing table to see whether a route exists to reach the neighbor.
- **Connect**
  - The router found a route to the neighbor and has completed the TCP three-way handshake.
- **Open sent**
  - An open message was sent, with the parameters for the BGP session.
- **Open confirm**
  - The router received agreement on the parameters for establishing a session.
  - Alternatively, the router goes into the active state if there is no response to the open message.
- **Established**
  - Peering is established and routing begins.

Idle state

- After you enter the **neighbor remote-as** command, BGP starts in the *idle* state, and the BGP process checks that it has a route to the IP address listed.
- BGP should be in the idle state for only a few seconds.
- However, if BGP does not find a route to the neighboring IP address, it stays in the idle state.

Connect state

- If it finds a route, it goes to the *connect* state when the TCP handshaking synchronize acknowledge (SYN ACK) packet returns (when the TCP three-way handshake is complete).
- After the TCP connection is set up, the BGP process creates a BGP open message and sends it to the neighbor.

open sent state

- After BGP dispatches this open message, the BGP peering session changes to the *open sent* state.

active state

- If there is no response for 5 seconds, the state changes to the *active* state.

open confirm state

- If a response does come back in a timely manner, BGP goes to the *open confirm* state and starts scanning (evaluating) the routing table for the paths to send to the neighbor.

established state

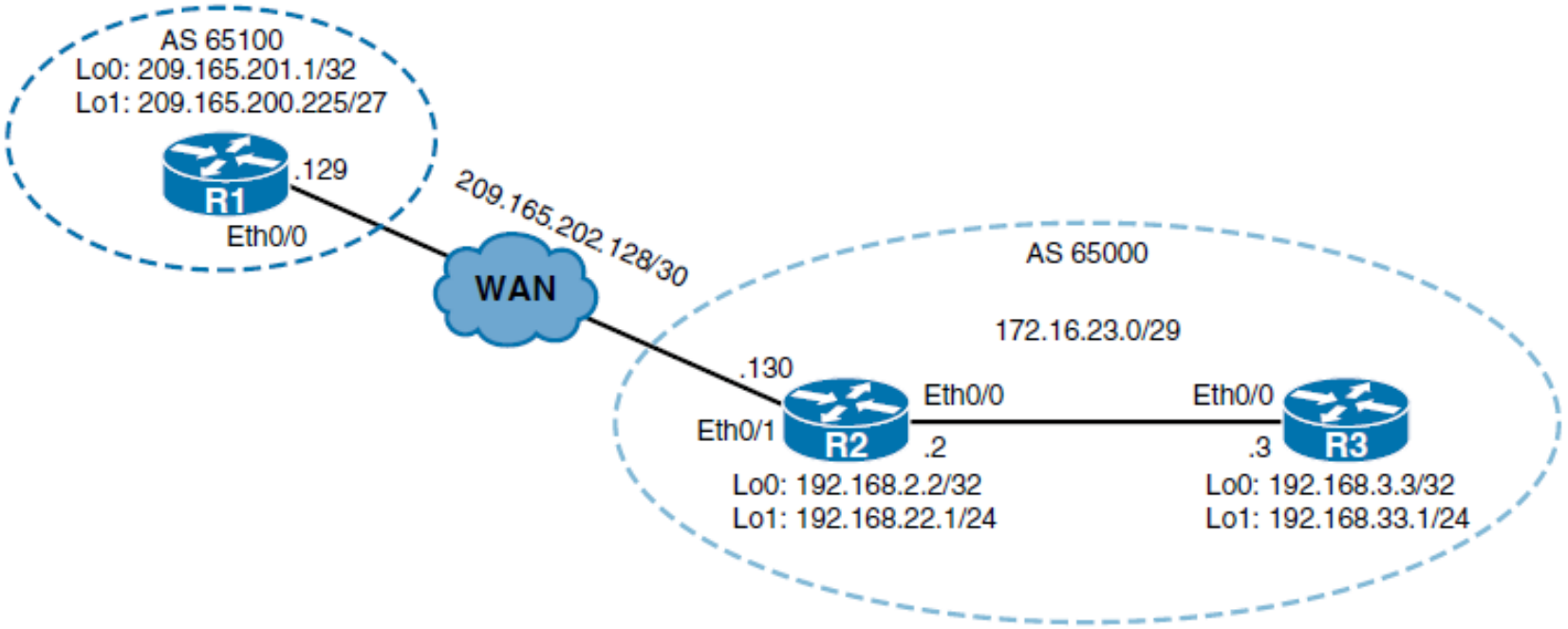
- When these paths have been found, BGP then goes to the *established* state and begins routing between the neighbors.

Note: The BGP state is shown in the last column of the **show ip bgp summary** command output.

iBGP Session Resilience

- In cases where multiple paths exist to reach an iBGP neighboring routers, the routers could peer with each other's loopback interface address and the BGP session would not be lost because loopback interfaces are always available as long as the router itself does not fail.
- This peering arrangement adds resiliency to the iBGP sessions because they are not tied into a physical interface, which might go down for any number of reasons.

iBGP Session Resilience




```
R2(config)# router bgp 65000
R2(config-router)# no neighbor 172.16.23.3
R2(config-router)# neighbor 192.168.3.3 remote-as 65000
R2(config-router)# neighbor 192.168.3.3 next-hop-self

R3(config)# router bgp 65000
R3(config-router)# no neighbor 172.16.23.2
R3(config-router)# neighbor 192.168.2.2 remote-as 65000
```

Sourcing iBGP from Loopback Address

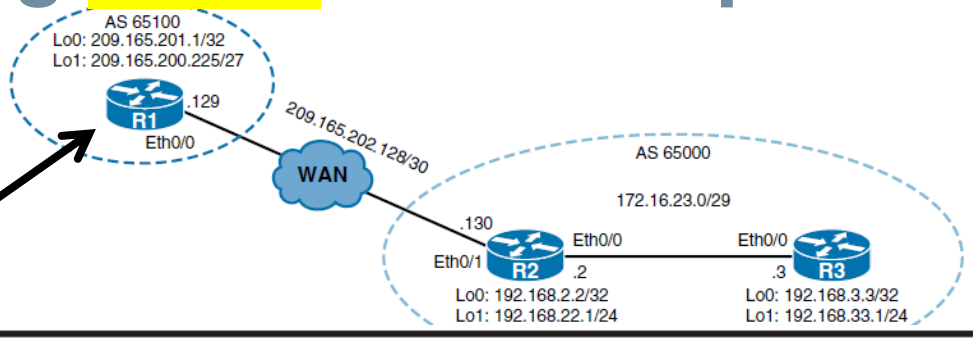
- For BGP packets, the source IP address must match the address in the corresponding **neighbor** statement on the other router. Otherwise, the routers will not be able to establish the BGP session, and the packet will be ignored.
- BGP does not accept unsolicited updates; it must be aware of every neighboring router and have a **neighbor** statement for it.

```
R2# show ip bgp summary
<Output omitted>

Neighbor        V     AS MsgRcvd MsgSent   TblVer  InQ  OutQ  Up/Down  State/PfxRcd
192.168.3.3     4  65000       0       0        1    0     0  00:14:59     Idle
209.165.202.129 4  65100    2980    2981        9    0     0  1d21h           1

R3# show ip bgp summary
<Output omitted>

Neighbor        V     AS MsgRcvd MsgSent   TblVer  InQ  OutQ  Up/Down  State/PfxRcd
192.168.2.2     4  65000       0       0        1    0     0  never       Idle
```

Sourcing iBGP from Loopback Address

```
R2(config)# router bgp 65000
R2(config-router)# neighbor 192.168.3.3 update-source Loopback 0

R3(config)# router bgp 65000
R3(config-router)# neighbor 192.168.2.2 update-source Loopback 0

R3# show ip bgp summary
<Output omitted>

Neighbor        V     AS    MsgRcvd  MsgSent   TblVer  InQ  OutQ    Up/Down  State/PfxRcd
192.168.2.2     4   65000          8        8       12    0     0    00:02:38            2
```

Sourcing eBGP from Loopback Address




```
R1(config)# ip route 192.168.2.2 255.255.255.255 209.165.202.130
R1(config)# router bgp 65100
R1(config-router)# no neighbor 209.165.202.130
R1(config-router)# neighbor 192.168.2.2 remote-as 65000
R1(config-router)# neighbor 192.168.2.2 update-source Loopback 0

R2(config)# ip route 209.165.201.1 255.255.255.255 209.165.202.129
R2(config)# router bgp 65000
R2(config-router)# no neighbor 209.165.202.129
R2(config-router)# neighbor 209.165.201.1 remote-as 65100
R2(config-router)# neighbor 209.165.201.1 update-source Loopback 0

R1# show ip bgp summary
<Output omitted>
Neighbor        V     AS MsgRcvd MsgSent   TblVer  InQ  OutQ  Up/Down  State/PfxRcd
192.168.2.2     4  65000       0       0        1    0     0  never       Idle
```

eBGP Multihop

- To fix this issue, you must also enable multihop eBGP, with the **neighbor** *ip-address* **ebgp-multihop** [*ttl*] router configuration command.

```
R1(config)# router bgp 65100
R1(config-router)# neighbor 192.168.2.2 ebgp-multihop

R2(config)# router bgp 65000
R2(config-router)# neighbor 209.165.201.1 ebgp-multihop

R1# show ip bgp summary
<Output omitted>

Neighbor        V    AS   MsgRcvd  MsgSent  TblVer  InQ  OutQ  Up/Down   State/PfxRcd
192.168.2.2     4 65000         6        5      12    0     0  00:00:30            2
```

- if no [*ttl*] value defined, it will use default for this command -> default is *ttl*=2

Resetting BGP Sessions

- **Hard Reset** of BGP Sessions
  - Resetting a session is a method of informing the neighbor or neighbors of a policy change. If BGP sessions are reset, all information received on those sessions is invalidated and removed from the BGP table. The remote neighbor detects a BGP session down state and, likewise, invalidates the received routes. After a period of 30 to 60 seconds, the BGP sessions are reestablished automatically, and the BGP table is exchanged again, but through the new filters. However, resetting the BGP session disrupts packet forwarding.
  - Use the **clear ip bgp *** or **clear ip bgp** { *neighbor-address* } privileged EXEC command to cause a hard reset of the BGP neighbors involved, where * indicates all sessions and the *neighbor-address* identifies the address of a specific neighbor for which the BGP sessions will be reset.

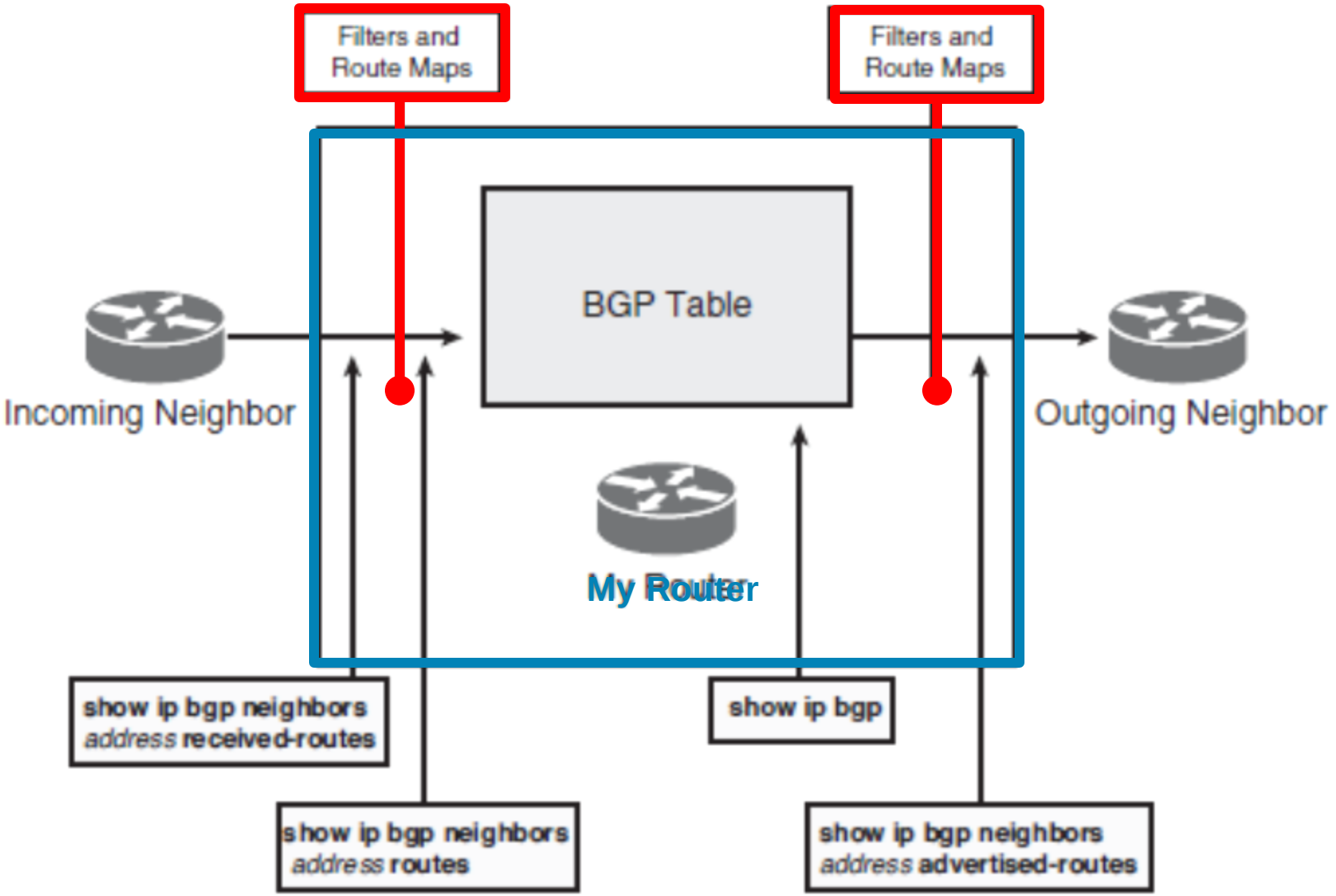
Resetting BGP Sessions

- **Soft Reset** or Route Refresh
  - Use the **clear ip bgp** {* | *neighbor-address* } **out** privileged EXEC command to cause BGP to do a soft reset for outbound updates.
  - The router on which this command is issued does not reset the BGP session. Instead, the router creates a new update and sends the whole table to the specified neighbors. This update includes withdrawal commands for networks that the neighbor will not see anymore, based on the new outbound policy.
  - Outbound BGP soft configuration does not have any memory overhead. This command is highly recommended when you are changing an outbound policy, but does not help if you are changing an inbound policy ...

Monitoring Soft Reconfiguration

- When a BGP session is reset using soft reconfiguration, the following commands can be useful for monitoring the BGP routes received, sent, or filtered
  - **`show ip bgp neighbors`** { *`address`* } **`received-routes:`** Displays all received routes (both accepted and **rejected**) from the specified neighbor.
  - **`show ip bgp neighbors`** { *`address`* } **`routes:`** Displays all routes that are received and **accepted** from the specified neighbor. This output is a subset of the output displayed by the **received-routes** keyword.
  - **`show ip bgp:`** Displays entries in the BGP table.
  - **`show ip bgp neighbors`** { *`address`* } **`advertised-routes:`** Displays all BGP routes that have been advertised to neighbors.

Monitoring Soft Reconfiguration



BGP Attributes and the Path-Selection Process

BGP Path Selection

- A router running BGP may receive updates about destinations from multiple neighbors, some in different autonomous systems, and therefore **multiple paths might exist to reach a given network**.
- These are kept in the BGP table.
- BGP chooses **only a single best path** to reach a specific destination.
- BGP is not designed to perform **load balancing**; paths are chosen **because of policy**, not based on bandwidth.
- The BGP selection process eliminates any multiple paths until a single best path is left.
- The best BGP path is submitted to the IP routing table manager process and is evaluated against any other routing protocols that can also reach that network.

BGP Path-Selection Process

show ip bgb:

Step 1. Prefer the route with the **highest** weight. (The weight is Cisco proprietary and is local to the router only.) **0, 32768**

Step 2. If multiple routes have the same weight, prefer the route with the **highest** local preference. (The local preference is used within an autonomous system.) **100**

Step 3. If multiple routes have the same local preference, prefer the route that was originated by the local router. (A locally originated route has a next hop of 0.0.0.0 in the BGP table.)

Step 4. If none of the routes were originated by the local router, prefer the route with the **shortest** AS-path.

Step 5. If the AS-path length is the same, prefer the **lowest**-origin code (IGP (***i***) < EGP (***e***) < incomplete (***?***)).

Step 6. If all origin codes are the same, prefer the path with the **lowest** MED. (The MED is exchanged between ASs)

BGP Path-Selection Process

Step 7. If the routes have the same MED, prefer **external BGP** paths (eBGP AD=20) over **internal** paths (iBGP AD=200).

Step 8.

a. If only internal (iBGP) paths remain, prefer the path through the **closest** IGP neighbor. This means that the router prefers the shortest internal path within the AS to reach the destination (the shortest path to the BGP next hop).

b. For eBGP paths, select the **oldest** route, to minimize the effect of routes going up and down (flapping).

Step 9. Prefer the route with the **lowest** neighbor BGP router ID value.

Step 10. If the BGP router IDs are the same, prefer the route with the **lowest** neighbor IP address.

BGP Path-Selection Process - summary

1. **Highest weight** (Cisco-proprietary BGP value).
2. **Highest local preference** (LOCAL_PREF).
3. **Prefer locally originated route** (Next Hop 0.0.0.0).
4. **Shortest AS_PATH is preferred** (Path).
5. **Choose route with lowest origin code. Internal paths are preferred over external paths, and external paths are preferred over paths with an origin of "incomplete"** (i < e < ?).
6. **Lowest multi-exit discriminator** (MED).
7. **External BGP routes** (AD20) **preferred over Internal BGP routes.**
8. **a**.) **If no external route, select path with lowest IGP cost to the next-hop router for iBGP.**

   b.) **If external route only - choose oldest** (most stable).
9. **Choose lowest BGP RID** (Router ID).
10. **Choose lowest neighbor IP address**

The Path-Selection Decision Process with a Multihomed Connection

- **Step 1** looks at weight, which by default is set to 0 for routes that were not originated by this router.
- **Step 2** compares local preference, which by default is set to 100 for all networks.
- Both Step 1 and Step 2 have an effect only if the network administrator configures the weight or local preference to a non-default value.
- **Step 3** looks at networks that are owned by this autonomous system. If one of the routes is injected (*next hop 0.0.0.0*) into the BGP table by the local router, the local router prefers it to any routes received from other BGP routers.

The Path-Selection Decision Process with a Multihomed Connection

- **Step 4** selects the path that has the fewest autonomous systems to cross. This is the most common reason a path is selected in BGP.
- **Step 5** looks at how a network was introduced into BGP. This introduction is usually either with **network** commands (*i* for an origin code) or through **redistribution** (*?* for an origin code).
- **Step 6** looks at MED to judge where the neighbor autonomous system wants this autonomous system to send packets for a given network. The Cisco IOS Software sets the MED to 0 by default. *Therefore, MED does not participate in path selection unless the network administrator of the neighbor autonomous system manipulates the paths using MED.*

The Path-Selection Decision Process with a Multihomed Connection

- The second most common decision point is **Step 7**, which states that an externally learned path from an eBGP neighbor is preferred over a path learned from an iBGP neighbor. *A router in an autonomous system prefers to use the ISP's bandwidth to reach a network rather than using internal bandwidth to reach an iBGP neighbor on the other side of its own autonomous system.*
- If the autonomous system path length is equal and the router in an autonomous system has no eBGP neighbors for that network (only iBGP neighbors), it makes sense to take the quickest path to the nearest exit point. **Step 8** looks for the closest iBGP neighbor; the IGP metric determines what closest means.

BGP Attributes

- The following are some terms defining how these attributes are implemented:
- An attribute is either well-known or optional, mandatory or discretionary, and transitive or nontransitive. An attribute might also be partial.
- Not all combinations of these characteristics are valid; path attributes fall into four separate categories:
  - Well-known mandatory
  - Well-known discretionary
  - Optional transitive
  - Optional nontransitive
- Only optional transitive attributes might be marked as partial.

Well-Known Attributes

- A well-known attribute is one that all BGP implementations must recognize and propagate to BGP neighbors.

- There are two types of well-known attributes:
  - **Well-known mandatory (povinné) attribute:** A well-known mandatory attribute *must* appear in all BGP update messages.
  - **Well-known discretionary (nepovinné) attribute:** A well-known discretionary attribute does not have to be present in all BGP update messages.

Optional Attributes

- Attributes that are not well known are called optional **(voliteľné)**.
- BGP routers that implement an optional attribute might propagate it to other BGP neighbors, depending on its meaning.
- Optional attributes are either transitive or nontransitive, as follows:
  - **Optional transitive:** BGP routers that do not implement an optional transitive attribute should pass it to other BGP routers **untouched** and mark the attribute as partial.
  - **Optional nontransitive:** BGP routers that do not implement an optional nontransitive attribute must delete the attribute and must not pass it to other BGP routers.

Defined BGP Attributes

The attributes defined by BGP include the following:

- **Well-known mandatory attributes (mONA)**
  - Origin
  - Next-hop
  - AS-path
- **Well-known discretionary attributes (daLA)**
  - Local preference
  - Atomic aggregate
- **Optional transitive attributes**
  - Aggregator
  - Community
- **Optional nontransitive attribute**
  - MED

In addition, Cisco has defined a **weight** attribute for BGP. The weight is configured locally on a router and is not propagated to any other BGP routers.

The Origin Attribute

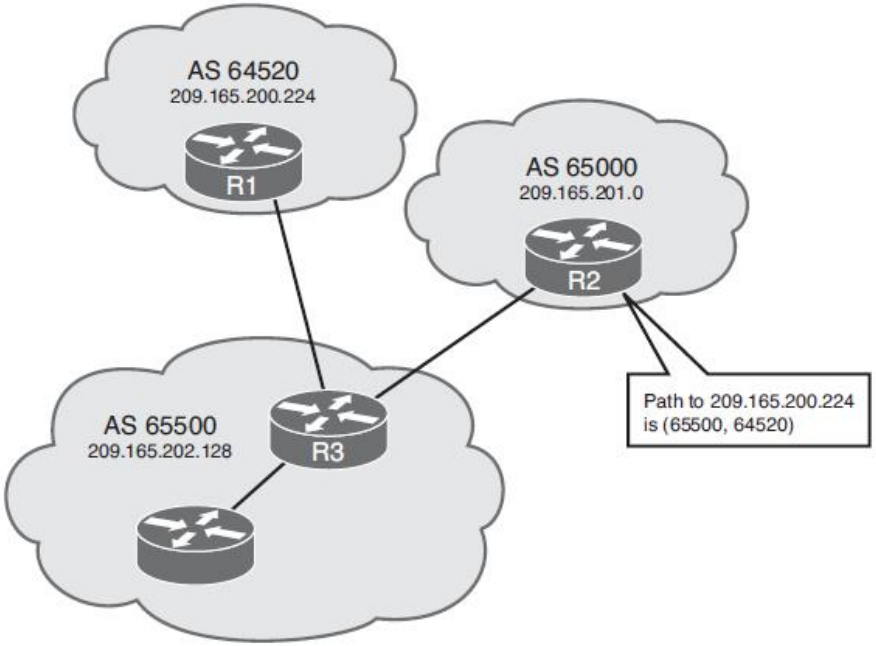
- The **ORIGIN** is a **well-known mandatory** attribute that defines the origin of the path information.
- The origin attribute can be one of three values:
- **IGP** (i)
  - This normally happens when a **network** command is used to advertise the route via BGP.
  - An origin of IGP is indicated with an *i* in the BGP table.
- **EGP** (e)
  - The route is learned via EGP.
  - EGP is considered a historic routing protocol and is not supported on the Internet.
- **Incomplete** (?)
  - The route's origin is unknown or is learned via some other means.
  - This usually occurs when a route is redistributed into BGP.

The Next-Hop Attribute

- The BGP **NEXT-HOP** attribute is a **well-known mandatory** attribute that indicates the next-hop IP address that is to be used to reach a destination.
- As discussed earlier:
  - for eBGP the next-hop address is the IP address of the neighbor that sent the update,
  - but for iBGP the next hop advertised by eBGP is carried into iBGP by default.

The AS-Path Attribute

- The **AS-PATH** attribute is the list of autonomous system numbers that a route has traversed to reach a destination, with the number of the autonomous system that originated the route at the end of the list.
- The AS-path attribute is a **well-known mandatory** attribute



The Local-Preference Attribute

- **Local preference** is **a well-known discretionary** attribute that indicates to routers in the autonomous system which path is preferred to exit the autonomous system.
- A path with a *higher* local preference is preferred.
- The term *local* refers to inside the autonomous system.
- The local preference attribute is sent only to iBGP neighbors; it is not passed to eBGP peers.
- The default value for local preference on a Cisco router is 100.

The Atomic aggregate and Aggregator Attribute

- **Atomic aggregate** is **a Well-known discretionary** attribute; it must be recognized by all BGP implementations and does not have to exist in all BGP updates.
- The purpose of the attribute is to alert BGP speakers along the path that some information have been lost due to the route aggregation process and that the aggregate path might not be the best path to the destination.
- When you use the Atomic Aggregate attribute, the BGP speaker has the option to send the **Aggregator** attribute (**Optional transitive**). The Aggregator attribute includes the AS number and the IP address of the router that originated the aggregated route. In Cisco routers, the IP address is the router ID of the router that performs the route aggregation.

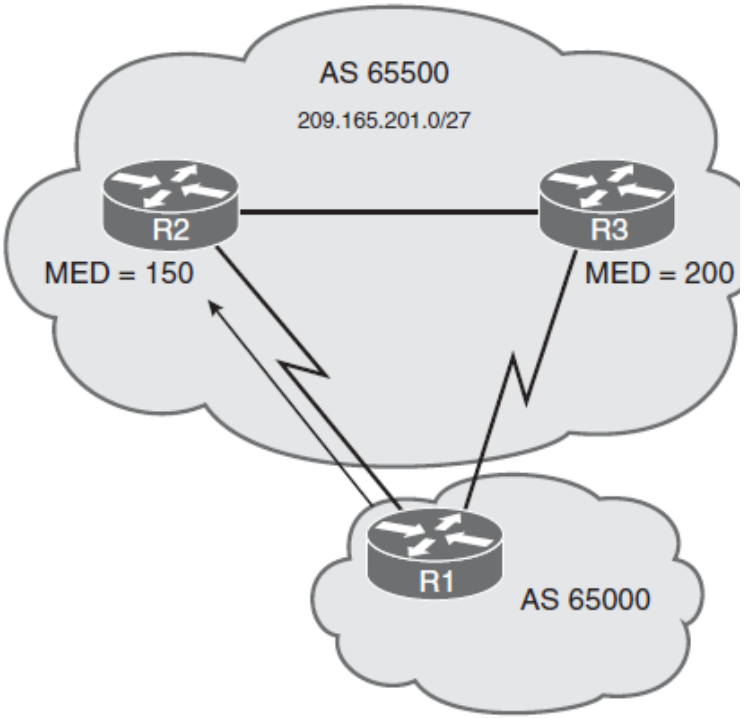
The Community Attribute

- Communities are **optional transitive** attributes.
- BGP communities allow routers to *tag* routes with an indicator (*the community*) and allow other routers to make decisions based on that tag.
- Any BGP router can tag routes in incoming and outgoing routing updates, or when doing redistribution.
- Any BGP router can filter routes in incoming or outgoing updates or can select preferred routes based on communities (the tag).
- Communities are not restricted to one network or one autonomous system, and they have no physical boundaries.

The MED Attribute

- The MED (multiple exit discriminator) attribute, also called the *metric*, is an **optional nontransitive** attribute.
- The MED indicates to *external* neighbors the preferred path *into* an autonomous system. This is a dynamic way for an autonomous system to **try to influence another autonomous system** as to which way it should choose to reach a certain route if there are multiple entry points into the autonomous system.
- A *lower* metric value is preferred.
- Unlike local preference, the MED is exchanged between autonomous systems. The MED is sent to eBGP peers; those routers propagate the MED within their autonomous system, and the routers within the autonomous system use the MED, but do not pass it on to the next autonomous system.
- When the same update is passed on to another autonomous system, the metric will be set back to the default of 0.

MED Example



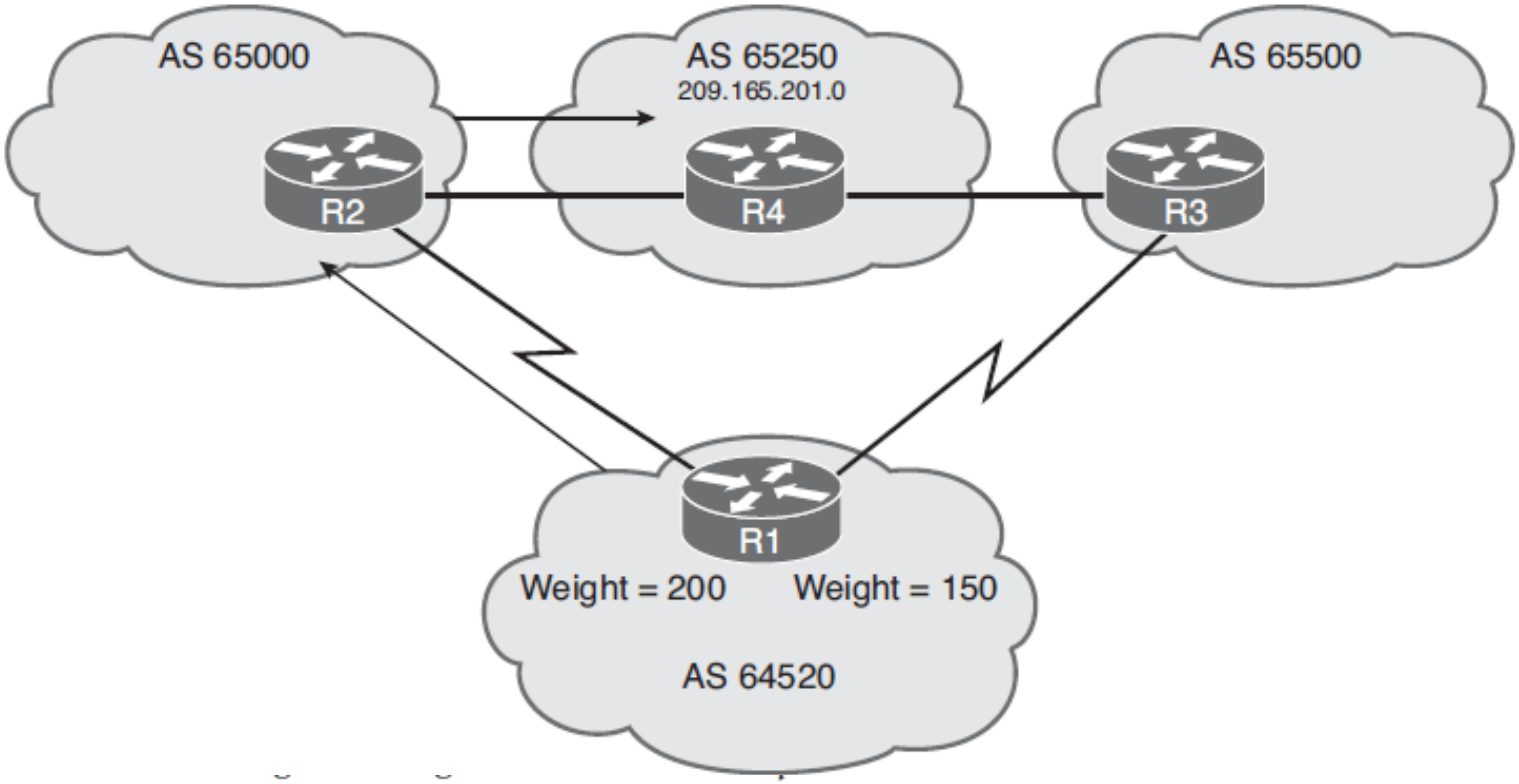

```
R2(config)# ip prefix-list PF1 permit 209.165.201.0/27
R2(config)# route-map SET-MED permit 10
R2(config-route-map)# match ip address prefix-list PF1
R2(config-route-map)# set metric 150
R2(config-route-map)# route-map SET-MED permit 20
R2(config-route-map)# exit
R2(config)# router bgp 65550
R2(config-router)# neighbor 209.165.202.129 route-map SET-MED out
```

```
R3(config)# ip prefix-list PF1 permit 209.165.201.0/27
R3(config)# route-map SET-MED permit 10
R3(config-route-map)# match ip address prefix-list PF1
R3(config-route-map)# set metric 200
R3(config-route-map)# route-map SET-MED permit 20
R3(config-route-map)# exit
R3(config)# router bgp 65550
R3(config-router)# neighbor 209.165.202.133 route-map SET-MED out
```

The Weight Attribute (Cisco Only)

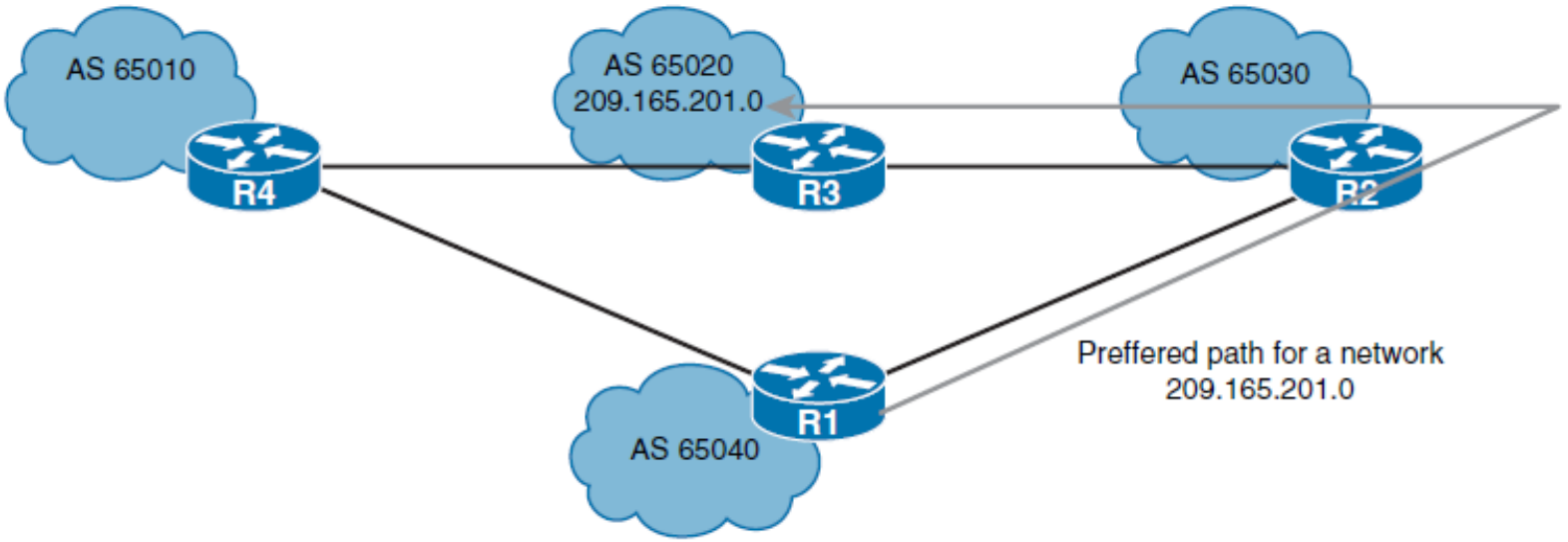
- The weight attribute is a **Cisco-defined** attribute used for the path-selection process.
- The weight attribute is configured locally and provides local routing policy only; it is *not* propagated to *any* BGP neighbors.
- Routes with a *higher* weight are preferred when multiple routes to the same destination exist.
- The weight can have a value from 0 to 65535. Paths that the router originates have a weight of 32768 by default, and other paths have a weight of 0 by default.

The Weight Attribute (Cisco Only)



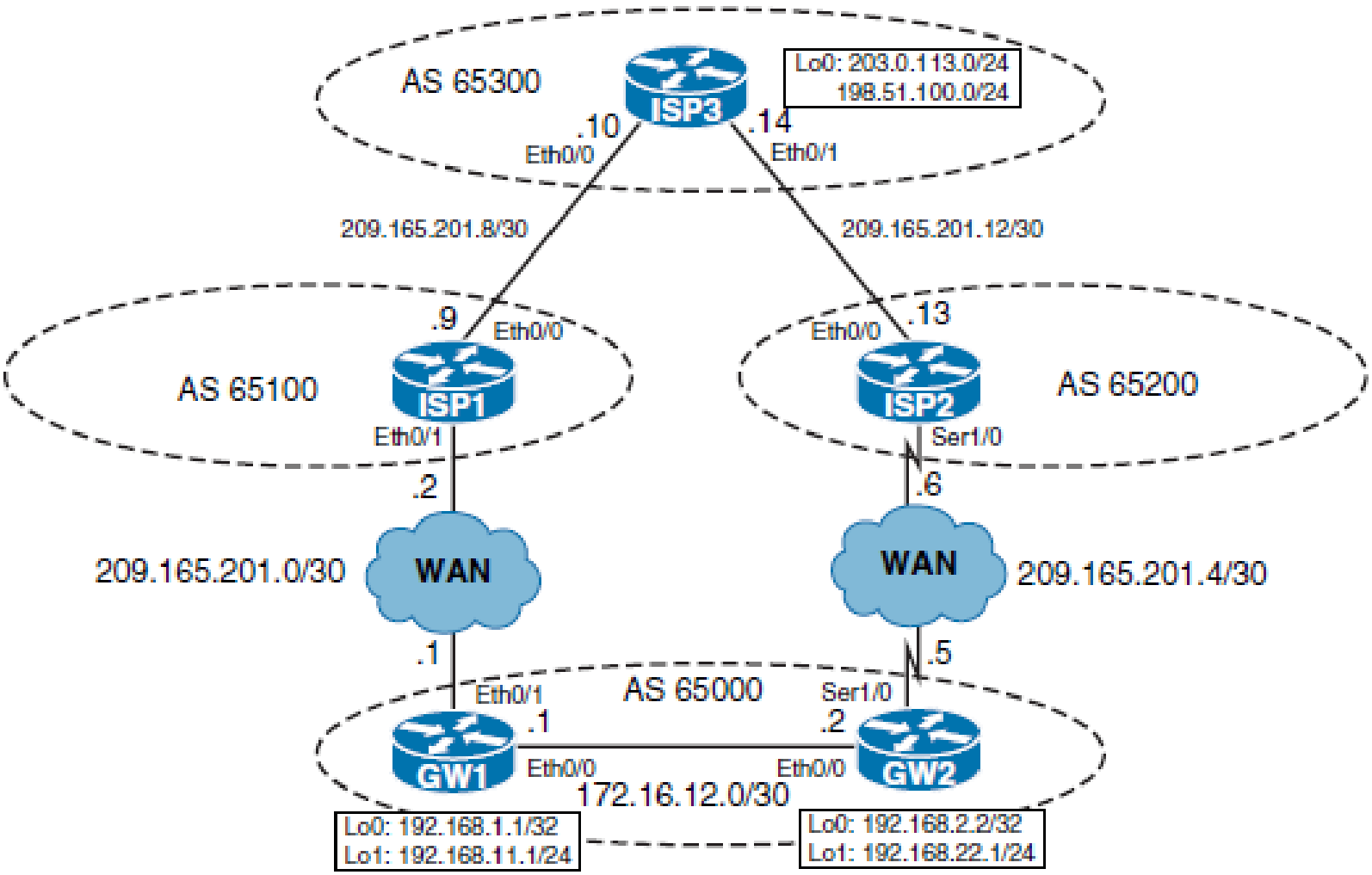
Parameter	Description
ip-address	The BGP neighbor's IP address.
weight	The weight to assign. Acceptable values are 0 to 65535. The default is 32768 for local routes (routes that the router originates). Other routes have a weight of 0 by default.

Changing the Weight Using Route Maps




```
R1(config)# ip prefix-list AS65020_ROUTES permit 209.165.201.0/24 le 28
R1(config)# route-map RM-SET-Weight permit 10
R1(config-route-map)# match ip address prefix-list AS65020_ROUTES
R1(config-route-map)# set weight 150
R1(config-route-map)# route-map RM-SET-Weight permit 20
R1(config-route-map)# set weight 100
R1(config-route-map)# exit
R1(config)# router bgp 65040
R1(config-router)# neighbor 209.165.202.129 route-map RM-SET-Weight in
```

Influencing BGP Path Selection



show ip bgp

```
GW1# show ip bgp
BGP table version is 20, local router ID is 209.165.201.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
 *>  192.168.11.0    0.0.0.0                  0         32768 i
 *>i 192.168.22.0    192.168.2.2              0    100      0 i
 *>  198.51.100.0    209.165.201.2                          0 65100 65300 i
 * i                  209.165.201.6            0    100      0 65200 65300 i
 *>  203.0.113.0     209.165.201.2                          0 65100 65300 i
 * i                  209.165.201.6            0    100      0 65200 65300 i
```

```
GW2# show ip bgp
BGP table version is 15, local router ID is 192.168.2.2
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
 *>i 192.168.11.0    192.168.1.1              0    100      0 i
 *>  192.168.22.0    0.0.0.0                  0         32768 i
 * i 198.51.100.0    209.165.201.2            0    100      0 65100 65300 i
 *>                  209.165.201.6                          0 65200 65300 i
 * i 203.0.113.0     209.165.201.2            0    100      0 65100 65300 i
 *>                  209.165.201.6                          0 65200 65300 i
```

show ip route bgp

```
GW1# show ip route bgp
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       + - replicated route, % - next hop override

Gateway of last resort is not set

B     192.168.22.0/24 [200/0] via 192.168.2.2, 23:40:37
B     198.51.100.0/24 [20/0] via 209.165.201.2, 01:18:10
B     203.0.113.0/24 [20/0] via 209.165.201.2, 01:18:10
```

```
GW2# show ip route bgp
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       + - replicated route, % - next hop override

Gateway of last resort is not set

B     192.168.11.0/24 [200/0] via 192.168.1.1, 23:56:54
B     198.51.100.0/24 [20/0] via 209.165.201.6, 01:50:22
B     203.0.113.0/24 [20/0] via 209.165.201.6, 01:50:22
```

traceroute

```
GW1# traceroute 198.51.100.1 source loopback 1
Type escape sequence to abort.
Tracing the route to 198.51.100.1
VRF info: (vrf in name/id, vrf out name/id)
  1 209.165.201.2 0 msec 0 msec 1 msec
  2 209.165.201.10 4 msec *  4 msec
GW1# traceroute 203.0.113.1 source loopback 1
Type escape sequence to abort.
Tracing the route to 203.0.113.1
VRF info: (vrf in name/id, vrf out name/id)
  1 209.165.201.2 1 msec 0 msec 1 msec
  2 209.165.201.10 0 msec *  1 msec
GW1#
```

```
GW2# traceroute 198.51.100.1 source loopback 1
Type escape sequence to abort.
Tracing the route to 198.51.100.1
VRF info: (vrf in name/id, vrf out name/id)
  1 209.165.201.6 8 msec 8 msec 8 msec
  2 209.165.201.14 8 msec *  6 msec

GW2# traceroute 203.0.113.1 source loopback 1
Type escape sequence to abort.
Tracing the route to 203.0.113.1
VRF info: (vrf in name/id, vrf out name/id)
  1 209.165.201.6 7 msec 9 msec 9 msec
  2 209.165.201.14 9 msec *  9 msec
GW2#
```

ISP3

```
ISP3# show ip bgp
<Output omitted>
     Network          Next Hop            Metric LocPrf Weight Path
 *   192.168.11.0     209.165.201.13                         0 65200 65000 i
 *>                   209.165.201.9                          0 65100 65000 i
 *   192.168.22.0     209.165.201.9                          0 65100 65000 i
 *>                   209.165.201.13                         0 65200 65000 i
 *>  198.51.100.0     0.0.0.0                  0         32768 i
 *>  203.0.113.0      0.0.0.0                  0         32768 i
```

```
ISP3# sh ip route bgp
<Output omitted>
Gateway of last resort is not set

B     192.168.11.0/24 [20/0] via 209.165.201.9, 00:10:53
B     192.168.22.0/24 [20/0] via 209.165.201.13, 00:10:56
```

Changing the Weight

```
GW2(config)# router bgp 65000
GW2(config-router)# neighbor 192.168.1.1 weight 10

GW2# show ip bgp
<Output omitted>
     Network          Next Hop            Metric LocPrf Weight Path
 *>i 192.168.11.0     192.168.1.1              0    100      0 i
 *>  192.168.22.0     0.0.0.0                  0         32768 i
 * i 198.51.100.0     209.165.201.2            0    100      0 65100 65300 i
 *>                   209.165.201.6                          0 65200 65300 i
 * i 203.0.113.0      209.165.201.2            0    100      0 65100 65300 i
 *>                   209.165.201.6                          0 65200 65300 i
GW2#
```

Changing the Weight

```
GW2# clear ip bgp 192.168.1.1 in
GW2# sh ip bgp
<Output omitted>
     Network          Next Hop            Metric LocPrf Weight Path
 *>i 192.168.11.0     192.168.1.1              0    100     10 i
 *>  192.168.22.0     0.0.0.0                  0         32768 i
 *>i 198.51.100.0     209.165.201.2            0    100     10 65100 65300 i
 *                    209.165.201.6                          0 65200 65300 i
 *>i 203.0.113.0      209.165.201.2            0    100     10 65100 65300 i
 *                    209.165.201.6                          0 65200 65300 i
GW2#

GW2# traceroute 198.51.100.1 source loopback 1
Type escape sequence to abort.
Tracing the route to 198.51.100.1
VRF info: (vrf in name/id, vrf out name/id)
  1 172.16.12.1 1 msec 1 msec 0 msec
  2 209.165.201.2 1 msec 0 msec 1 msec
  3 209.165.201.10 3 msec *  5 msec
GW2#
```

Changing Local Preference

```
GW2(config)# router bgp 65000
GW2(config-router)# no neighbor 192.168.1.1 weight 10

GW1(config)# route-map prefer_isp1 permit 10
GW1(config-route-map)# set local-preference 150
GW1(config-route-map)# router bgp 65000
GW1(config-router)# neighbor 209.165.201.2 route-map prefer_isp1 in
```

Changing Local Preference

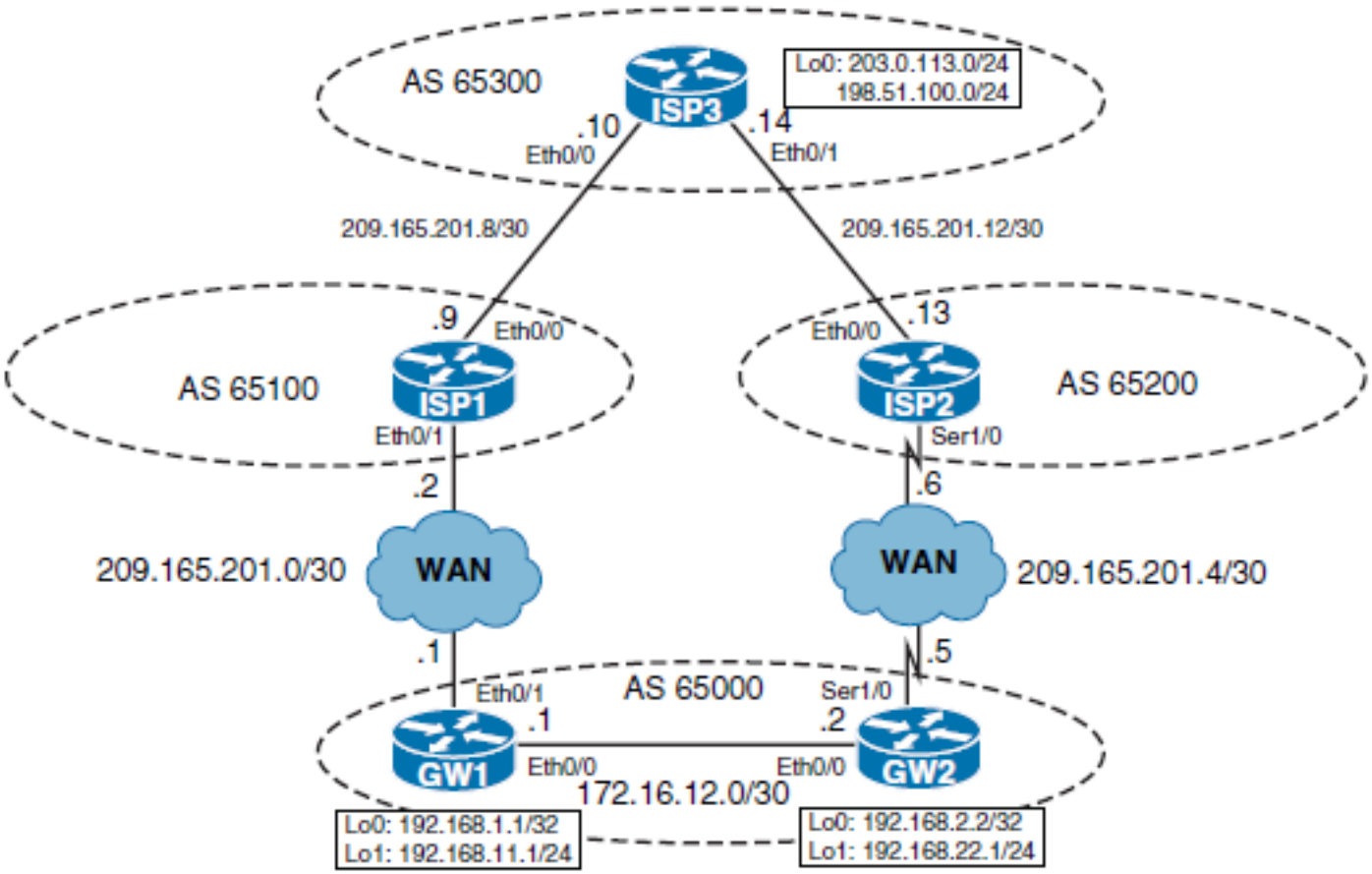
```
GW1# clear ip bgp 209.165.201.2 in

GW1# show ip bgp
<Output omitted>
     Network          Next Hop            Metric LocPrf Weight Path
 *>  192.168.11.0     0.0.0.0                  0         32768 i
 *>i 192.168.22.0     192.168.2.2              0    100      0 i
 *>  198.51.100.0     209.165.201.2                 150      0 65100 65300 i
 *>  203.0.113.0      209.165.201.2                 150      0 65100 65300 i
GW1#
```

Setting the AS-Path

- It is complicated to influence other autonomous systems to select a particular path for traffic that is returning to a specific autonomous system
- One way that an autonomous system can attempt to influence incoming traffic flow is by sending out eBGP updates with an extended AS-path attribute for undesired paths

```
GW2(config)# route-map ASPath-Prepend permit 10
GW2(config-route-map)# set as-path prepend 65000
GW2(config-route-map)# router bgp 65000
GW2(config-router)# neighbor 209.165.201.6 route-map ASPath-Prepend out

GW2# clear ip bgp 209.165.201.6 out
```

Setting the AS-Path



Setting the AS-Path

```
ISP3# show ip bgp
<Output omitted>
     Network          Next Hop            Metric LocPrf Weight Path
 *   192.168.11.0     209.165.201.13                         0 65200 65000 65000 i
 *>                   209.165.201.9                          0 65100 65000 i
 *>  192.168.22.0     209.165.201.9                          0 65100 65000 i
 *                    209.165.201.13                         0 65200 65000 65000 i
 *>  198.51.100.0     0.0.0.0                  0         32768 i
 *>  203.0.113.0      0.0.0.0                  0         32768 i
ISP3#

ISP3# trace 192.168.22.1 source loopback0
Type escape sequence to abort.
Tracing the route to 192.168.22.1
VRF info: (vrf in name/id, vrf out name/id)
  1 209.165.201.9 1 msec 0 msec 1 msec
  2 209.165.201.1 0 msec 1 msec 0 msec
  3 172.16.12.2 0 msec *  1 msec
ISP3#
```

Controlling BGP Routing Updates

Controlling BGP Routing Updates

- If there are multiple paths between your network and ISP, you may need to filter certain information during the exchange of BGP updates to influence the route selection or to enforce an administrative policy.
- BGP peer groups are used to group peers with similar policies together for a simpler, more efficient configuration.

Filtering BGP Routing Updates

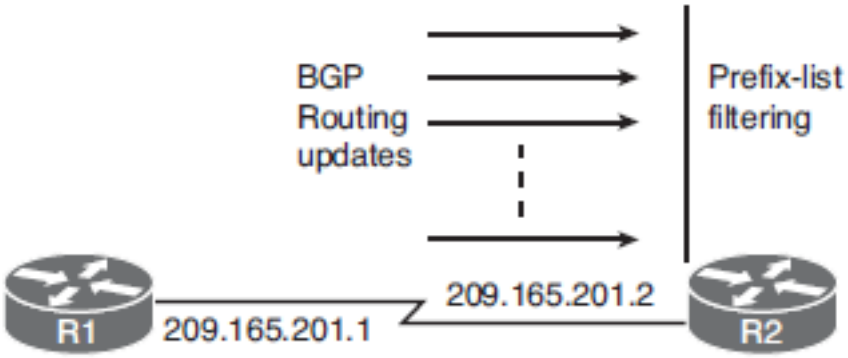
- BGP Filtering Using Prefix Lists
- BGP Filtering Using AS-Path Access Lists
- BGP Filtering Using Route Maps
- Filtering Order

BGP Filtering Using Prefix Lists

- The **neighbor** *ip-address* **prefix-list** *prefix-list-name* { **in** | **out** } router configuration command is used to apply a prefix list to routes from or to a neighbor.

Parameter	Description
ip-address	IP address of the BGP neighbor.
prefix-list-name	Name of a prefix list.
in	Prefix list is applied to incoming advertisements.
out	Prefix list is applied to outgoing advertisements.

BGP Filtering Using Prefix Lists - Example




```
router bgp 65001
  neighbor 209.165.201.1 remote-as 65002
  neighbor 209.165.201.1 prefix-list ANY-8to24-NET in
!
ip prefix-list ANY-8to24-NET permit 0.0.0.0/0 ge 8 le 24
```

BGP Filtering Using AS-Path Access Lists

Parameter	Description
access-list-number	Number from 1 to 500 that specifies the AS-path access list number.
permit | **deny**	Indicates whether this entry allows or blocks if the regular expression is true.
regexp	Regular expression that defines the AS-path filter. The autonomous system number is expressed in the range from 1 to 65535.

- The autonomous system path access list is defined by the **`ip as-path access-list`** *`accesslist-number`* **`{ permit | deny }`** *regexp* global configuration command

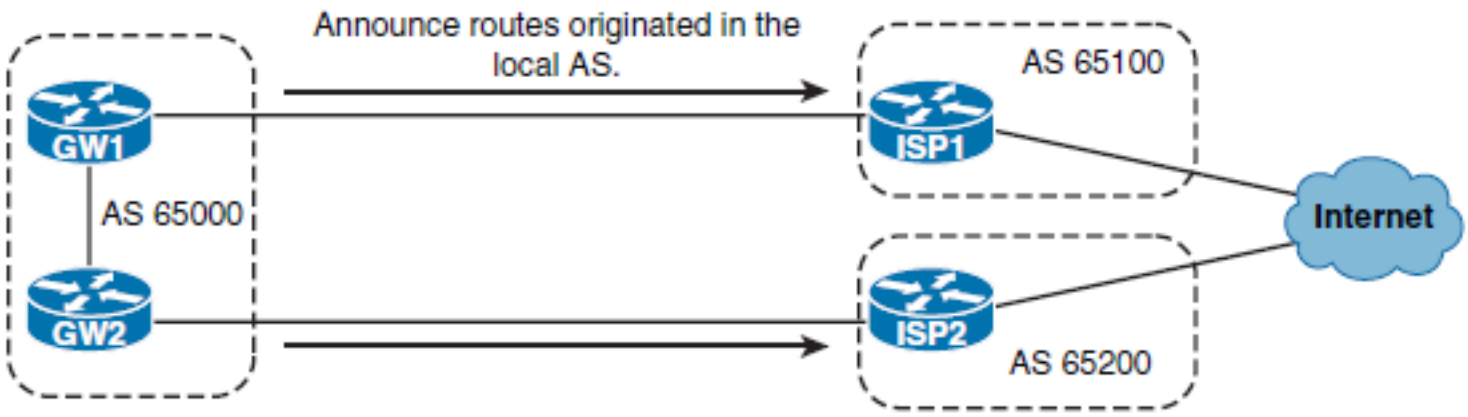
Parameter	Description
.	Matches any single character
*	Matchers 0 or more sequences of a pattern
^	Matches the beginning of a string
$	Matches the end of the string
_ (underscore)	Matches a comma, left brace, right brace, left parenthesis, right parenthesis, the beginning of a string, the end of a string, or a space

BGP Filtering Using AS-Path Access Lists

- The **neighbor** *ip-address* **filter-list** *access-list-number* { **in** | **out** } router configuration command is used to apply an AS-path access list to routes from or to a neighbor.

Parameter	Description
ip-address	IP address of the BGP neighbor.
access-list-number	Number of an AS-path access list.
in	Access list is applied to incoming routes.
out	Access list is applied to outgoing routes.

BGP Filtering Using AS-Path Access Lists - Example




```
GW1(config)# ip as-path access-list 1 permit ^$
GW1(config)# router bgp 65000
GW1(config-router)# neighbor 209.165.201.1 filter-list 1 out
```

```
GW2(config)# ip as-path access-list 1 permit ^$
GW2(config)# router bgp 65000
GW2(config-router)# neighbor 209.165.201.5 filter-list 1 out
```

BGP Filtering Using Route Maps

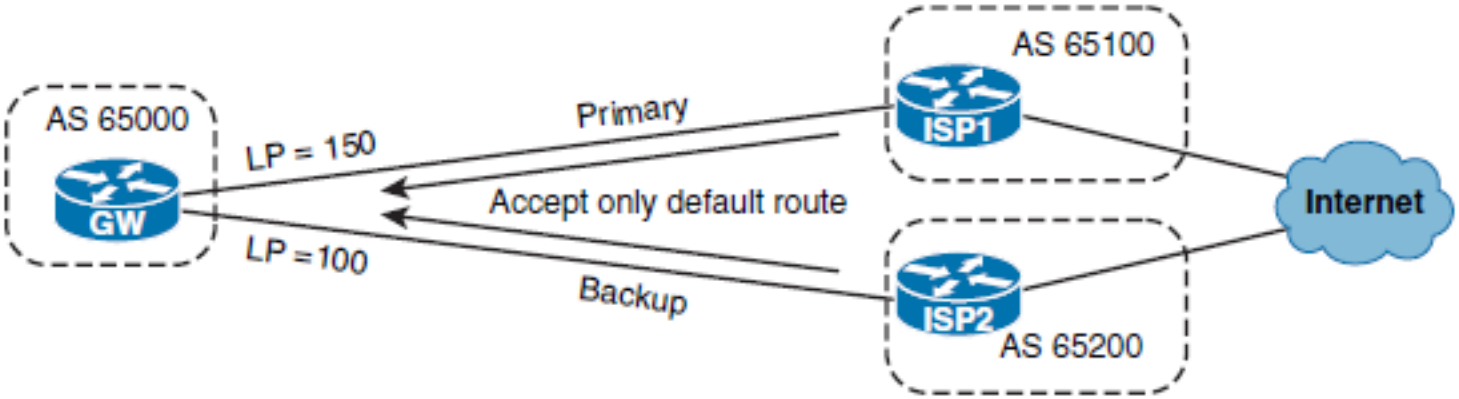
A route map can match and set several different BGP attributes, including the following:

- Origin
- Next hop
- Community
- Local preference
- MED

Route maps can match based on other items, including the following:

- Network number and subnet mask (with an IP prefix list)
- Route originator
- Tag an IGP route
- AS-path
- Route type (internal or external)

BGP Filtering Using Route Maps




```
router bgp 65000
  neighbor 209.165.201.1 remote-as 65100
  neighbor 209.165.201.1 route-map FILTER in
  neighbor 209.165.201.5 remote-as 65200
  neighbor 209.165.201.5 route-map FILTER in
!
route-map FILTER permit 10
 match ip address prefix-list default-only
 match as-path 10
 set local-preference 150
!
route-map FILTER permit 20
 match ip address prefix-list default-only
!
ip as-path access-list 10 permit ^65100$
ip prefix-list default-only permit 0.0.0.0/0
```

BGP Peer Groups

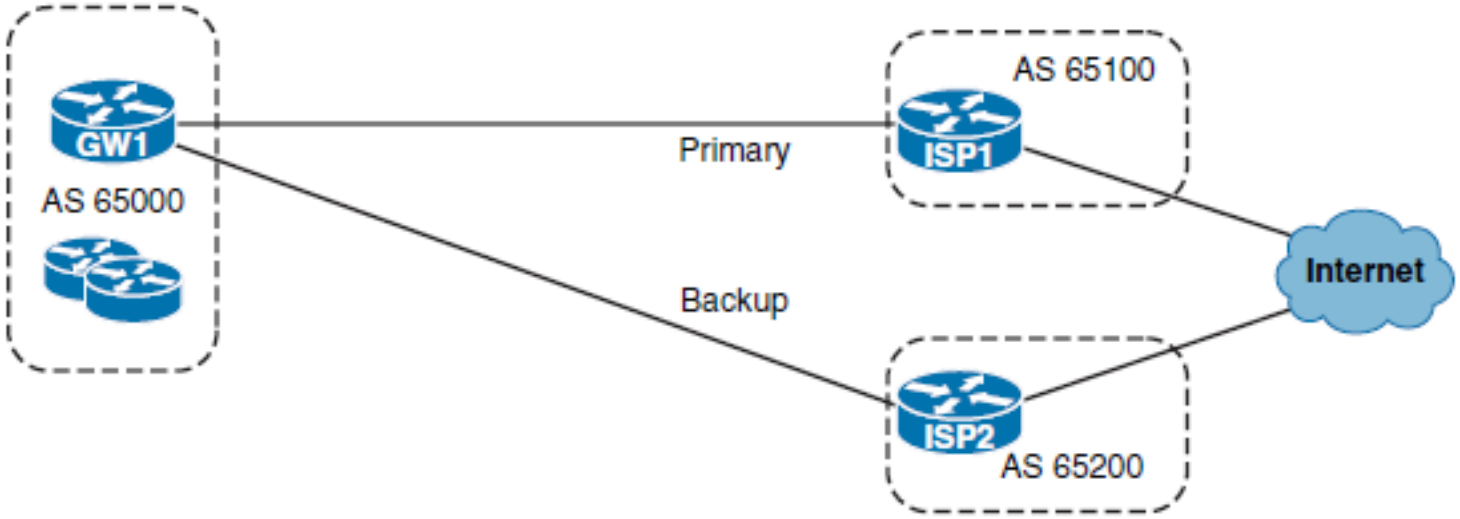
- In BGP, many neighbors are often configured with the same update policies.
- On a Cisco IOS router, neighbors with the same update policies can be grouped into peer groups to simplify configuration and, more important, to make updating more efficient and improve performance.
- A peer group's configuration can have many BGP features, including the following:
  - update-source
  - next-hop-self
  - ebgp-multihop
  - Authentication of the BGP sessions
  - Change the weight of routes received
  - Filter incoming or outgoing routes

Peer Group Configuration

- The **neighbor** *peer-group-name* **peer-group** router configuration command is used to create a BGP peer group.
- The **neighbor** *ip-address* **peer-group** *peer-group-name* router configuration command, is used to assign neighbors as part of the group after the group has been created.

Parameter	Description
ip-address	The IP address of the neighbor that is to be assigned as a member of the peer group
peer-group-name	The name of the BGP peer group

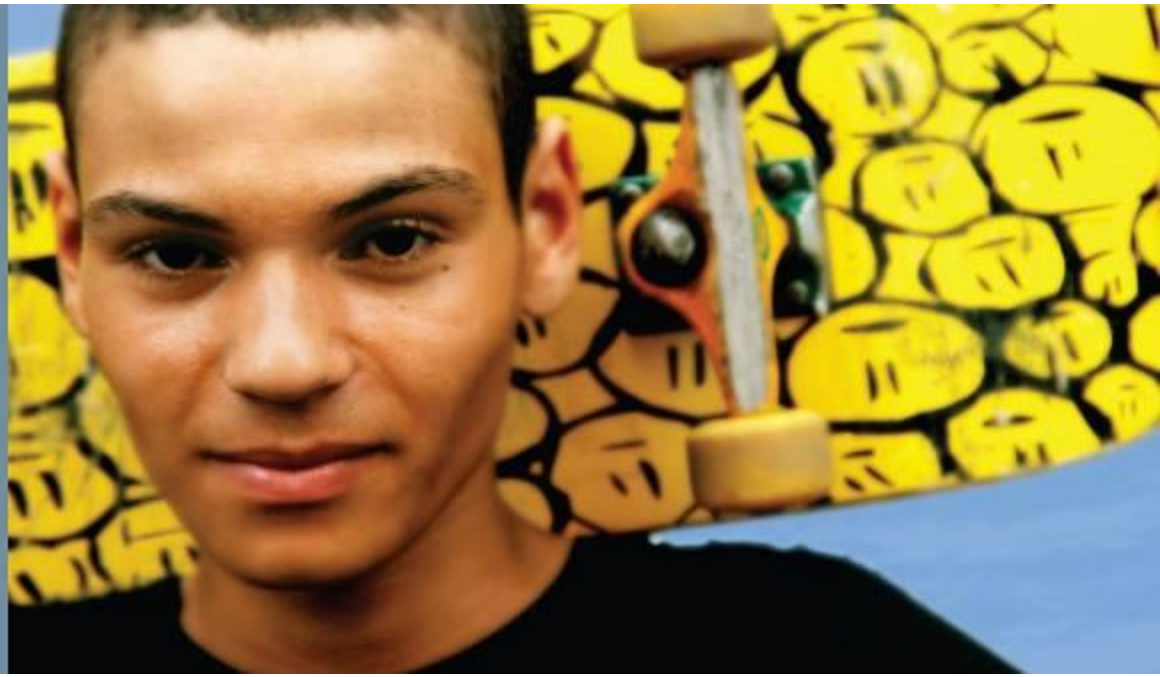
Peer Group Configuration Example



Peer Group Configuration Example

```
router bgp 65000
 neighbor ISP peer-group
 neighbor ISP filter-list 10 out
 neighbor ISP prefix-list desired-subnets in
 neighbor ISP route-map FILTER in
!
neighbor 209.165.201.1 remote-as 65100
neighbor 209.165.201.1 peer-group ISP
neighbor 209.165.201.5 remote-as 65200
neighbor 209.165.201.5 peer-group ISP
!
route-map FILTER permit 10
 match as-path 20
 set local-preference 150
!
route-map FILTER permit 20
!
ip as-path access-list 10 permit ^$
ip as-path access-list 20 permit ^65100_
!
ip prefix-list desired-subnets permit 0.0.0.0/0
ip prefix-list desired-subnets permit 0.0.0.0/0 ge8 le 24
```

Implementing BGP for IPv6 Internet Connectivity



Implementing BGP for IPv6 Internet Connectivity

This section covers the following topics:

- MP-BGP support for IPv6
- Exchanging IPv6 routes over an IPv4 session
- Exchanging IPv6 routes over an IPv6 session
- BGP for IPv6 configuration and verification
- Comparing IPv4 to Dual (IPv4/IPv6) BGP transport
- BGP filtering mechanisms for IPv6

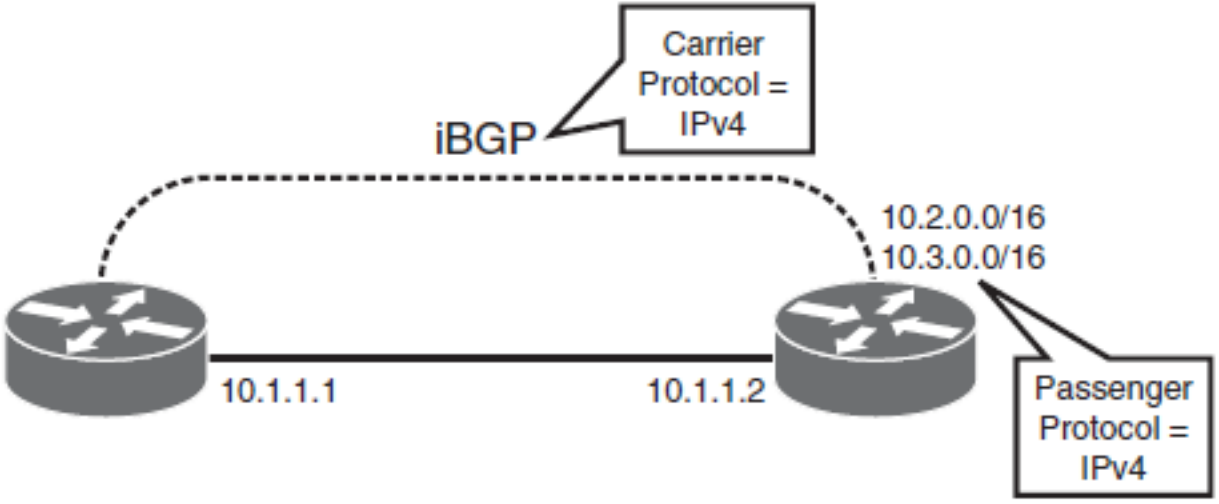
MP-BGP Support for IPv6

Multiprotocol extensions are defined as new attributes. IPv6-specific extensions incorporated into MBGP include the following:

- A new identifier for the IPv6 address family.
- Scoped addresses. The next-hop attribute contains a global IPv6 address or a link-local address.
- The next-hop attribute and NLRI are expressed as IPv6 addresses and prefixes

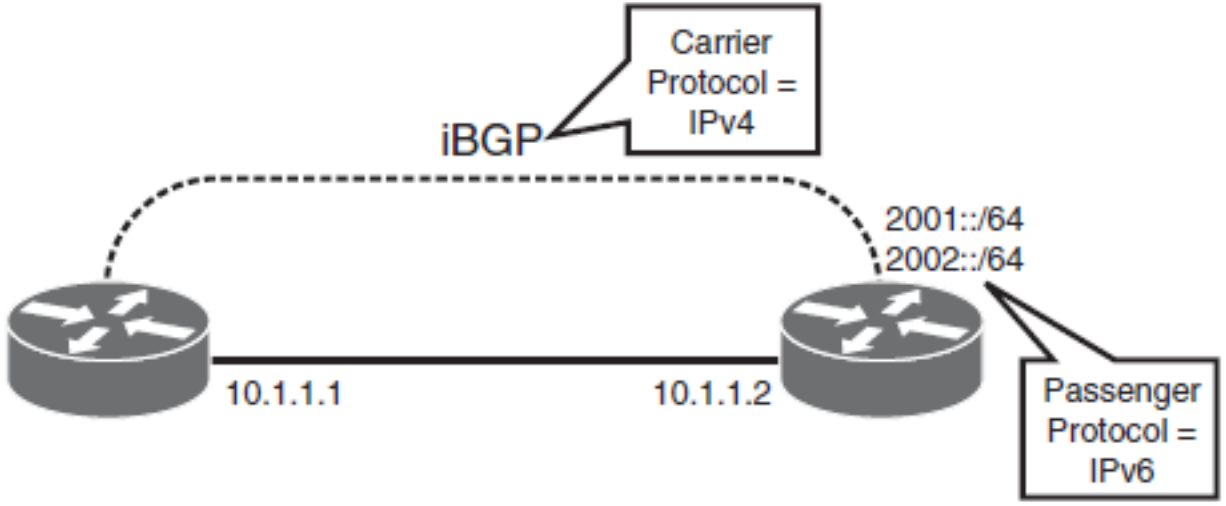
MP-BGP

- MP-BGP can, of course, operate with multiple protocols. It operates by identifying two separate protocols: the carrier protocol and the passenger protocol.
- In an all-IPv4 environment, BGP establishes sessions using IPv4 (using TCP port 179); IPv4 is the carrier protocol.
- The routes that BGP advertises, which is the passenger protocol, are also IPv4.



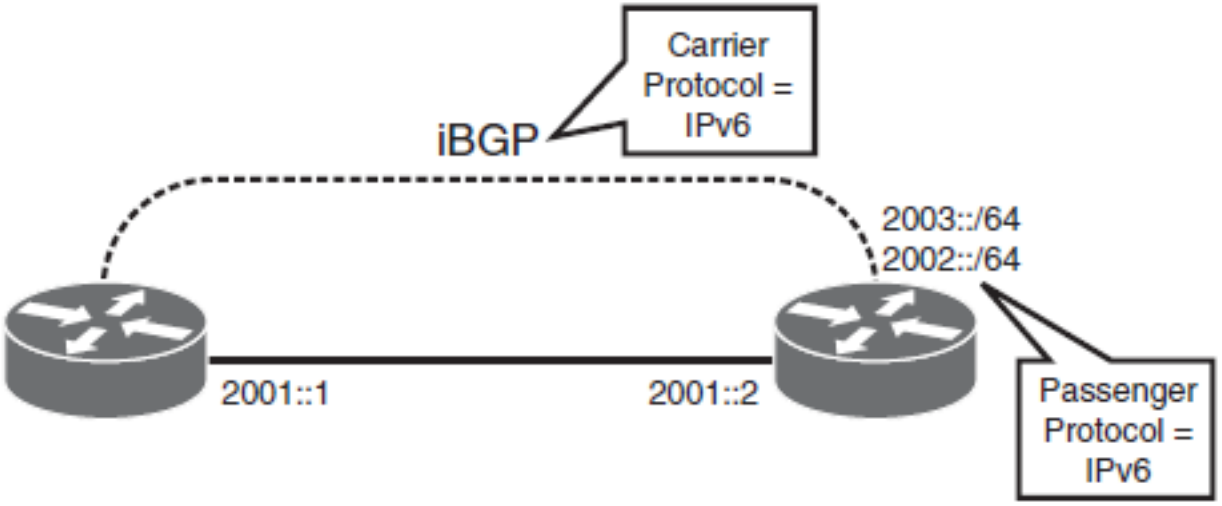
MP-BGP

- Protocols other than IPv4, including IPv6, also need to advertise reachability information.
- MP-BGP extensions allow these other protocols to be carried using BGP.



MP-BGP

- In an all-IPv6 environment, BGP can be used as both the carrier and passenger protocol.
- In this case, IPv6 is used to establish BGP sessions, and BGP advertises IPv6 prefixes.



Exchanging IPv6 Routes over an IPv4 Session

- Existing IPv4 TCP sessions can carry IPv6 routing information when adding IPv6 support to a network.
- An existing neighbor can be activated for the IPv6 address family and IPv6 routing information will be sent over the same neighbor session.
- MP-BGP allows the use of many address families to define the type of addresses being carried.

Exchanging IPv6 Routes over an IPv4 Session

- The **`address-family {ipv4 | ipv6} [ unicast | multicast ]`** router configuration command enters address family configuration mode for configuring BGP routing sessions.

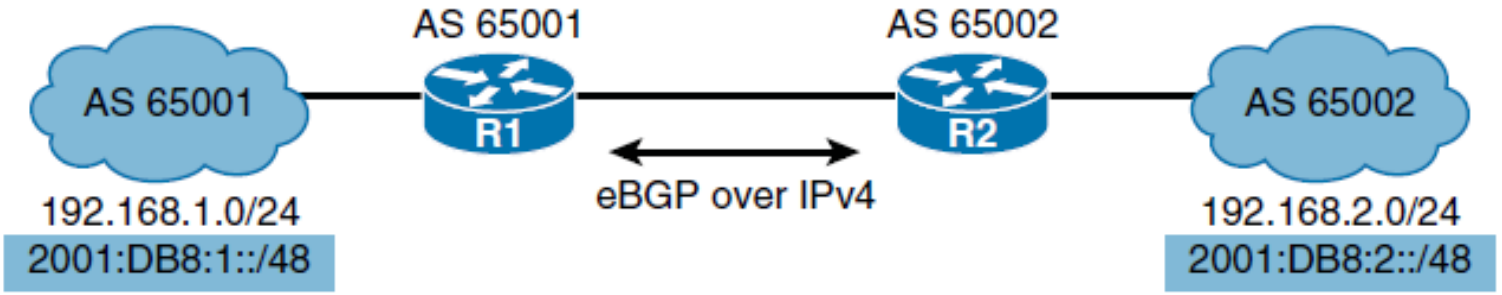
Parameter	Description
ipv4	Used for a routing session using IPv4 address prefixes.
ipv6	Used for a routing session using IPv6 address prefixes.
unicast	(Optional) Specifies unicast address prefixes. This is the default for both ipv4 and ipv6 keywords.
multicast	(Optional) Specifies multicast address prefixes.

Exchanging IPv6 Routes over an IPv4 Session

- In an IPv6 address family, a neighbor needs to be activated using the **`neighbor`** { *`IPv4 address`* | *`IPv6 address`* } **`activate`** address-family configuration command.
- The exchange of addresses with BGP neighbors is enabled for the IPv4 address family by default.
- The **`network`** *`ipv6-address/prefix-length`* command, this time in address family configuration mode, is used to specify the networks to be advertised.
- This command injects a prefix into the BGP database only for the specified address family

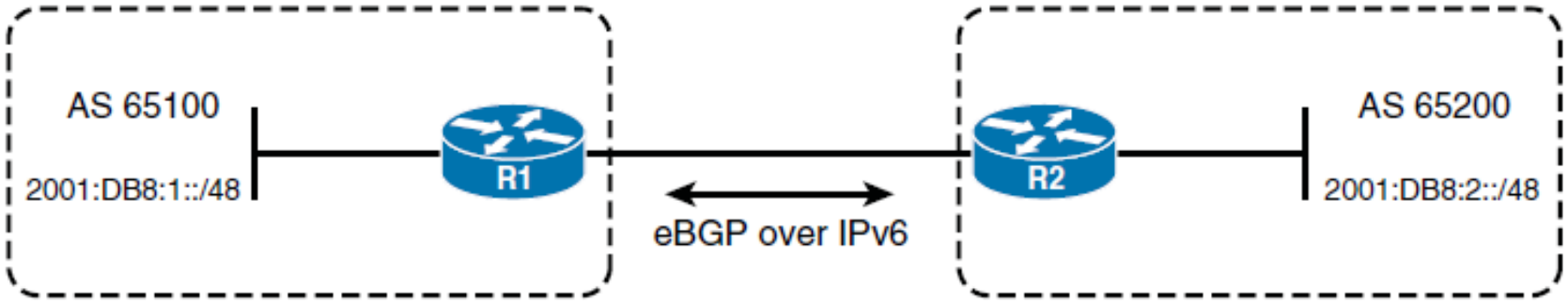
Parameter	Description
ipv6-address	The IPv6 address to be used.
prefix-length	The length of the IPv6 prefix. A decimal value that indicates how many of the high-order contiguous bits of the address comprise the prefix (the network portion of the address). A slash mark must precede the decimal value.

Exchanging IPv6 Routes over an IPv4 Session



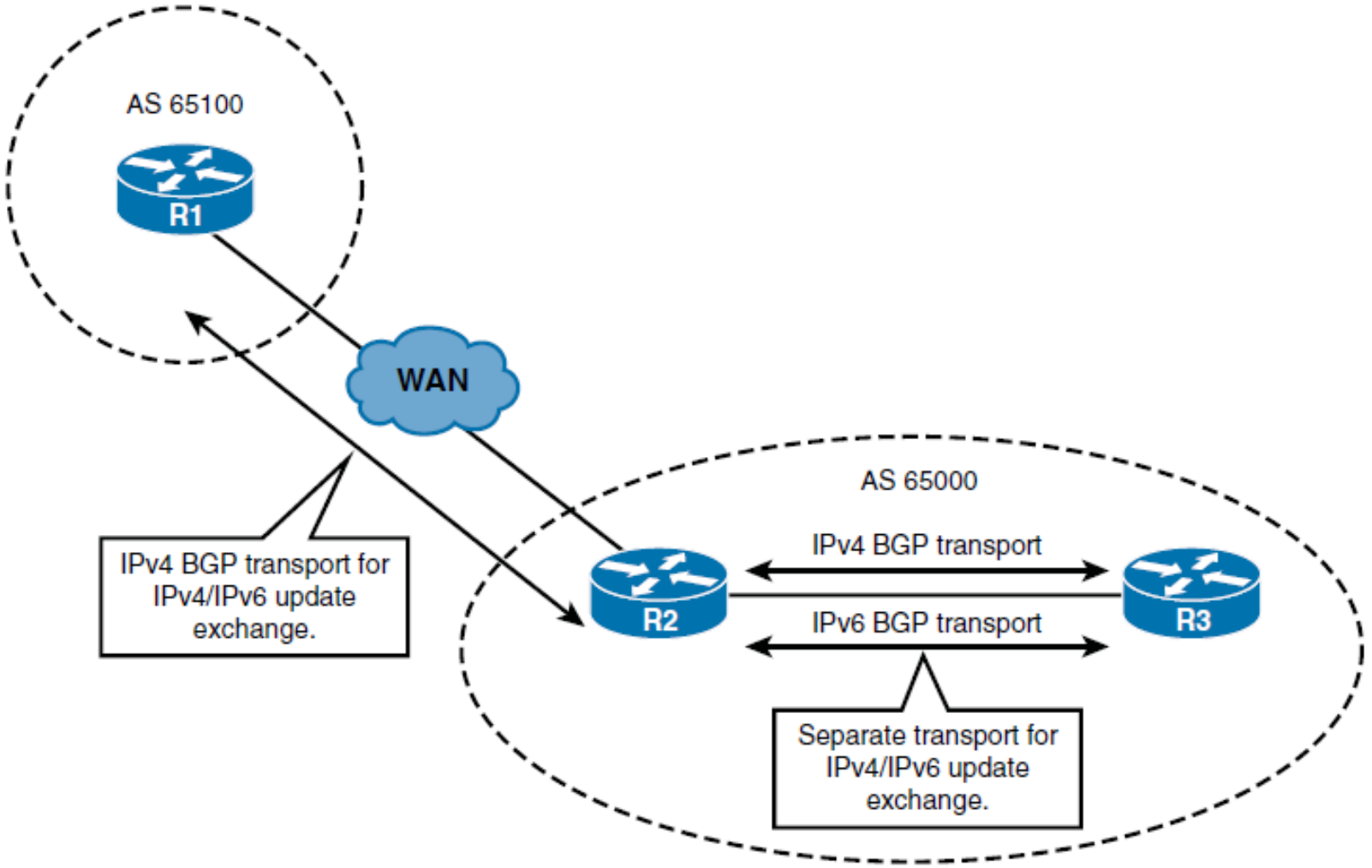

```
router bgp 65001
 neighbor 192.168.2.2 remote-as 65002
!
 address-family ipv4 unicast
  network 192.168.1.0 mask 255.255.255.0
 address-family ipv6 unicast
  neighbor 192.168.2.2 activate
  network 2001:db8:1::/48
```

Exchanging IPv6 Routes over an IPv6 Session

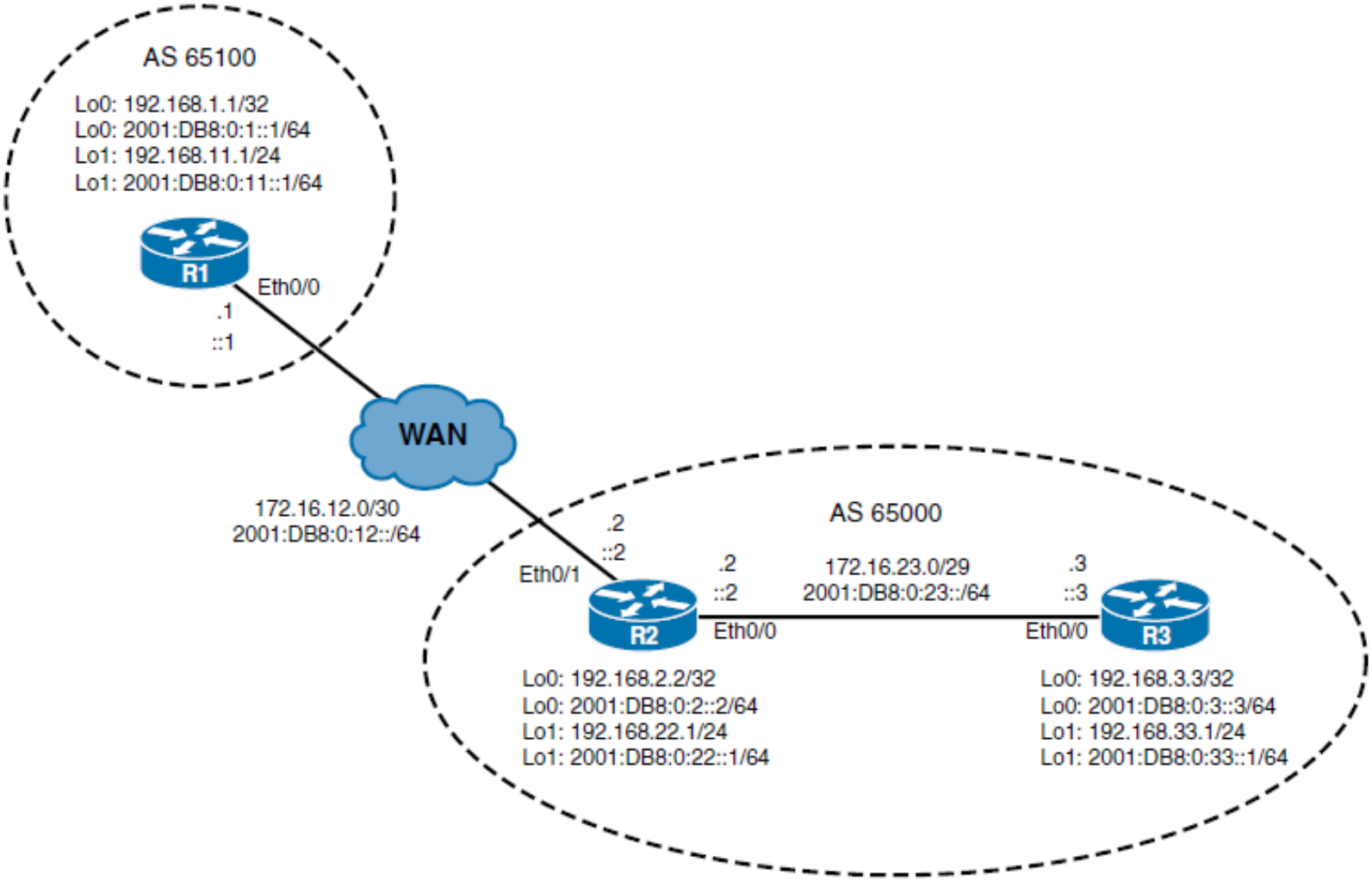



```
router bgp 65100
 bgp router-id 1.1.1.1
 neighbor 2001:db8:2::2 remote-as 65200
!
address-family ipv6 unicast
  neighbor 2001:db8:2::2 activate
  network 2001:db8:1::/48
```

BGP for IPv6 Configuration and Verification



BGP for IPv6 Configuration and Verification



BGP for IPv6 Configuration and Verification

```
R1# show running-config | section router bgp
router bgp 65100
 bgp log-neighbor-changes
 neighbor 172.16.12.2 remote-as 65000
 !
 address-family ipv4
  network 192.168.11.0
  neighbor 172.16.12.2 activate
 exit-address-family
 !
 address-family ipv6
  network 2001:DB8:0:11::/64
  neighbor 172.16.12.2 activate
  neighbor 172.16.12.2 route-map nh out
 exit-address-family
```

BGP for IPv6 Configuration and Verification

```
R1# show bgp ipv4 unicast
<Output omitted>
     Network          Next Hop            Metric LocPrf Weight Path
*>   192.168.11.0     0.0.0.0                  0         32768 i
*>   192.168.22.0     172.16.12.2              0             0 65000 i
*>   192.168.33.0     172.16.12.2                            0 65000 i

R1# show bgp ipv6 unicast
BGP table version is 2, local router ID is 192.168.11.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
 *>  2001:DB8:0:11::/64
                      ::                       0         32768 i
```

BGP for IPv6 Configuration and Verification

```
R1# show bgp ipv4 unicast summary
BGP router identifier 192.168.11.1, local AS number 65100
BGP table version is 8, main routing table version 8
3 network entries using 444 bytes of memory
3 path entries using 192 bytes of memory
3/3 BGP path/bestpath attribute entries using 408 bytes of memory
1 BGP AS-PATH entries using 24 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 1068 total bytes of memory
BGP activity 4/0 prefixes, 6/2 paths, scan interval 60 secs

Neighbor        V     AS MsgRcvd MsgSent   TblVer  InQ  OutQ  Up/Down State/PfxRcd
172.16.12.2     4  65000      72      71        8    0     0  01:01:03        2
```

BGP for IPv6 Configuration and Verification

```
R1# show bgp ipv6 unicast summary
BGP router identifier 192.168.11.1, local AS number 65100
BGP table version is 2, main routing table version 2
1 network entries using 172 bytes of memory
1 path entries using 88 bytes of memory
1/1 BGP path/bestpath attribute entries using 136 bytes of memory
1 BGP AS-PATH entries using 24 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 420 total bytes of memory
BGP activity 4/0 prefixes, 6/2 paths, scan interval 60 secs

Neighbor        V     AS MsgRcvd MsgSent   TblVer  InQ  OutQ  Up/Down  State/PfxRcd
172.16.12.2     4  65000       0       0        1    0     0  never       (NoNeg)
```

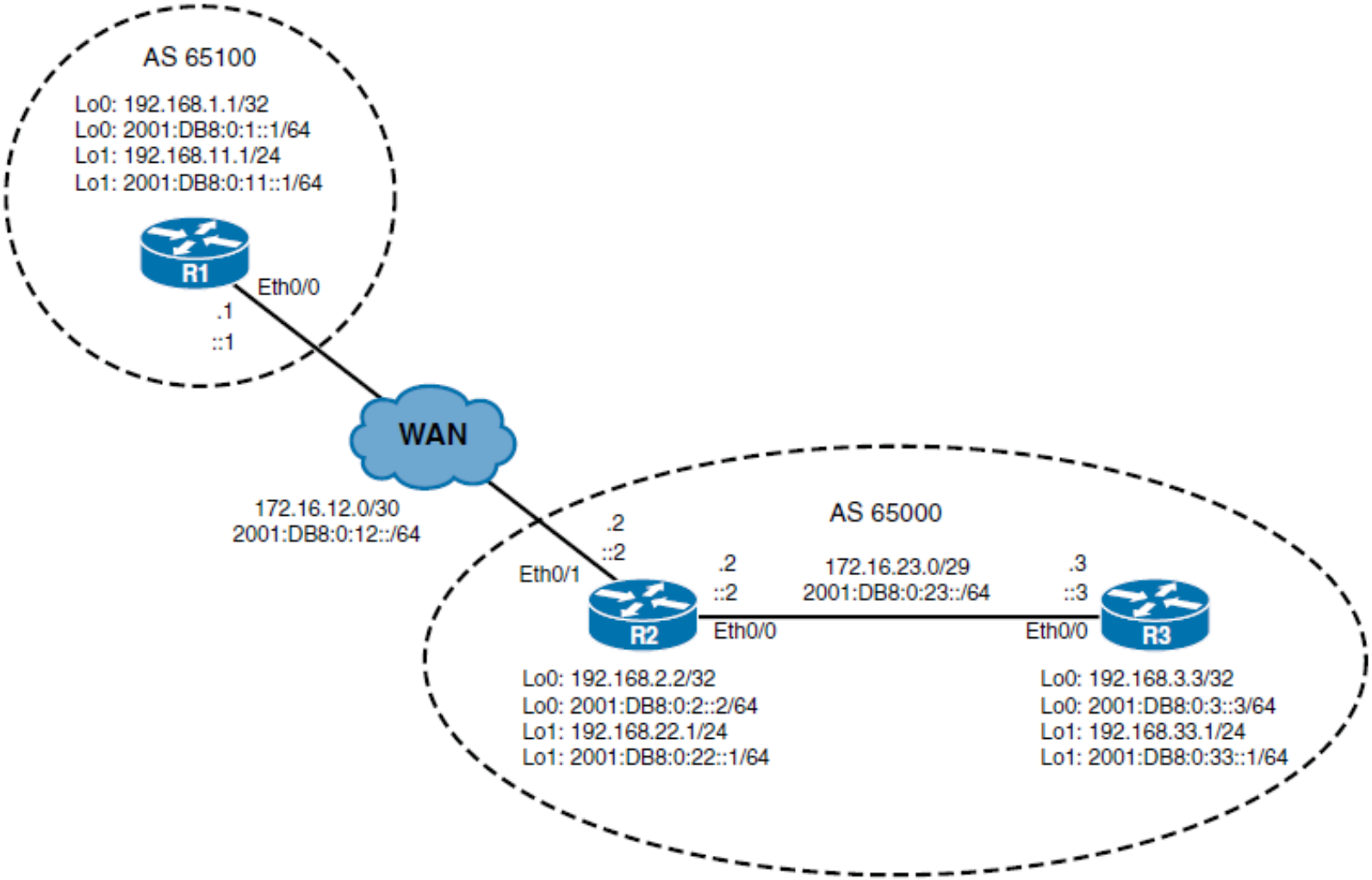
BGP for IPv6 Configuration and Verification

```
R2# show running-config | section router bgp
router bgp 65000
 bgp log-neighbor-changes
 network 192.168.22.0
 neighbor 172.16.12.1 remote-as 65100
 neighbor 192.168.3.3 remote-as 65000
 neighbor 192.168.3.3 update-source Loopback0
 neighbor 192.168.3.3 next-hop-self

R2# show ip bgp
<Output omitted>
     Network          Next Hop            Metric LocPrf Weight Path
 *>  192.168.11.0     172.16.12.1              0             0 65100 i
 *>  192.168.22.0     0.0.0.0                  0         32768 i
 *>i 192.168.33.0     192.168.3.3              0    100      0 i
```

BGP for IPv6 Configuration and Verification

```
R3# show running-config | section router bgp
router bgp 65000
 bgp log-neighbor-changes
 network 192.168.33.0
 neighbor 192.168.2.2 remote-as 65000
 neighbor 192.168.2.2 update-source Loopback0

R3# show ip bgp
<Output omitted>
     Network          Next Hop            Metric LocPrf Weight Path
 *>i 192.168.11.0     192.168.2.2              0    100      0 65100 i
 *>i 192.168.22.0     192.168.2.2              0    100      0 i
 *>  192.168.33.0     0.0.0.0                  0         32768 i

R3# ping 192.168.11.1 source loopback 1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.11.1, timeout is 2 seconds:
Packet sent with a source address of 192.168.33.1
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
```

Enable eBGP IPv6 Route Exchange



Enable eBGP IPv6 Route Exchange

```
R2(config)# router bgp 65000
R2(config-router)# address-family ipv6 unicast
R2(config-router-af)# neighbor 172.16.12.1 activate
R2(config-router-af)# network 2001:DB8:0:22::/64

R2# show running-config | section router bgp
router bgp 65000
 bgp log-neighbor-changes
 neighbor 172.16.12.1 remote-as 65100
 neighbor 192.168.3.3 remote-as 65000
 neighbor 192.168.3.3 update-source Loopback0
 !
 address-family ipv4
  network 192.168.22.0
  neighbor 172.16.12.1 activate
  neighbor 192.168.3.3 activate
  neighbor 192.168.3.3 next-hop-self
 exit-address-family
 !
 address-family ipv6
  network 2001:DB8:0:22::/64
  neighbor 172.16.12.1 activate
 exit-address-family
```

Enable eBGP IPv6 Route Exchange

```
R1# show bgp ipv6 unicast summary
BGP router identifier 192.168.11.1, local AS number 65100
BGP table version is 4, main routing table version 4
2 network entries using 344 bytes of memory
2 path entries using 176 bytes of memory
2/1 BGP path/bestpath attribute entries using 272 bytes of memory
1 BGP AS-PATH entries using 24 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 816 total bytes of memory
BGP activity 23/18 prefixes, 46/41 paths, scan interval 60 secs

Neighbor        V    AS    MsgRcvd  MsgSent  TblVer  InQ  OutQ  Up/Down   State/PfxRcd
172.16.12.2     4  65000        41       41       4    0     0  00:31:14            1
```

Enable eBGP IPv6 Route Exchange

```
R1# show bgp ipv6 unicast
BGP table version is 4, local router ID is 192.168.11.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
 *>  2001:DB8:0:11::/64
                       ::                       0         32768 i
 *   2001:DB8:0:22::/64
                       ::FFFF:172.16.12.2
                                                0             0 65000 i
```

Enable eBGP IPv6 Route Exchange

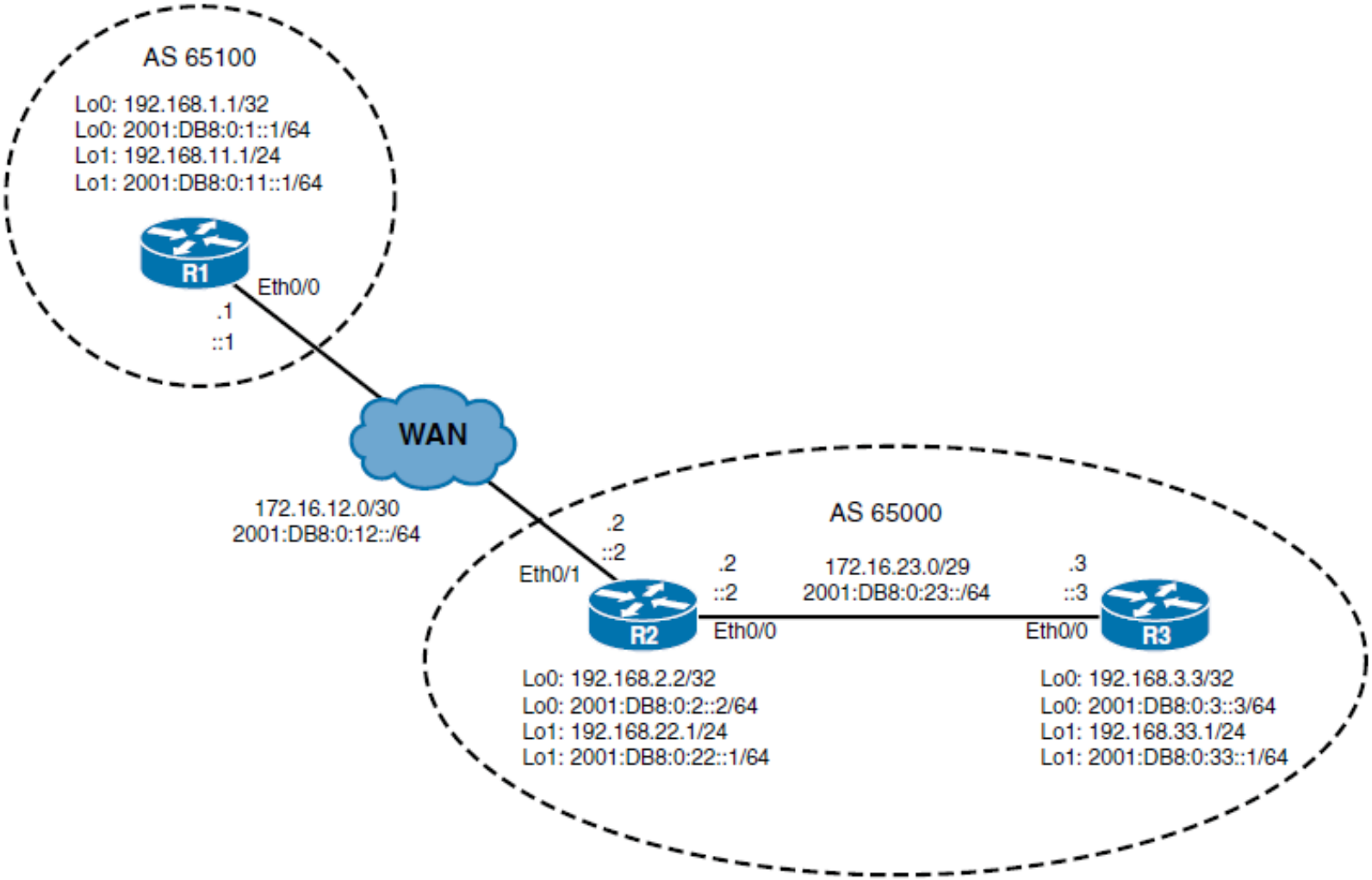
- The eBGP IPv6 update received on R1 is not marked as best (using the > sign) in the BGP table, because the next-hop address is not reachable.
- Also the next-hop address, ::FFFF:172.16.12.2, is an IPv6 address derived from the IPv4 next-hop address.
- This neighbor relationship is an IPv4 neighbor relationship, carrying IPv6 routes.
- Because an IPv6 route must have an IPv6 next hop, BGP dynamically created this IPv6 next-hop address from the actual IPv4 next-hop address.
- However, this is not a reachable IPv6 address; therefore, the route is not marked as best in the BGP table, and it does not appear in the IPv6 routing table.

Enable eBGP IPv6 Route Exchange

```
R2(config)# route-map NH-R1
R2(config-route-map)# set ipv6 next-hop 2001:DB8:0:12::2
R2(config-route-map)# router bgp 65000
R2(config-router)# address-family ipv6 unicast
R2(config-router-af)# neighbor 172.16.12.1 route-map NH-R1 out
```

```
R1# show bgp ipv6 unicast
<Output omitted>

     Network          Next Hop            Metric LocPrf Weight Path
 *>  2001:DB8:0:11::/64
                      ::                       0         32768 i
 *>  2001:DB8:0:22::/64
                      2001:DB8:0:12::2
                                               0             0 65000 i
```

Enable iBGP IPv6 Route Exchange



Enable iBGP IPv6 Route Exchange

```
R2(config)# router bgp 65000
R2(config-router)# neighbor 2001:DB8:0:3::3 remote-as 65000
R2(config-router)# neighbor 2001:DB8:0:3::3 update-source Loopback 0
R2(config-router)# address-family ipv6 unicast
R2(config-router-af)# neighbor 2001:DB8:0:3::3 activate
R2(config-router-af)# neighbor 2001:DB8:0:3::3 next-hop-self
```

```
R3(config-router-af)# router bgp 65000
R3(config-router)# neighbor 2001:DB8:0:2::2 remote-as 65000
R3(config-router)# neighbor 2001:DB8:0:2::2 update-source Loopback 0
R3(config-router)# address-family ipv6 unicast
R3(config-router-af)# neighbor 2001:DB8:0:2::2 activate
R3(config-router-af)# neighbor 2001:DB8:0:2::2 next-hop-self
R3(config-router-af)# network 2001:DB8:0:33::/64
```

Enable iBGP IPv6 Route Exchange

```
R3# show bgp ipv6 unicast
<Output omitted>
     Network          Next Hop            Metric LocPrf Weight Path
 *>i 2001:DB8:0:11::/64
                      2001:DB8:0:2::2          0    100      0 65100 i
 *>i 2001:DB8:0:22::/64
                      2001:DB8:0:2::2          0    100      0 i
 *>  2001:DB8:0:33::/64
                      ::                       0         32768 i
R3# show ipv6 route bgp
<Output omitted>
B   2001:DB8:0:11::/64 [200/0]
     via 2001:DB8:0:2::2
B   2001:DB8:0:22::/64 [200/0]
     via 2001:DB8:0:2::2
```

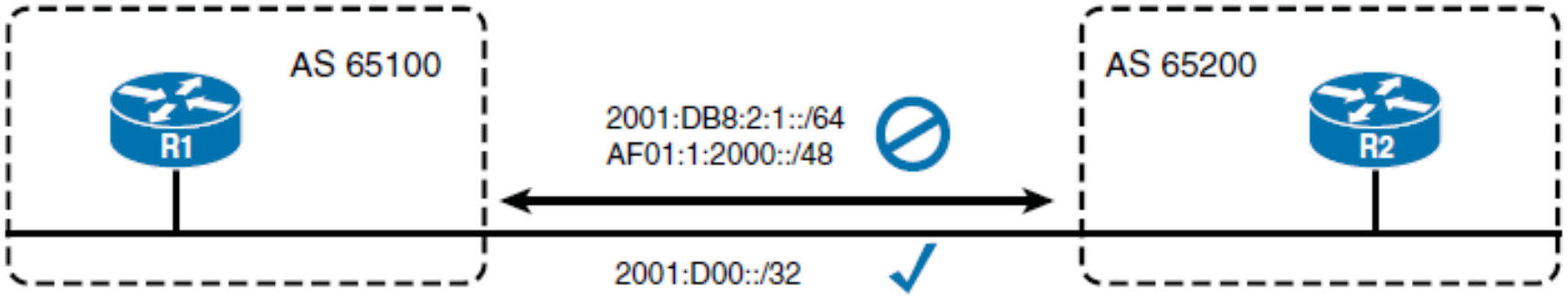
Comparing IPv4 to Dual (IPv4/IPv6) BGP Transport

- As you have seen, both IPv4 and IPv6 address families can use a single IPv4 neighbor or two separate sessions can be established, one for each address family.
- There are advantages to both approaches.
- Using a single IPv4 neighbor reduces the number of neighbor sessions. In an environment where a lot of neighbors are configured, this can significantly reduce the size and complexity of configuration.
- However, running IPv6 over an IPv4 session requires modification of the next-hop attribute.
- In contrast, when using two separate sessions for IPv4 and IPv6, there is no need to implement route maps to overwrite the next-hop parameter.
- Exchange of IPv4 and IPv6 routes is completely independent; neighbor configuration and handling is duplicated.
- Note that IPv6 neighbors are not seen in the **show ip bgp summary** command output; use the **show bgp ipv6 unicast summary** command instead.

BGP Filtering Mechanisms for IPv6

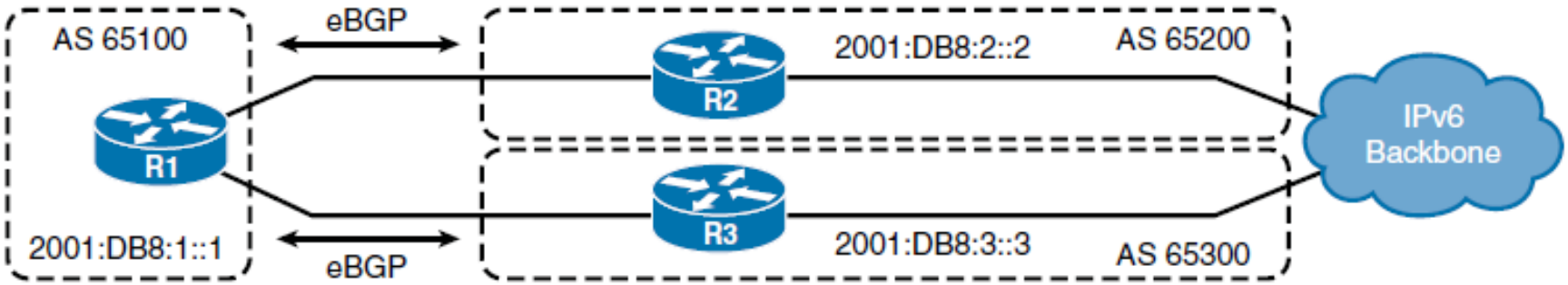
- IPv6 Prefix List Filtering
- IPv6 Path Selection with BGP Local Preference

IPv6 Prefix List Filtering




```
ipv6 prefix-list large_networks seq 5 permit 2000::/3 le 48
!
router bgp 65100
 bgp router-id 1.1.1.1
 neighbor 2001:DB8:2::2 remote-as 65200
 address-family ipv6
  neighbor 2001:DB8:2::2 activate
  neighbor 2001:DB8:2::2 prefix-list large_networks in
  neighbor 2001:DB8:2::2 prefix-list large_networks out
  network 2001:D00::/24
```

IPv6 Path Selection with BGP Local Preference




```
router bgp 65100
 bgp router-id 1.1.1.1
 neighbor 2001:DB8:2::2 remote-as 65200
 neighbor 2001:DB8:3::3 remote-as 65300
 address-family ipv6
  neighbor 2001:DB8:2::2 activate
  neighbor 2001:DB8:3::3 activate
  neighbor 2001:DB8:2::2 route-map LP200 in
  neighbor 2001:DB8:3::3 route-map LP50 in
  network 2001:D00::/24
!
route-map LP50 permit 10
 set local-preference 50
!
route-map LP200 permit 10
 set local-preference 200
```

Appendix C

BGP Supplement

- BGP Route Summarization
- Communities
- Route Reflectors
- Advertising a Default Route
- Not Advertising Private Autonomous System Numbers

BGP Route Summarization

Two BGP attributes are related to aggregate addressing:

- **Atomic aggregate**
  - A well-known discretionary attribute that informs the neighbor autonomous system that the originating router has aggregated the routes
- **Aggregator**
  - An optional transitive attribute that specifies the BGP router ID and autonomous system number of the router that performed the route aggregation

BGP Route Summarization

- By default, the aggregate route is advertised as coming from the autonomous system that did the aggregation and has the atomic aggregate attribute set to show that information might be missing.
- The autonomous system numbers from the nonaggregated routes are not listed.
- You can configure the router to include the unordered list of all autonomous systems contained in all paths that are being summarized.

Network Boundary Summarization

- BGP works differently than the other protocols, the **`network`** *`network-number`* [**`mask`** *`network-mask`*] router configuration command for BGP permits BGP to advertise a network if it is present in the IP routing table.
- This command allows classless prefixes. The router can advertise individual subnets, networks, or supernets. The default mask is the classful mask and results in only the classful network number being announced.
- Note that at least one subnet of the specified major network must be present in the IP routing table for BGP to start announcing the classful network. However, if you specify the **`mask`** *`network-mask`* , an exact match to the network (both address and mask) must exist in the routing table for the network to be advertised.

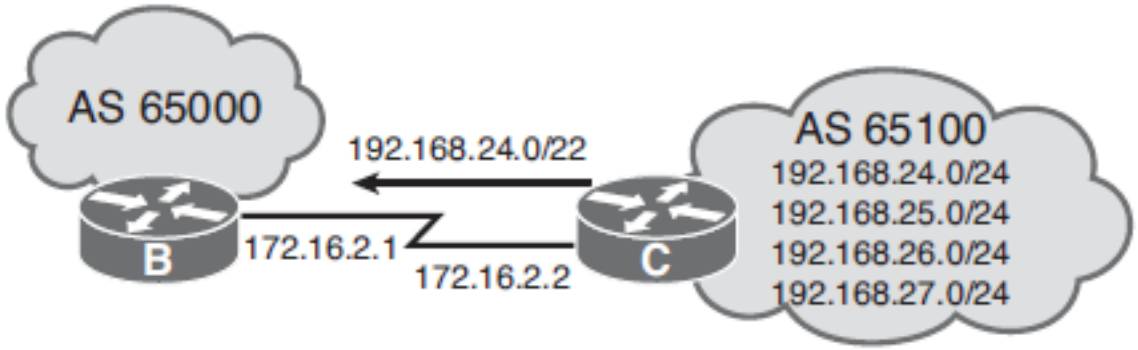
Network Boundary Summarization

- The BGP **auto-summary** command determines how BGP handles redistributed routes.
- The **no auto-summary** router configuration command turns off BGP autosummarization.
- When summarization is enabled (with **auto-summary**), all redistributed subnets are summarized to their classful boundaries in the BGP table.
- When summarization is disabled (with **no auto-summary**), all redistributed subnets are present in their original form in the BGP table.

BGP Route Summarization Using the network Command

- To advertise a simple classful network number, use the **network** *network-number* router configuration command without the **mask** option.
- To advertise an aggregate of prefixes that originate in this autonomous system, use the **network** *network-number* [**mask** *network-mask*] router configuration command with the **mask** option.
- Remember that the prefix must exactly match [both address and mask] an entry in the IP routing table for the network to be advertised

Cautions When Using the network Command for Summarization




```
router bgp 65100
 network 192.168.24.0
 network 192.168.25.0
 network 192.168.26.0
 network 192.168.27.0
 network 192.168.24.0 mask 255.255.252.0
 neighbor 172.16.2.1 remote-as 65000
```

Cautions When Using the network Command for Summarization

- Each of the four Class C networks is announced because each already exists in the routing table. These networks are summarized with the **`network 192.168.24.0 mask 255.255.252.0`** command on Router C.
- However, with this command the 192.168.24.0/22 route is *not* announced by default because that route is not in the routing table

- Correct way for summarization with the network command:

```
router bgp 65100
  network 192.168.24.0 mask 255.255.252.0
  neighbor 172.16.2.1 remote-as 65000
ip route 192.168.24.0 255.255.252.0 null 0
```

Creating a Summary Address in the BGP Table Using the `aggregate-address` Command

- The **`aggregate-address`** *`ip-address mask`* **`[ summary-only ] [ as-set ]`** router configuration command is used to create an aggregate, or summary, entry in the BGP table.

Parameter	Description
ip-address	Identifies the aggregate address to be created.
mask	Identifies the mask of the aggregate address to be created.
summary-only	(Optional) Causes the router to advertise only the aggregated route. The default is to advertise both the aggregate and the more specific routes.
as-set	(Optional) Generates AS-Path information with the aggregate route to include all the autonomous system numbers listed in all the paths of the more specific routes. The default for the aggregate route is to list only the autonomous system number of the router that generated the aggregate route.

Compare…

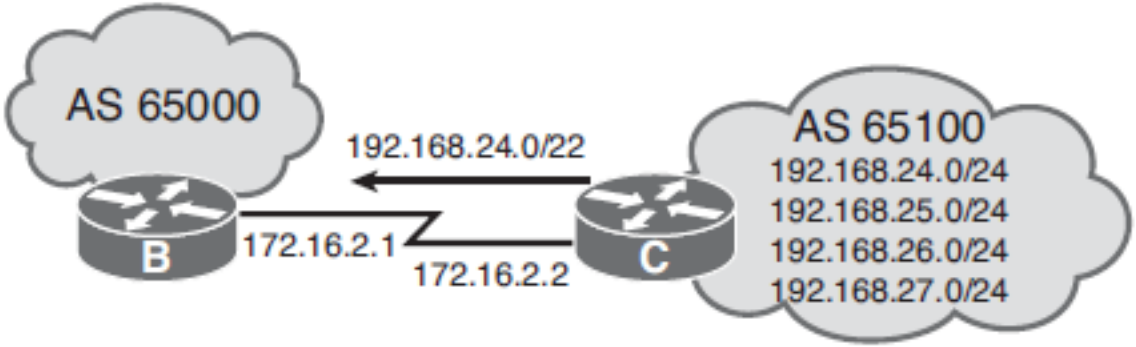
Notice the difference between the **aggregate-address** and the **network** command:

- The **aggregate-address** command aggregates only networks that are already in the *BGP table* .
- With the BGP **network** command, the network must exist in the *IP routing table* for the summary network to be advertised.

Using the aggregate-address Command

- When you use the **aggregate-address** command without the **as-set** keyword, the aggregate route is advertised as coming from your autonomous system, and the atomic aggregate attribute is set to show that information might be missing. The atomic aggregate attribute is set unless you specify the **as-set** keyword.
- Without the **summary-only** keyword, the router still advertises the individual networks.
- When the **aggregate-address** command is used, a BGP route to null 0 is automatically installed in the IP routing table for the summarized route.
- For BGP to announce a summary route using the **aggregate-address** command, at least one of the more specific routes must be in the BGP table.

aggregate-address Example




```
router bgp 65100
  network 192.168.24.0
  network 192.168.25.0
  network 192.168.26.0
  network 192.168.27.0
  neighbor 172.16.2.1 remote-as 65000
  aggregate-address 192.168.24.0 255.255.252.0 summary-only
```

aggregate-address Example

```
RouterC# show ip bgp
BGP table version is 28, local router ID is 172.16.2.1
Status codes: s = suppressed, * = valid, > = best, and i = internal
Origin codes : i = IGP, e = EGP, and ? = incomplete
Network              Next Hop          Metric  LocPrf   Weight   Path
*>192.168.24.0/22    0.0.0.0                0            32768   i
s>192.168.24.0       0.0.0.0                0            32768   i
s>192.168.25.0       0.0.0.0                0            32768   i
s>192.168.26.0       0.0.0.0                0            32768   i
s>192.168.27.0       0.0.0.0                0            32768   i
```

Communities

- The BGP communities function allows routers to tag routes with an indicator (the *community*) and allows other routers to make decisions (filter) based on that tag.
- BGP communities are used for destinations (routes) that share some common properties and that, therefore, share common policies.
- Routers, therefore, act on the community, rather than on individual routes.
- Communities are not restricted to one network or autonomous system, and they have no physical boundaries.

Community Attribute

- The community attribute is an optional transitive attribute. If a router does not understand the concept of communities, it passes it on to the next router. However, if the router does understand the concept, it must be configured to propagate the community. Otherwise, communities are dropped by default.
- Each network can be a member of more than one community.
- The community attribute is a 32-bit number.
- The upper 16 bits indicate the autonomous system number of the autonomous system that defined the community.
- The lower 16 bits are the community number and have local significance.

Setting the Communities Configuration

- Route maps can be used to set the community attributes.
- The **set community** {[*community-number*] [*well-known-community*] [**additive**]} | **none** route map configuration command is used within a route map to set the BGP community attribute.

Parameter	Description
community-number	The community number. Values are 1 to 4,294,967,200.
well-known-community	The following are predefined, well-known communities: ■ internet: Advertises this route to the Internet community and any router that belongs to it ■ no-export: Does not advertise to eBGP peers ■ no-advertise: Does not advertise this route to any peer ■ local-AS: Does not send outside the local autonomous system
additive	(Optional) Specifies that the community is to be added to the existing communities.
none	Removes the community attribute from the prefixes that pass the route map.

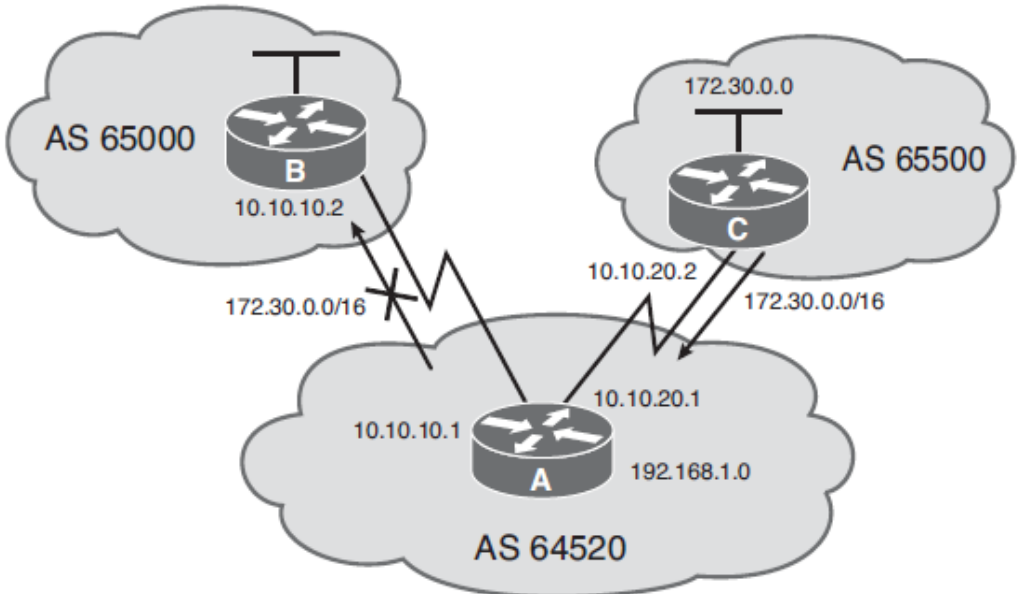
Sending the Communities Configuration

- The **`set community`** command is used along with the **`neighbor route-map`** command to apply the route map to updates.
- The **`neighbor`** { *`ip-address`* | *`peer-group-name`* } **`send-community`** router configuration command is used to specify that the BGP communities attribute should be sent to a BGP neighbor.

Parameter	Description
ip-address	The IP address of the BGP neighbor to which the communities attribute is sent
peer-group-name	The name of a BGP peer group

- By default, the communities attribute is not sent to any neighbor. (Communities are stripped in outgoing BGP updates.)

BGP Communities Example




```
router bgp 65500
  network 172.30.0.0
  neighbor 10.10.20.1 remote-as 64520
  neighbor 10.10.20.1 send-community
  neighbor 10.10.20.1 route-map SETCOMM out
!
route-map SETCOMM permit 10
  match ip address 1
  set community no-export
!
access-list 1 permit 0.0.0.0 255.255.255.255
```

Using the Communities Configuration

- The **ip community-list** *community-list-number* { **permit** | **deny** } *community-number* global configuration command is used to create a community list for BGP and to control access to it.

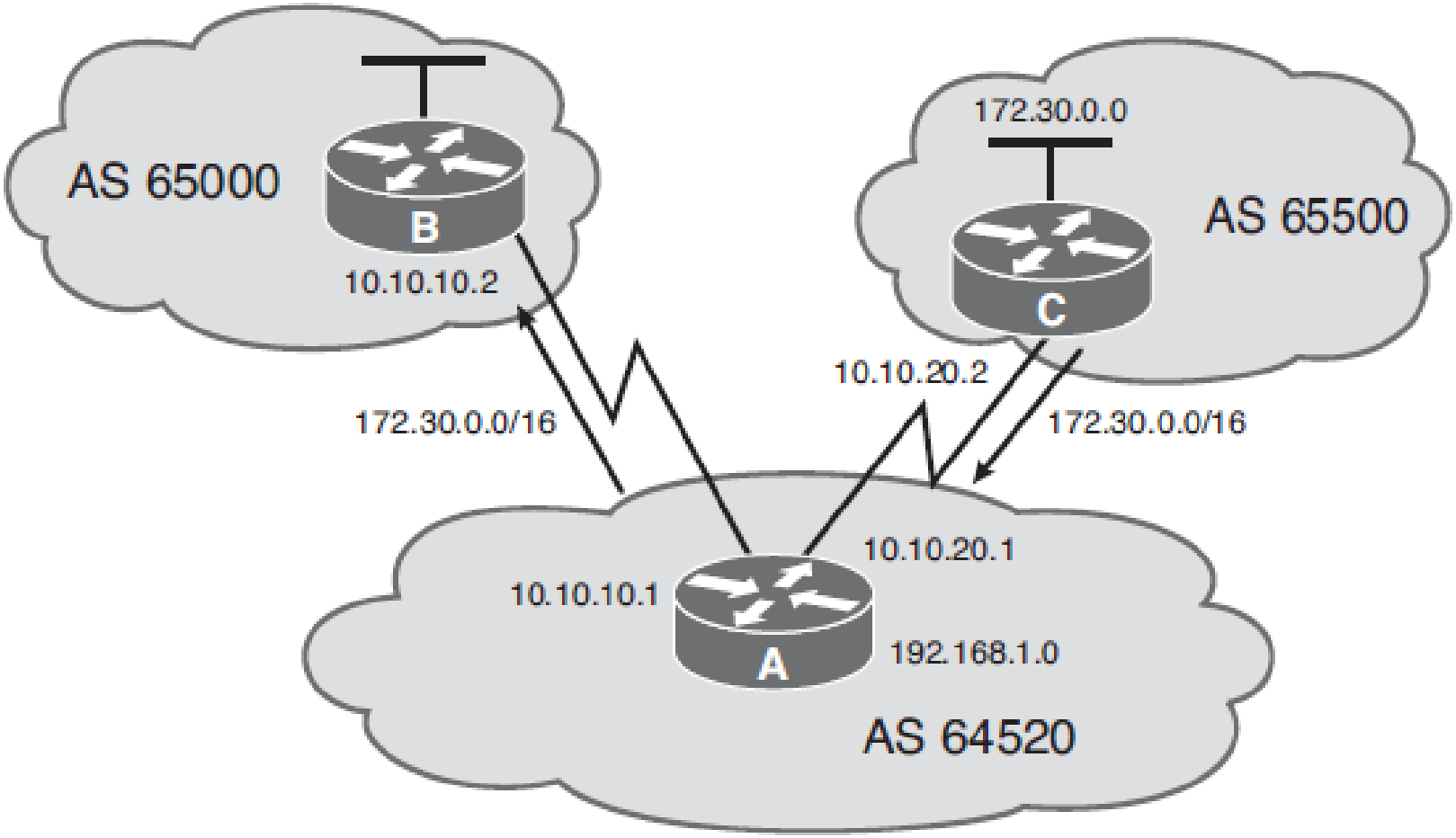
Parameter	Description
community-list-number	The community list number, in the range of 1 to 99
permit | deny	Permits or denies access for a matching condition
community-number	The community number or well-known-community configured by a set community command

Using the Communities Configuration

- The **`match community`** *`community-list-number`* **`[ exact ]`** route map configuration command enables you to match a BGP community attribute to a value in a community list.

Parameter	Description
community-list-number	The community list number, in the range of 1 to 99, that is used to compare the community attribute.
exact	(Optional) Indicates that an exact match is required. All the communities and only those communities in the community list must be present in the community attribute.

BGP Communities Example Using Weight



BGP Communities Example Using Weight

```
router bgp 65500
  network 172.30.0.0
  neighbor 10.10.20.1 remote-as 64520
  neighbor 10.10.20.1 send-community
  neighbor 10.10.20.1 route-map SETCOMM out
!
route-map SETCOMM permit 10
  match ip address 1
  set community 100 additive
!
access-list 1 permit 0.0.0.0 255.255.255.255
```

```
router bgp 64520
  neighbor 10.10.20.2 remote-as 65500
  neighbor 10.10.20.2 route-map CHKCOMM in
!
route-map CHKCOMM permit 10
  match community 1
  set weight 20
route-map CHKCOMM permit 20
  match community 2
!
ip community-list 1 permit 100
ip community-list 2 permit internet
```

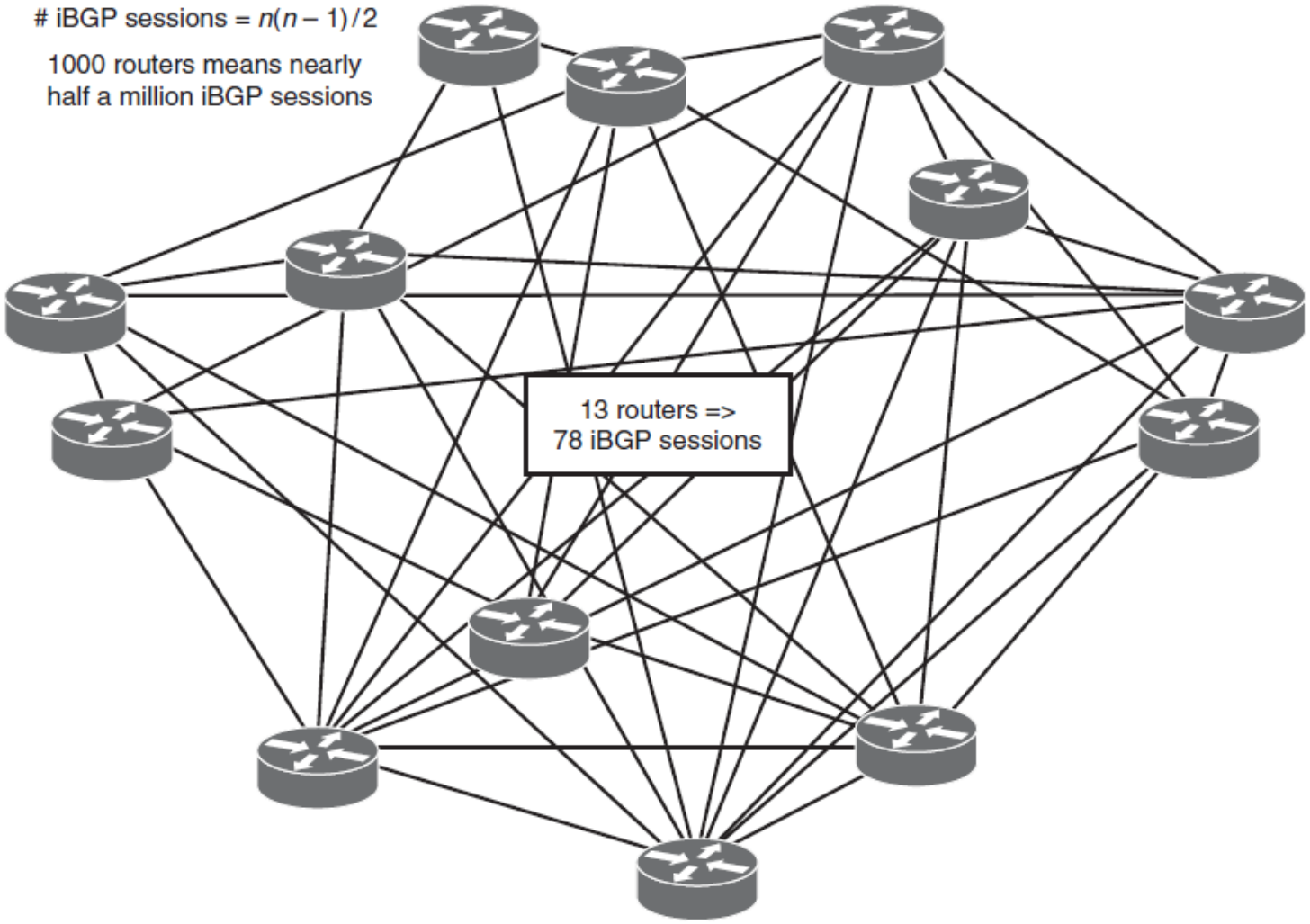
BGP Communities Example Using Weight

```
RtrA # show ip bgp 172.30.0.0/16
BGP routing Table entry for 172.30.0.0/16, version 2
Paths: (1 available, best #1)
  Advertised to non peer-group peers:
    10.10.10.2
  65500
    10.10.20.2 from 10.10.20.2 (172.30.0.1)
      Origin IGP, metric 0, localpref 100, weight 20, valid, external, best, ref 2
 Community: 100
```

Route Reflectors

- BGP specifies that routes learned via iBGP are never propagated to other iBGP peers.
- The result is that a full mesh of iBGP peers is required within an autonomous system.
- With only 13 routers, 78 iBGP sessions would need to be maintained.
- As the number of routers increases, so does the number of sessions required, governed by the following formula, in which *n* is the number of routers:
- Number of iBGP sessions = $n (n – 1) / 2$

Route Reflectors (PANIC MODE ☹)

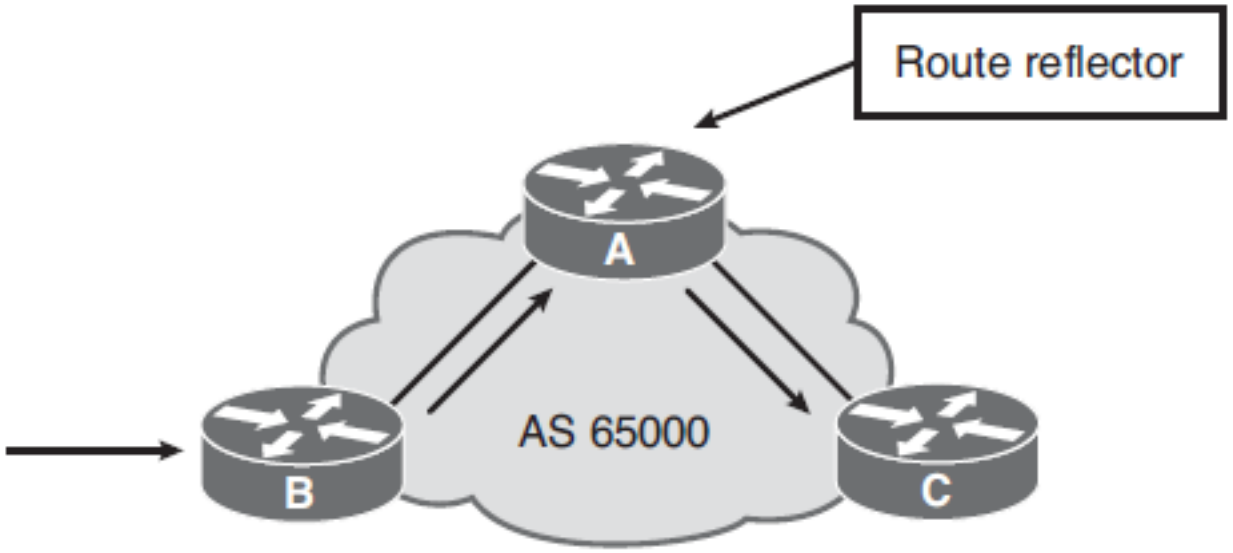


Route Reflectors

- In addition to the number of BGP TCP sessions that must be created and maintained, the amount of routing traffic might also be a problem.
- Depending on the autonomous system topology, traffic might be replicated many times on some links as it travels to each iBGP peer.
- For example, if the physical topology of a large autonomous system includes some WAN links, the iBGP sessions running over those links might consume a significant amount of bandwidth.
- A solution to this problem is the use of route reflectors (RRs).
- This section describes what an RR is, how it works, and how to configure it.
- RRs modify the BGP rule by allowing the router configured as the RR to propagate routes learned by iBGP to other iBGP peers

Route Reflectors

- This saves on the number of BGP TCP sessions that must be maintained and also reduces the BGP routing traffic.



Route Reflector Benefits

- With a BGP RR configured, a full mesh of iBGP peers is no longer required.
- The RR is allowed to propagate iBGP routes to other iBGP peers.
- Route reflectors reduce the number of BGP neighbor relationships in an autonomous system by having key routers replicate updates to their RR clients.
- Route reflectors do not affect the paths that IP packets follow.
- Only the path that routing information is distributed on is affected.
- An autonomous system can have multiple RRs, both for redundancy and for grouping to further reduce the number of iBGP sessions required.
- Migrating to RRs involves a minimal configuration and does not have to be done all at one time, because routers that are not RRs can coexist with RRs within an autonomous system.

Route Reflector Terminology

- **Route Reflector:**
  - Router that is configured to be the router allowed to advertise (or reflect) routes it learned via iBGP to other iBGP peers.
- ***Clients**:*
  - Routers peering with the RR has a partial iBGP
  - Peering between the clients is not needed, because the route reflector passes advertisements between the clients.
- ***Cluster:***
  - The combination of the RR and its clients
- ***Nonclients:***
  - Other iBGP peers of the RR that are not clients
- ***Originator ID:***
  - Is an optional, nontransitive BGP attribute that is created by the RR.
  - This attribute carries the router ID of the route's originator in the local autonomous system.
  - If the update comes back to the originator because of poor configuration, the originator ignores it.

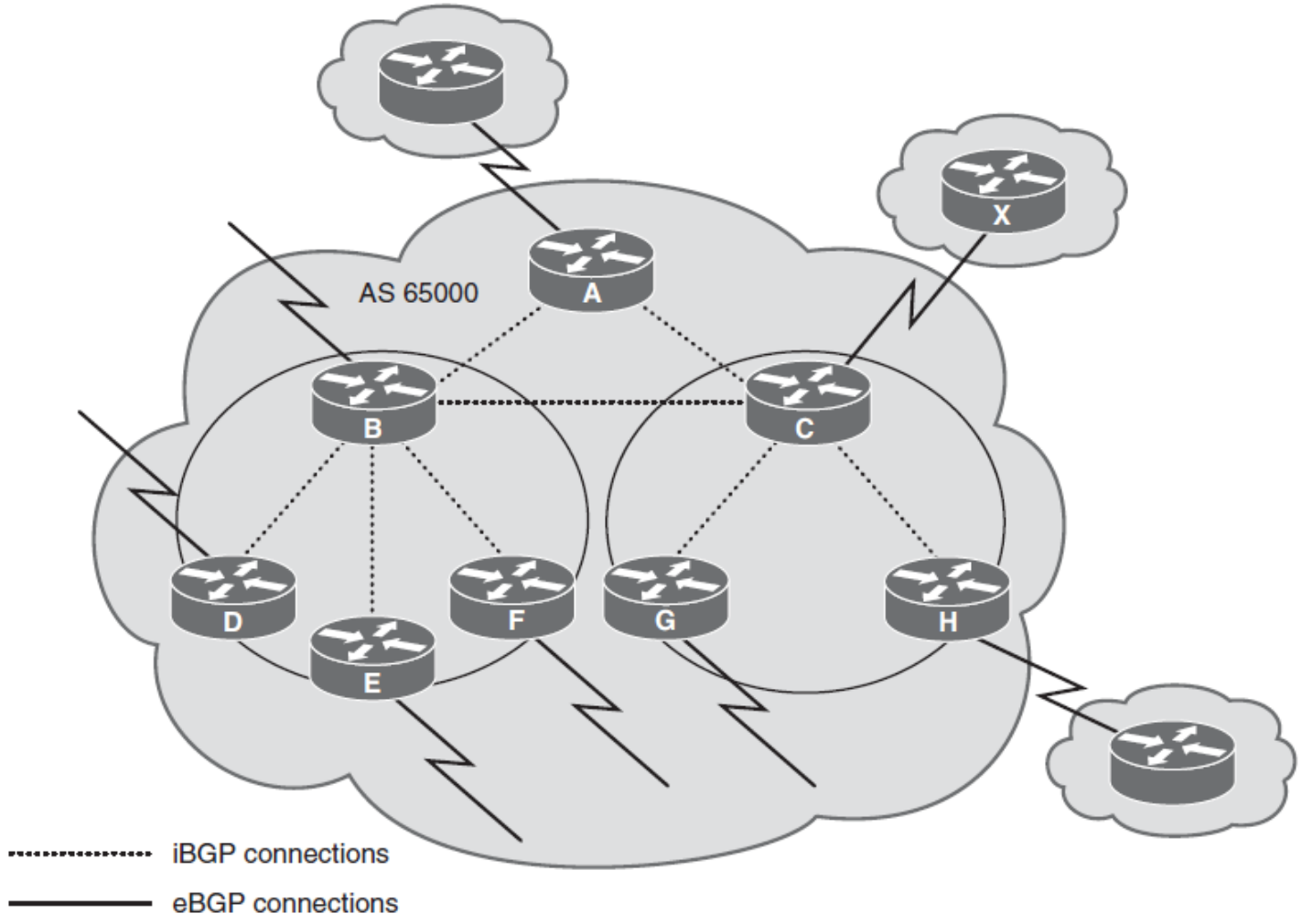
Route Reflector Terminology

- **Cluster ID :**
  - Usually a cluster has a single RR, in which case the cluster is identified by the RR's router ID.
  - To increase redundancy and avoid single points of failure, a cluster might have more than one RR. When this occurs, all the RRs in the cluster need to be configured with a Cluster ID.
  - The cluster ID allows route reflectors to recognize updates from other RRs in the same cluster.
- ***Cluster list***
  - Is a sequence of cluster IDs that the route has passed. When an RR reflects a route from its clients to nonclients outside the cluster, it appends the local cluster ID to the cluster list. If the update has an empty cluster list, the RR creates one.
  - Using this attribute, an RR can tell whether the routing information is looped back to the same cluster because of poor configuration.
  - If the local cluster ID is found in an advertisement's cluster list, the advertisement is ignored.
  - The originator ID, cluster ID, and cluster list help prevent routing loops in RR configurations.

Route Reflector Design

- When using RRs in an autonomous system, you can divide the autonomous system into multiple clusters, each having at least one RR and a few clients. Multiple RRs can exist in one cluster for redundancy.
- The RRs must be fully meshed with iBGP to ensure that all routes learned are propagated throughout the autonomous system.
- An IGP is still used, just as it was before RRs were introduced, to carry local routes and next-hop addresses.
- Normal split-horizon rules still apply between an RR and its clients. Thus an RR that receives a route from a client does not advertise that route back to that client.

Route Reflector Design Example



Route Reflector Operation

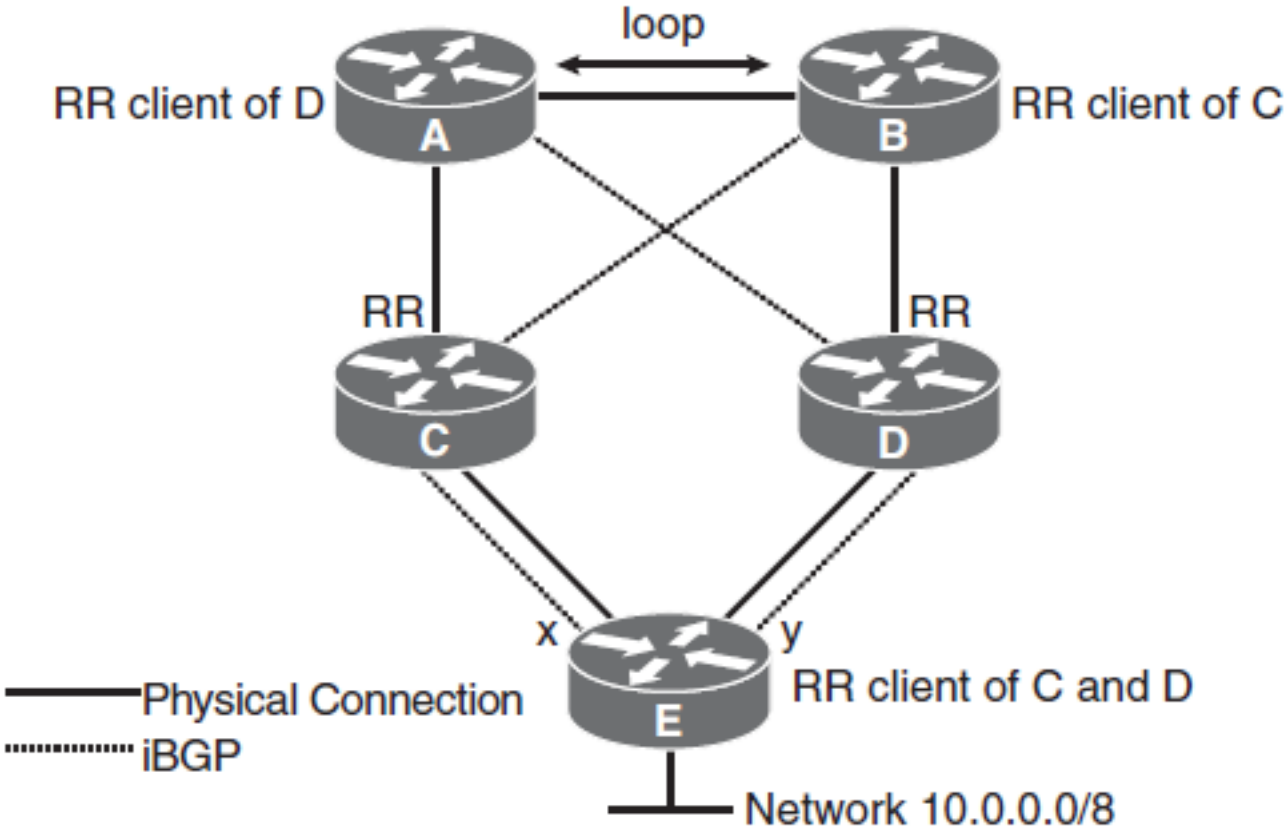
When an RR receives an update, it takes the following actions, depending on the type of peer that sent the update:

- If the update is from a client peer, it sends the update to all nonclient peers and to all client peers (except the route's originator).
- If the update is from a nonclient peer, it sends the update to all clients in the cluster.
- If the update is from an eBGP peer, it sends the update to all nonclient peers and to all client peers.

Route Reflector Migration Tips

- When migrating to using RRs, the first consideration is which routers should be the reflectors and which should be the clients.
- Following the physical topology in this design decision ensures that the packet-forwarding paths are not affected.
- Not following the physical topology (for example, configuring RR clients that are not physically connected to the route reflector) might result in routing loops.

Bad Route Reflector Design

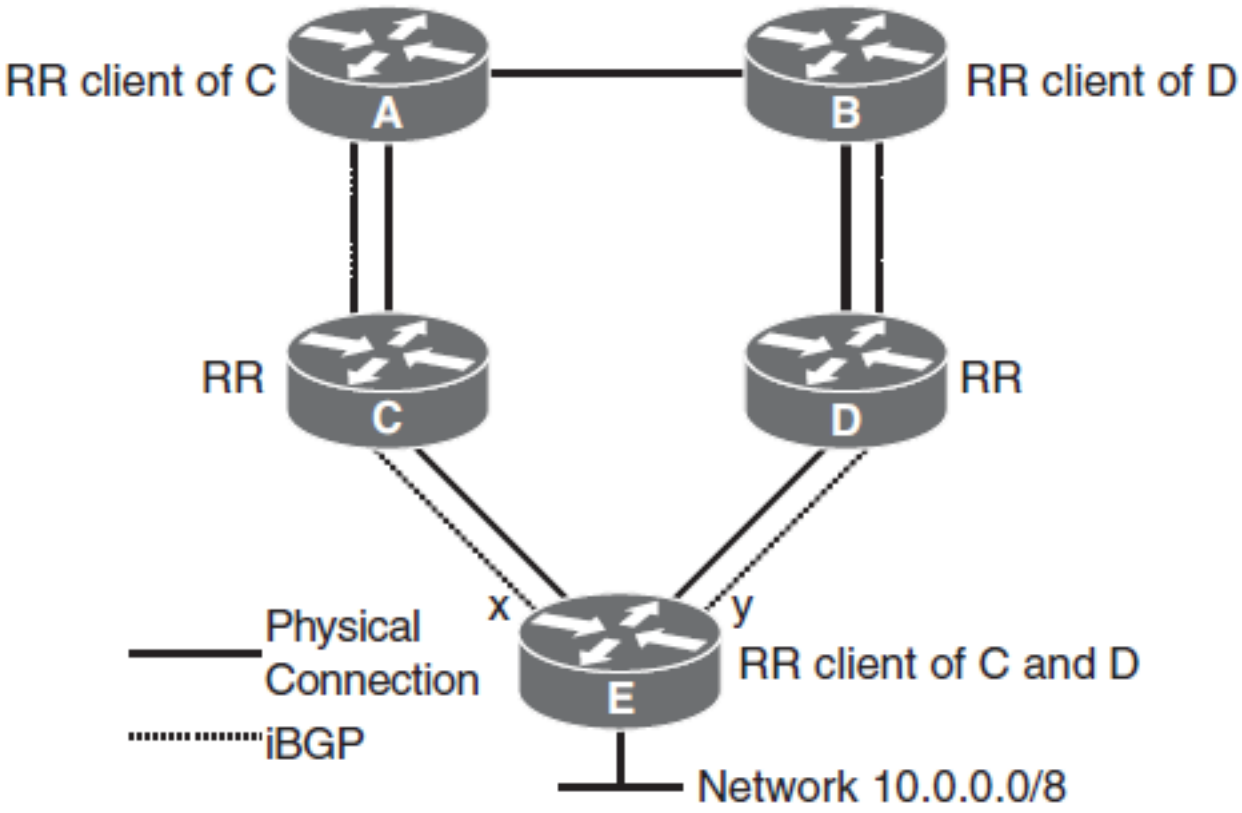


Bad Route Reflector Design

In this *bad design* , which does not follow the physical topology, the following happens:

- Router B knows that the next hop to get to 10.0.0.0 is *x* (because it learns this from its RR, Router C).
- Router A knows that the next hop to get to 10.0.0.0 is *y* (because it learns this from its RR, Router D).
- For Router B to get to *x* , the best route might be through Router A, so Router B sends a packet destined for 10.0.0.0 to Router A.
- For Router A to get to *y* , the best route might be through Router B, so Router A sends a packet destined for 10.0.0.0 to Router B.
- This is a routing loop.

Good Route Reflector Design

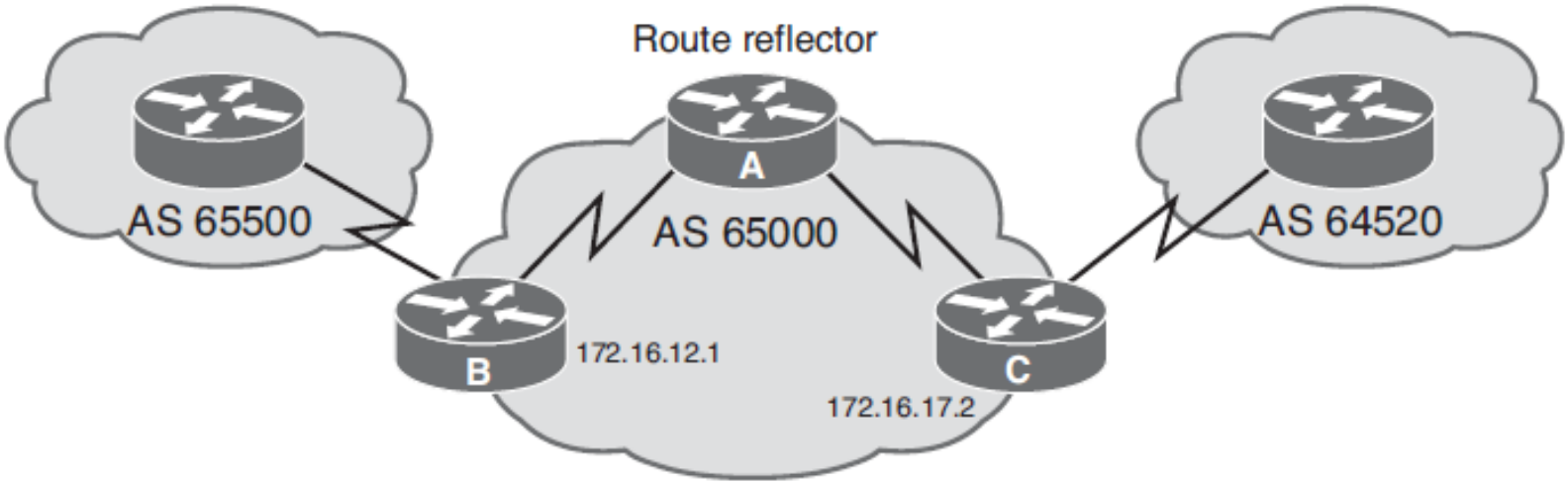


Good Route Reflector Design

In this *good design* , which follows the physical topology, the following are true:

- Router B knows that the next hop to get to 10.0.0.0 is *y* (because it learns this from its RR, Router D).
- Router A knows that the next hop to get to 10.0.0.0 is *x* (because it learns this from its RR, Router C).
- For Router A to get to *x* , the best route is through Router C, so Router A sends a packet destined for 10.0.0.0 to Router C, and Router C sends it to Router E.
- For Router B to get to *y* , the best route is through Router D, so Router B sends a packet destined for 10.0.0.0 to Router D, and Router D sends it to Router E.
- There is no routing loop.

Route Reflector Configuration




```
RTRA(config)# router bgp 65000
RTRA(config-router)# neighbor 172.16.12.1 remote-as 65000
RTRA(config-router)# neighbor 172.16.12.1 route-reflector-client
RTRA(config-router)# neighbor 172.16.17.2 remote-as 65000
RTRA(config-router)# neighbor 172.16.17.2 route-reflector-client
```

Advertising a Default Route

- The **neighbor** { *ip-address* | *peer-group-name* } **default-originate** [**route-map** *map-name*] router configuration command can be used for a BGP router to send the default route 0.0.0.0 to a neighbor, for its use as a default route.

Parameter	Description
ip-address	The IP address of the BGP neighbor
peer-group-name	The name of a BGP peer group
route-map *map-name*	(Optional) Identifies a route map, to allow the default route to be sent to the neighbor conditionally

Not Advertising Private Autonomous System Numbers

- IANA defines private autonomous system numbers 64512 through 65534 to be used for private purposes.
- Only public autonomous system numbers should be sent to eBGP neighbors on the Internet.

Not Advertising Private Autonomous System Numbers

- Use the **`neighbor`** { *`ip-address`* | *`peer-group-name`* } **`remove-private-as`** [**`all`** [**`replace-as`**]] router configuration command to remove private autonomous system numbers from the AS-Path attribute; this command is available only for eBGP neighbors

Parameter	Description
ip-address	The IP address of the BGP neighbor.
peer-group-name	The name of a BGP peer group.
all	(Optional) Removes all private autonomous system numbers from the autonomous system path in outgoing updates.
replace-as	(Optional) Only valid with the all keyword, the replace-as keyword causes all private autonomous system numbers in the autonomous system path to be replaced with the router's local autonomous system number. This ensures that the length of the AS-path attribute (used in the BGP path selection process) remains the same as it was before the autonomous system numbers were replaced.

Chapter 7 Summary

- BGP terminology and concepts, including the following:
- BGP's use between autonomous systems and how it is different than other routing protocols described in this book
- BGP's classification as a path vector protocol and its use of TCP protocol 179
- BGP's loop-free guarantee, because it does not accept a routing update that already includes its autonomous system number in the AS-path list
- The three tables used by BGP: the BGP table, IP routing table, and BGP neighbor table
- The four BGP message types: open, keepalive, update, and notification
- When to use BGP: if the autonomous system allows packets to transit through it to reach other autonomous systems, if the autonomous system has multiple connections to other autonomous systems, or if the routing policy and route selection for traffic entering and leaving the autonomous system must be manipulated
- The use of full-mesh iBGP on all routers in the transit path within the autonomous system

Chapter 7 Summary

- When not to use BGP: if there is only a single connection to the Internet or another autonomous system, if edge routers have a lack of memory or processing power, if you have a limited understanding of route filtering and the BGP path-selection process, or if the routing policy that will be implemented in an autonomous system is consistent with the policy implemented in the ISP autonomous system
- BGP neighbor (peer) relationships:
- iBGP, when BGP runs between routers in the same autonomous system
- eBGP, when BGP runs between routers that are in different autonomous systems.
- eBGP neighbors are typically directly connected .

Chapter 7 Summary

- Basic BGP configuration, including the relationship between the BGP table, the IP routing table and the **network** command: The **network** command allows a BGP router to inject a network that is in its IP routing table into its BGP table and advertise that network to its BGP neighbors. BGP neighbors exchange their best BGP routes.
- The neighbor router that receives that network information puts the information in its BGP table and selects its best BGP route for that network. The best route is offered to its IP routing table.
- Using BGP features, including next-hop-self, update source, and eBGP multihop.
- Understanding and troubleshooting the BGP states: idle, connect, active, open sent, open confirm, and established.
- Performing hard and soft resets of BGP sessions, required after a neighbor policy is changed.

Chapter 7 Summary

- The BGP attributes that can be either well-known or optional, mandatory or discretionary, and transitive or nontransitive. An attribute might also be partial. The BGP attributes are the following:
  - **AS-path:** Well-known mandatory. The list of autonomous system numbers that a route has traversed to reach a destination, with the number of the autonomous system that originated the route at the end of the list.
  - **Next hop:** Well-known mandatory. Indicates the next-hop IP address that is to be used to reach a destination. For eBGP, the next hop is the IP address of the neighbor that sent the update; for iBGP, the next hop advertised by eBGP is carried into iBGP by default.
  - **Origin:** Well-known mandatory. Defines the origin of the path information; can be IGP, EGP, or incomplete.
  - **Local preference:** Well-known discretionary. Indicates to routers in the autonomous system which path is preferred to exit the autonomous system. The path with a *higher* local preference is preferred. Sent only to iBGP neighbors.
  - **Atomic aggregate:** Well-known discretionary. Informs the neighbor autonomous system that the originating router has aggregated the routes.

Chapter 7 Summary

- **Aggregator:** Optional transitive. Specifies the BGP router ID and autonomous system number of the router that performed the route aggregation.
- **Community:** Optional transitive. Allows routers to tag routes with an indicator (the community) and allows other routers to make decisions based on that tag.
- **MED:** Optional nontransitive. Also called metric. Indicates to external neighbors the preferred path into an autonomous system. A *lower* value is preferred; exchanged between autonomous systems.
- **Weight:** Cisco defined; provides local routing policy only and is not propagated to any BGP neighbors. Routes with a *higher* weight are preferred.

Chapter 7 Summary

- The 11-step BGP route-selection decision process is as follows:
  - **1.** Prefer the highest weight.
  - **2.** Prefer the highest local preference.
  - **3.** Prefer the route originated by the local router.
  - **4.** Prefer the shortest AS-path.
  - **5.** Prefer the lowest origin code.
  - **6.** Prefer the lowest MED.
  - **7.** Prefer the eBGP path over the iBGP path.
  - **8.** Prefer the path through the closest IGP neighbor.
  - **9.** Prefer the oldest route for eBGP paths.
  - **10.** Prefer the path with the lowest neighbor BGP router ID.
  - **11.** Prefer the route with the lowest neighbor IP address.

Chapter 7 Summary

- Verifying BGP configuration.
- BGP path manipulation and filtering, including changing the weight, local preference, AS-path, and MED attributes. Prefix lists, distribute lists, filter lists, and route maps may be used.
- Configuring BGP peer groups, a group of BGP neighbors of the router being configured that all have the same update policies.
- Implementing MP-BGP for IPv6, including the following:
- Exchanging IPv6 routes over an IPv4 session
- Exchanging IPv6 routes over an IPv6 session
- BGP filtering mechanisms used for IPv6.

Chapter 7 Labs

- **CCNPv7 ROUTE Lab7.1 BGP Config**
- **CCNPv7 ROUTE Lab7.2 BGP AS PATH**
- **CCNPv7 ROUTE Lab7.3 IBGP EBGP LocalPref MED**
- **CCNPv7 ROUTE Lab7.4 IBGP EBGP Synchronization**
- **CCNPv7 ROUTE Lab7.5 MP-BGP**

Acknowledgment

- Some of images and texts are from Implementing Cisco IP Routing (ROUTE) Foundation Learning Guide by Diane Teare, Bob Vachon and Rick Graziani (1587204568)
- Copyright © 2015 – 2016 Cisco Systems, Inc.
- Special Thanks to *Bruno Silva*