



You make **possible**



# BGP General Operation and Overview

Steven Moore @smoore\_bits  
Oliver Boehmer @oboehmer

TECRST-1310

**CISCO** *Live!*

Barcelona | January 27-31, 2020



# Abstract

This introductory level Techtorial will cover foundational aspects of the Border Gateway Protocol (BGP) and explore many details of the protocol, including best practices needed to ensure a successful deployment of BGP into your network.

Topics of our session include BGP General Operation, Attributes and Policy Control, BGP Path Selection Algorithm, Applying Policy with BGP, BGP Route-Reflectors, Multi-protocol BGP and some common BGP Deployment Scenarios. We will end with a short lab demonstrating some of the items that were presented during the session.

We do not assume any prior knowledge of BGP in particular, but some knowledge of general IP routing concepts is required.

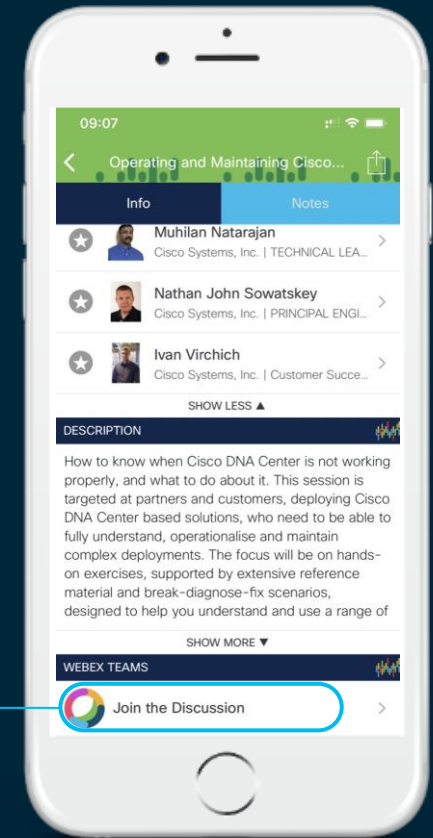
# Cisco Webex Teams

## Questions?

Use Cisco Webex Teams to chat with the speaker after the session

## How

- 1 Find this session in the Cisco Events Mobile App
- 2 Click “Join the Discussion”
- 3 Install Webex Teams or go directly to the team space
- 4 Enter messages/questions in the team space



# Agenda

- BGP General Operation

- Overview
- EBGP and IBGP

- Attributes and Best Path Selection Algorithm

- Route Origination
- AS-PATH
- NEXTHOP
- Communities

- Controlling Traffic

- Controlling Outbound Traffic
- BGP Multipath
- Controlling Inbound Traffic

*What is BGP?  
How does BGP work?  
What problem does  
BGP solve?*

- Route Reflectors

- Multiprotocol BGP

- Common BGP Deployments

- Securing BGP

- BGP Routing Convergence

- Show and Tell/Demo Lab

# Routing Protocol Background

- Routing protocols share the same fundamental, essential components.
  - Establish Communication
    - Who are they exchanging information with, and how?
  - Exchange Routes
    - What information is sent, and how?
  - Perform Computation
    - What algorithm is used to compute loop free paths?
  - Route Installation
    - What routes are the best? Can we install them?
- BGP is no exception!
- Understanding how BGP implements each of these will help us learn, use, and operate networks with BGP.

# IGP vs EGP

- IGP – Interior Gateway Protocol
  - Exchange routes within Autonomous Systems
  - Limited Scalability
  - Sub-second convergence
  - EIGRP, IS-IS, OSPF etc.
  
- EGP (BGP) – Exterior Gateway Protocol
  - Exchange routes between Autonomous Systems
  - Slower convergence in exchange for scalability
  - eBGP, iBGP\*

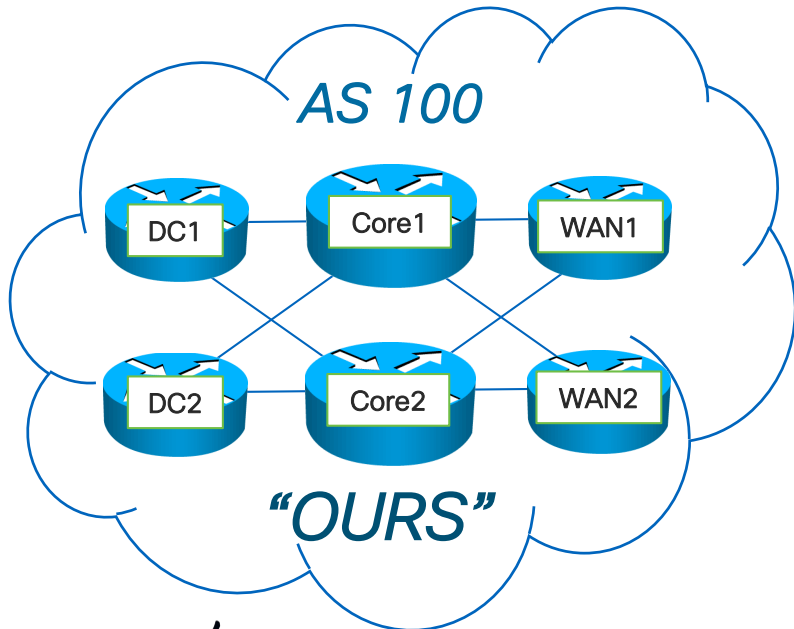
*\*Some may refer to them as EBGP and IBGP, e.g. IETF*



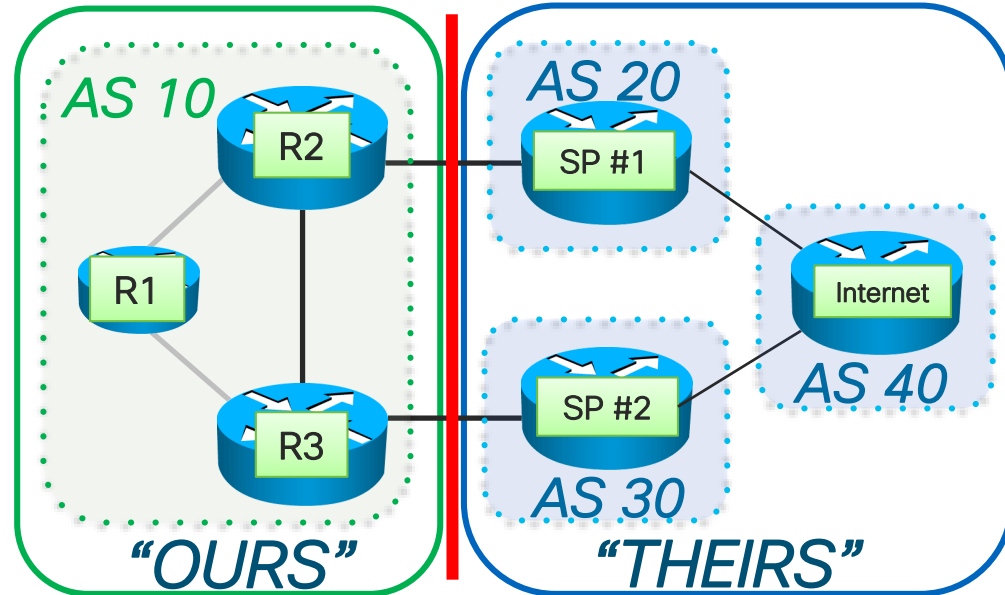
# IGP vs BGP: Autonomous System

An Autonomous System is usually under single administrative control

## EIGRP



## BGP

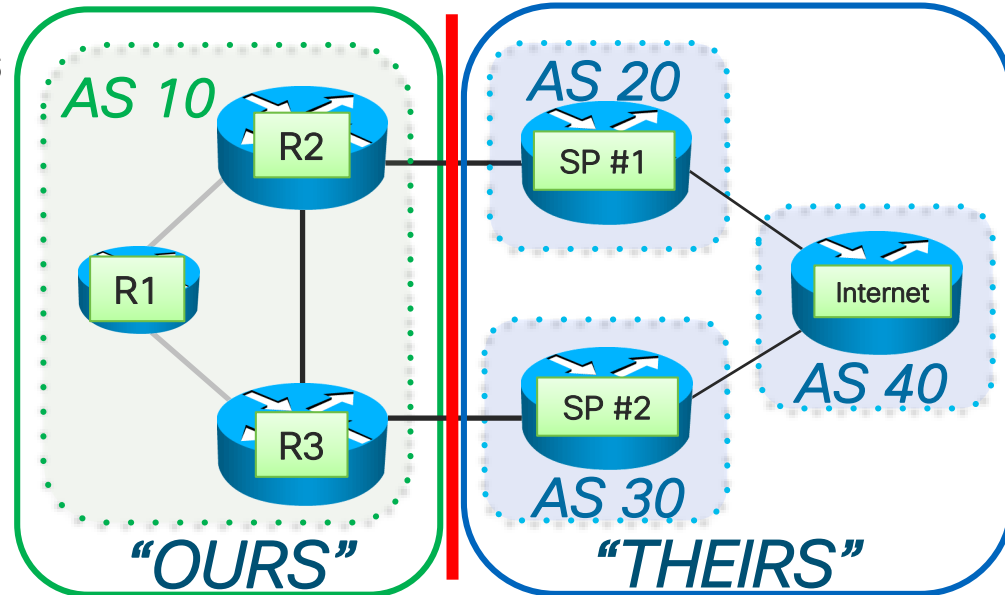




# BGP Autonomous System

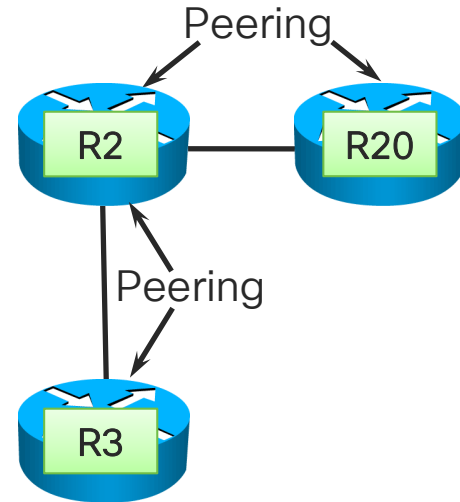
- A network sharing the same routing policy
  - Usually under single administrative control
  - Possibly multiple IGPs
- An AS originates their own routes
- AS Numbers
  - Historically 2 bytes
    - 1 To 65535 (64512-65535 are private)
    - Running out of AS numbers...
  - RFC 4893
    - 4-byte AS number
    - Unique AS for every IPv4 address

*BGP*

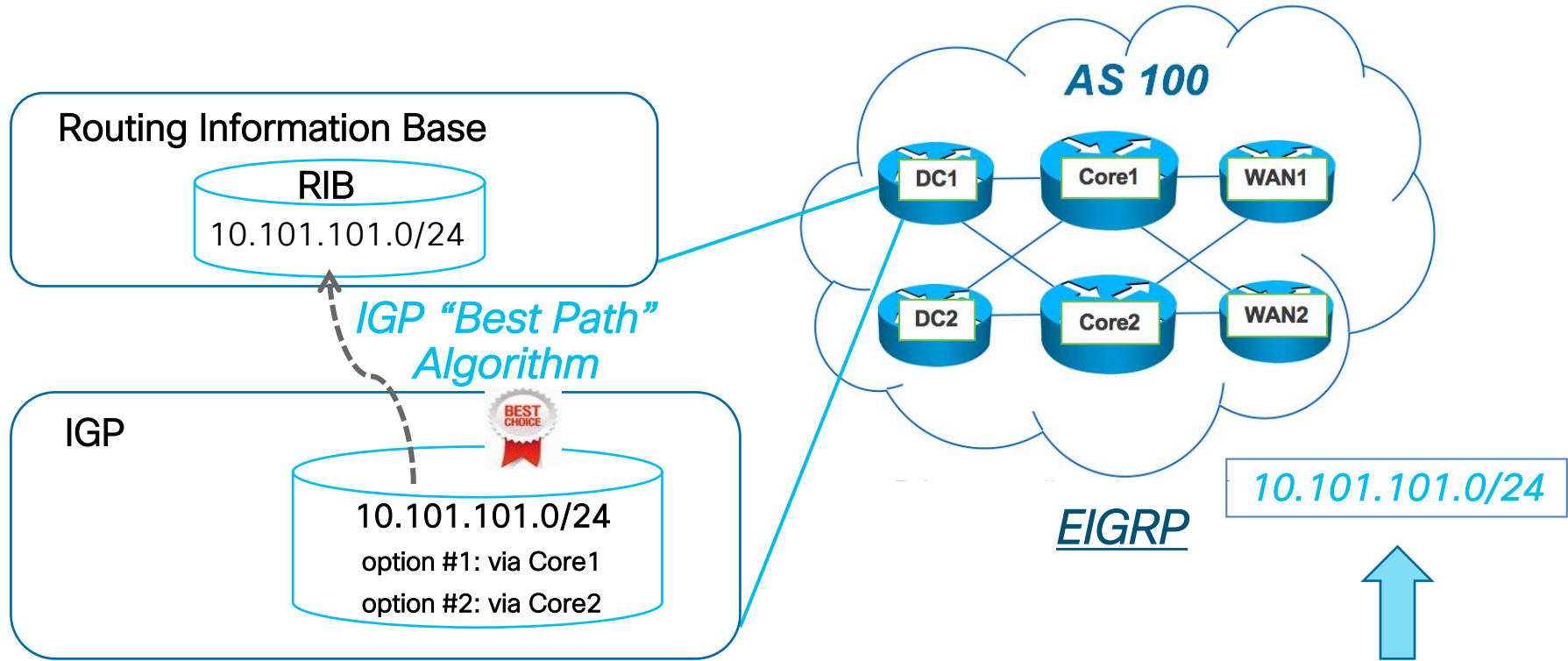


# BGP General Operation: Overview

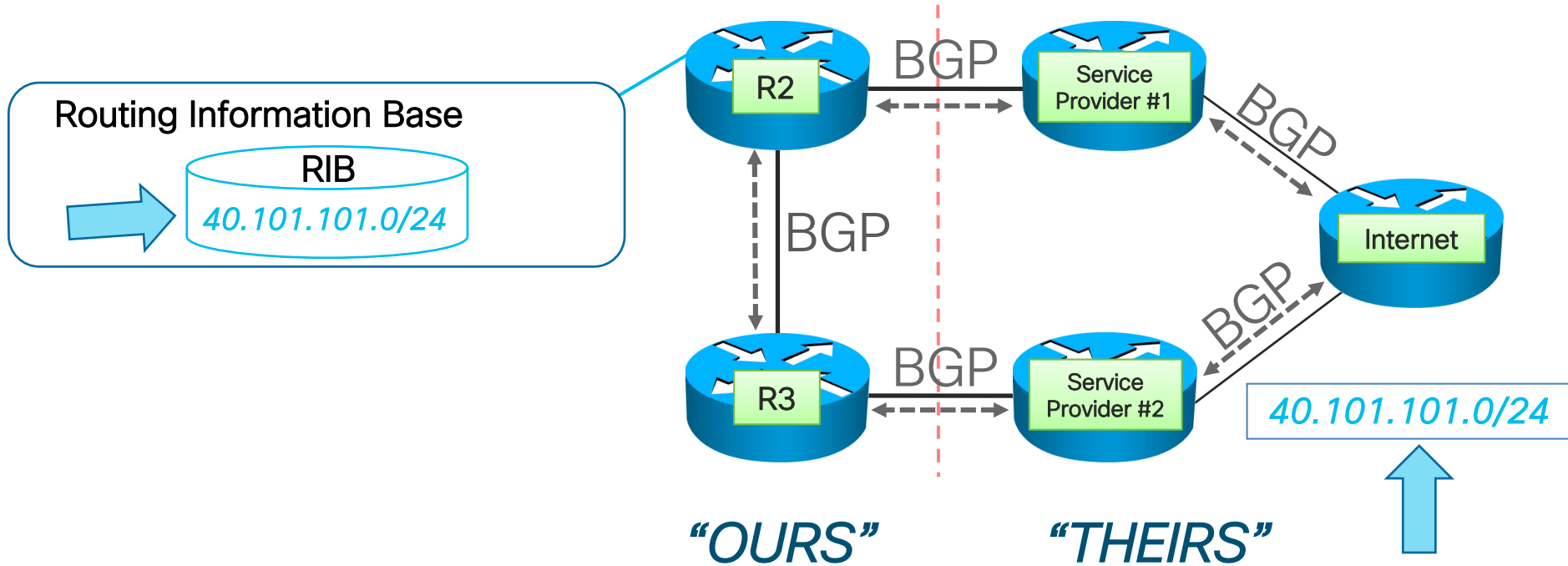
- BGP peers with other BGP speakers
  - Peer is also called “neighbor”
  - Uses TCP port 179
  - Negotiates communication
- BGP peers exchange routes via UPDATES
- UPDATES have Attributes describing the route
- Picks the Best Path
  - Installs in the routing/forwarding table
  - Advertises to BGP peers via UPDATES
- Routing policies are used to tweak Attributes to influence Best Path selection



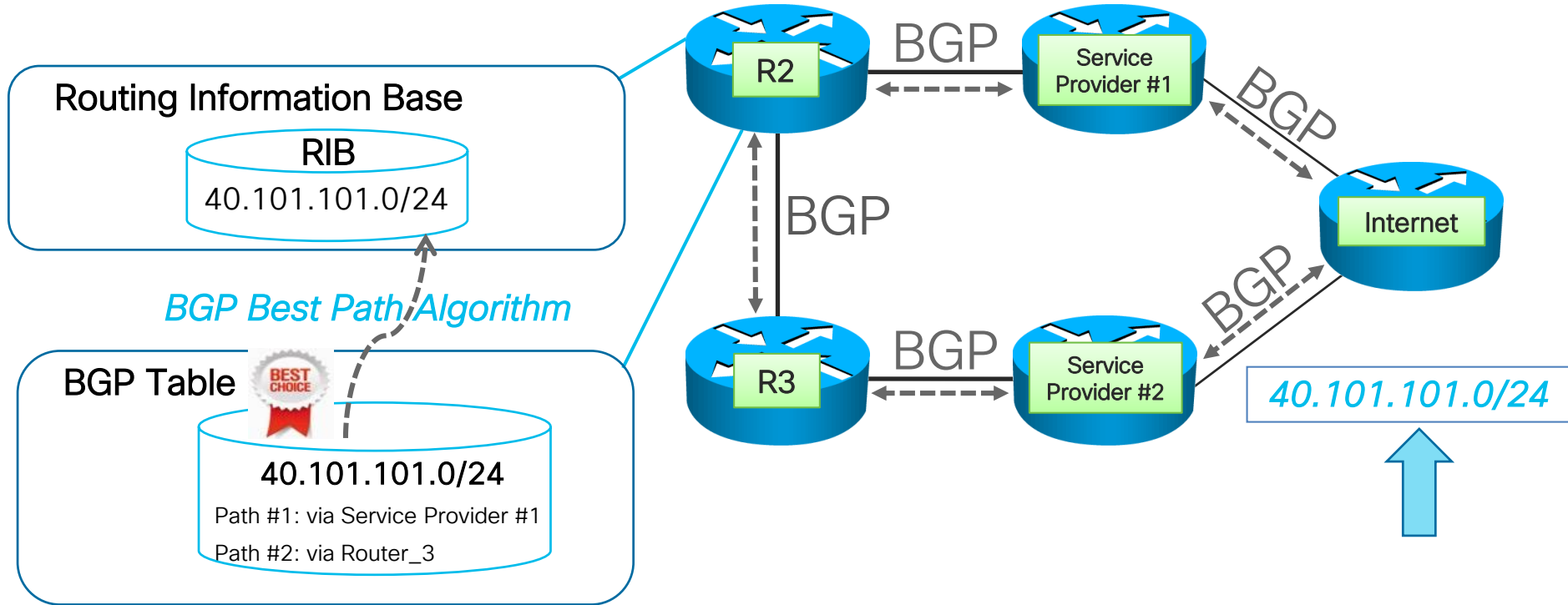
# IGP Best Paths



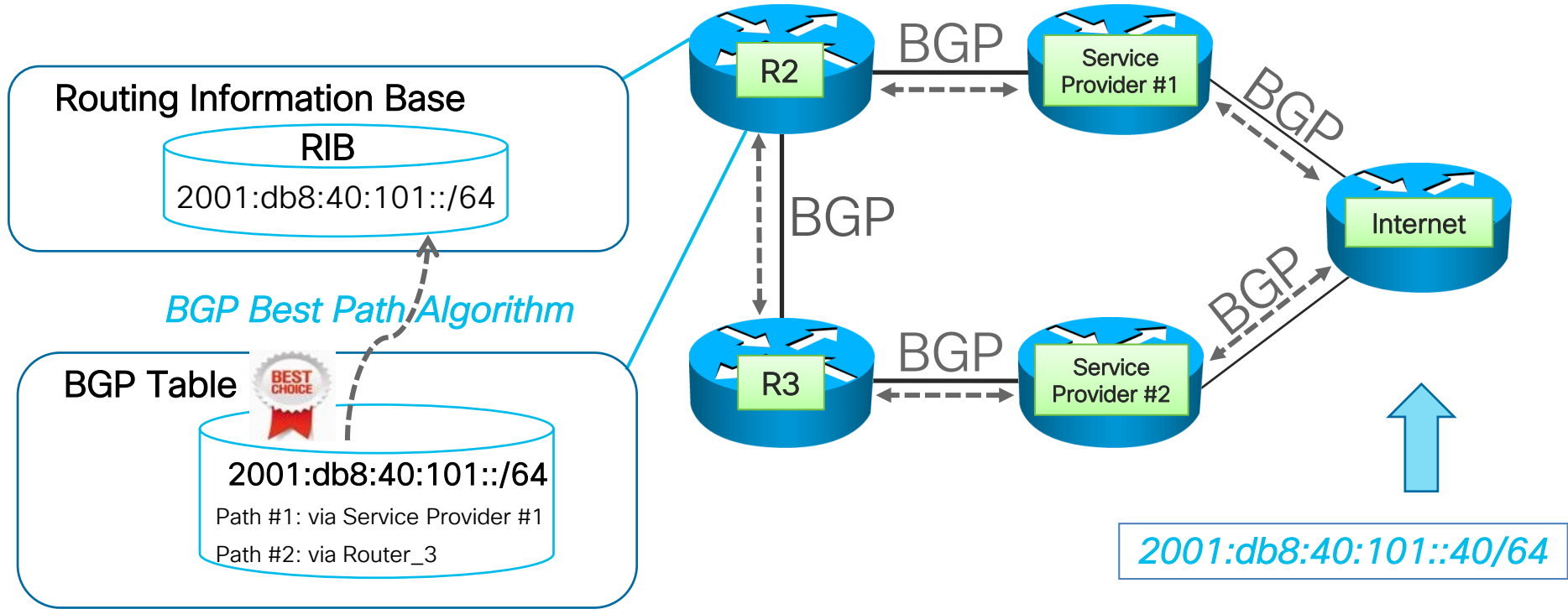
# Overview



# Overview

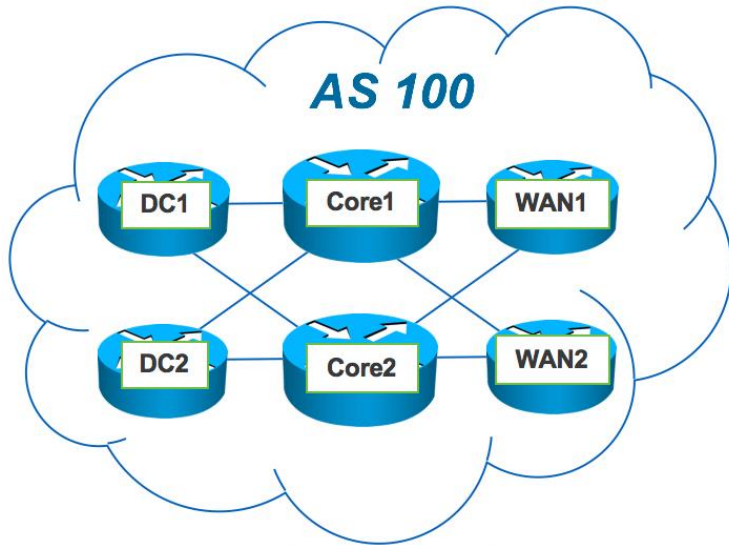


# Overview



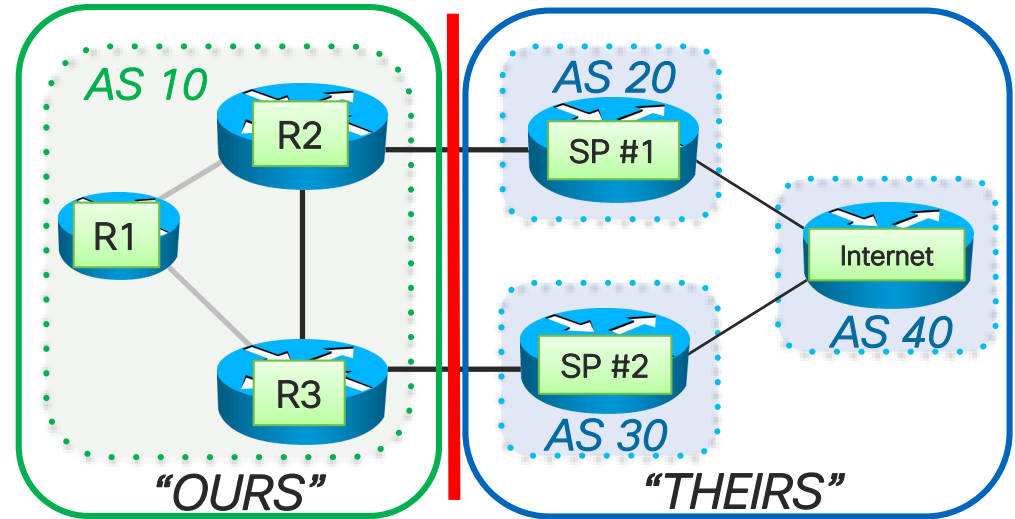
# IGP vs BGP: Best Paths and Attributes

## EIGRP



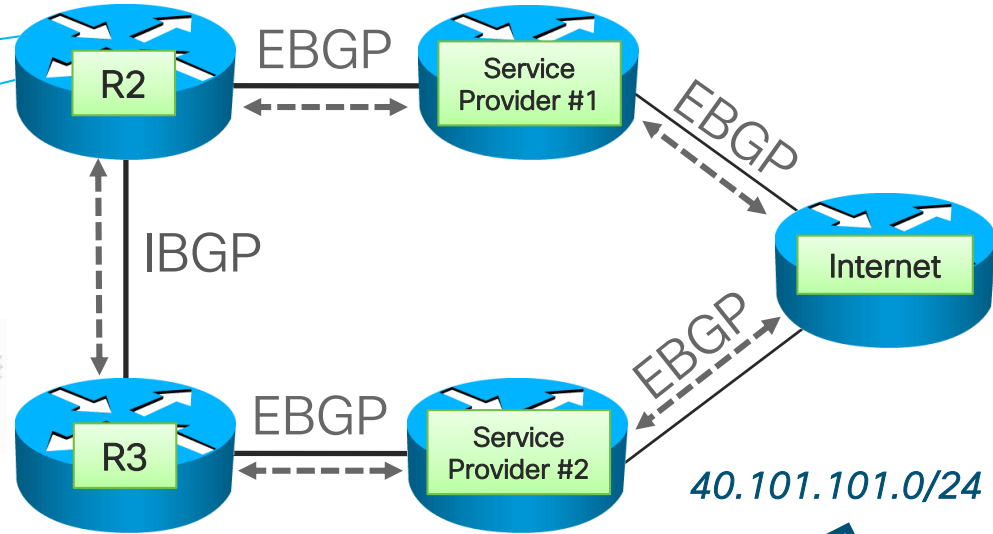
- Primary attribute is a cost/metric
- The path with the lowest metric is the best...nice and easy

## BGP



- Routing Policy between AS is usually more complex
- Shortest path is not necessarily the best one
- The “Best Path Algorithm” compares attributes between different paths to select the best
- Route-policies are used to tweak attributes to influence outcome of Best Path

# BGP: Best Paths and Attributes



```
R2#show ip bgp 40.101.101.0
```

```
BGP routing table entry for 40.101.101.0/24
```

```
Paths: (2 available, best #1, table default)
```

```
Advertised to update-groups:
```

```
1
```

```
Refresh Epoch 1
```

```
20 40 ← as-path
```

```
20.2.20.20 from 20.2.20.20 (20.100.100.20)
```

```
Origin IGP, localpref 100, valid, external, best
```

```
rx pathid: 0, tx pathid: 0x0
```

```
Refresh Epoch 1
```

```
30 40 ← as-path
```

```
10.100.100.3 (metric 2) from 10.100.100.3 (10.100.100.3)
```

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

```
rx pathid: 0, tx pathid: 0
```



Externally learned (EBGP)

Internally learned (IBGP)

40.101.101.0/24





# BGP CLI Command Structure



- It used to be that BGP was used strictly for IPv4.
- We now have IPv6, VPN, multicast, and many, many other types of information which BGP can transport. (Address Families, and Sub-Address Families)
- Many of the commands from IOS are historically focused around IPv4, but we've included a reference with a small sample of examples for the current, modern cli structure from IOS.
- Since the video, and supporting details use the older familiar style, we have tried to maintain consistency. Please familiarize yourself with the new structure.



```
r#show bgp ipv4 unicast
```

```
BGP table version is 22, local router ID is 1.1.1.1
```

```
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,  
              t secondary path, L long-lived-stale,
```

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

```
RPKI validation codes: V valid, I invalid, N Not found
```

	Network	Next Hop	Metric	LocPrf	Weight	Path
*>	1.1.1.1/32	0.0.0.0	0		32768	i
*>	2.2.2.2/32	8.8.8.2	0		0 20	i
*>	3.3.3.3/32	9.9.9.2	0		0 30	i
*	8.8.8.0/29	8.8.8.2	0		0 20	i
*>		0.0.0.0	0		32768	i
*	9.9.9.0/29	9.9.9.2	0		0 30	i
*>		0.0.0.0	0		32768	i



**r#show ip bgp summary**

BGP router identifier 1.1.1.1, local AS number 10  
BGP table version is 22, main routing table version 22  
5 network entries using 1240 bytes of memory  
7 path entries using 952 bytes of memory  
3/3 BGP path/bestpath attribute entries using 840 bytes of memory  
2 BGP AS-PATH entries using 48 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 3080 total bytes of memory  
BGP activity 17/7 prefixes, 36/22 paths, scan interval 60 secs

Neighbor	V	AS	MsgRcvd	MsgSent	TblVer	InQ	OutQ	Up/Down	State/PfxRcd
8.8.8.2	4	20	2718	2725	22	0	0	1d17h	2
9.9.9.2	4	30	4123	4128	22	0	0	2d14h	2

**r#show bgp ipv4 unicast summary**

BGP router identifier 1.1.1.1, local AS number 10  
BGP table version is 22, main routing table version 22  
5 network entries using 1240 bytes of memory  
7 path entries using 952 bytes of memory  
3/3 BGP path/bestpath attribute entries using 840 bytes of memory  
2 BGP AS-PATH entries using 48 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 3080 total bytes of memory  
BGP activity 17/7 prefixes, 36/22 paths, scan interval 60 secs

Neighbor	V	AS	MsgRcvd	MsgSent	TblVer	InQ	OutQ	Up/Down	State/PfxRcd
8.8.8.2	4	20	2718	2725	22	0	0	1d17h	2
9.9.9.2	4	30	4123	4128	22	0	0	2d14h	2

r#



```
r#show bgp ip?
```

```
ipv4  ipv6
```

```
r#show bgp ipv4 ?
```

```
  flowspec  Address Family modifier
```

```
  mdt       Address Family modifier
```

```
  multicast Address Family modifier
```

```
  mvpn      Address Family modifier
```

```
  unicast   Address Family modifier
```

```
r#show bgp vpn?
```

```
vpn4  vpn6
```



```
r#show ipv6 bgp summary?  
% Unrecognized command
```

```
r#show bgp ipv6 unicast summary
```

```
BGP router identifier 1.1.1.1, local AS number 10  
BGP table version is 5088, main routing table version 5088  
5 network entries using 1360 bytes of memory  
7 path entries using 1064 bytes of memory  
3/3 BGP path/bestpath attribute entries using 840 bytes of memory  
2 BGP AS-PATH entries using 48 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 3312 total bytes of memory  
BGP activity 17/7 prefixes, 36/22 paths, scan interval 60 secs
```

Neighbor	V	AS	MsgRcvd	MsgSent	TblVer	InQ	OutQ	Up/Down	State/PfxRcd
2008:DA6::2	4	20	16	20	5088	0	0	00:10:17	2
2009:DA6::2	4	30	2833	2808	5088	0	0	1d18h	2

```
r#
```

# BGP General Operation: Updates

- UPDATE message is how routes and other information is shared between peers
  - Efficiently packed for scale/Attributes
  - Contains NLRI, or Network Layer Reachability Info
- Once BGP sends a route to a peer, it assumes the peer will keep it. There is no periodic refresh
- New UPDATES are sent when
  - Best path change (Better or Worse)
  - Peer bounces
  - Route-Refresh

```

No.    Outer Source  Outer Destination  Protoc  Information
-----
13    30.3.30.30      30.3.30.3         BGP     OPEN Message, KEEPALIVE Message
14    30.3.30.30      30.3.30.3         BGP     KEEPALIVE Message
15    30.3.30.30      30.3.30.3         BGP     KEEPALIVE Message
16    30.3.30.30      30.3.30.3         BGP     UPDATE Message, UPDATE Message, UPDATE Message, UPDATE Message

```

▶ Frame 16: 250 bytes on wire (2000 bits), 250 bytes captured (2000 bits)  
▶ Ethernet II, Src: fa:16:3e:34:89:43 (fa:16:3e:34:89:43), Dst: fa:16:3e:22:35:cf (fa:16:3e:22:35:cf)  
▶ Internet Protocol Version 4, Src: 30.3.30.3, Dst: 30.3.30.3  
▶ Transmission Control Protocol, Src Port: 20576, Dst Port: 179, Seq: 96, Ack: 79, Len: 196  
▶ Border Gateway Protocol - UPDATE Message  
▼ Border Gateway Protocol - UPDATE Message  
 Marker: ffffffffffffffffffffffffffffffffff  
 Length: 66  
 Type: UPDATE Message (2)  
 Withdrawn Routes Length: 0  
 Total Path Attribute Length: 28  
 Path attributes  
 ▶ Path Attribute - ORIGIN: IGP  
 ▶ Path Attribute - AS\_PATH: 10 20 40  
 ▶ Path Attribute - NEXT\_HOP: 30.3.30.3  
 Network Layer Reachability Information (NLRI)  
 ▼ 40.1.1.0/24  
 NLRI prefix length: 24  
 NLRI prefix: 40.1.1.0  
 ▶ 40.40.40.0/24  
 ▶ 103.103.0.0/16  
 ▶ 103.103.40.0/24  
▶ Border Gateway Protocol - UPDATE Message  
▶ Border Gateway Protocol - UPDATE Message

# EBGP and IBGP

# BGP Peering

## Internal vs External

- Often confusing: IBGP and EBGP – not different protocols
- Different rules for communicating BETWEEN Autonomous Systems and WITHIN an Autonomous System
  - Trust
  - Complexity
- Depending on who we're talking to, some of the characteristics of the peering communication change by default.
- Further confusing, *Internal* BGP is not intended to be an *IGP*

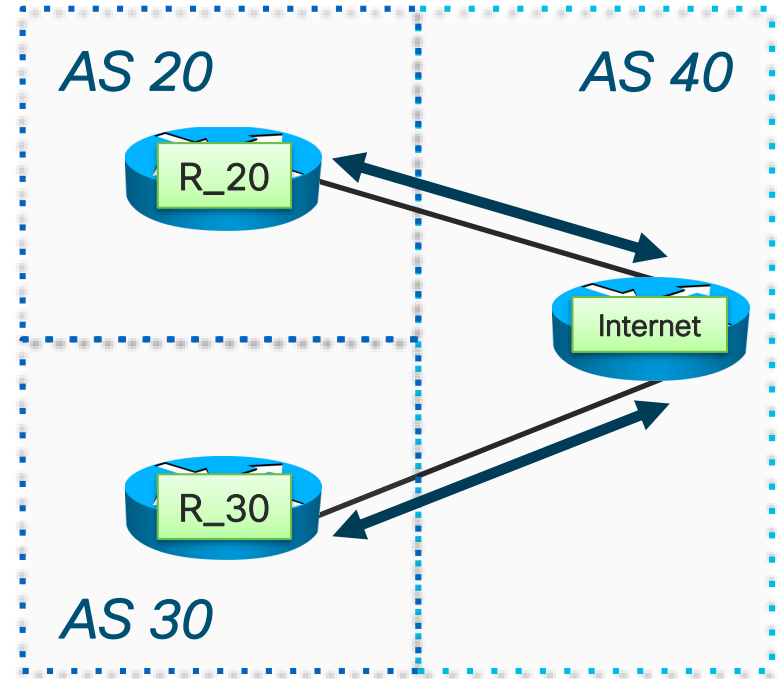


# EBGP Peering

# EBGP - External BGP

- BGP neighbor is in a different AS
- Usually directly connected
- NEXTHOP set to self (own ip)

	External eBGP
AS #s (Autonomous System Numbers)	Ours $\neq$ Peers
TTL (Time to Live)	1 (default)
Next Hop	Change
Directly Connected Check	Enabled (default)



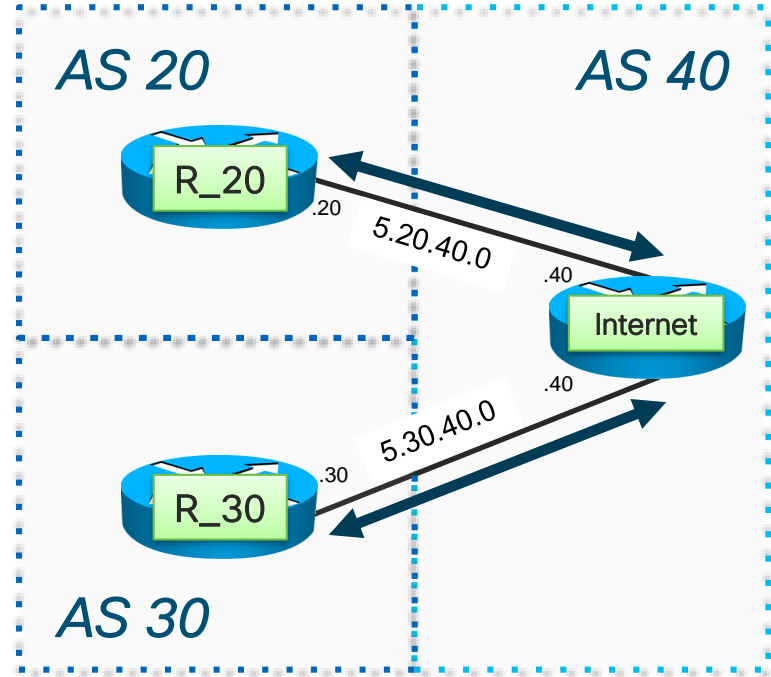
# EBGP - External BGP

## R\_20

```
router bgp 20
  bgp router-id 20.100.100.20
  neighbor 5.20.40.40 remote-as 40
  neighbor 5.20.40.40 send-community
```

## Internet

```
router bgp 40
  router-id 40.100.100.40
  neighbor 5.20.40.20 remote-as 20
  address-family ipv4 unicast
  send-community
```



```
Router_20#sh ip bgp summary
```

```
BGP router identifier 20.100.100.20, local AS number 20
```

```
<snip>
```

Neighbor	V	AS	MsgRcvd	MsgSent	TblVer	InQ	OutQ	Up/Down	State/PfxRcd
5.20.40.40	4	40	468	510	266	0	0	07:26:43	53

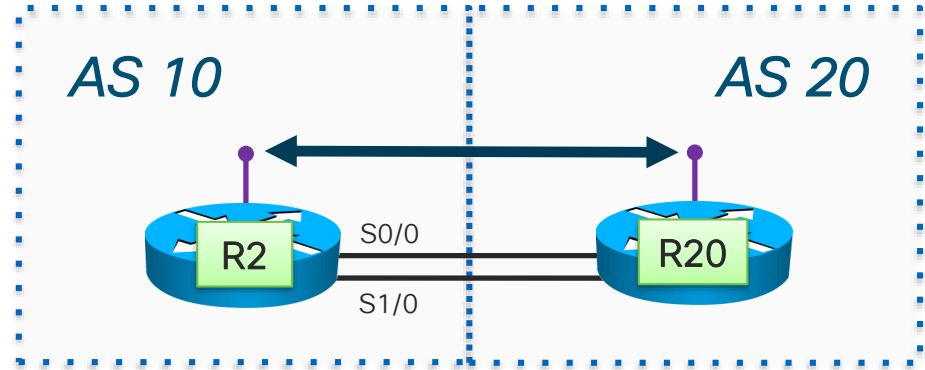
# EBGP Multihop

- Often used to load-balance traffic over multiple links
- Loopbacks typically used then for eBGP peering

## R\_2

```
router bgp 10
  neighbor 10.1.20.1 remote-as 20
  neighbor 10.1.20.1 update-source loop0
  neighbor 10.1.20.1 ebgp-multihop 2
```

```
ip route 10.1.20.1 255.255.255.255 s0/0
ip route 10.1.20.1 255.255.255.255 s1/0
```



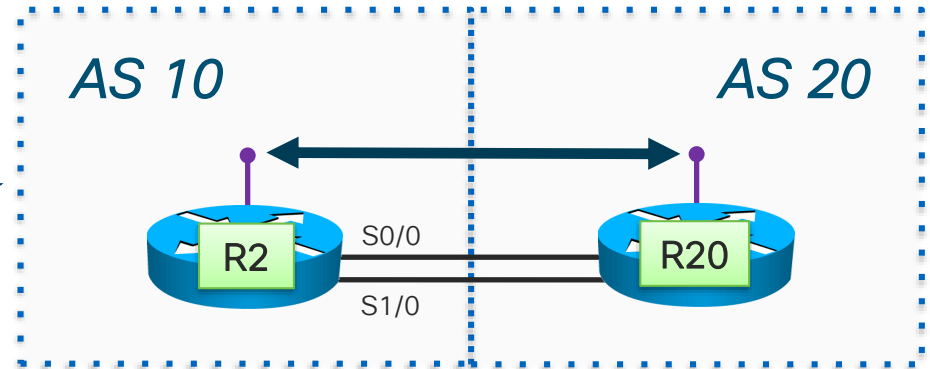
Question: Why set the TTL to 2 when it is just one hop away?

# EBGP Multihop & Disable-Connected-Check

## R\_2

```
router bgp 10
  neighbor 10.1.20.1 remote-as 20
  neighbor 10.1.20.1 update-source loop0
  neighbor 10.1.20.1 disable-connected-check
```

```
ip route 10.1.20.1 255.255.255.255 s0/0
ip route 10.1.20.1 255.255.255.255 s1/0
```



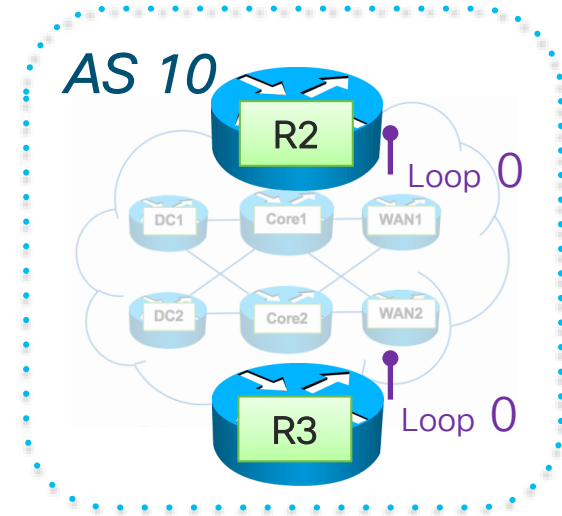
\*<http://www.networkingwithfish.com/clearing-up-some-misinformation-re-ebgp-multihop-and-ttl/>

# IBGP Peering

# IBGP - Internal BGP

- BGP Neighbor in same AS
- Usually neighbor over an IGP
- Peer to loopbacks
- NEXTHOP is unchanged

	Internal iBGP
AS #'s (Autonomous System Numbers)	Ours = Peers
TTL (Time to Live)	255
Next Hop	unchanged
Directly Connected Check	disabled



# IBGP – Loopback Peering

## Best Practice

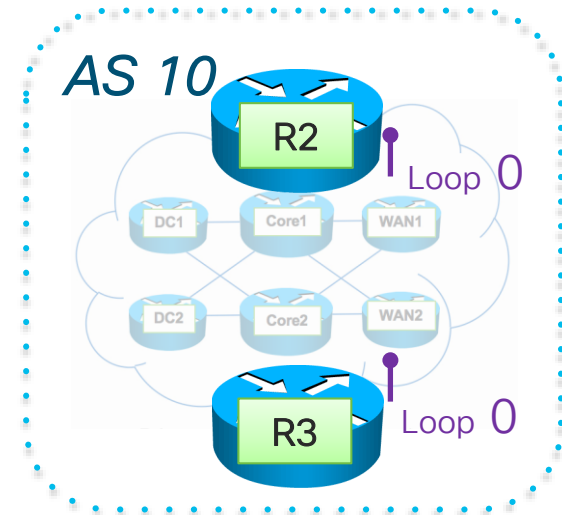
- ✓ Loopbacks should be /32s
- ✓ Have an IGP route to loopbacks

### R\_2

```
router bgp 10
  bgp router-id 10.100.100.2
  neighbor 10.100.100.3 remote-as 10
  neighbor 10.100.100.3 update-source Loopback0
  neighbor 10.100.100.3 next-hop-self
```

### R\_3

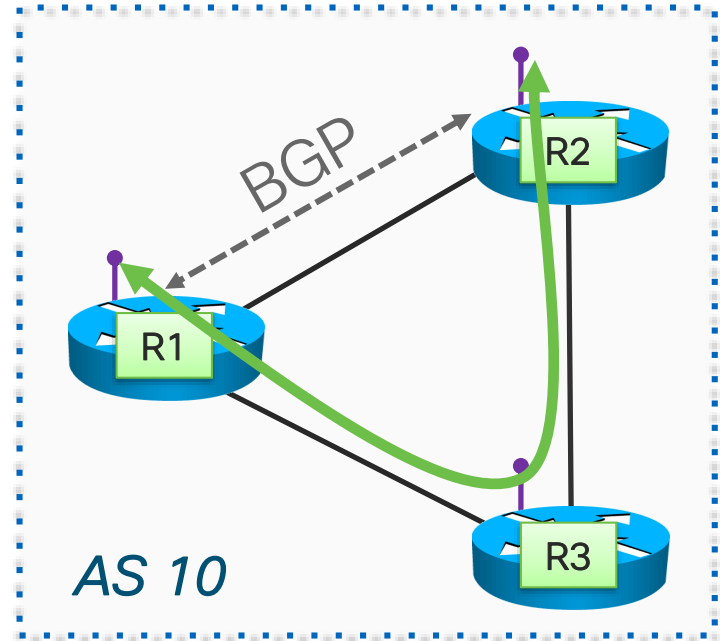
```
router bgp 10
  bgp router-id 10.100.100.3
  neighbor 10.100.100.2 remote-as 10
  neighbor 10.100.100.2 update-source Loopback0
  neighbor 10.100.100.2 next-hop-self
```





# IBGP – Loopback Peering

- ✓ Loopback peering promotes stability
  1. R1 & R2 are BGP peers
  2. The physical link connecting them *fails*
- If the R1-R2 BGP Peer is tied to the IP addresses of the physical link
  - ✗ BGP peer would fail
  - ✗ Churn would occur
- If the R1-R2 BGP Peer was between loopbacks
  - ✓ As long as the 2 loopbacks can reach each other through R3 the BGP peer will stay up
  - ✓ Let the IGP do its job of quick convergence



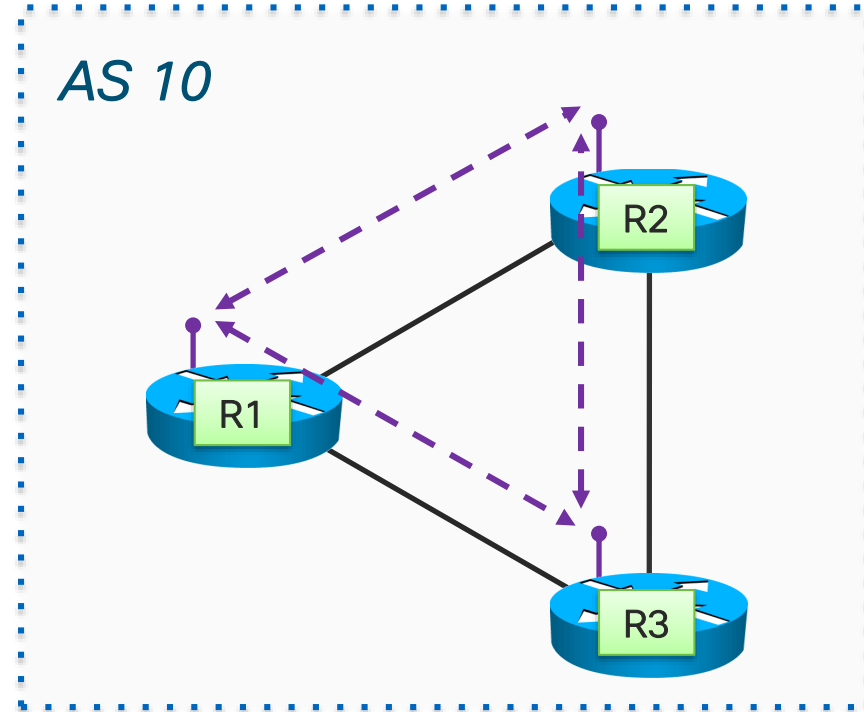
# IBGP – Internal BGP “Rule”

**Cannot** advertise route

**received** from one IBGP peer

**to** another IBGP peer

- Implications:
  - Full IBGP mesh is required
  - $n*(n-1)/2$  peering mesh – scaling problem!
  - Route-Reflectors relax this constraint



# Attributes and Best Path Selection Algorithm

# Agenda

- BGP General Operation
  - Overview
  - EBGP and IBGP
- Attributes and Best Path Selection Algorithm
  - Route Origination
  - AS-PATH
  - NEXTHOP
  - Communities
- Controlling Traffic
  - Controlling Outbound Traffic
  - BGP Multipath
  - Controlling Inbound Traffic
- Route Reflectors
- Multiprotocol BGP
- Common BGP Deployments
- Securing BGP
- BGP Routing Convergence
- Show and Tell/Demo Lab

# BGP Route vs. BGP Path

- BGP can have multiple paths per route
- Here we have 2 paths to the 40.100.101.0/24 prefix

```
R2#show ip bgp 40.101.101.0
```

```
BGP routing table entry for 40.101.101.0/24
```

```
Paths: (2 available, best #1, table default)
```

```
Advertised to update-groups:
```

```
1
```

```
Refresh Epoch 1
```

```
20 40
```

```
20.2.20.20 from 20.2.20.20 (20.100.100.20)
```

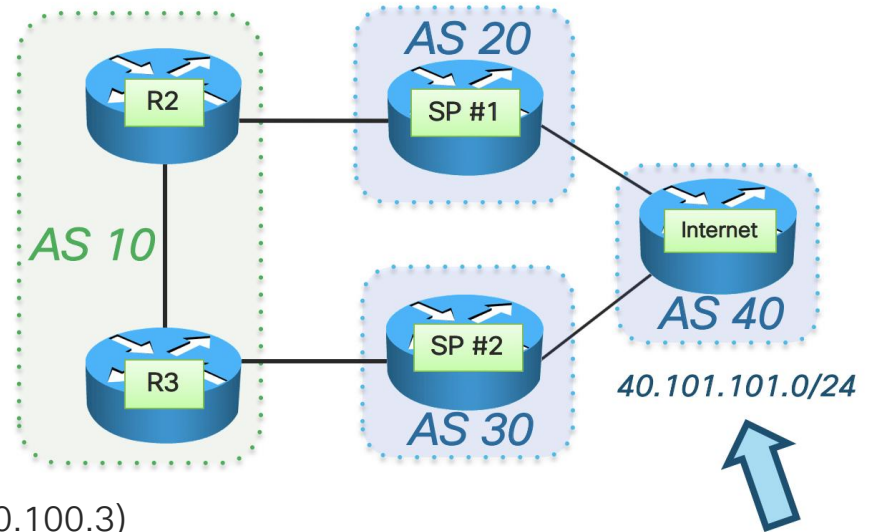
```
Origin IGP, localpref 100, valid, external, best  
rx pathid: 0, tx pathid: 0x0
```

```
Refresh Epoch 1
```

```
30 40
```

```
10.100.100.3 (metric 2) from 10.100.100.3 (10.100.100.3)
```

```
Origin IGP, metric 0, localpref 100, valid, internal  
rx pathid: 0, tx pathid: 0
```



# BGP Route vs. BGP Path

- “show ip bgp summary” provides the total number of routes and paths
- Paths and routes both consume memory
- The more paths you have per route, the more memory consumed

```
R3#sh ip bgp summary
```

```
BGP router identifier 10.100.100.3, local AS number 10
```

```
BGP table version is 268, main routing table version 268
```

```
62 network entries using 15376 bytes of memory
```

```
119 path entries using 16184 bytes of memory
```

```
21/12 BGP path/bestpath attribute entries using 5880 bytes of memory
```

```
31 BGP AS-PATH entries using 2464 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 39904 total bytes of memory
```

```
BGP activity 243/125 prefixes, 481/254 paths, scan interval 60 secs
```

# Attributes

## IGP

- Primary attribute is a cost/metric
- The path with the lowest metric is the best...nice and easy

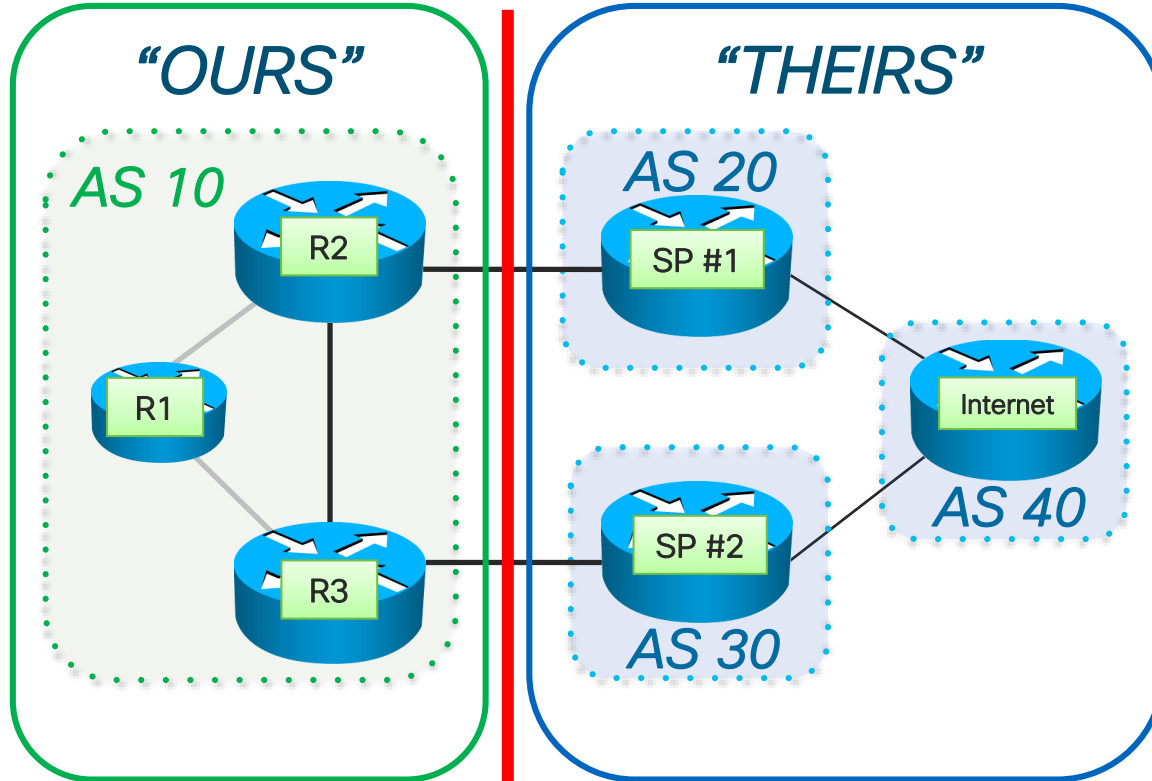
$$x < y$$

## BGP

- Routing Policy between AS is usually more complex
- Shortest path is not necessarily the best one
- Has many attributes to describe reachability to a destination
- The “Best Path Algorithm” compares attributes between different paths to select the best
- Route-policies are used tweak attributes to influence outcome of Best Path: routing



# Attributes





# BGP Path Selection Algorithm

	Attribute	Logic
1	Weight	Higher is better. Local to the router...not really an attribute.
2	Local Preference	Local to an AS...higher is better
3	Locally Originated	Corner case..."network 10.0.0.0" vs. "aggregate 10.0.0.0" vs. "redistribute" on the same router
4	AS-PATH	Shorter AS-PATH is better
5	ORIGIN	IGP < EGP < Incomplete
6	MED	Is often a reflection of IGP metrics so lower is better
7	EBGP vs. IBGP	Prefer EBGP path over IBGP path

# BGP Path Selection Algorithm (cont'd)

	Attribute	Logic
8	IGP cost to NEXTHOP	Lower is better
9	Lowest Router ID	Lower is better
10	Shortest CLUSTER_LIST	Lower is better
11	Lowest neighbor IP address	Lower is better

All details at <http://www.cisco.com/c/en/us/support/docs/ip/border-gateway-protocol-bgp/13753-25.html>

# BGP Path Selection Algorithm

BGP Attribute	“Fish”ism
Weight	Wise
Local Preference	Lip
Locally Originated	Lovers
AS-PATH	Apply
ORIGIN	Oral
MED	Medication
EBGP vs. IBGP	Every
NEXTHOP IGP Cost	Night



# BGP Path Selection Algorithm



## N WLLA OMNI – Wendell Odom

### *N WLLA OMNI*

- N *Next hop reachability*
- W *Weight, bigger is better*
- L *Local preference, bigger is better*
- L *Locally injected preferred over BGP learned*
- A *AS path length, shorter is better*
  
- O *Origin, (iGP is better than eGP is better than incomplete)*
- M *MED, lower is better*
- N *Neighbor type, EBGP better than IBGP*
- I *IGP metric to BGP next-hop, lower is better*

# Attributes and Best Path

```
R2#show ip bgp 40.101.101.0
BGP routing table entry for 40.101.101.0/24
Paths: (2 available, best #1, table default)
  Advertised to update-groups:
```

1

Refresh Epoch 1

20 40 ← as-path

20.2.20.20 from 20.2.20.20 (20.100.100.20)

Origin IGP, localpref 100, valid, external, best

rx pathid: 0, tx pathid: 0x0

Refresh Epoch 1

30 40 ← as-path

10.100.100.3 (metric 2) from 10.100.100.3 (10.100.100.3)

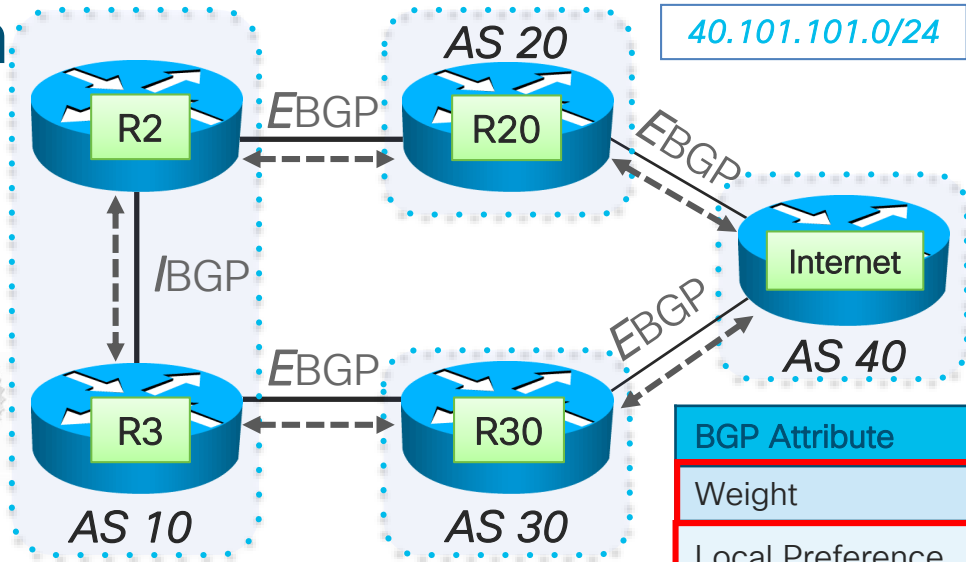
Origin IGP, metric 0, localpref 100, valid, internal

rx pathid: 0, tx pathid: 0



Externally learned (eBGP)

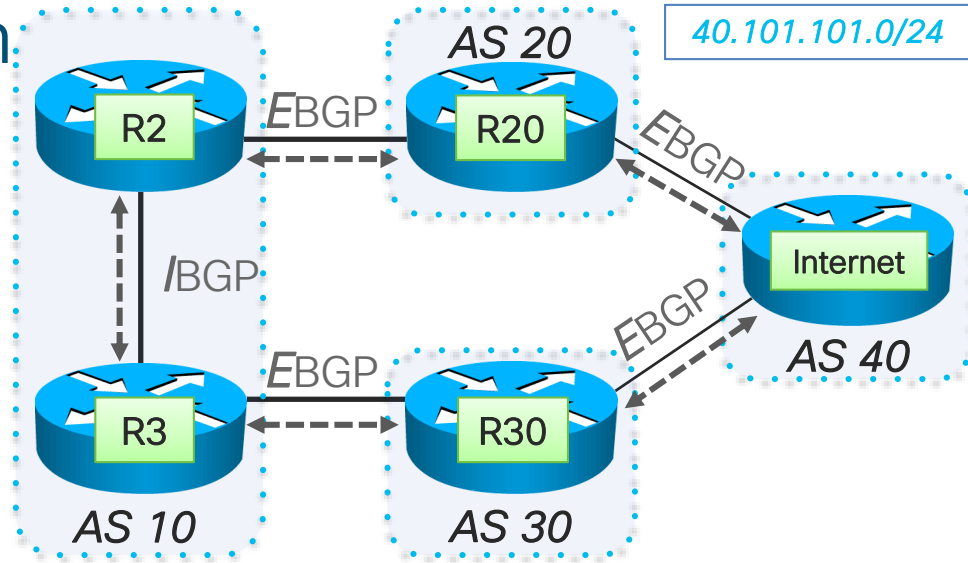
Internally learned (iBGP)



40.101.101.0/24

BGP Attribute
Weight
Local Preference
Locally Originated
AS-PATH
ORIGIN
MED
EBGP vs. IBGP
NEXTHOP IGP Cost

# Attributes and Best Path



```
R2#sh ip bgp
```

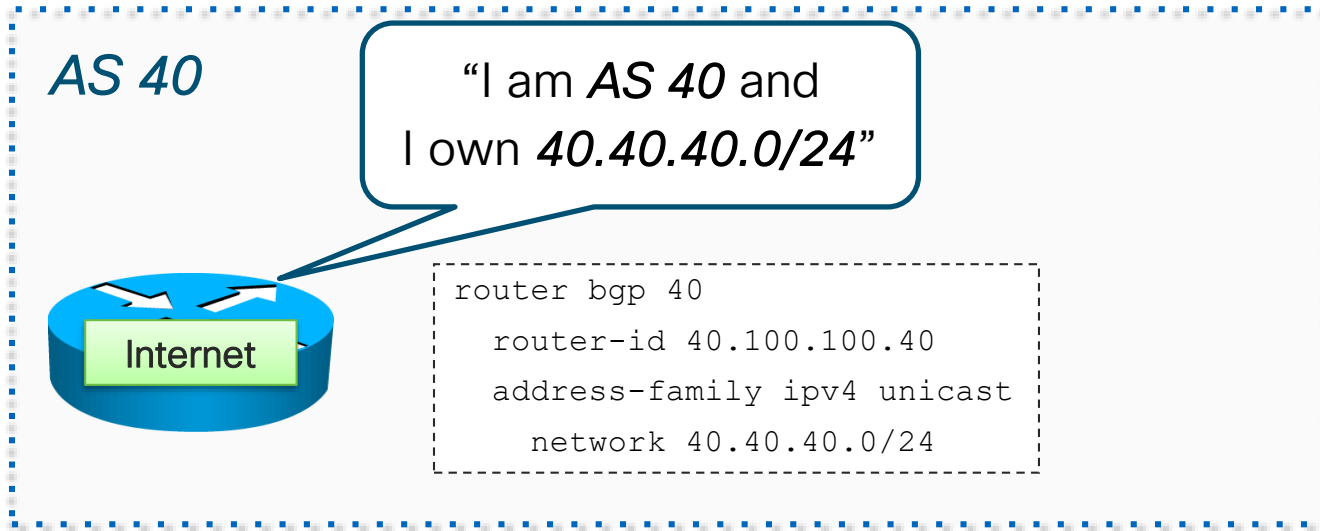
```
BGP table version is 320, local router ID is 10.100.100.2 Status codes: s suppressed,
d damped, h history, * valid, > best, i - internal, r RIB-failure, S Stale, m multipath, (...)
Origin codes: i - IGP, e - EGP, ? - incomplete (...)
```

Network	Next Hop	Metric	LocPrf	Weight	Path
*> 40.101.101.0/24	20.2.20.20			0	20 40 i
* i	10.100.100.3	0	100	0	30 40 i



# Route Origination

# Route Origination



- An AS “originates” routes in their address space
- Three ways to originate a route...



# Route Origination – Network Statements

```
router bgp 40
  router-id 40.100.100.40
  address-family ipv4 unicast
    network 40.40.40.0/24 ←
ip route 40.40.40.0 255.255.255.0 Null0 250 ←
```

- **NOT like OSPF or EIGRP network statement!!**
- Easiest and Cleanest method to determine/control what you are originating
- Network 40.40.40.0 mask 255.255.255.0
  - Originates 40.40.40.0/24 ←
  - Requires 40.40.40.0/24 to be in the RIB
  - Floating static route to Null0 is common ←

# Route Origination – Network Statements

```
Internet# show ip bgp 40.40.40.0
BGP routing table information for VRF default, address family IPv4
Unicast
BGP routing table entry for 40.40.40.0/24, version 27
Paths: (1 available, best #1)
Flags: (0x080002) on xmit-list, is not in urib
  Advertised path-id 1
  Path type: local, path is valid, is best path
  AS-Path: NONE, path locally originated
  0.0.0.0 (metric 0) from 0.0.0.0 (40.100.100.40)
  Origin IGP, MED not set, localpref 100, weight 32768
  Path-id 1 advertised to peers:
    5.20.40.20          5.30.40.30
Internet#
```

- The Origin is IGP
- Weight is 32768
- “0.0.0.0 from 0.0.0.0”

# Route Origination – Redistribution

- Routes can be redistributed into BGP
- Pros
  - Easy to configure and setup
- Cons
  - IGP instability is passed along to BGP
  - It is not always obvious what routes you are originating
  - “Redistribute static” is especially dangerous
    - What if someone configures a static route for Google’s address space?
    - You could blackhole Google’s traffic

*Use route-maps to control what you redistribute!!*

# Route Origination – Redistribution

- NEXTHOP uses the IGP nexthop of 10.1.1.14
- ORIGIN is set to “Incomplete”
- “metric” here means MED
  - Uses the IGP metric of 11
- Weight is 32768

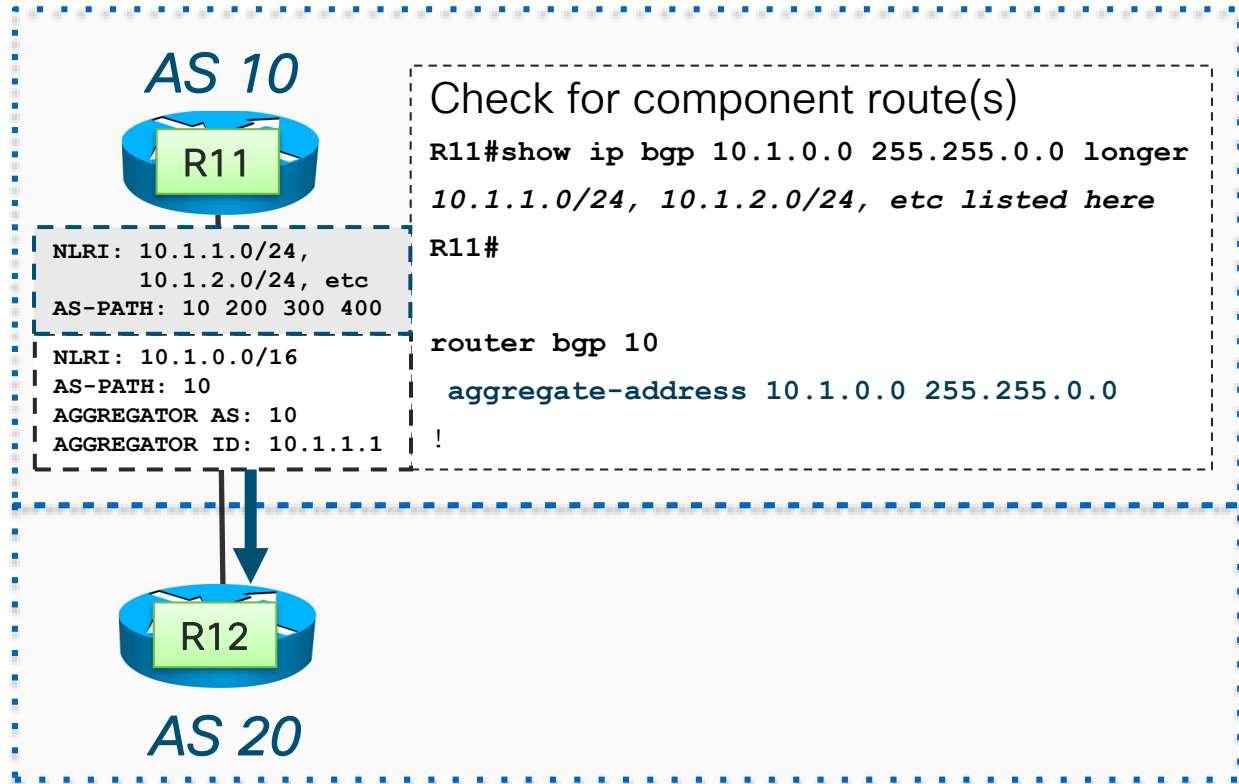
```
R10#show ip bgp 10.1.1.3
BGP routing table entry for 10.1.1.3/32, version 5
Paths: (1 available, best #1, table default)
  Advertised to update-groups:
    9
  Local
    10.1.1.14 from 0.0.0.0 (10.1.1.2)
      Origin incomplete, metric 11, localpref 100, weight 32768, valid,
sourced, best
```

# Route Origination – Aggregation

- Typically used by ISPs to summarize their address space
- Reduces number of routes in global BGP table
- Adds AGGREGATOR attribute
  - Contains Router-ID and AS of the router that did the aggregation
  - Used for troubleshooting

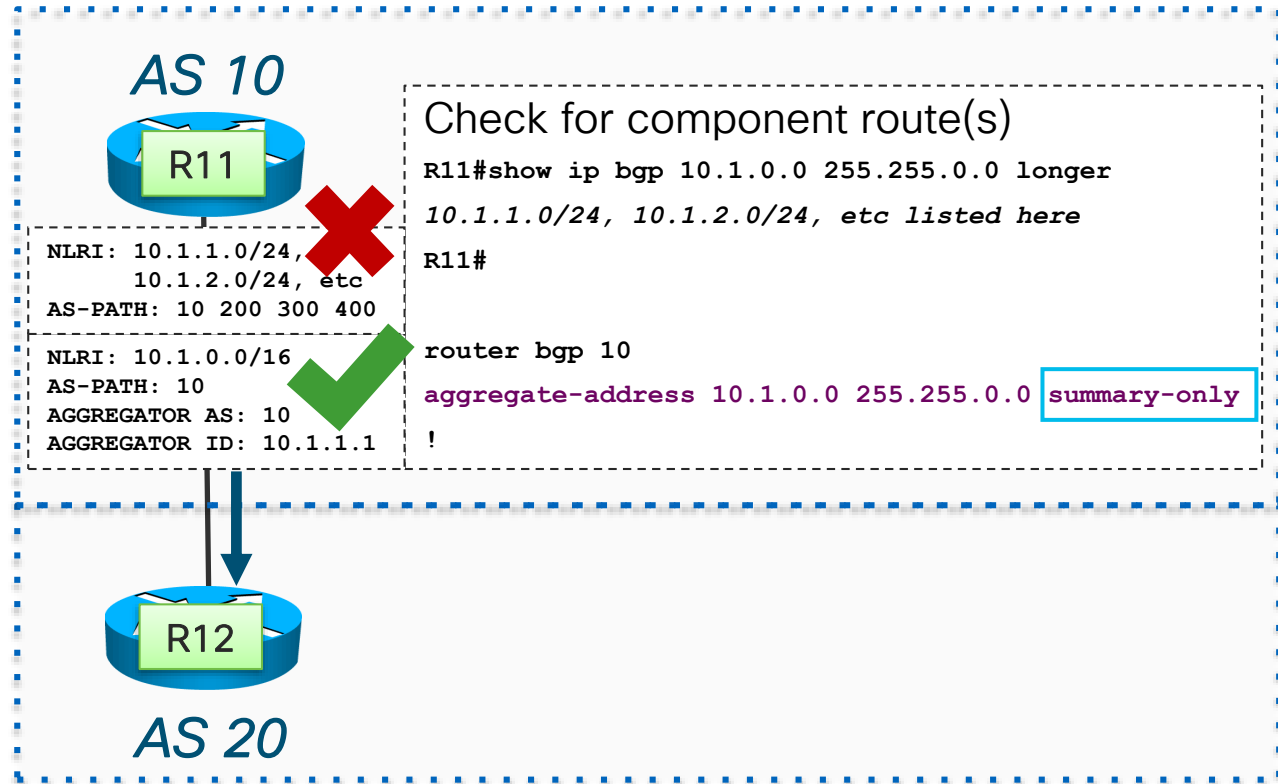
# Route Origination – Aggregation

- Configure an “aggregate-address” statement
- BGP table must have component route(s)
  - Components are the longer length prefixes that fall within the aggregate’s range
  - Use “show ip bgp x.x.x.x y.y.y.y longer” to check for components
- Component routes are still advertised



# Route Origination – Aggregation

- Adding the “*summary-only*” keyword
  - causes BGP to suppress the components of the aggregate



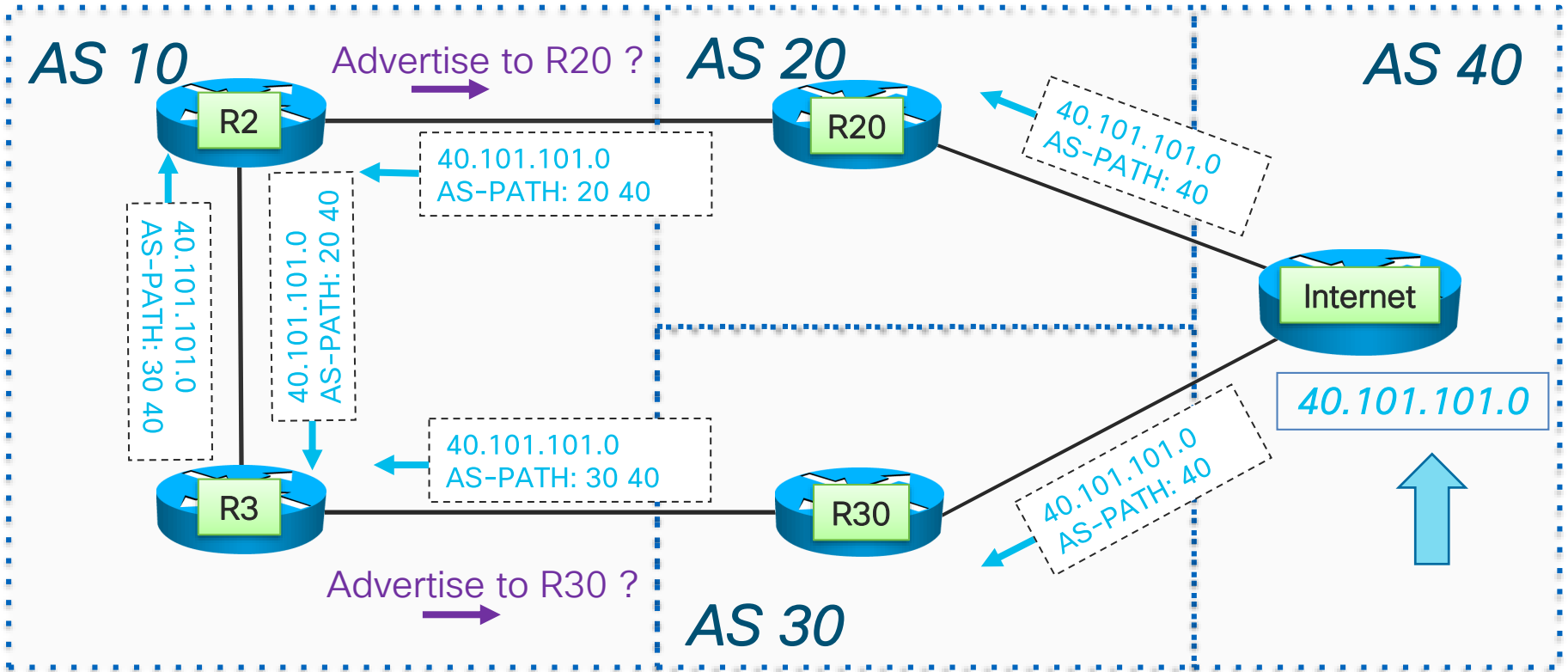
# AS-PATH



# AS-PATH

- The AS-PATH tells the story of which AS a route has traversed
- AS-Path is used for loop detection on the border of the AS
  - BGP drops an external update if it sees its own AS in the path
- BGP prepends his own AS# to the AS-PATH when advertising to an EBGP peer
- When viewing the AS-PATH, the most recent AS is on the left, the originating AS is on the far right
- Shortest AS-PATH is often the tie-breaker for best path selection

# AS-PATH



# AS-PATH

```
R2#show ip bgp 40.101.101.0
```

BGP routing table entry for 40.101.101.0/24

Paths: (2 available, best #1, table default)

Advertised to update-groups:

1

Refresh Epoch 1

20 40

20.2.20.20 from 20.2.20.20 (20.100.100.20)

Origin IGP, localpref 100, valid, **external**, best

rx pathid: 0, tx pathid: 0x0

Refresh Epoch 1

30 40

10.100.100.3 (metric 2) from 10.100.100.3 (10.100.100.3)

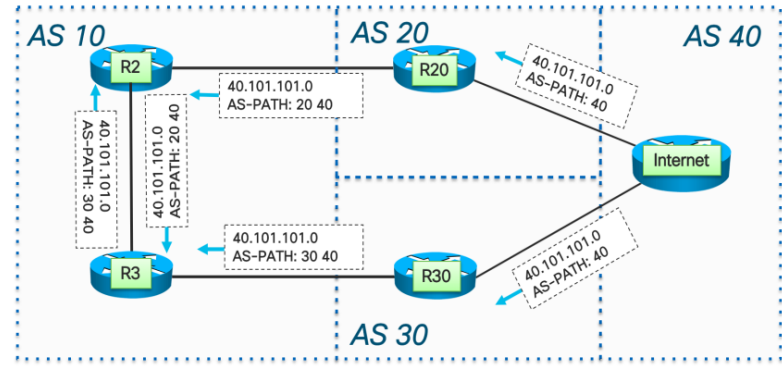
Origin IGP, metric 0, localpref 100, valid, **internal**

rx pathid: 0, tx pathid: 0



Externally learned (EBGP)

Internally learned (IBGP)



BGP Attribute
Weight
Local Preference
Locally Originated
AS-PATH
ORIGIN
MED
EBGP vs. IBGP
NEXTHOP IGP Cost

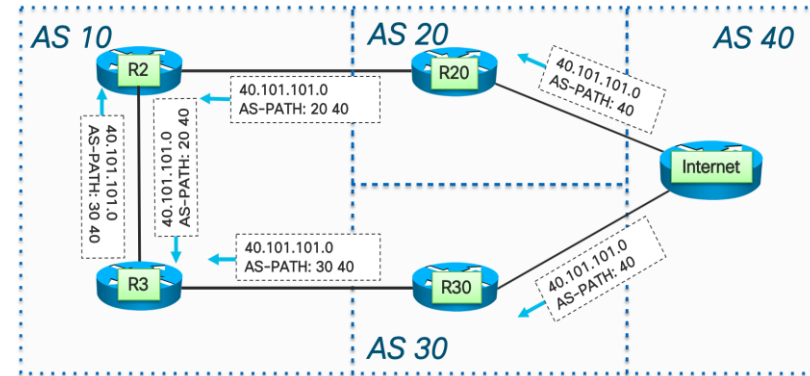
# AS-PATH

Q: Should R2 advertise 40.101.101.0 to R20? **NO**

Q: Should R3 advertise 40.101.101.0 to R30? **NO**

BGP picks the best path

- Installs in the routing/forwarding table
- Advertises to BGP peers via UPDATES



R2 picked R20 (externally learned over internal) as the best path to 40.101.101.0

R3 picked R30 (externally learned over internal) as the best path to 40.101.101.0

# NEXTHOP

# NEXTHOP

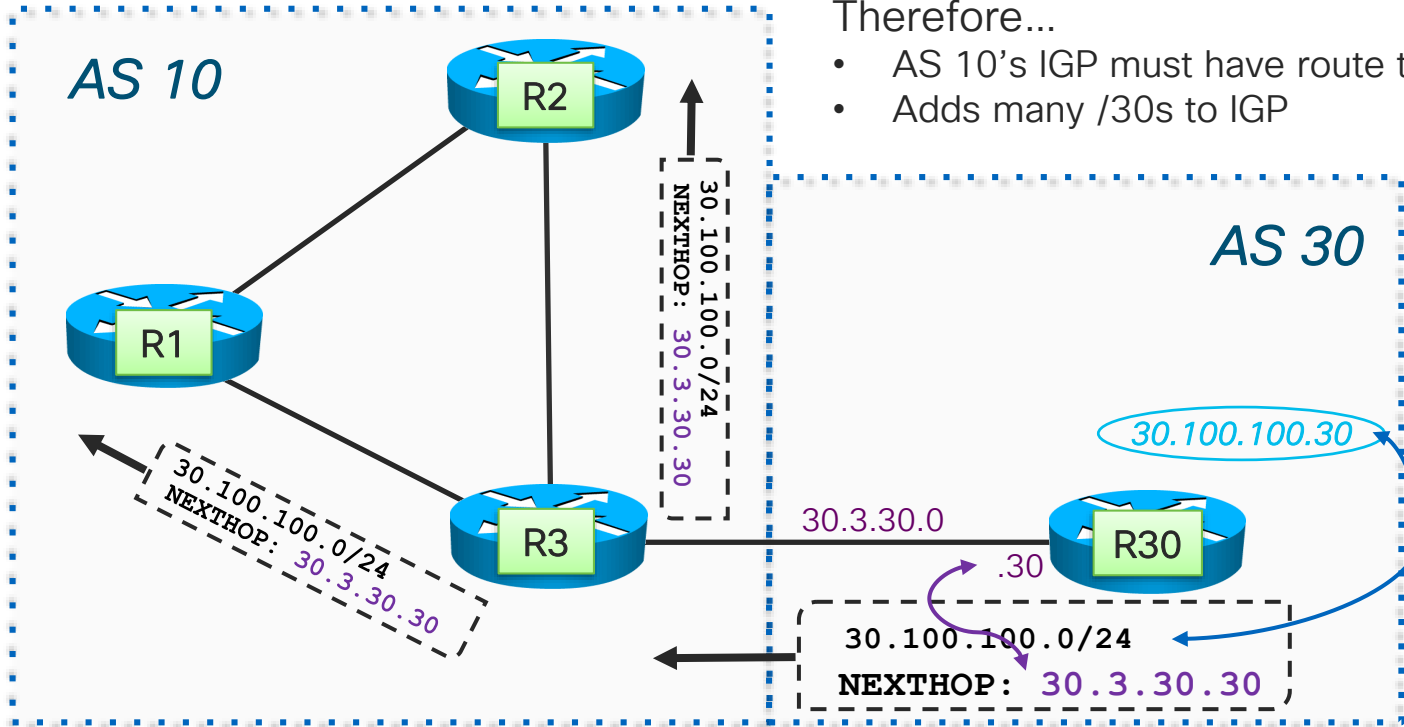
- NEXTHOP is the address that we must route towards in order to reach the BGP prefix
  - Paths where the next-hop is unreachable are not considered for best-path calculation
- EBGP does “next-hop-self” automatically
  - Multiple EBGP peers on the same subnet is an exception
- IBGP does not modify the NEXTHOP by default
  - NEXTHOP will remain as the IP of the EBGP peer
  - Forces BGP speakers in an AS to have routes for the EBGP facing links
  - Would need to carry many /30 eBGP facing links in our IGP
  - Best practice is to use “next-hop-self” to avoid this

# IBGP *without* next-hop-self

IBGP default – *leave Next Hop Alone*

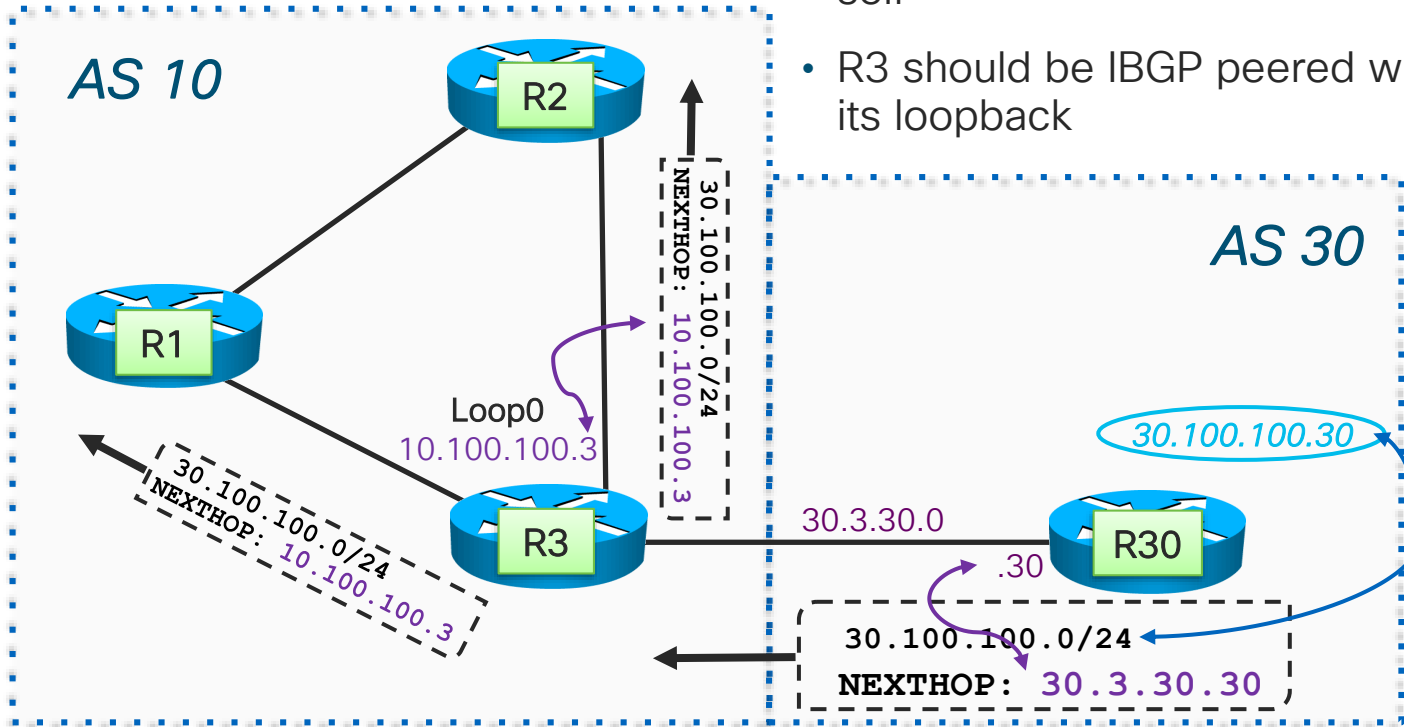
Therefore...

- AS 10's IGP must have route to 30.3.30.30
- Adds many /30s to IGP



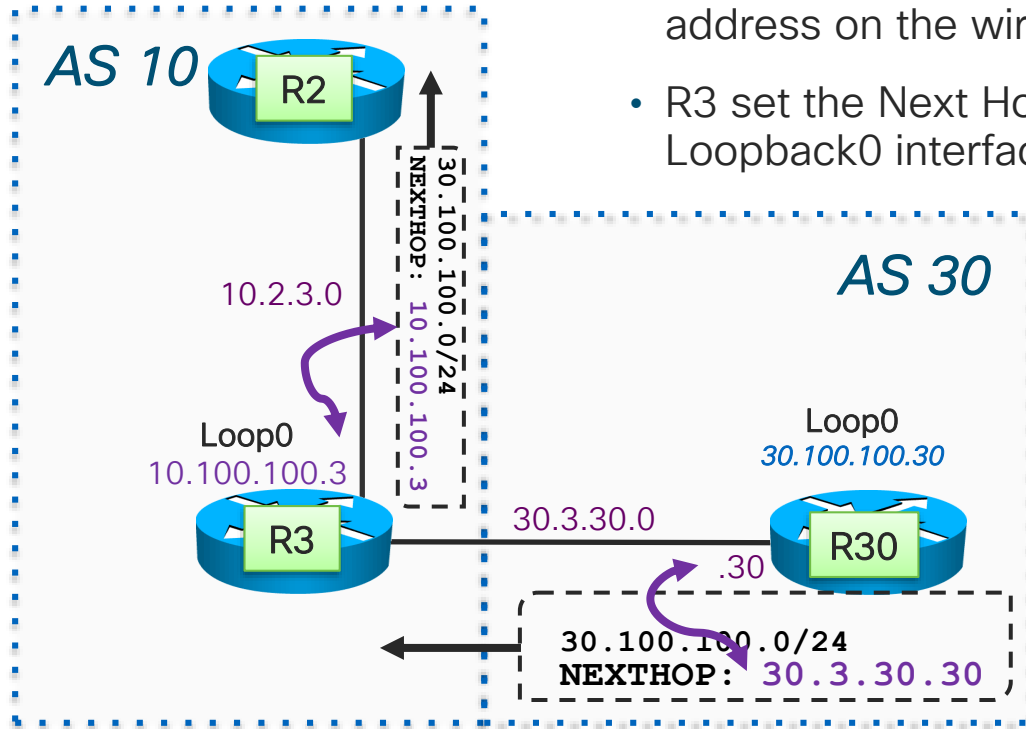
# IBGP *with* next-hop-self

- R3's BGP with R1 and R2 is set with "next hop self"
- R3 should be IBGP peered with R1 and R2 via its loopback





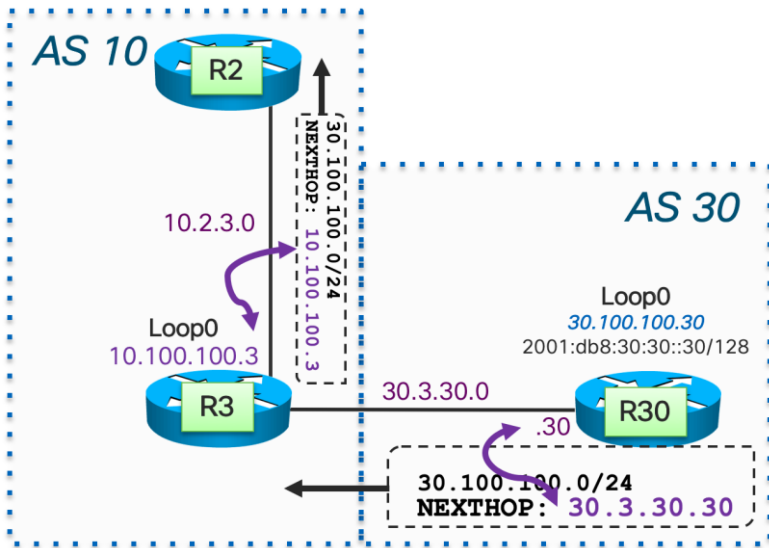
# Question: Which IP Address to set as “Self”



??  
Why  
?

- R30 set Next Hop address to 30.3.30.30 – its IP address on the wire connecting to R3
- R3 set the Next Hop address to 10.100.100.3 – its Loopback0 interface IP address

# Question: Which IP Address to set as “Self”



- R30 sets NextHop to 30.3.30.30 when advertising to R3
- R3 sets NextHop to 10.100.100.3 when advertising to R2

“Self” in relationship to setting Next Hop is the IP address used for the BGP peering with that neighbor.

```
R3#sh tcp brief | include 179
```

```
7FCB80694E40 2001:DB8:3:3::3.179 2001:DB8:2:2::2.44437 ESTAB
7FCB8068F0C8 10.100.100.3.179 iBGP w/ R2 10.100.100.2.15078 ESTAB
7FCB7FA26460 30.3.30.3.179 eBGP w/ R30 30.3.30.30.51769 ESTAB
7FCB7FA25A40 2001:DB8:3:30::3.179 2001:DB8:3:30::30.41069 ESTAB
```

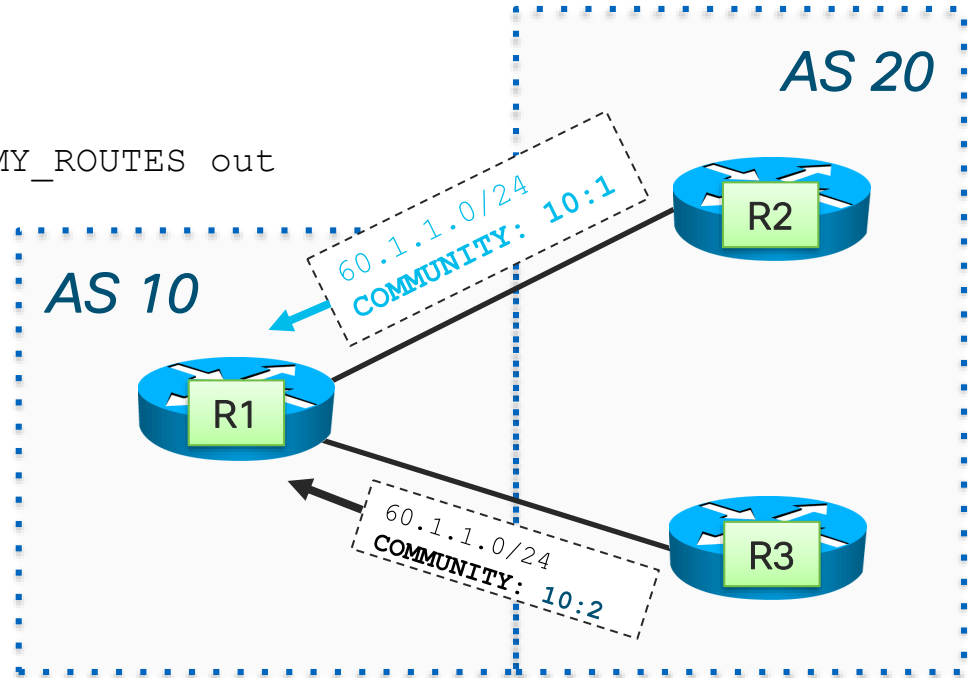
# Communities

# Communities

- A COMMUNITY is an attribute that stores a number
  - 4-byte number that is usually displayed in X:Y notation
  - “ip bgp-community new-format” triggers X:Y notation
- A community by itself does nothing
  - Tagging a prefix with 100:1 or 100:2 will not change routing in any way
- Set communities via a route-map
- Communities are not advertised by default

# Sending Communities

```
R2#
router bgp 20
  neighbor 10.1.1.2 remote-as 10
  neighbor 10.1.1.2 send-community
  neighbor 10.1.1.2 route-map TAG_MY_ROUTES out
!
ip bgp-community new-format
!
route-map TAG_MY_ROUTES permit 10
  set community 10:1
!
R2#
```



# Receiving Communities

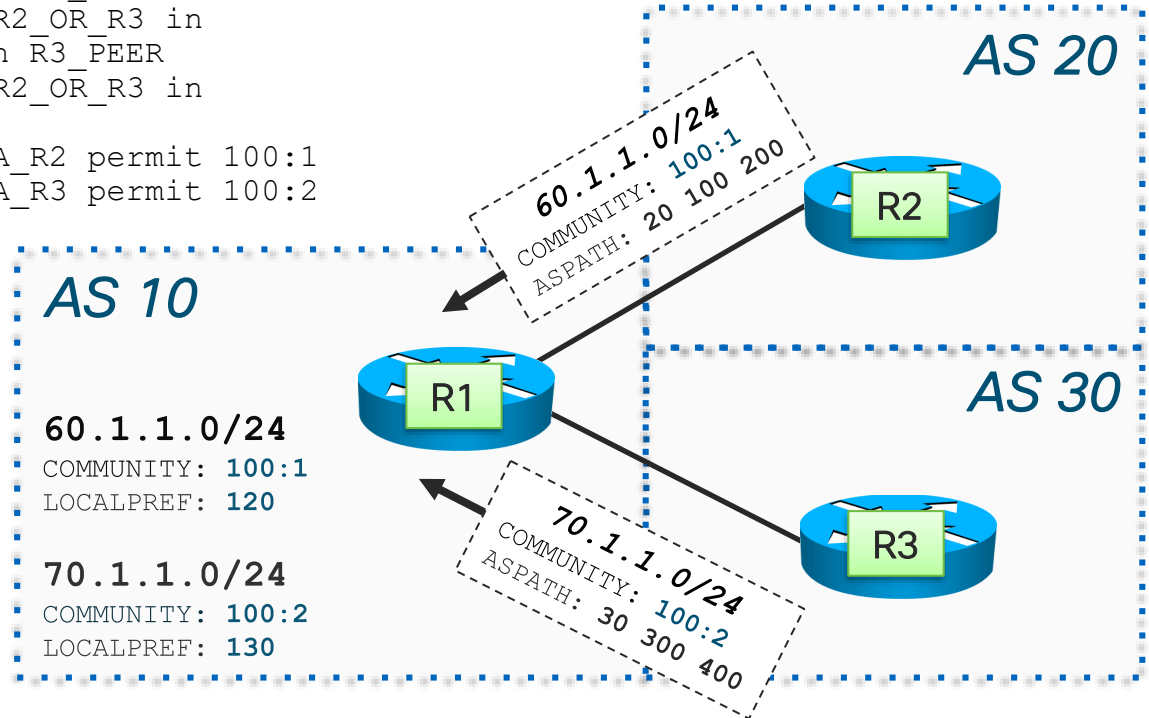
- Applying Policy towards communities does impact routing
- Use route-maps and community-list to
  - Match against a certain community
  - Modify a BGP attribute as a result
    - LOCALPREF, ASPATH prepending, etc
- You can impact 1000s of prefixes by applying policy based on a single community

# Communities

```
R1#
router bgp 10
  neighbor 20.1.1.1 description R2_PEER
  neighbor 20.1.1.1 route-map R2_OR_R3 in
  neighbor 30.1.1.1 description R3_PEER
  neighbor 30.1.1.1 route-map R2_OR_R3 in

ip community-list standard VIA_R2 permit 100:1
ip community-list standard VIA_R3 permit 100:2
```

```
route-map R2_OR_R3 permit 10
  match community VIA_R2
  set local-preference 120
route-map R2_OR_R3 permit 20
  match community VIA_R3
  set local-preference 130
route-map R2_OR_R3 permit 30
```



# Well Known Communities

- “A community by itself does nothing”
  - There are exceptions to every rule 😊
- Well Known Communities do have an automatic impact

Community	Impact
<i>local-AS</i>	<i>Do not send to EBGp peers</i>
<i>no-advertise</i>	<i>Do not advertise to any peer</i>
<i>no-export</i>	<i>Do not export outside AS/confed</i>



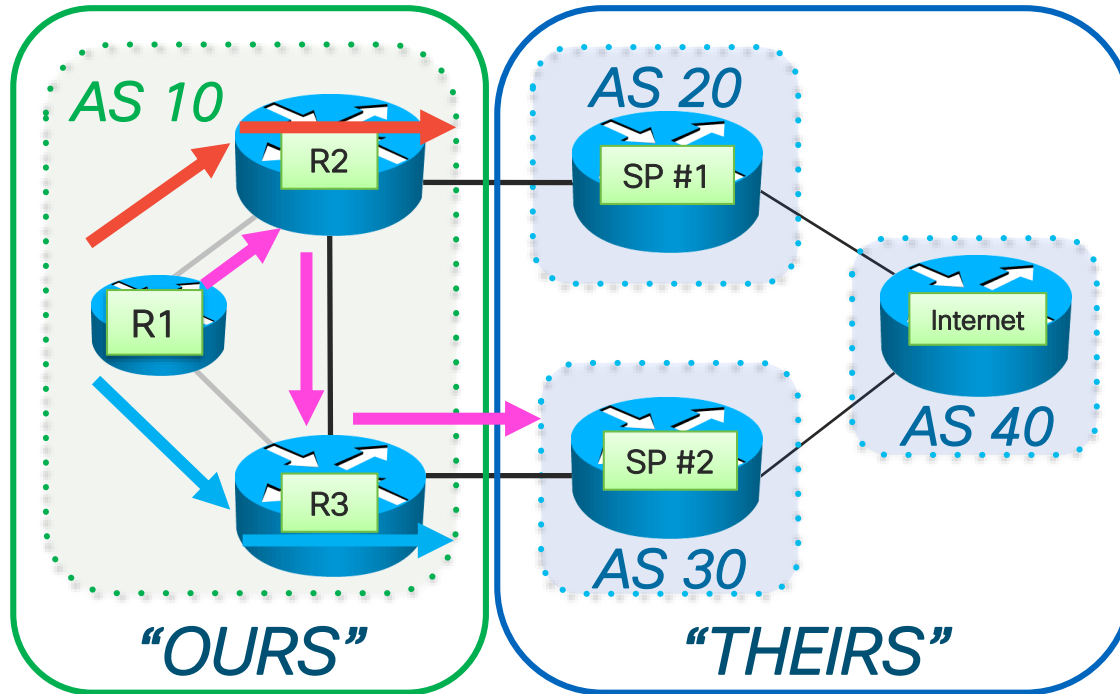
# Controlling Traffic

# Agenda

- BGP General Operation
  - Overview
  - EBGP and IBGP
- Attributes and Best Path Selection Algorithm
  - Route Origination
  - AS-PATH
  - NEXTHOP
  - Communities
- Controlling Traffic
  - Controlling Outbound Traffic
  - BGP Multipath
  - Controlling Inbound Traffic
- Route Reflectors
- Multiprotocol BGP
- Common BGP Deployments
- Securing BGP
- BGP Routing Convergence
- Show and Tell/Demo Lab

# Controlling Outbound Traffic

# Controlling Outbound Traffic



BGP Attribute
Weight
Local Preference
Locally Originated
AS-PATH
ORIGIN
MED
EBGP vs. IBGP
NEXTHOP IGP Cost

# Local Preference

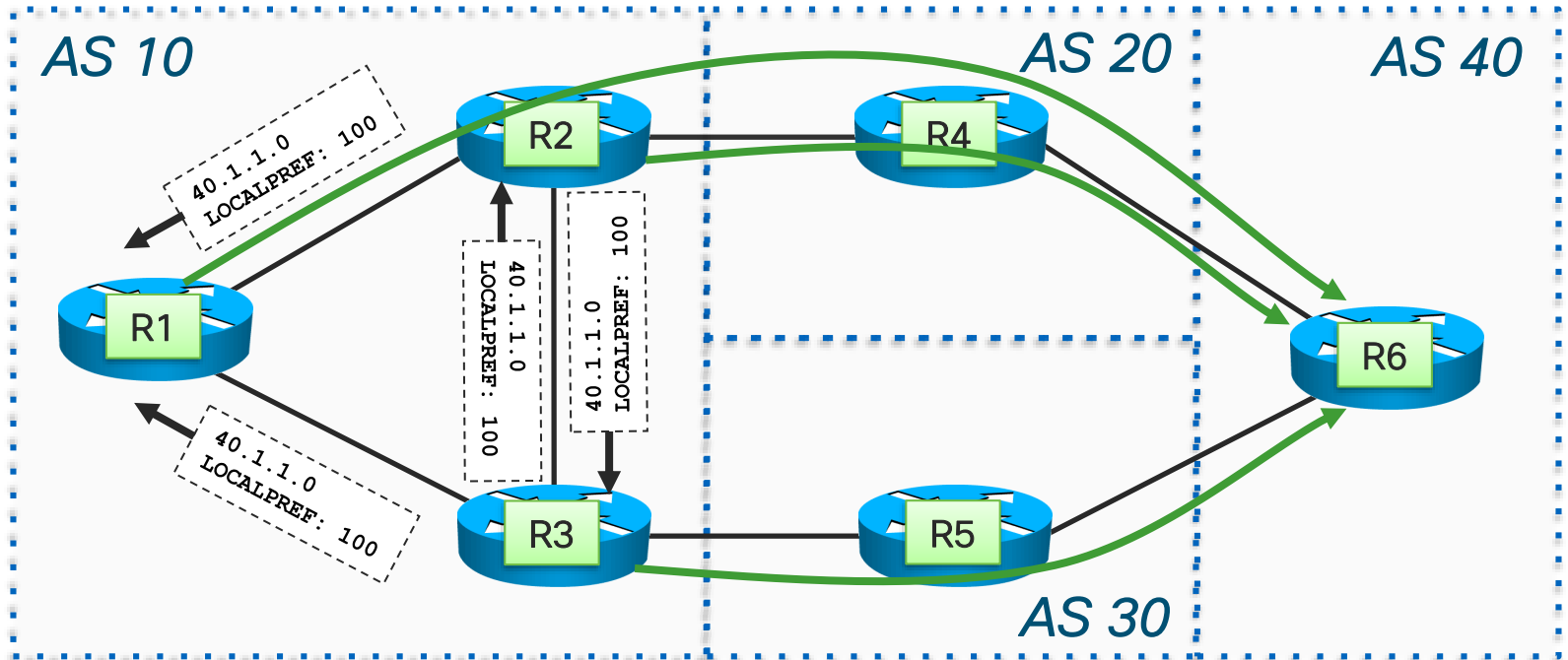
## *Local* to an AS

- Local preference is never transmitted to an eBGP peer
  - A default LP of 100 is applied to routes from eBGP peers
- 
- A common attribute used to influence outbound traffic
  - Higher LOCAL\_PREF is preferred
  - Is compared very early in the Best Path Algorithm
  - Think of it as indicating which exit from your AS is preferred.

BGP Attribute
Weight
Local Preference
Locally Originated
AS-PATH
ORIGIN
MED
EBGP vs. IBGP
NEXTHOP IGP Cost

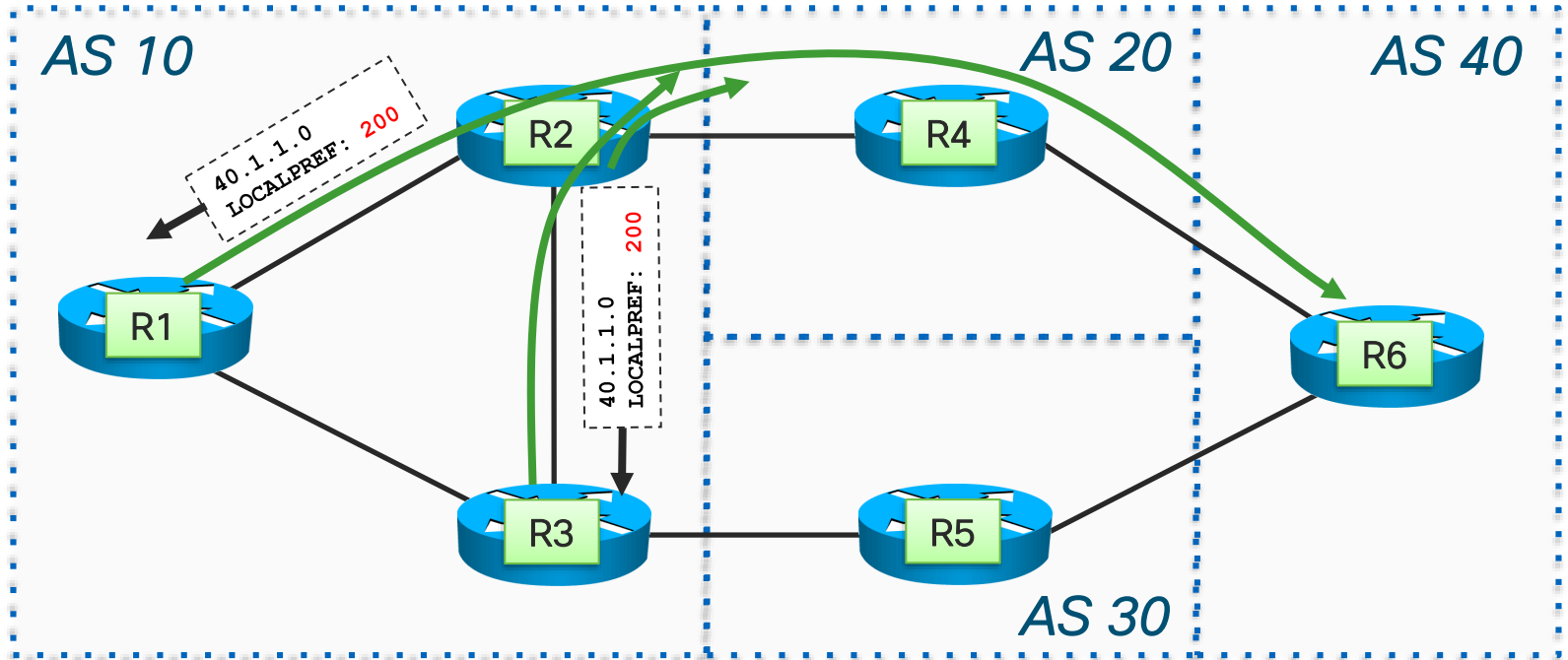
# Local Preference

- Default behavior: LOCALPREF 100
- R2 and R3 prefer EBGP path
- R1 prefers path from R2 over R3 (lower neighbor IP)

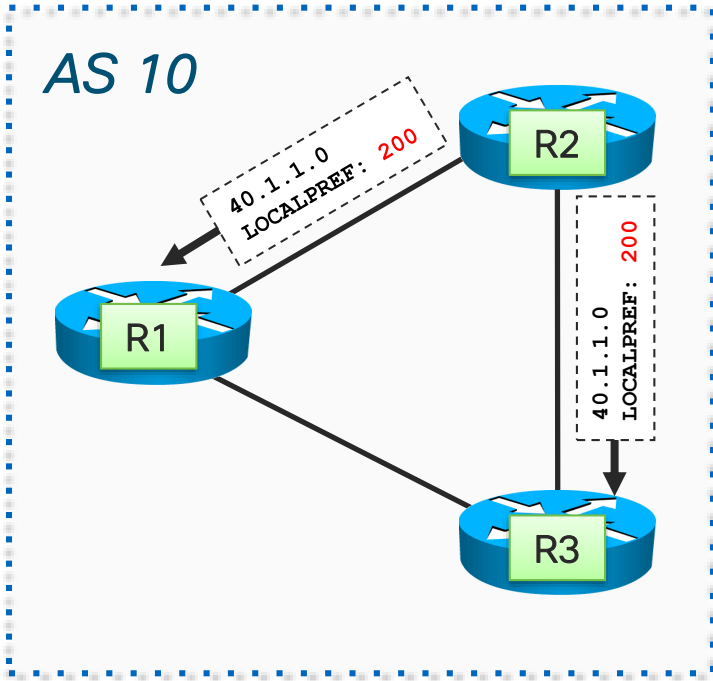


# Local Preference

- R2 advertises LOCALPREF of 200
- R1, R2, and R3 all prefer the R2 exit



# Local Preference



R2#

!

```
router bgp 10
```

```
  neighbor 10.1.1.1 remote-as 10
```

```
  neighbor 10.1.1.1 route-map SET_LOCAL_PREF out
```

```
  neighbor 10.1.1.3 remote-as 10
```

```
  neighbor 10.1.1.3 route-map SET_LOCAL_PREF out
```

!

```
route-map SET_LOCAL_PREF permit 10
```

```
  set local-preference 200
```

!

Or: set localpref inbound on EBGP session



# Alternatives to Local Preference

- Local preference is a very “heavy” attribute to influence routing, as it is evaluated very early in best path algorithm
- Especially with Internet routing, AS path length can be very important (how “far” is the destination?)
- Hence, evaluate attributes for best path manipulation for *your* design
- No one-size fits all; there are *lots* of ways to implement BGP routing policies...



BGP Attribute
Weight
Local Preference
Locally Originated
AS-PATH
ORIGIN
MED
EBGP vs. IBGP
NEXTHOP IGP Cost

# Applying BGP Policy

- Policy based on various attributes:
  - ASPATH
  - Community
  - Destination prefix
  - Many, many others...
- Tools (IOS):
  - Distribute-list or prefix-list
  - Filter-list (as-path access-list)
  - Community-list
  - Route-maps
- Actions
  - Reject/accept selected routes
  - Set attributes to influence path selection

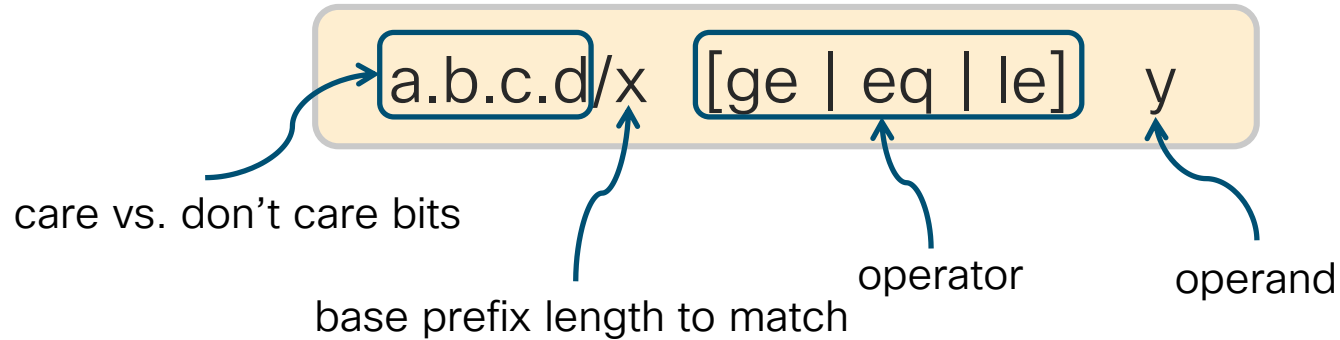


# Policy Control - Prefix List

- Per-peer prefix filter, inbound or outbound
- Allows coverage for ranges of prefix lengths (ge, le)
- Based upon network numbers in NLRI (using familiar IPv4 address/mask format)

```
router bgp 200
  neighbor 220.200.1.1 remote-as 210
  neighbor 220.200.1.1 prefix-list PEER-IN in
  neighbor 220.200.1.1 prefix-list PEER-OUT out
!
ip prefix-list PEER-IN deny 218.10.0.0/16
ip prefix-list PEER-IN permit 0.0.0.0/0 le 32
ip prefix-list PEER-OUT permit 215.7.0.0/16
ip prefix-list PEER-OUT deny 0.0.0.0/0 le 32
```

# Policy Control - Prefix List



`ip prefix-list PEER-IN permit 10.0.0.0/8 le 32`

All 10.x.x.x subnets, regardless of mask length (e.g. 10.1.2.4/24, 10.1.1.1/32, 10.1.0.0/16)

- 0.0.0.0/0 eq 32 = All /32 prefixes (e.g. 1.2.3.4/32)
- 192.168.1.0/24 = 192.168.1.0/24 eq 24 (**ONLY 192.168.1.0/24**)
- 172.16.0.0/16 ge 28 = all subnets from 172.16.0.0/16 that have a mask length of /28 or greater (e.g. 172.16.4.0/28)

# Policy Control – Filter List

- Filter routes based on AS path
- Inbound or Outbound
- Example Configuration:

```
router bgp 100
  neighbor 220.200.1.1 filter-list 5 out
  neighbor 220.200.1.1 filter-list 6 in
!
ip as-path access-list 5 permit ^200$
ip as-path access-list 6 permit ^150$
```

# Policy Control - Regular Expressions

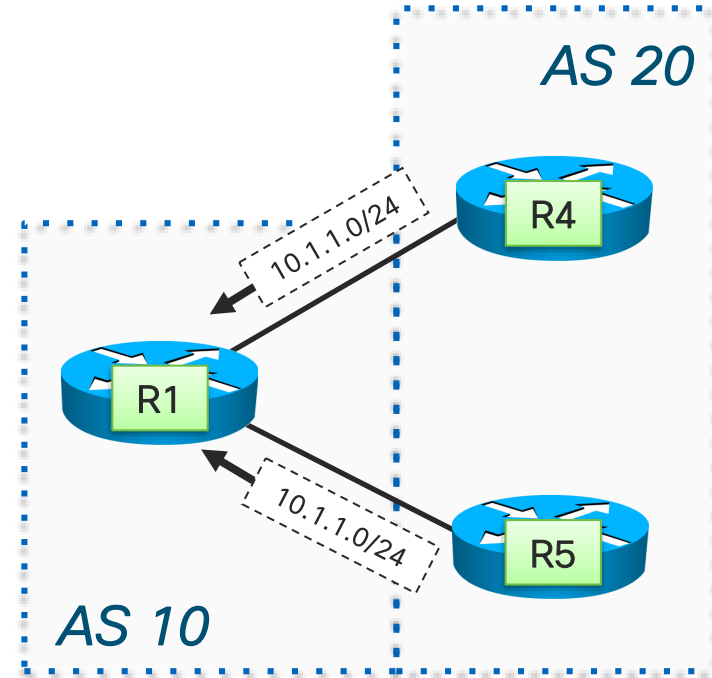
## Simple Examples:

.*	Match anything
^\$	Match routes local to this AS (as-path is empty)
_1800\$	Originated by 1800 (as-path ends with 1800)
^1800_	Received from 1800 (as-path starts with 1800)
_1800_	AS 1800 is somewhere in the as-path
_790_1800_	Passing through 790 then 1800
1800	Literal "1800" is somewhere (e.g. matches <i>1800</i> , <i>21800</i> , <i>18001</i> , etc)

# BGP Multipath

# BGP Multipath

- R1 receives two paths from AS20 (via R2 and R3)
- Best-path algorithm selects one and installs it in routing table
  - By default, all of the traffic goes via one link only
  - Assuming all attributes are equal, uses the one from the lower neighbor IP address
- We could do some manual load-sharing via localpref/MED, but that's cumbersome



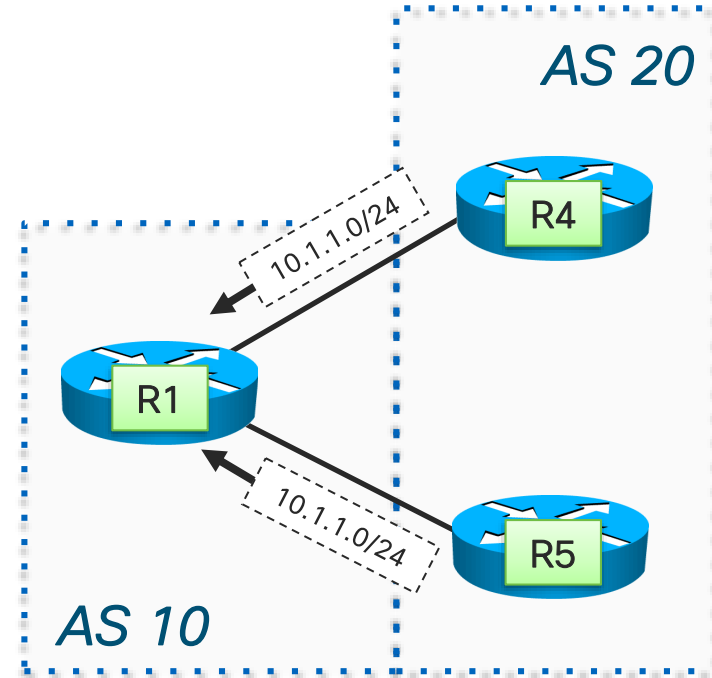


# EBGP Multipath

- Enable *EBGP* multipath on R1 to install both paths

```
router bgp 10
  maximum-paths 2
```

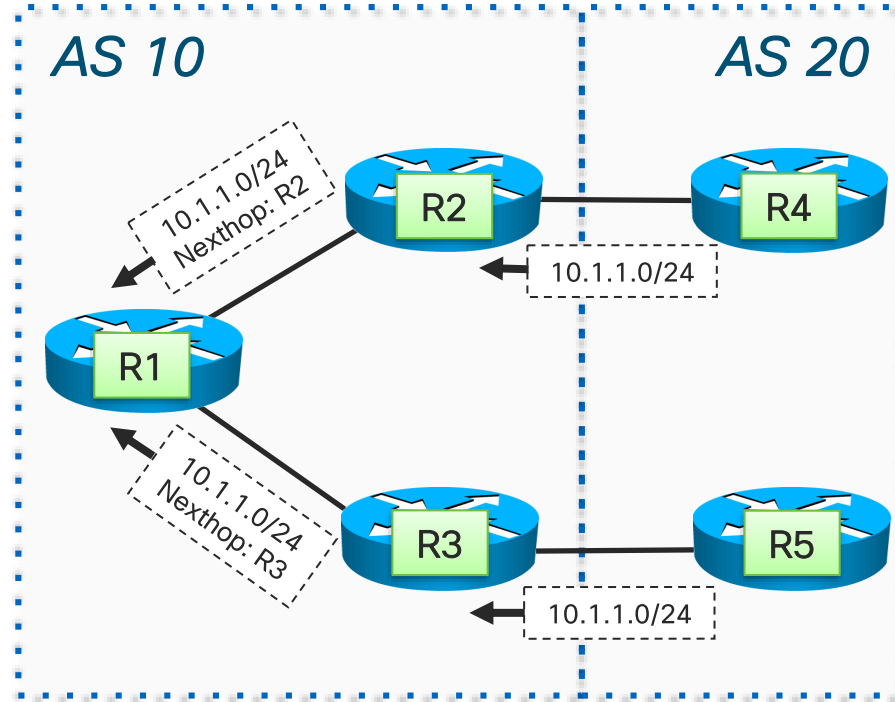
- Multipath selection is part of the Best Path algorithm
  - Evaluated before the more arbitrary tie breakers like IP address/etc.
- Only paths with identical AS\_PATH will be considered
  - Hidden knob “bgp bestpath as-path multipath relax” changes this, but be aware of what you’re doing



# IBGP Multipath

- In this topology, *EBGP* Multipath will not help
- R1 will choose one of the internal paths, and will select one: R2 *or* R3
- *If* R1's IGP cost to R2 and R3 is equal, and all other path attributes are the same, *IBGP* multipath can be used

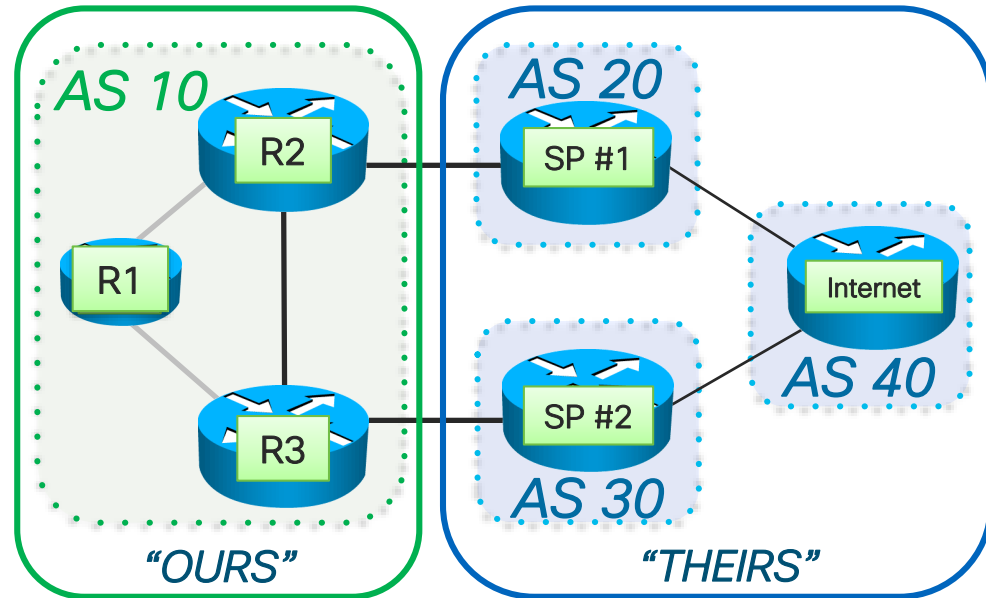
```
router bgp 10
maximum-paths ibgp 2
```



# Controlling Inbound Traffic

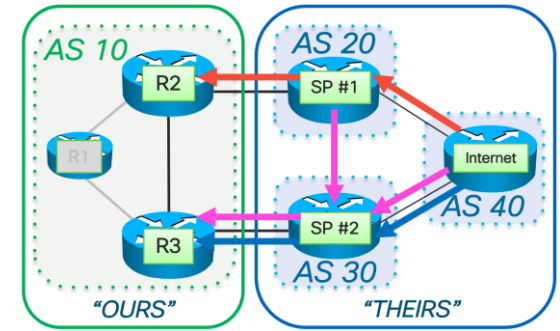
# Controlling Inbound Traffic

- The 1<sup>st</sup> rule of controlling inbound traffic...
  - You do not have ultimate control of how traffic enters your AS
  - They may have outbound policies that will override *all of your attempts* to influence inbound traffic
- That said, what are your options?
  - Leaking more-specific routes
  - MED
  - AS-PATH Prepending
  - Community/Localpref agreement



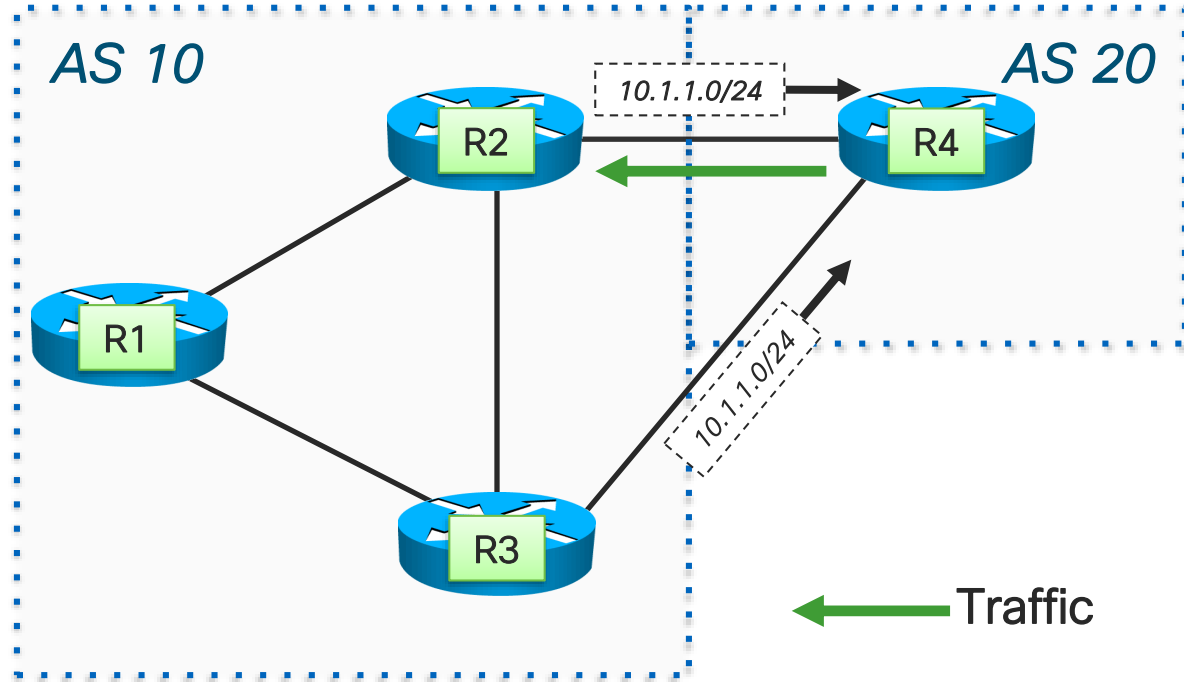
# Leaking Specific Routes

- A RIB lookup always looks for the most specific match
  - A route for 10.1.1.1/32 will be used over 10.1.1.0/24
- You can leak more specific routes to one ISP but not the other
- If the routes are not filtered this will draw the traffic in through the preferred ISP
- Some argue: Advertising more specifics to the global Internet is not “nice” as it causes the Internet BGP table to bloat, and everyone has to bear the costs
- Many ISPs filter routes that are too specific
  - You can’t advertise /32s for your entire address space
  - These will obviously be filtered
  - Check RFC7454 for an overview of BGP Operational Best Practices



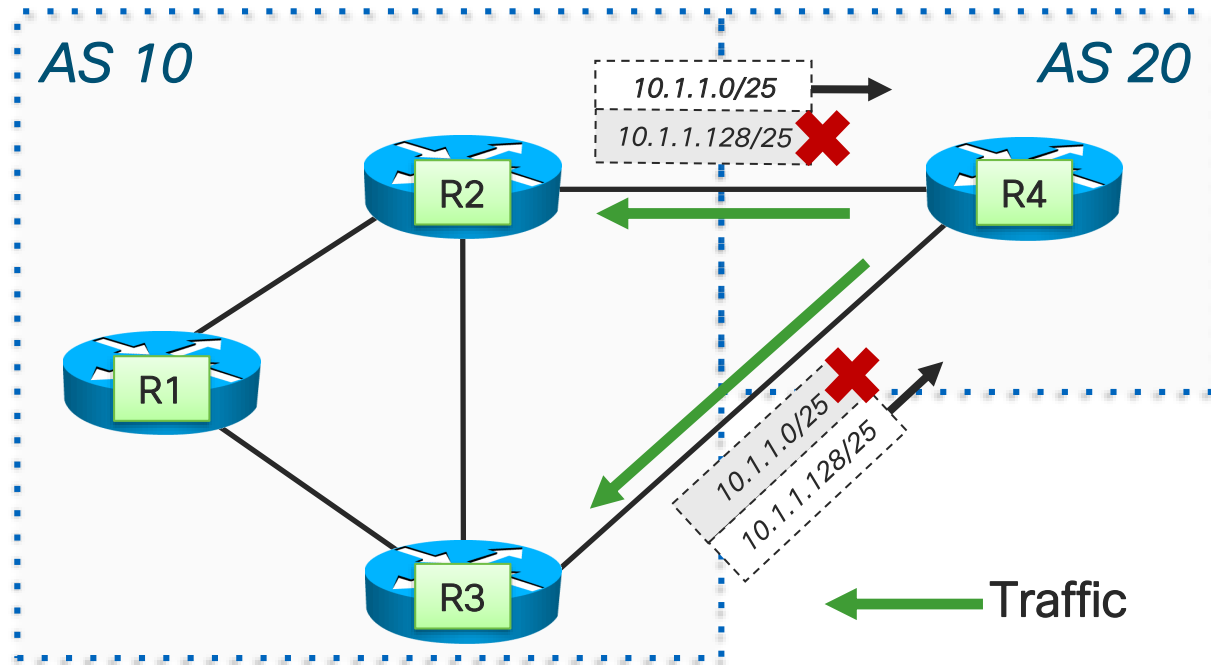
# Leaking Specific Routes

- You are *AS 10*
- *AS 10* owns 10.1.1.0/24
- *AS 20* only uses one link to send traffic to *AS 10* ☹️
- You want to utilize both links



# Leaking Specific Routes

- Split your /24 in two /25s
- R2
  - *advertise* 10.1.1.0/25
  - *suppress* 10.1.1.128/25
- R3
  - *suppress* 10.1.1.0/25
  - *advertise* 10.1.1.128/25
- AS 20 will now send traffic on both links



# Leaking Specific Routes

## Question:

What happens if R2 or R3 die?

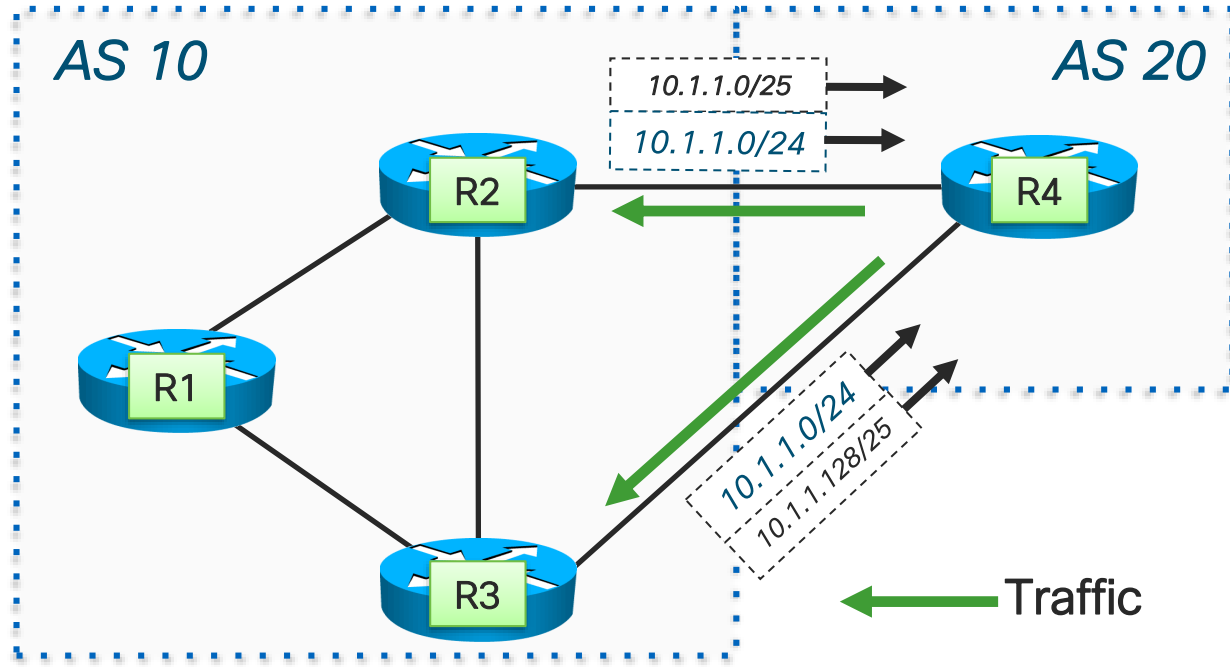
## Answer:

R4 will not know about that /25

## Solution:

Advertise

- full prefix
- plus the more specific





# Leaking Specific Routes

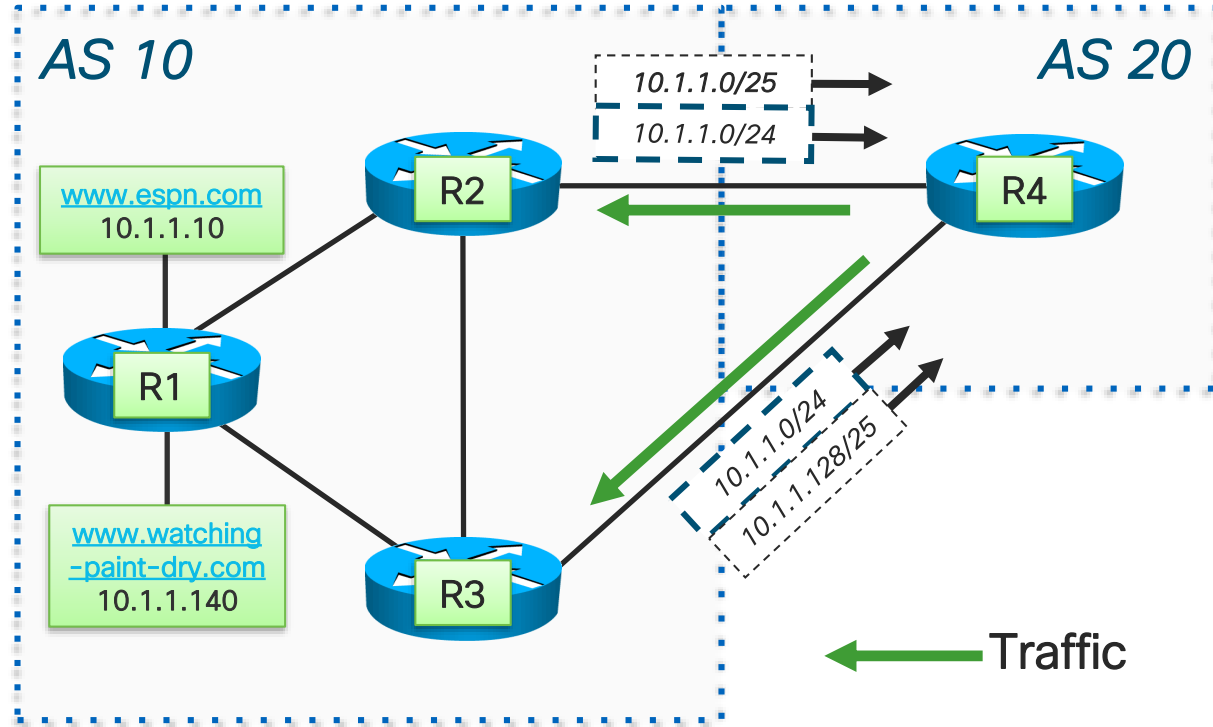
## Question:

Will inbound traffic split 50/50 on your two links?

## Answer:

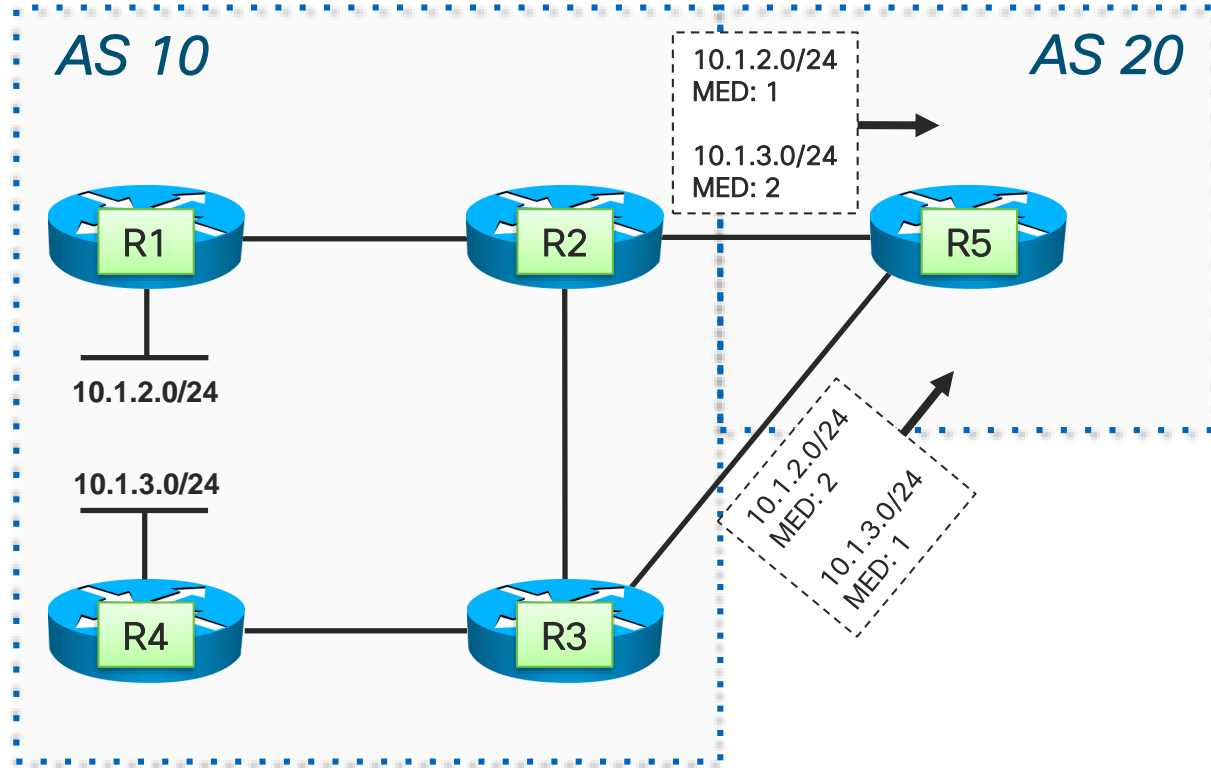
In this case it is likely the

- R2 link will receive more traffic than the R3 link



# MED

- Officially “Multi Exit Discriminator”
- An attribute used to *attempt* influence inbound traffic
- Lower MED is better
  - MED is designed to be a reflection of IGP metrics
  - A lower IGP metric is always preferred
  - Therefore lower MED is preferred
- Used to attempt to bring traffic into the AS on the EBGP speaker closest to the destination



# MED

MEDs can be set manually

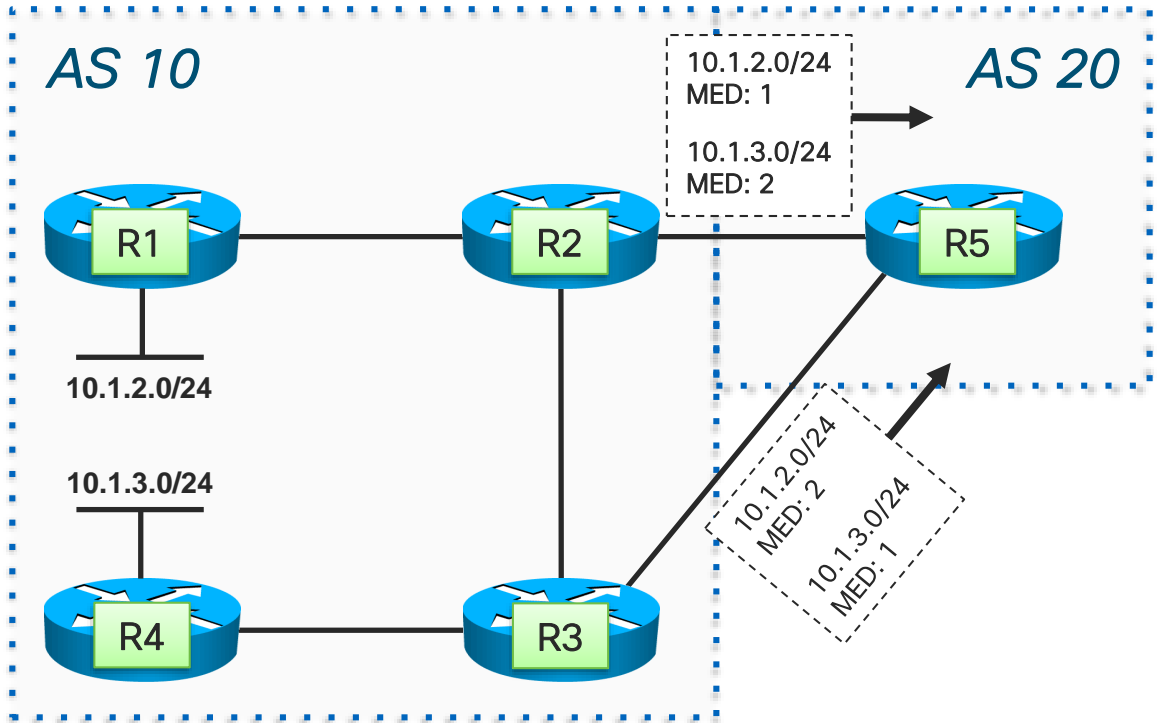
“set metric-type internal” sets MED dynamically

Uses IGP cost to prefix as the MED value

R2 has an IGP cost of 1 to 10.1.2.0

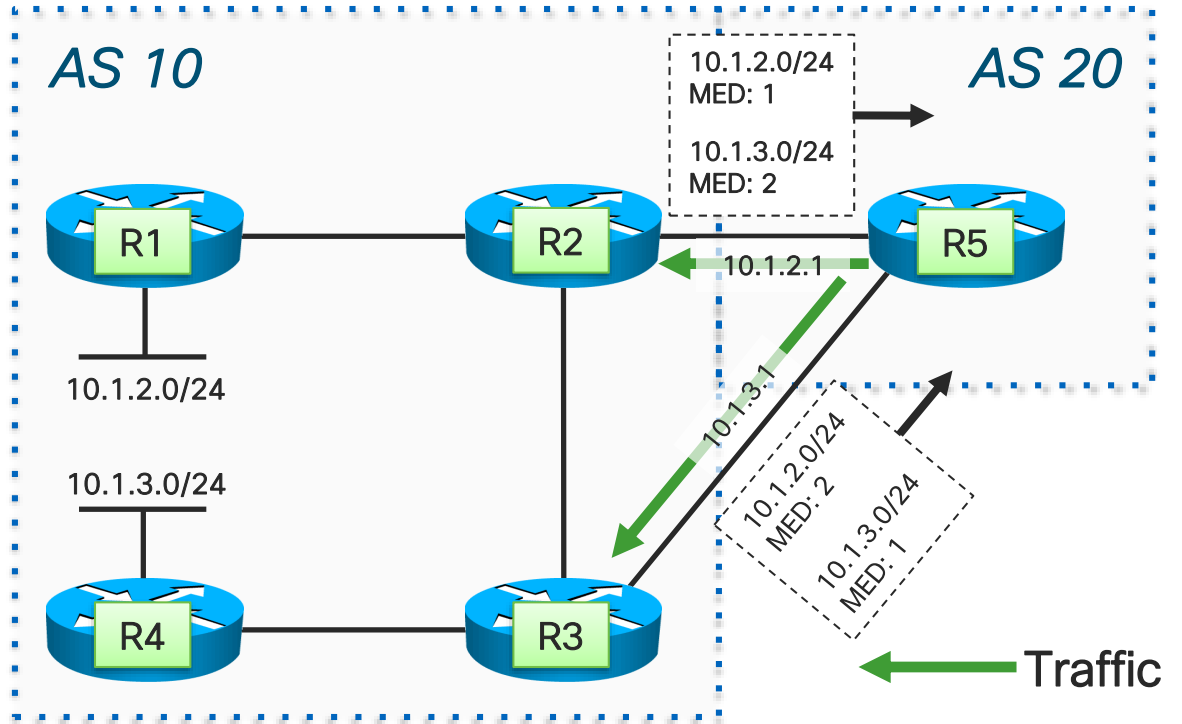
R2 has an IGP cost of 2 to 10.1.3.0

```
R2#  
router bgp 10  
  neighbor 10.1.1.5 remote-as 20  
  neighbor 10.1.1.5 SET_MED out  
!  
route-map SET_MED permit 10  
  set metric-type internal  
!
```



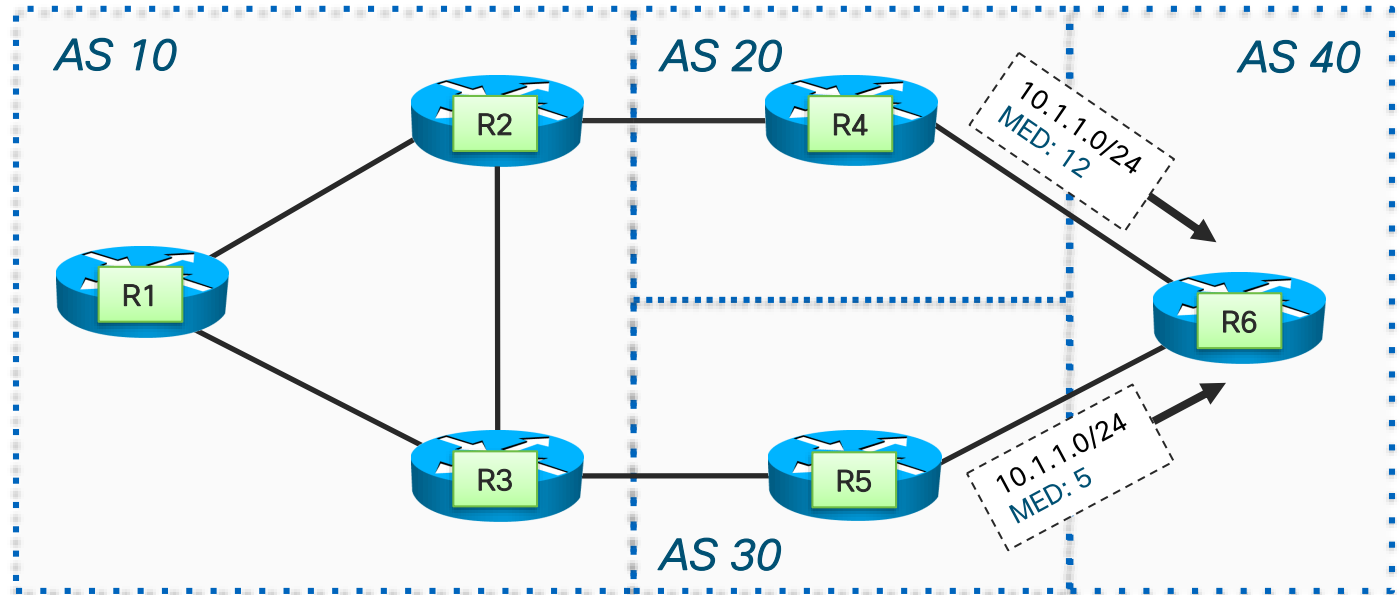
# MED

- Traffic for 10.1.2.0/24 uses the R2 link
- Traffic for 10.1.3.0/24 uses the R3 link



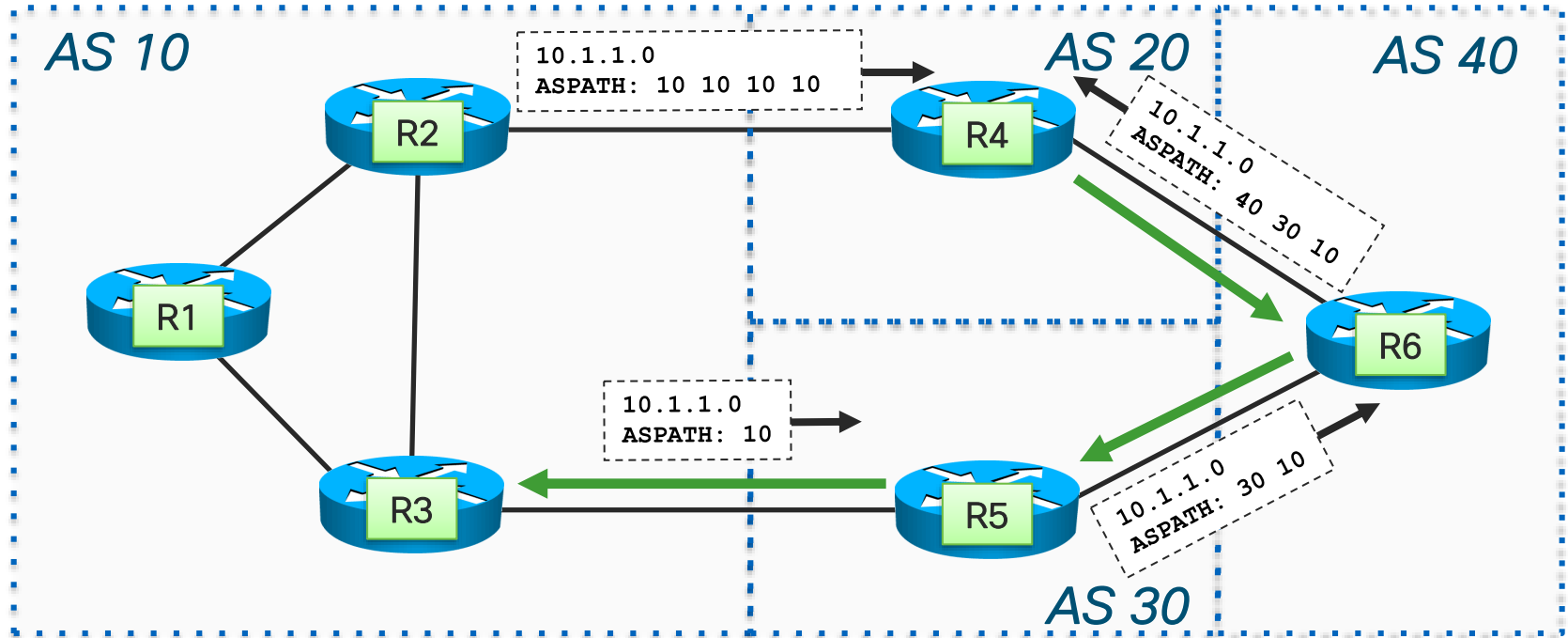
# MED – bgp always-compare-med

- MEDs are only compared if received from the same AS
- Makes sense as you can't necessarily compare routing policies across different AS
- R6 does not compare MEDs for the paths received from AS20 and AS30 unless “bgp always-compare-med” is configured



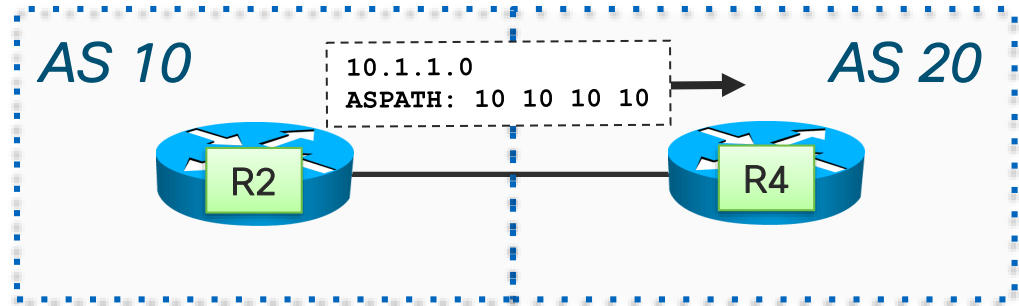
# AS-PATH Prepending

- AS 10 can force traffic into R3 by prepending from R2 → R4
- A shorter ASPATH is preferred



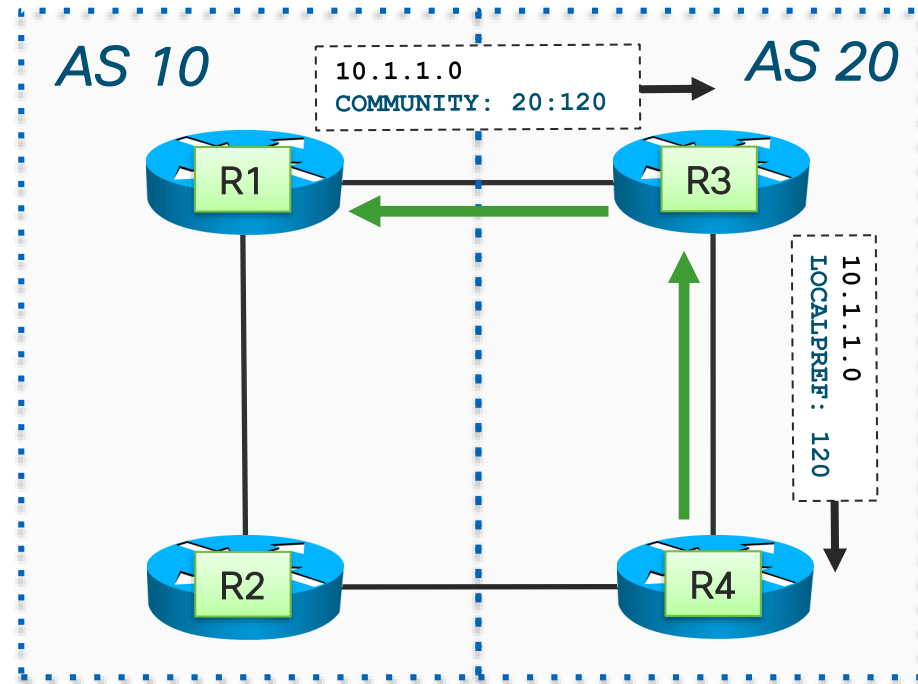
# AS-PATH Prepending

```
R2#  
router bgp 10  
  neighbor 10.1.1.4 remote-as 20  
  neighbor 10.1.1.4 route-map PREPEND_3X out  
!  
route-map PREPEND_3X permit 10  
  set as-path prepend 10 10 10  
!
```



# Community/Local Pref Agreement

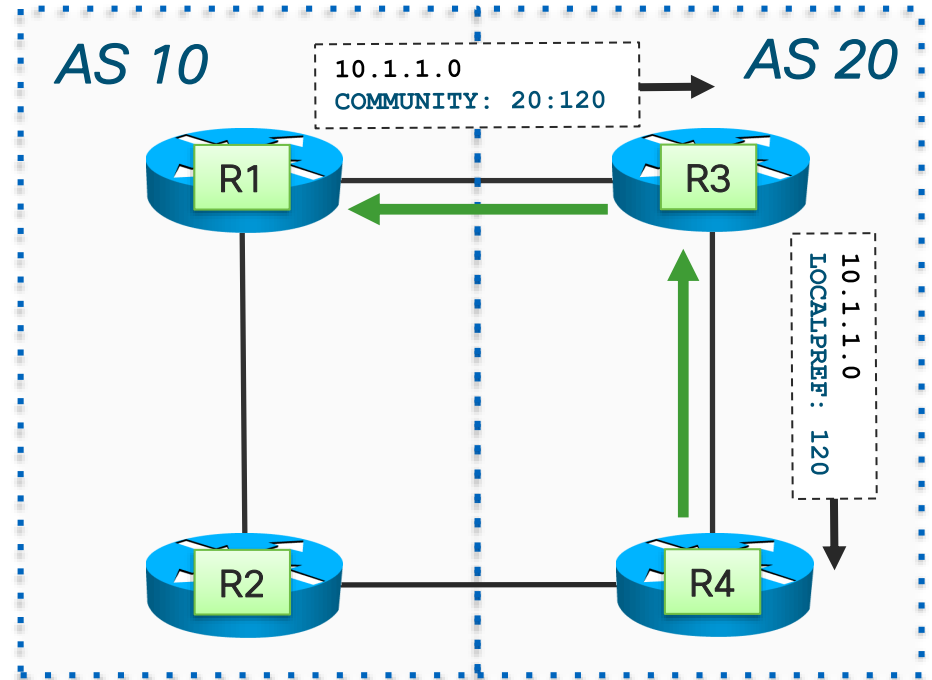
- Many providers accept communities from their customers to give customers some control on inbound traffic.
- Example:
  - Customer sends community 20:80, ISP sets the LOCALPREF to 80
  - Customer sends community 20:120, ISP sets the LOCALPREF to 120





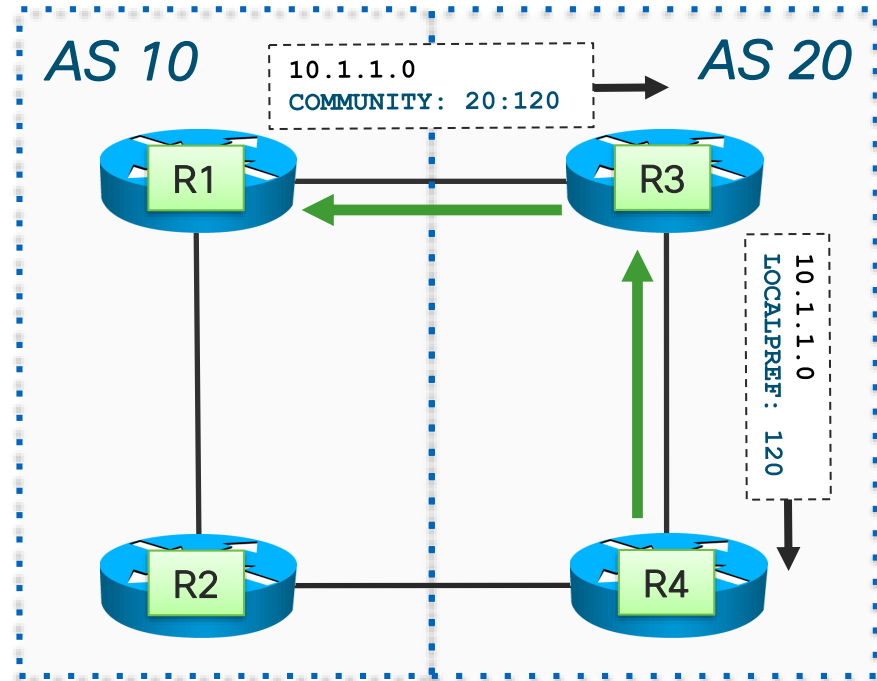
# Community/LOCALPREF Agreement

```
R1#  
router bgp 10  
  neighbor 10.1.1.3 remote-as 20  
  neighbor 10.1.1.3 route-map SET_COMMUNITY out  
  neighbor 10.1.1.3 send-community  
!  
route-map SET_COMMUNITY permit 10  
  set community 20:120  
!
```



# Community/LOCALPREF Agreement

```
R3#
router bgp 20
  neighbor 10.1.1.1 remote-as 10
  neighbor 10.1.1.1 route-map COMMUNITY_TO_LOCALPREF in
!
ip community-list standard LP_80 permit 20:80
ip community-list standard LP_120 permit 20:120
!
route-map COMMUNITY_TO_LOCALPREF permit 10
  match community LP_80
  set local-preference 80
!
route-map COMMUNITY_TO_LOCALPREF permit 20
  match community LP_120
  set local-preference 120
!
route-map COMMUNITY_TO_LOCALPREF permit 30
!
```



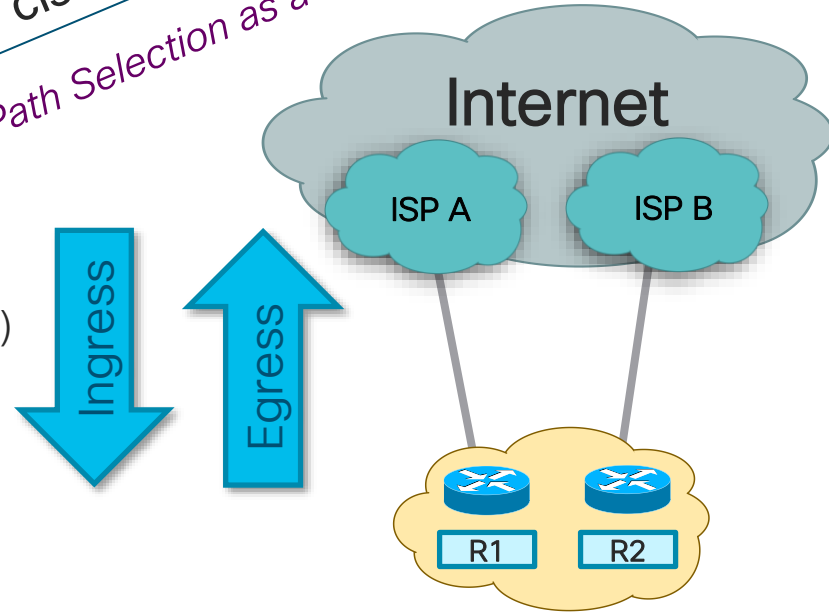
# Enterprise Multi-Homed Internet Edge Architectures

Zig Zsiga, BRKRST-2044

Watch live or the recording on [ciscolive.com](https://www.ciscolive.com)

Think of Path Selection as a Two-way street.

- Single Router, 1 Link
- Single Router, 2 Links (Equal and Unequal BW)
- Multiple Routers, Multiple Links (Equal and Unequal BW)
- Multiple Routers, Multiple Firewalls, Multi-Site



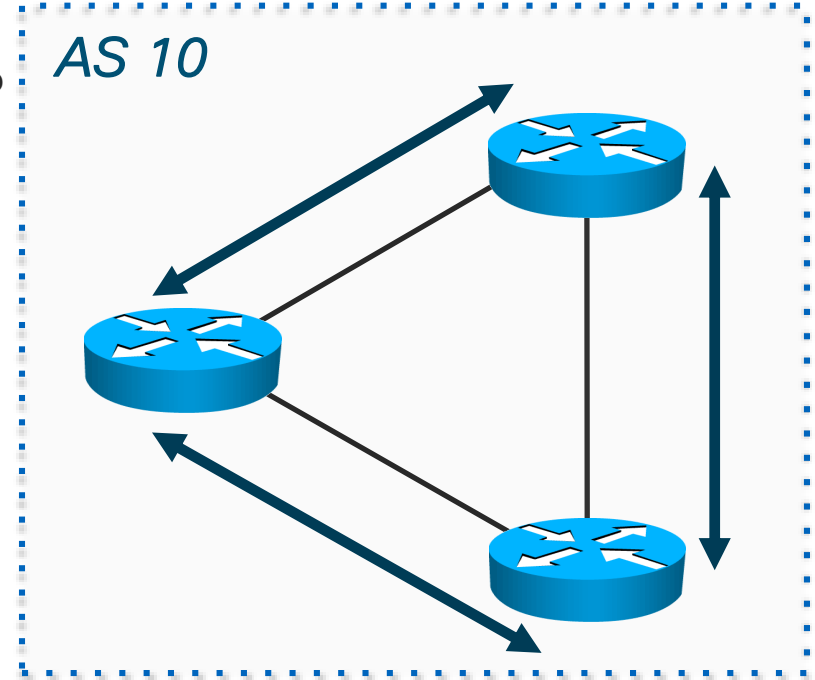
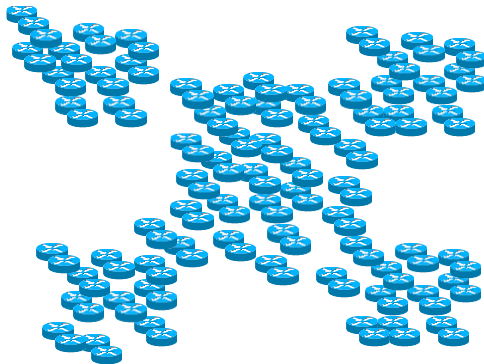
# Route Reflectors

# Agenda

- BGP General Operation
  - Overview
  - EBGP and IBGP
- Attributes and Best Path Selection Algorithm
  - Route Origination
  - AS-PATH
  - NEXTHOP
  - Communities
- Controlling Traffic
  - Controlling Outbound Traffic
  - BGP Multipath
  - Controlling Inbound Traffic
- Route Reflectors
- Multiprotocol BGP
- Common BGP Deployments
- Securing BGP
- BGP Routing Convergence
- Show and Tell/Demo Lab

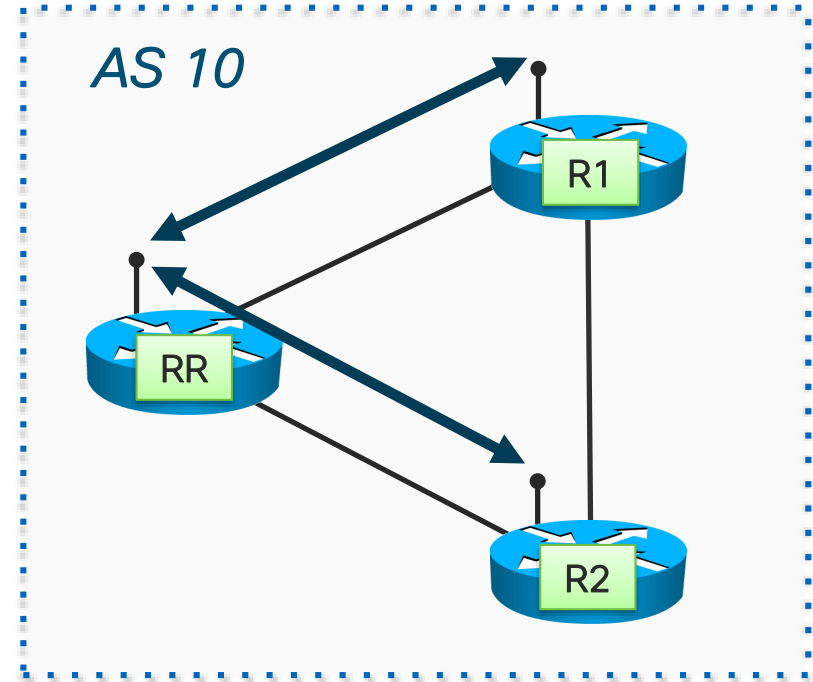
# BGP Advertising Rule to Prevent Loops

- A route received from one iBGP peer will NOT be advertised to another iBGP peer
- Full iBGP mesh is required
- $n*(n-1)/2$  peering mesh – scaling problem!



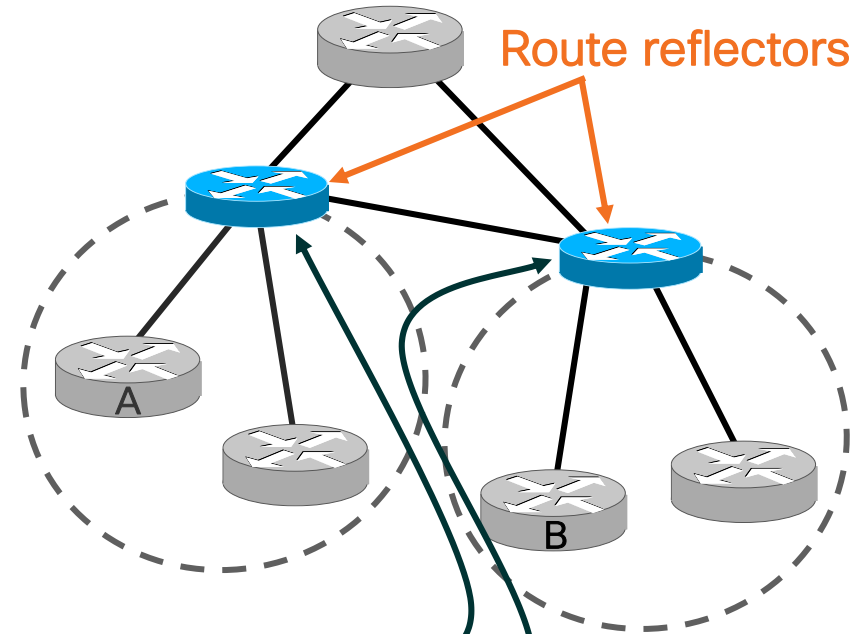
# Route Reflectors

- A route received from one iBGP peer will NOT be advertised to another iBGP peer
  - Full iBGP mesh is required ( $(n-1)/2$  peering mesh – scaling problem!
- Route-Reflectors relax this constraint**



# Route Reflector Basics

- A route reflector is an iBGP speaker that reflects routes learned from iBGP peers to other iBGP peers
- Route reflectors are designated by configuring some of their iBGP peers as route reflector clients

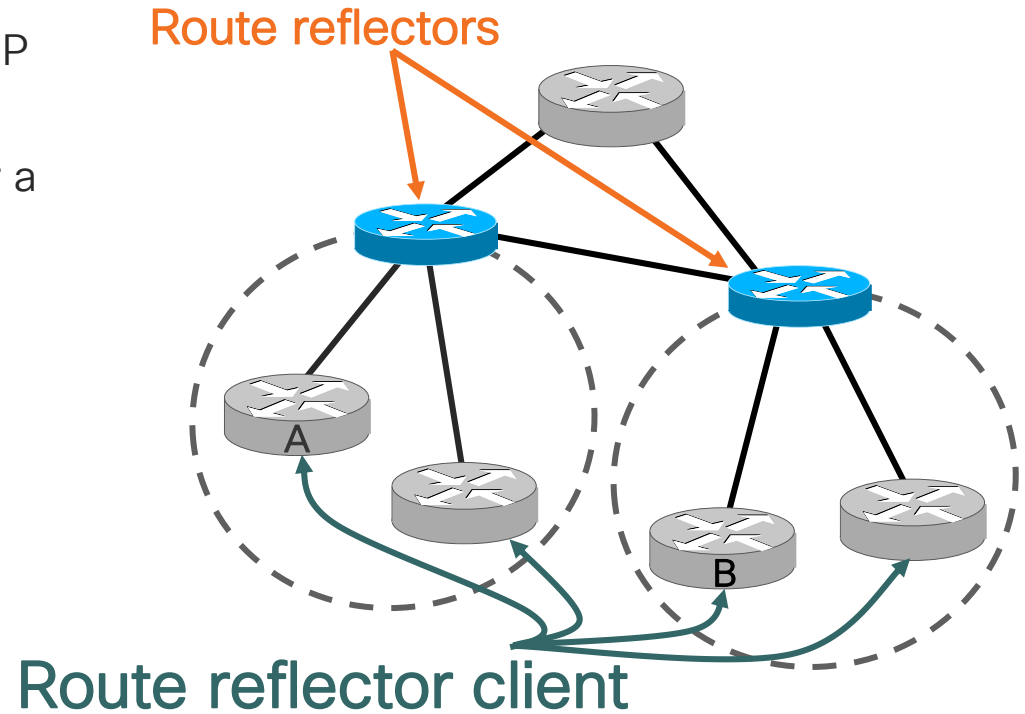


```
neighbor <A> route-reflector-client  
neighbor <B> route-reflector-client
```



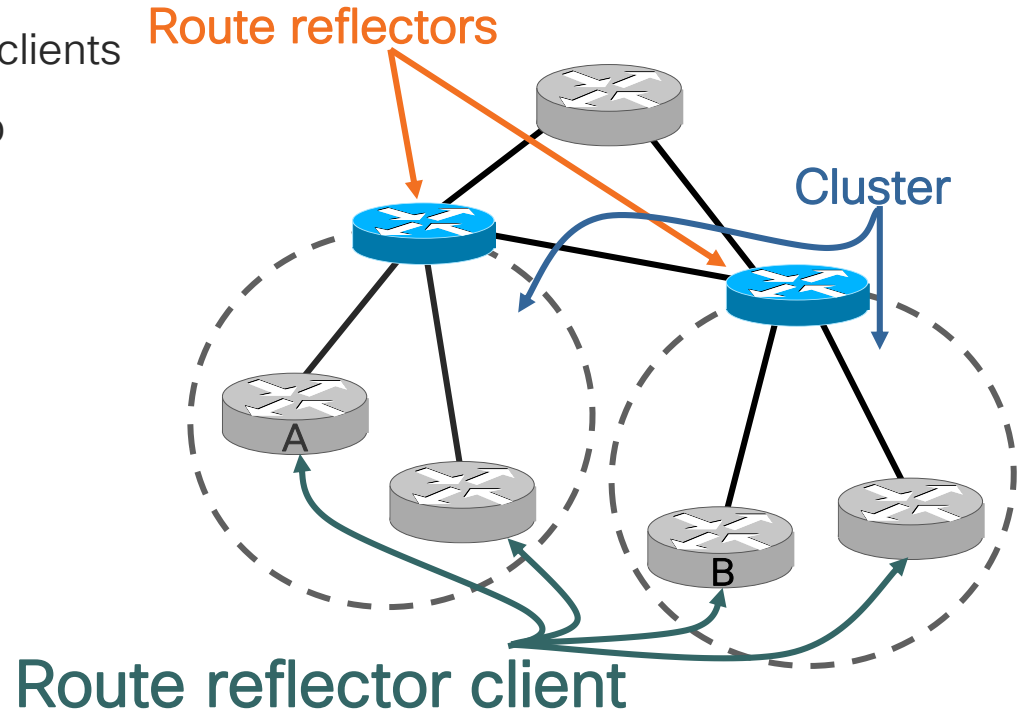
# Route Reflector Basics

- A route reflector client is just an iBGP speaker
- There is no special configuration for a route reflector client



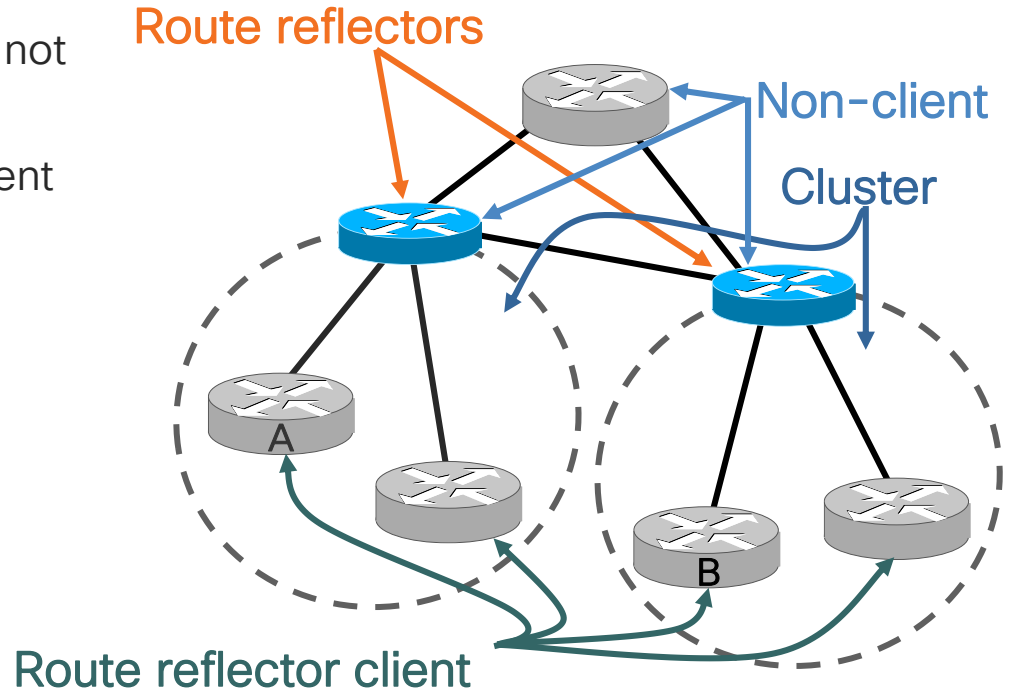
# Route Reflector Basics

- A cluster is a route reflector and its clients
- Route reflector clusters may overlap



# Route Reflector Basics

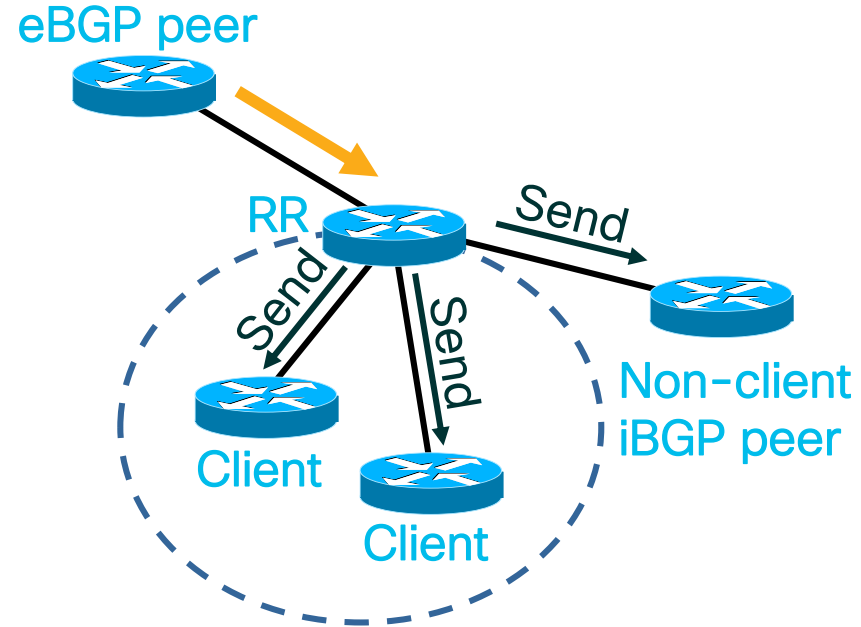
- A non-client is any iBGP peer that is not a route reflector client
- Each route reflector is also a non-client of each other route reflector in this network
- Route reflectors must be fully iBGP meshed



# Route Reflector – Advertisement Rules

If a Route Reflector Receives a Route from an eBGP Peer what will it do?  
(Always assuming this route was elected as best path)

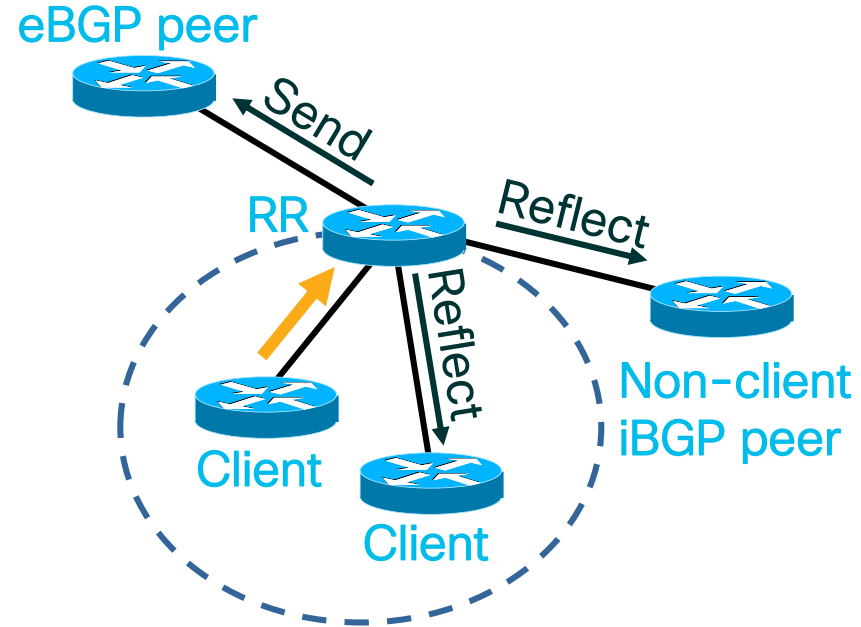
- Send the route to ALL BGP peers (iBGP and eBGP)



# Route Reflector – Advertisement Rules

If a Route Reflector Receives a Route from a Client what will it do?

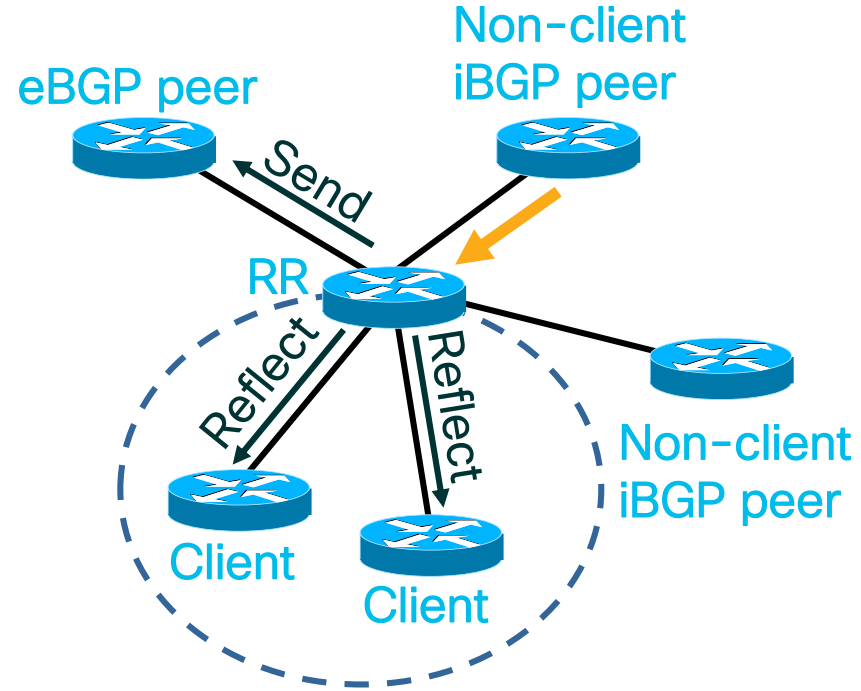
- Reflect the route to all clients
- Reflect the route to all non-clients
- Send the route to all eBGP peers



# Route Reflector – Advertisement Rules

If a Route Reflector Receives a Route from a Non-Client what will it do?

- Reflect the route to all clients
- Send the route to all eBGP peers



# Route Reflector Design and Redundancy

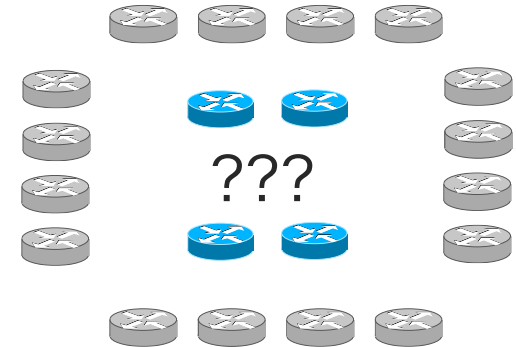


A client may peer with more than one reflector

- A client that peers to only one reflector has a single point of failure
- Clients should peer to at least two reflectors to provide redundancy

Questions:

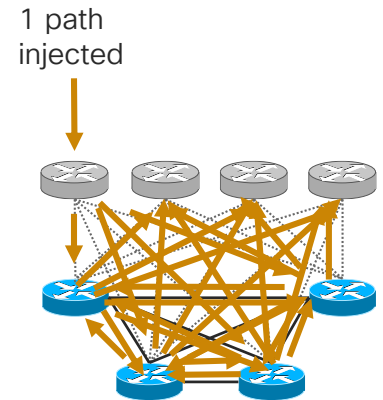
- How many reflectors should a single client be peered to?
- Where should the RRs be placed in the network?
- How many RRs are needed?



# Route Reflector Design and Redundancy



- Redundancy is needed but....
- Too much burns memory on RR-Clients (RRCs) because the client learns the same information from each RR
- Also burns memory on the RRs because they learn multiple paths for each route introduced by a RRC
- Two route reflectors per client should be plenty...
- ...but this is not a hard and fast rule
- As with everything else... "it depends"
  - PEs, RRs, SLAs, network size, network topology, etc.



*for illustrative purpose only*



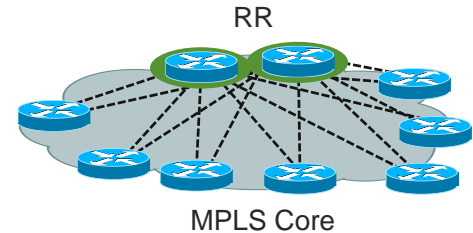
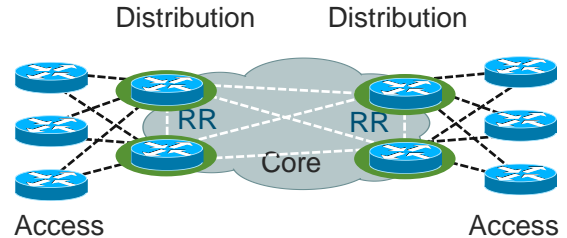
# A word of reason



- Most routers sold in the last decade can easily run 100 or more sessions (all depends on number of prefixes carried)
- ASR1000-RP2 scales to thousands of sessions (Isocore tested 20 Million routes with 1000 RR clients)
- So RP performance is often not the limiting factor of a full iBGP mesh, it's rather the manageability adding/removing nodes from the mesh
- So don't over-engineer it...

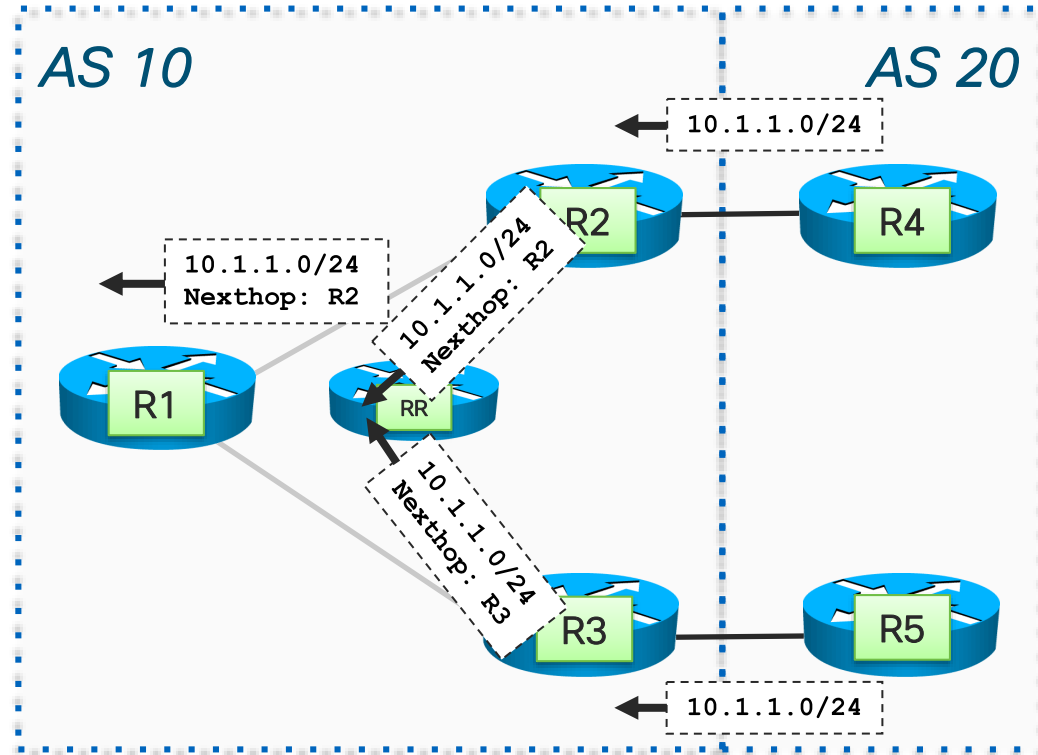
# Common Route Reflector Deployment Scenarios

- Non-MPLS core mandate RRs **follow the topology** to avoid loops, so putting them at (or directly on) distribution layer is a straight-forward approach
- With MPLS, route reflector placement can be more flexible, we commonly see dedicated RRs (not in data path), placement/number dictated by number of PEs/clients and path diversity requirements
  - Core routers can be BGP-free, they just switch labels



# Route Reflectors and Information Hiding

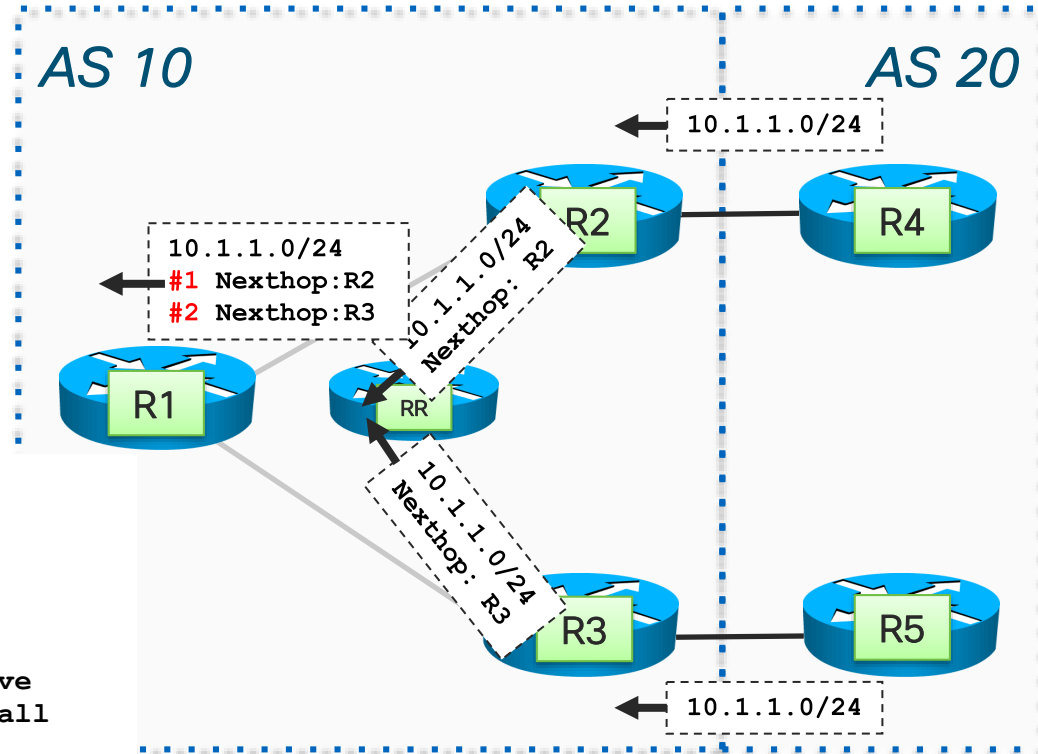
- RR will only forward only its best path
- R1 will only see one path to AS20
- This stops iBGP Multipath from working
- Even worse, can negatively affect failover times in larger deployments (see later)



# Route Reflectors and Information Hiding Solution 1: BGP AddPath

- BGP Protocol Enhancement to send more than one path
- Requires support for this feature on both RR and peers

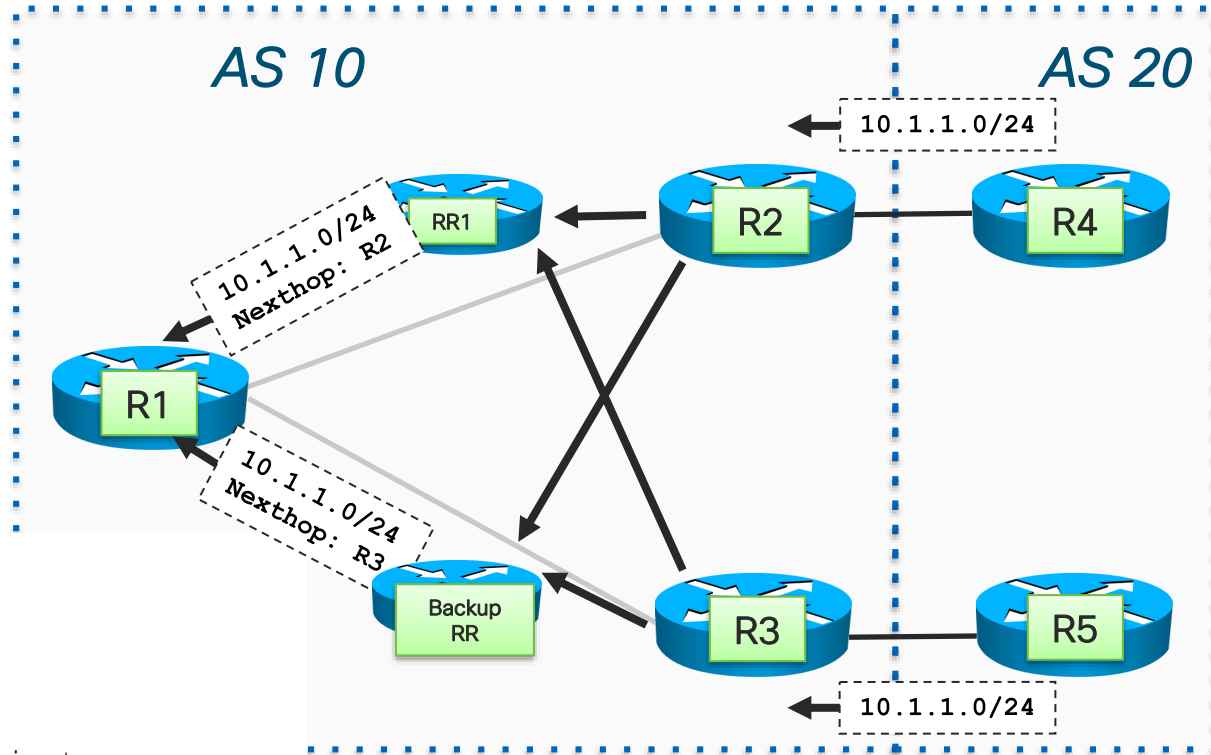
```
RR#  
router bgp 10  
  address-family ipv4 unicast  
    bgp additional-paths select all  
  neighbor 10.1.1.1 remote-as 10  
  neighbor 10.1.1.1 route-reflector-client  
  neighbor 10.1.1.1 additional-paths send receive  
  neighbor 10.1.1.1 advertise additional-paths all  
[...]  
!
```



# Route Reflectors and Information Hiding

## Solution 2: BGP Diverse-Paths

- Denote one of the RRs in a cluster as "Backup-RR"
- Configure this one to advertise the 2<sup>nd</sup> best path
- Can also work by configuring two iBGP sessions between RR and each client



```
Backup-RR#  
router bgp 10  
maximum-paths ibgp 4  
bgp bestpath igp-metric ignore  
bgp additional-paths select backup  
! bgp additional-paths install  
neighbor 10.1.1.1 route-reflector-client  
neighbor 10.1.1.1 advertise diverse-path backup
```

# Multiprotocol BGP

# Agenda

- BGP General Operation
  - Overview
  - EBGP and IBGP
- Attributes and Best Path Selection Algorithm
  - Route Origination
  - AS-PATH
  - NEXTHOP
  - Communities
- Controlling Traffic
  - Controlling Outbound Traffic
  - BGP Multipath
  - Controlling Inbound Traffic
- Route Reflectors
- Multiprotocol BGP
- Common BGP Deployments
- Securing BGP
- BGP Routing Convergence
- Show and Tell/Demo Lab

# Routing Protocol Reachability Announcements

- Interior routing protocols announce networks and topology on how to reach them
  - Network reachability and topology information is often closely coupled
- BGP also advertises network reachability, but leaves out how to reach the Next Hop
  - This allows to extend the announcements to much more than IP destinations, using the exact same protocol



Hey, I am your EIGRP neighbor and you can reach 10.1.1.0/24 through me using metric X



Hello everyone! I am OSPF router John, here are my OSPF neighbors so you know how to find me, and I also own 10.20.2.0/24



Good day! Glad you're speaking BGP with me. I can tell you where to reach IPv4 network 10.30.0.0/16, or would you rather learn about IPv6 networks or MAC addresses?



# Control-plane Evolution

- Many services are moving towards BGP to disseminate control-plane information
- Operator's and Designer's familiarity with BGP is an important factor
- But so is policy control, scale and the extensibility of the protocol

Service/transport	"Traditional"	Today
IDR (Peering)	BGP	BGP (IPv6)
SP L3VPN	BGP	BGP + FRR + Scalability
SP Multicast VPN	PIM	BGP Multicast VPN
DDOS mitigation	CLI	BGP flowspec
Network Monitoring	SNMP	BGP monitoring protocol
Security	Filters	BGP Sec (RPKI), DDoS Mitigation
Proximity		BGP connected app API
SP-L3VPN-DC		BGP Inter-AS, VPN4DC
Business & CE L2VPN	LDP	BGP PW Sign (VPLS)
DC Interconnect L2VPN		BGP MAC Sign (EVPN)
MPLS transport	LDP	BGP + Label (Unified MPLS)
Data Center	OSPF/ISIS	BGP + Multipath
Massive Scale DMVPN	NHRP / EIGRP	BGP + Path Diversity
Campus/Ent L3VPN	BGP	BGP

# BGP Address and Sub-Address Families

- BGP can advertise multiple Network Protocols' reachability information (NLRI) → Multi-Protocol-BGP
- **Address Family (AF):** Network Protocol Type (ex: IPv4, IPv6, CLNS, etc.)
- **Subsequent Address Family (SAF):** Additional semantic to the above, for example, unicast or multicast, MPLS VPN addresses, etc.
- Some examples (far from exhaustive):

AFI	SAFI	Description
1	1	IPv4 Unicast
1	2	IPv4 Multicast
1	4	Labeled IPv4
2	1	IPv6 Unicast
2	2	IPv6 Multicast
2	4	Labeled IPv6 (aka 6PE)

AFI	SAFI	Description
1	128	L3VPN IPv4 unicast
1	129	L3VPN IPv4 multicast
3	128	CLNS VPN
1	133	Flow-Spec
25	65	BGP-VPLS
25	70	EVPN
16388	71	BGP Link-State

See <http://www.iana.org/assignments/address-family-numbers/address-family-numbers.xhtml> and <http://www.iana.org/assignments/safi-namespace/safi-namespace.xhtml> for full list.

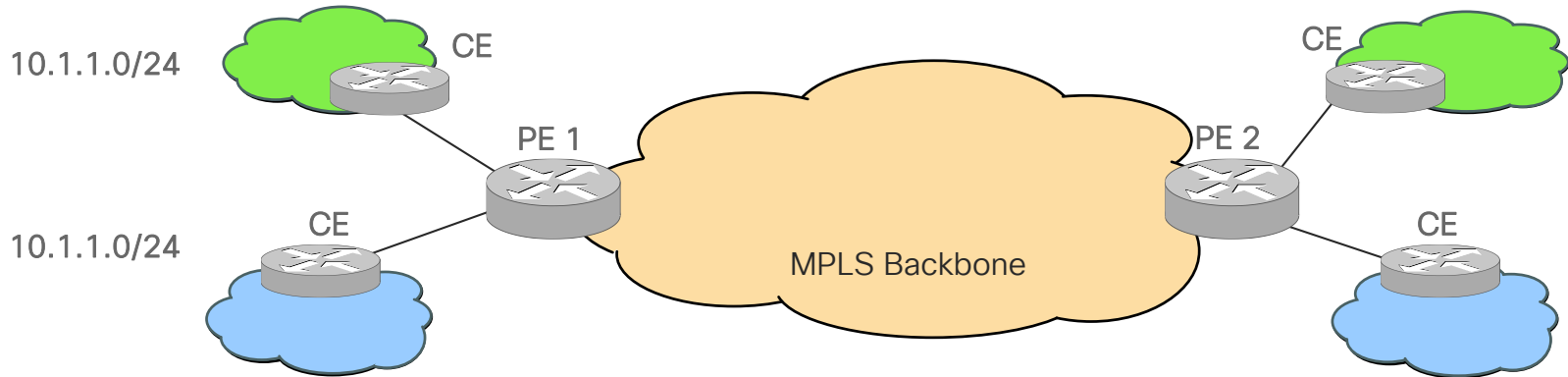
Not all denote to AFI/SAFI used in BGP.

# MP-BGP: Address-Family-Identifier Syntax

Original syntax	AFI/SAFI syntax
<pre>router bgp 20   bgp router-id 20.100.100.20   bgp log-neighbor-changes   neighbor 10.20.40.40 remote-as 20   neighbor 10.20.40.40 update-source Loopback0   network 10.100.100.0 mask 255.255.255.0</pre>	<pre>router bgp 20   bgp router-id 20.100.100.20   bgp log-neighbor-changes   neighbor 10.20.40.40 remote-as 20   neighbor 10.20.40.40 update-source Loopback0   !   address-family ipv4     network 10.100.100.0 mask 255.255.255.0     neighbor 10.20.40.40 activate     neighbor 10.20.40.40 send-community   exit-address-family</pre>
<p>Note: We can advertise multiple AFI/SAFI over the same session!</p>	<pre>!   address-family vpnv4     neighbor 10.20.40.40 activate     neighbor 10.20.40.40 send-community   extended</pre>

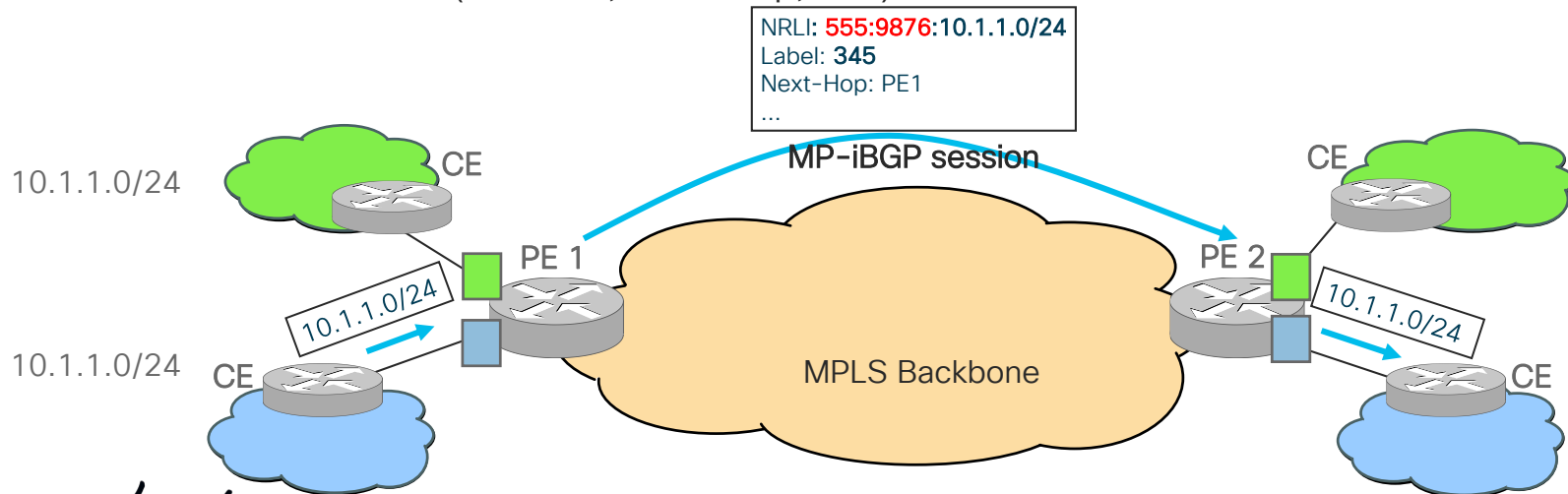
# Use Case 1: MPLS-VPN- L3VPN

- Layer 3 VPN carries customer routing information across an MPLS core
- Customer addresses can overlap (think: multiple enterprise customers all using 10.0.0.0/8)
- Problem: How do we differentiate them on the control plane (BGP) and on the forwarding plane (within the backbone)



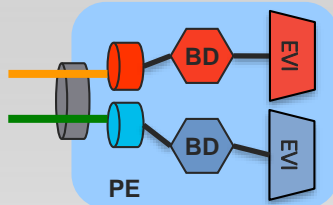
# Use Case 1: MPLS-VPN- L3VPN

- Solution:
  1. Control Plane: Make addresses distinguishable by adding an additional identifier:  
**Route Distinguisher (RD):** 555:9876:10.1.1.0/24
  2. Forwarding Plane: Add routing contexts on PEs, and carry packets as MPLS **labeled** packets across the backbone.
- BGP advertises VPNv4 addresses (8 byte RD + 4 byte IPv4 addresses) and a label as NLRI
- Other BGP attributes (AS-Path, Next-Hop, etc.) are included as seen before



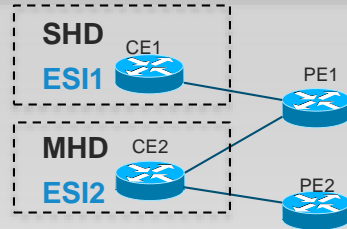
# Use Case 2: EVPN - Ethernet VPN

## EVPN Instance (EVI)



- EVI identifies a VPN in the network
- Encompass one or more bridge-domains, depending on service interface type
  - Port-based
  - VLAN-based (shown above)
  - VLAN-bundling

## Ethernet Segment



- Represents a 'site' connected to one or more PEs
- Uniquely identified by a 10-byte global Ethernet Segment Identifier (ESI)
- **Could be a single device or an entire network**
  - Single-Homed Device (SHD)
  - Multi-Homed Device (MHD)
  - Single-Homed Network (SHN)
  - Multi-Homed Network (MHN)

## BGP Routes

Route Types
[1] Ethernet Auto-Discovery (AD) Route
[2] MAC/IP Advertisement Route
[3] Inclusive Multicast Route
[4] Ethernet Segment Route
[5] IP Prefix Advertisement Route

- **New SAFI [70]**
- **Routes serve control plane purposes, including:**
  - MAC address reachability
  - MAC mass withdrawal
  - Split-Horizon label adv.
  - Aliasing
  - Multicast endpoint discovery
  - Redundancy group discovery
  - Designated forwarder election
  - IP address reachability
  - L2/L3 Integration

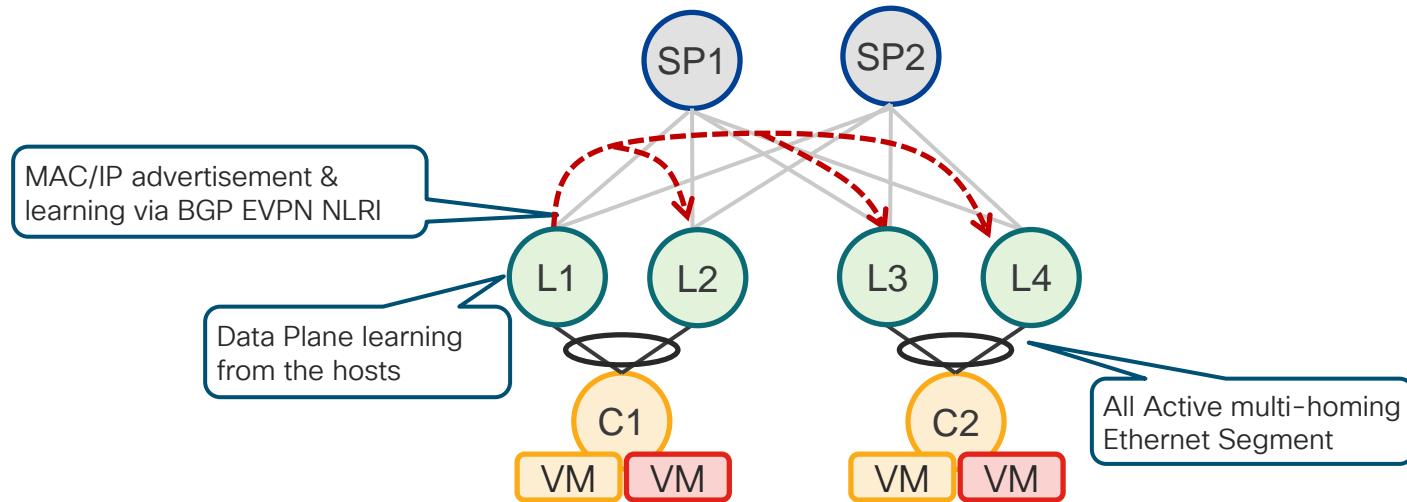
## BGP Route Attributes

Extended Communities
ESI MPLS Label
ES-Import
MAC Mobility
Default Gateway
Encapsulation

- **New BGP extended communities defined**
- **Expand information carried in BGP routes, including:**
  - MAC address moves
  - Redundancy mode
  - MAC / IP bindings of a GW
  - Split-horizon label encoding
  - Data plane Encapsulation

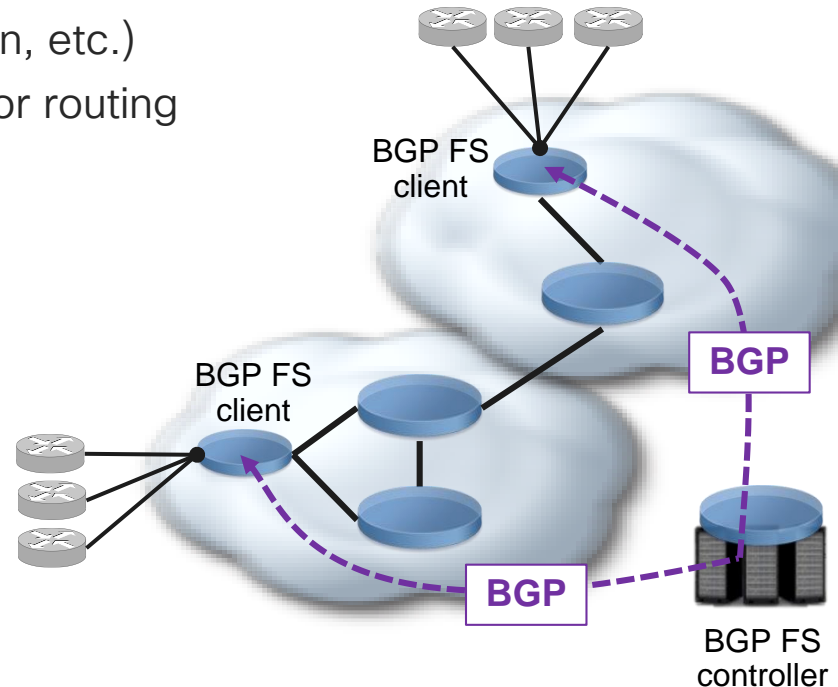
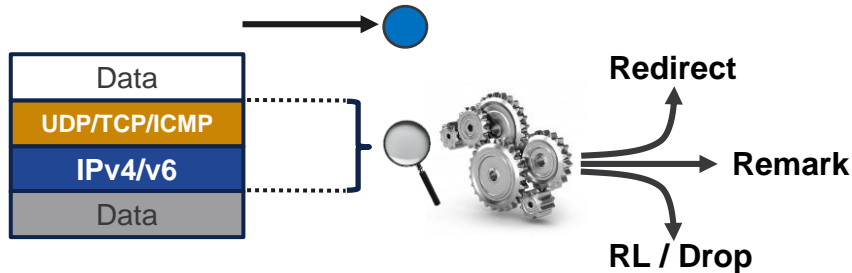
# EVPN - Ethernet VPN

- Leafs run Multi-Protocol BGP to advertise & learn MAC/IP addresses over the Network Fabric
- MAC/IP addresses are advertised to rest of Leafs



# BGP FlowSpec Overview

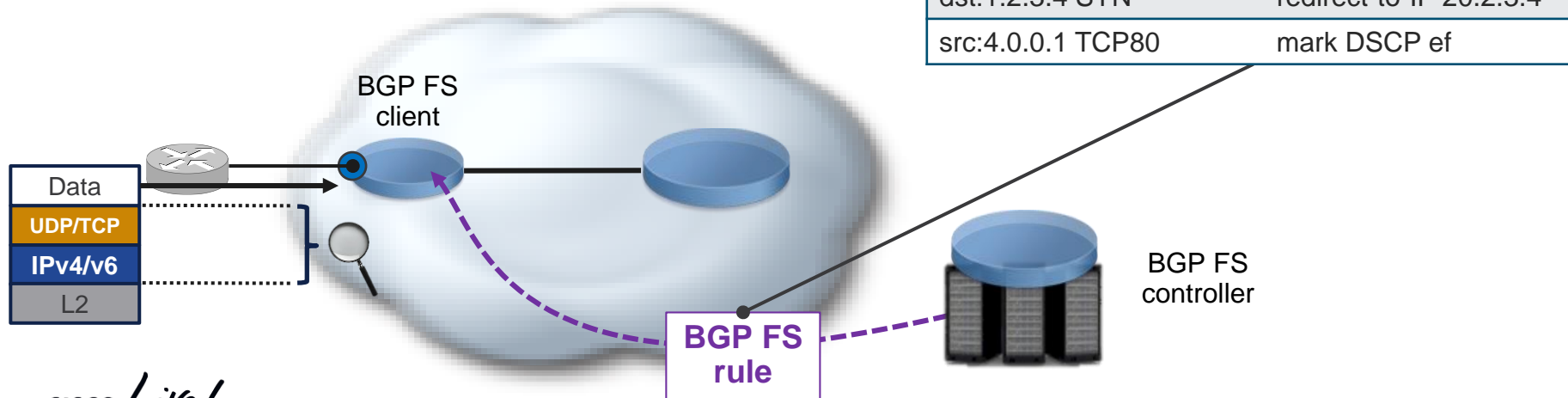
- Using BGP, a "Controller" can dynamically control the treatment of packets across the network, for example:
  - Drop packets (Denial of Service mitigation, etc.)
  - Redirect traffic to a different destination or routing context (VRF)
  - Apply QoS markings
  - Rate-limiting





# BGP FS Rule Made of Description and Action

- BGP is used to program remotely a rule made of:
  - A traffic description (v4/v6 L3/L4)
  - An action
- Traffic received on client matching the Description will be applied the Action



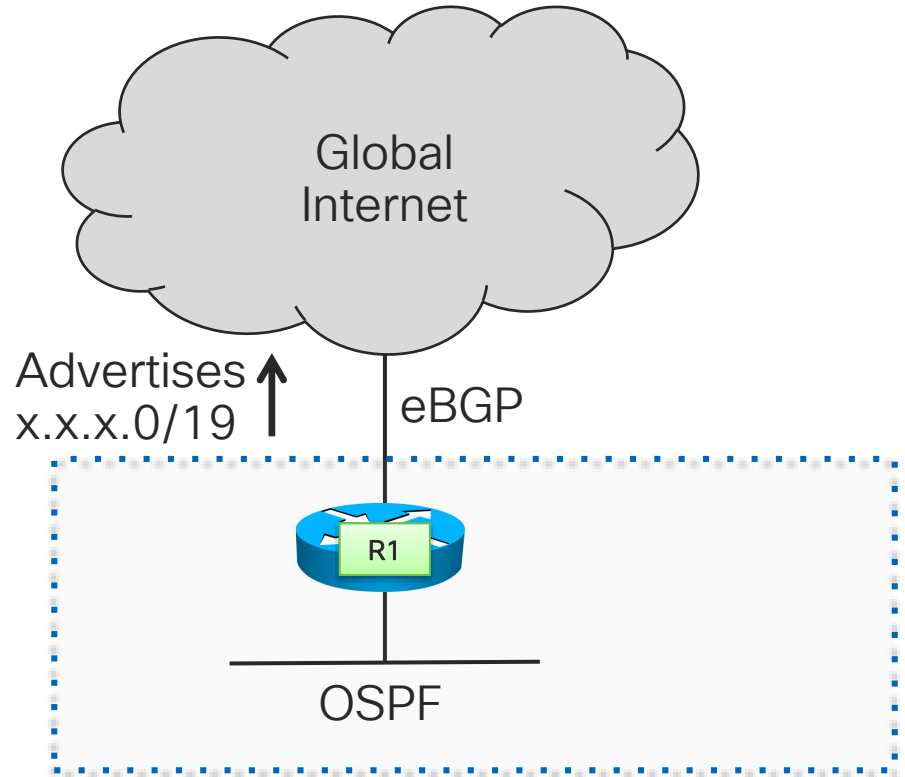
# Some Common BGP Deployment Scenarios

# Agenda

- BGP General Operation
  - Overview
  - EBGP and IBGP
- Attributes and Best Path Selection Algorithm
  - Route Origination
  - AS-PATH
  - NEXTHOP
  - Communities
- Controlling Traffic
  - Controlling Outbound Traffic
  - BGP Multipath
  - Controlling Inbound Traffic
- Route Reflectors
- Multiprotocol BGP
- Common BGP Deployments
- Securing BGP
- BGP Routing Convergence
- Show and Tell/Demo Lab

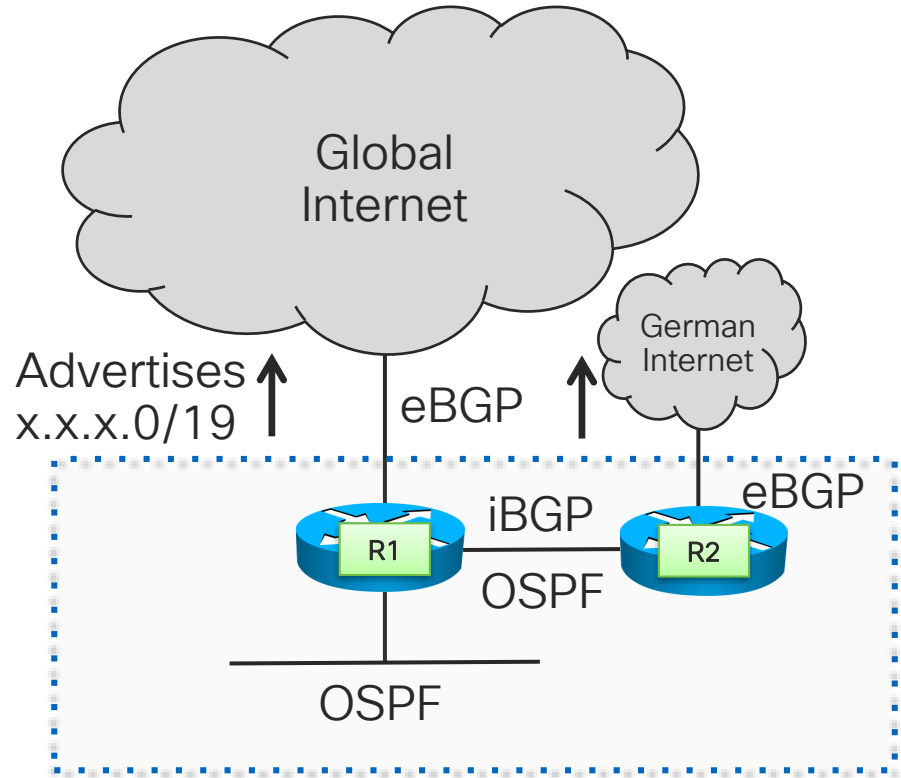
# My First BGP Deployment (circa 1995)

- Advertise our own /19 prefix to a US Service Provider
- That's it.
- Why did we do this?  
Because we could.



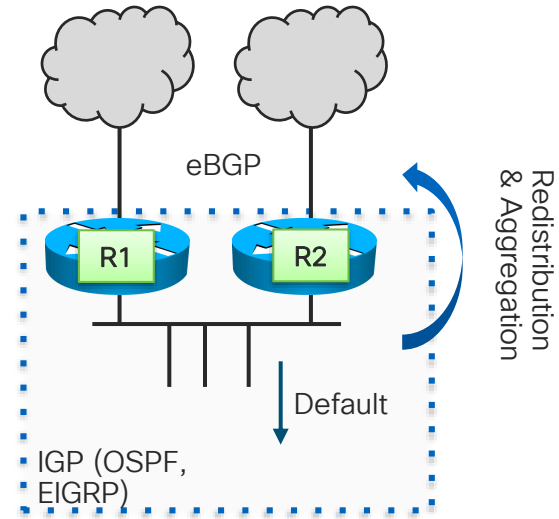
# My Second BGP Deployment (circa 1995 + a few months)

- Add a connection to DE-CIX, German IXP in Frankfurt
- Now this started to make real sense!



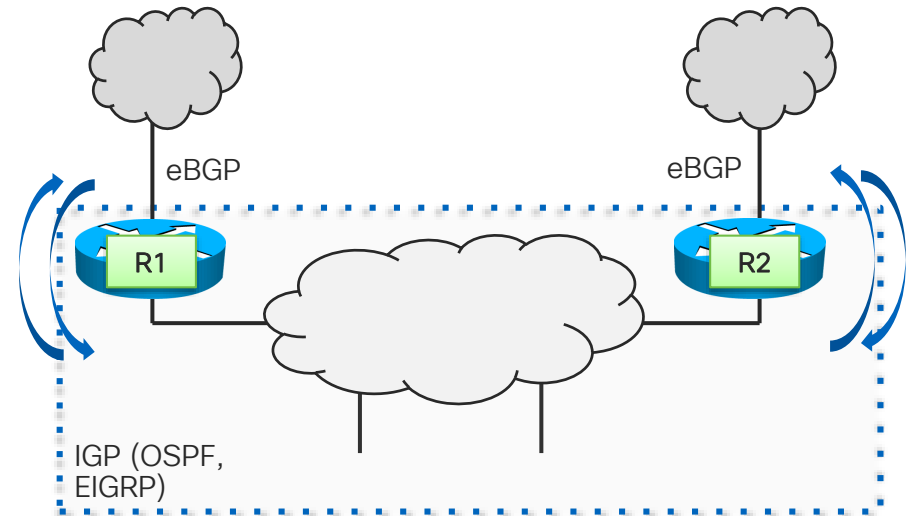
# Enterprise BGP Deployments – Central Exit

- Run eBGP towards external networks (Internet, partners) in a central place
- Originate static default route inside the AS to attract traffic
- Redistribute & aggregate your own prefix(es)



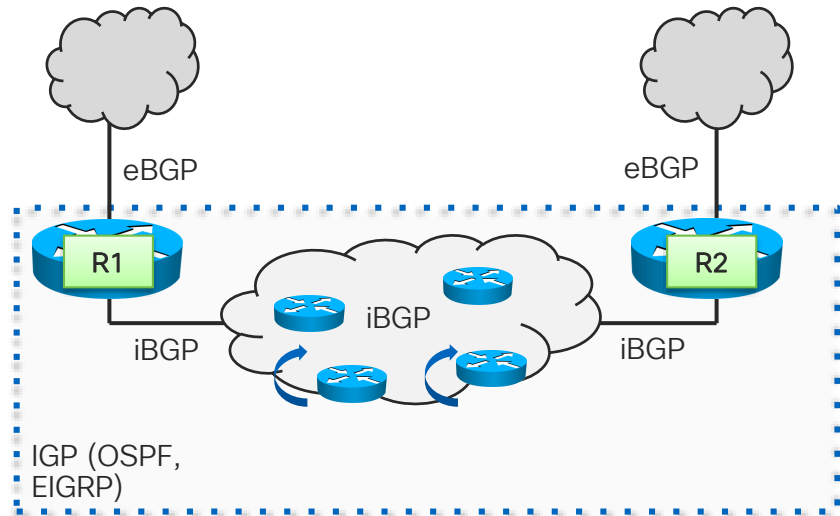
# Enterprise BGP Deployments – Distributed Exits

- Spread external access across multiple sites
- Originate dynamic default route inside the IGP to attract traffic
  - Conditional default based on external BGP routes
  - Or ask your eBGP peer to advertise a default
- Redistribute & aggregate your own prefix into eBGP
- Do NOT redistribute eBGP routes into your IGP unless you know exactly what you're doing (now and in the future)



# Enterprise BGP Deployments – Running iBGP within the AS

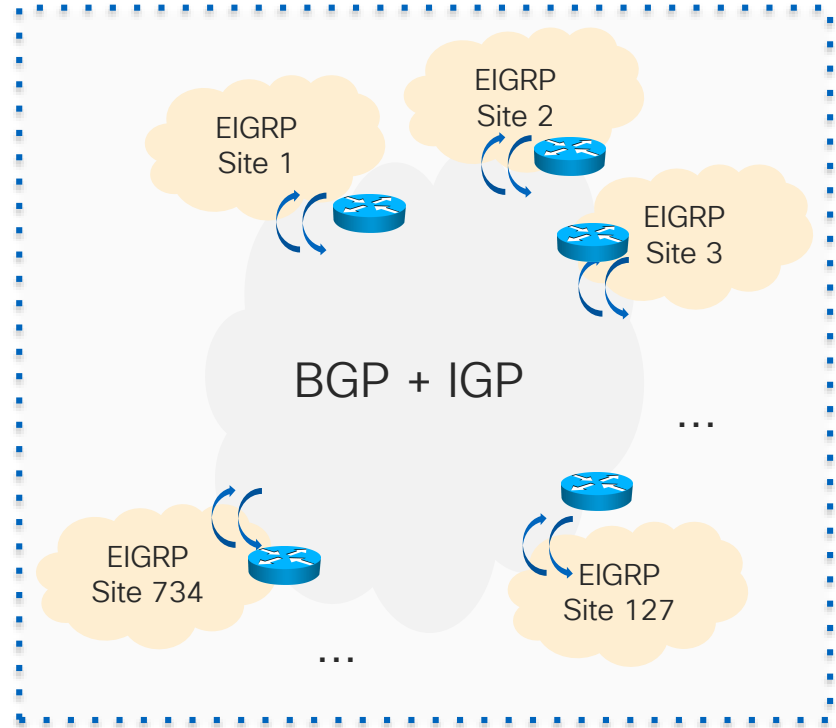
- Here we start to run iBGP within the AS and carry (selected?) eBGP-learned external destinations there
  - IGP continues to carry internal destinations
  - Redistribution/Aggregation of your own prefix can be done anywhere within your iBGP AS
- Amount of route and forwarding memory in core network device determine scale
  - Typically you don't want to carry a full routing table of ~700.000 entries in your network
  - MPLS/LDP can help to keep BGP off your core routers





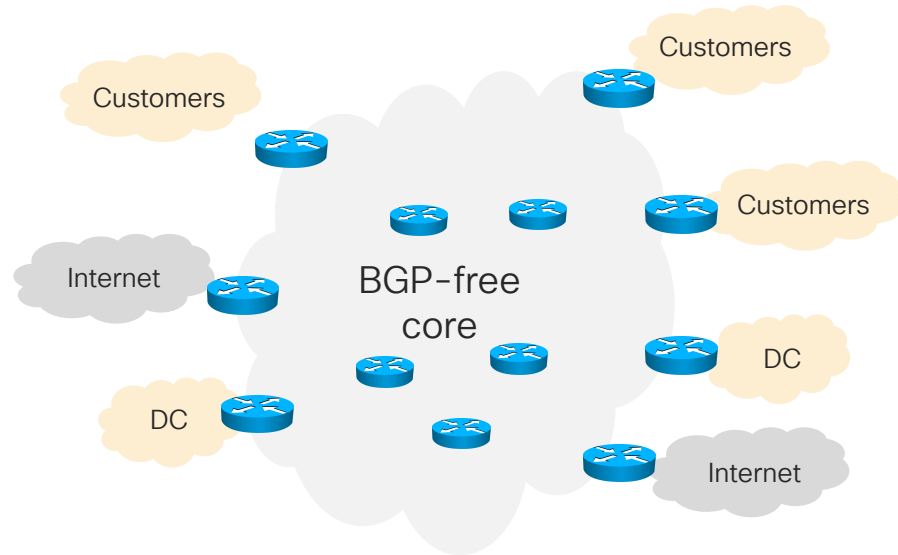
# Enterprise BGP Deployments – Scaling your IGP

- Another use case has started to evolve when IGP networks have started to grow and scale globally
- When IGP scale mechanisms (route aggregation/summarization, etc.) no longer work, folks started to partition their IGP domains and use BGP
- As always, be cautious when doing 2-way route redistribution
- The very same idea is used to scale MPLS-VPN



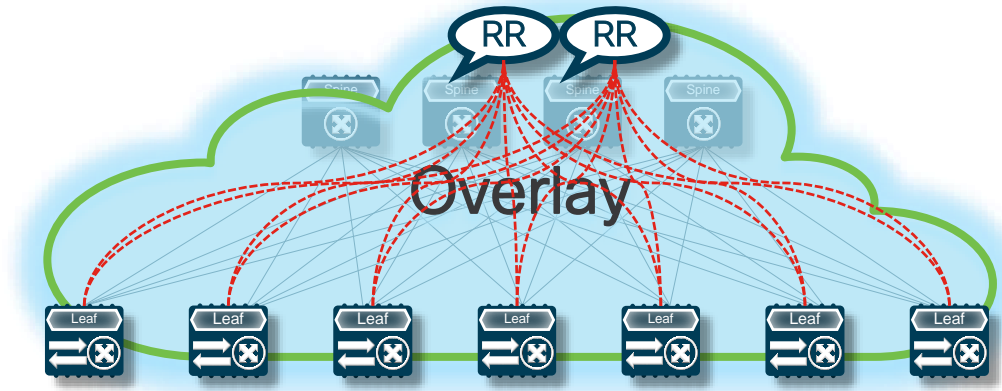
# Service Provider BGP Deployments – Designed for Scale

- Goal: Keep your IGP as lean as possible
  - → Move everything into BGP
  - → IGP is only used to carry infrastructure addresses (BGP next-hops)
- MPLS or Segment Routing allows to run BGP-free core
- Does this design achieve enough scale for today's networks?
  - No, but this is beyond this session's scope
  - Read up on “Unified MPLS” if you're interested

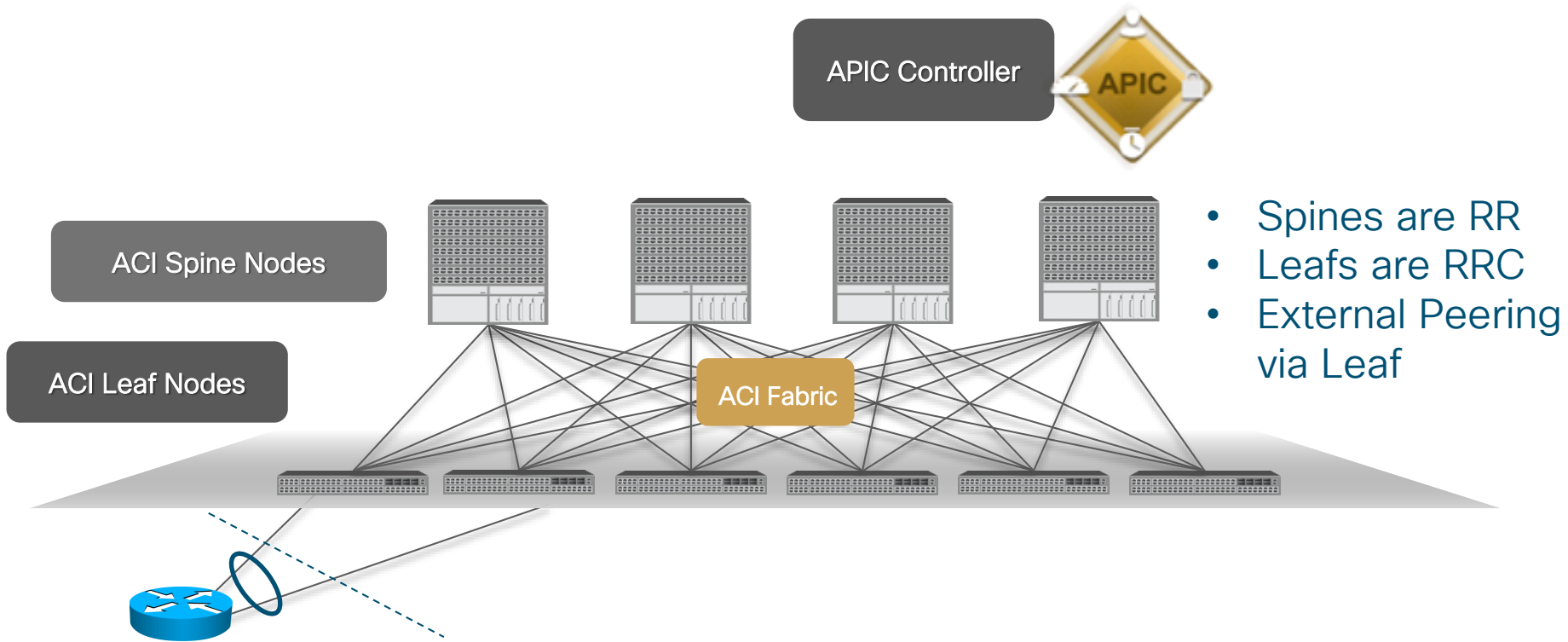


# BGP in the Data Center

- EVPN – MP-BGP distributes host and endpoint information
- Route-Reflectors (RR) deployed for scaling purposes
- Leaf nodes run BGP, advertise attached endpoint host route/information



# Overview of the ACI Fabric



# Securing BGP

*With great power comes great  
responsibility*

Voltaire (as well as Peter Parker's Uncle Ben)

# Primer: Securing BGP & Operations (1)

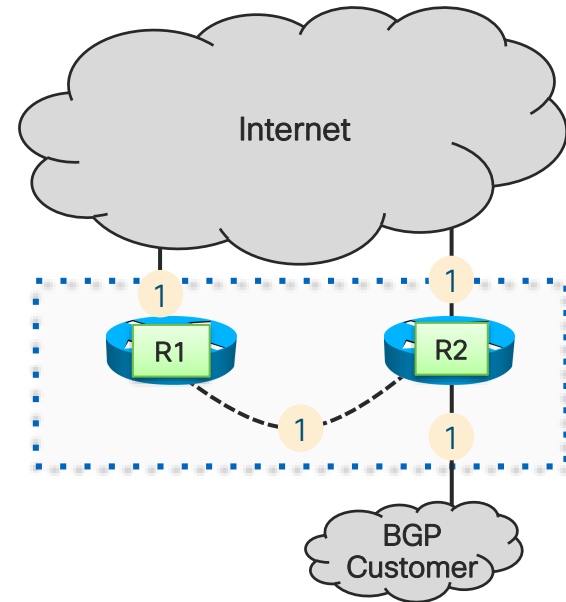
If we can ask you for three things before you deploy BGP towards the Internet:

1) Authenticate **all** your BGP sessions (use different passwords on eBGP)

```
router bgp 10
  neighbor xxxxxx password yyyyyyy
```

*or use TCP Authentication Option if available on both peers*

```
router bgp 10
  neighbor xxxxxx ao keychain_name
```



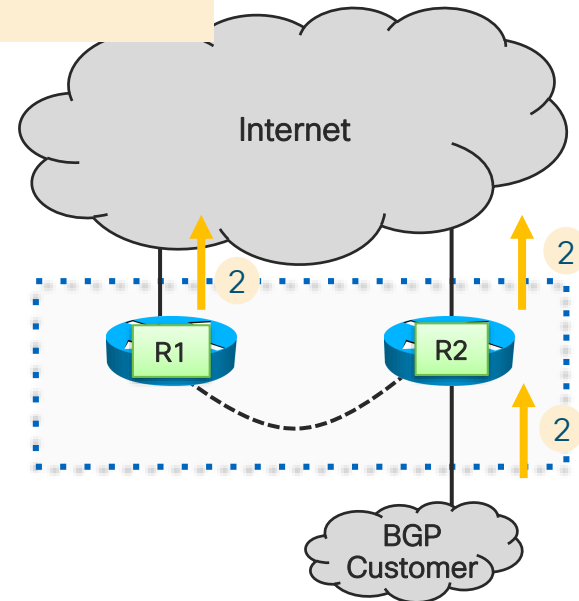
# Primer: Securing BGP & Operations (2)

## 2) Filter announcements on **all eBGP** connections

To the internet and from downstream customers: **very restrictive prefix filters** (with a final, potentially implicit “deny any”)

```
ip prefix-list cust seq 5 permit 10.0.0.0/24 le 32
ip prefix-list cust seq 10 permit 10.0.4.0/23 le 32
!
route-map from-customer permit 10
  match ip address prefix-list cust
!
ip prefix-list my-nets seq 5 permit 172.17.32.0/27
ip prefix-list my-nets seq 10 permit 10.0.0.0/24
ip prefix-list my-nets seq 15 permit 10.0.4.0/23
!
route-map to-internet permit 10
  match ip address prefix-list my-nets

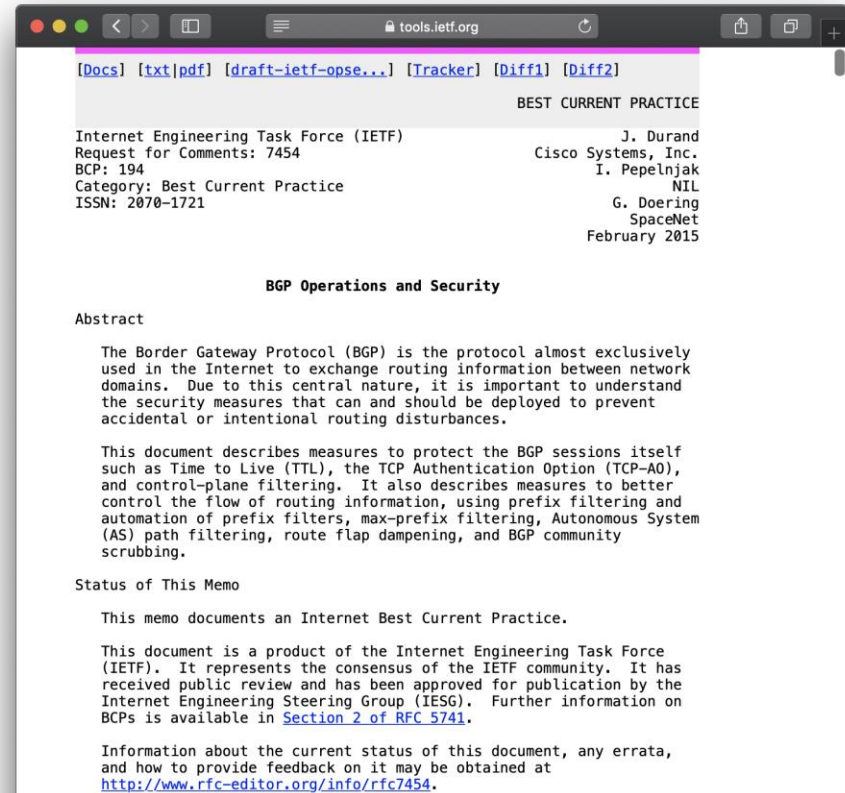
router bgp 10
  address-family ipv4 unicast
  neighbor <customer> route-map from-customer in
  neighbor <internet> route-map to-internet out
```





# Primer: Securing BGP & Operations (3)

3) Read RFC7454 / BCP194 to learn and apply this and more security mechanisms to your BGP deployment



[Docs] [txt|pdf] [draft-ietf-opse...] [Tracker] [Diff1] [Diff2]

BEST CURRENT PRACTICE

Internet Engineering Task Force (IETF)  
Request for Comments: 7454  
BCP: 194  
Category: Best Current Practice  
ISSN: 2070-1721

J. Durand  
Cisco Systems, Inc.  
I. Pepelnjak  
NIL  
G. Doering  
SpaceNet  
February 2015

**BGP Operations and Security**

**Abstract**

The Border Gateway Protocol (BGP) is the protocol almost exclusively used in the Internet to exchange routing information between network domains. Due to this central nature, it is important to understand the security measures that can and should be deployed to prevent accidental or intentional routing disturbances.

This document describes measures to protect the BGP sessions itself such as Time to Live (TTL), the TCP Authentication Option (TCP-AO), and control-plane filtering. It also describes measures to better control the flow of routing information, using prefix filtering and automation of prefix filters, max-prefix filtering, Autonomous System (AS) path filtering, route flap dampening, and BGP community scrubbing.

**Status of This Memo**

This memo documents an Internet Best Current Practice.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on BCPs is available in [Section 2 of RFC 5741](#).

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at <http://www.rfc-editor.org/info/rfc7454>.

# BGP Routing Convergence

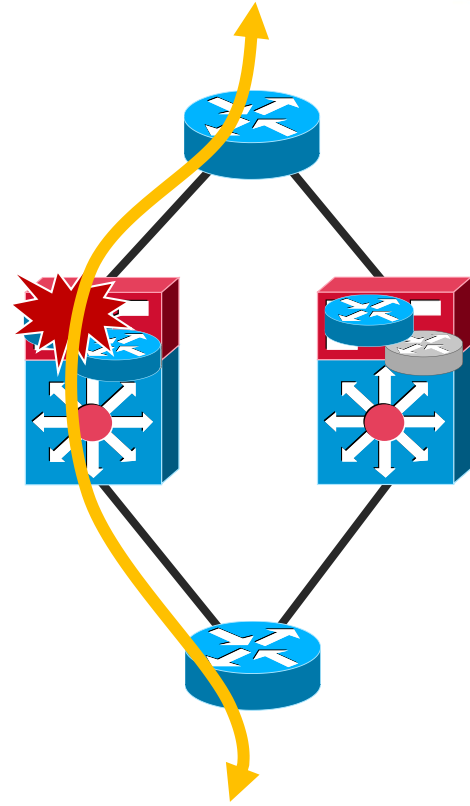
# Agenda

- BGP General Operation
  - Overview
  - EBGP and IBGP
- Attributes and Best Path Selection Algorithm
  - Route Origination
  - AS-PATH
  - NEXTHOP
  - Communities
- Controlling Traffic
  - Controlling Outbound Traffic
  - BGP Multipath
  - Controlling Inbound Traffic
- Route Reflectors
- Multiprotocol BGP
- Common BGP Deployments
- Securing BGP
- BGP Routing Convergence
- Show and Tell/Demo Lab

# What is Routing High Availability?



- Set of technologies & features to enable traffic to continue to flow **through** a device during a fault
- Routing HA **maintains** the logical network topology while the faulty device **recovers**
- Routing HA helps to address failures within the **control plane** of a routing device
- Routing HA increases the **resiliency** of a single system

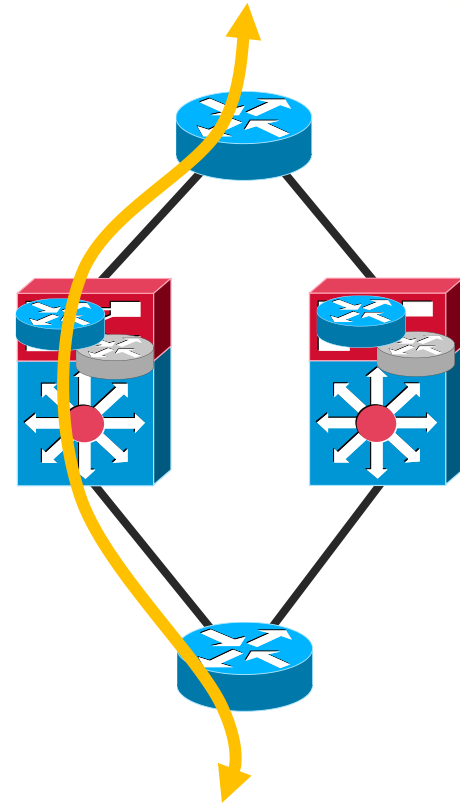


We're not covering this today!

# What is Routing Fast Convergence?



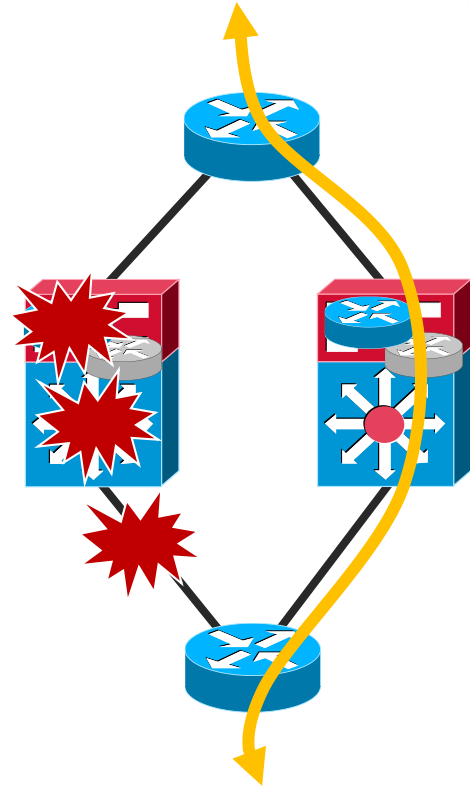
- Set of technologies & features to enable traffic to continue to flow **around** a device during a fault
- Routing FC **adapts** the logical network topology to avoid the faulty component
- Routing FC targets to address **any component** failure within a routing device
- Routing FC increases the **resiliency** of the network



# What is Routing Fast Convergence?



- Set of technologies & features to enable traffic to continue to flow **around** a device during a fault
- Routing FC **adapts** the logical network topology to avoid the faulty component
- Routing FC targets to address **any component** failure within a routing device
- Routing FC increases the **resiliency** of the network



# IGP vs. BGP Convergence

- IGP (OSPF/ISIS) deals with hundreds routes
  - Max a few thousands, but only a few hundreds are really important/relevant
- BGP is designed to carry millions of routes
  - and these days several customers carry that amount of prefixes!
- We can tune IGP to converge in  $\ll 1$  second
- But how about BGP?

# BGP Control-Plane Convergence Components

Convergence =

Failure Detection + Event Propagation + Routing Process + FIB Update

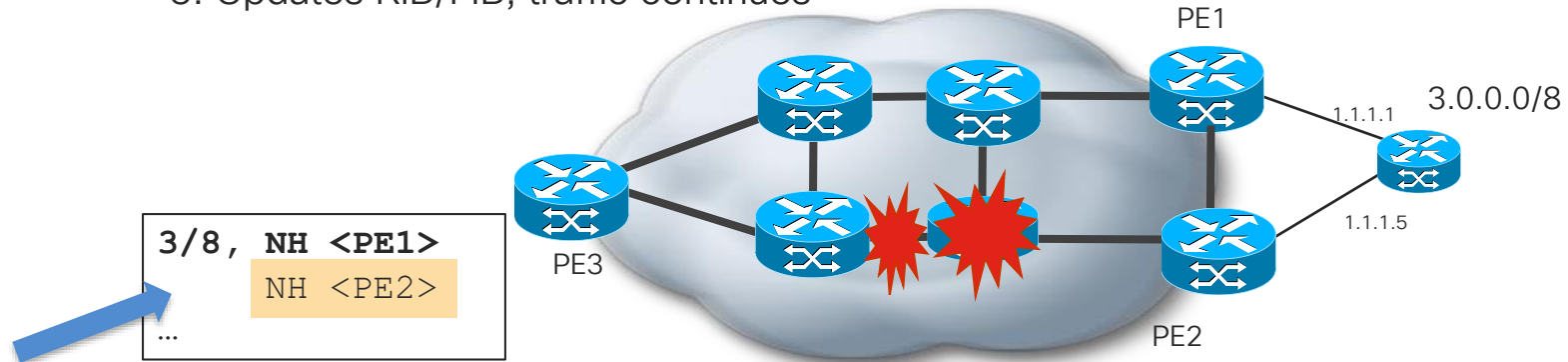




# BGP Control-Plane Convergence Components I: Core failure

1. Core Link or node goes down
2. IGP notices failure, computes new paths to PE1/PE2
3. IGP notifies BGP that a path to a next-hop has changed
4. PE3 identifies affected paths, runs best path, path to PE2 no longer as good as the one to PE1
5. Updates RIB/FIB, traffic continues

```
3/8, NH 1.1.1.1
      NH <PE2>
...
```



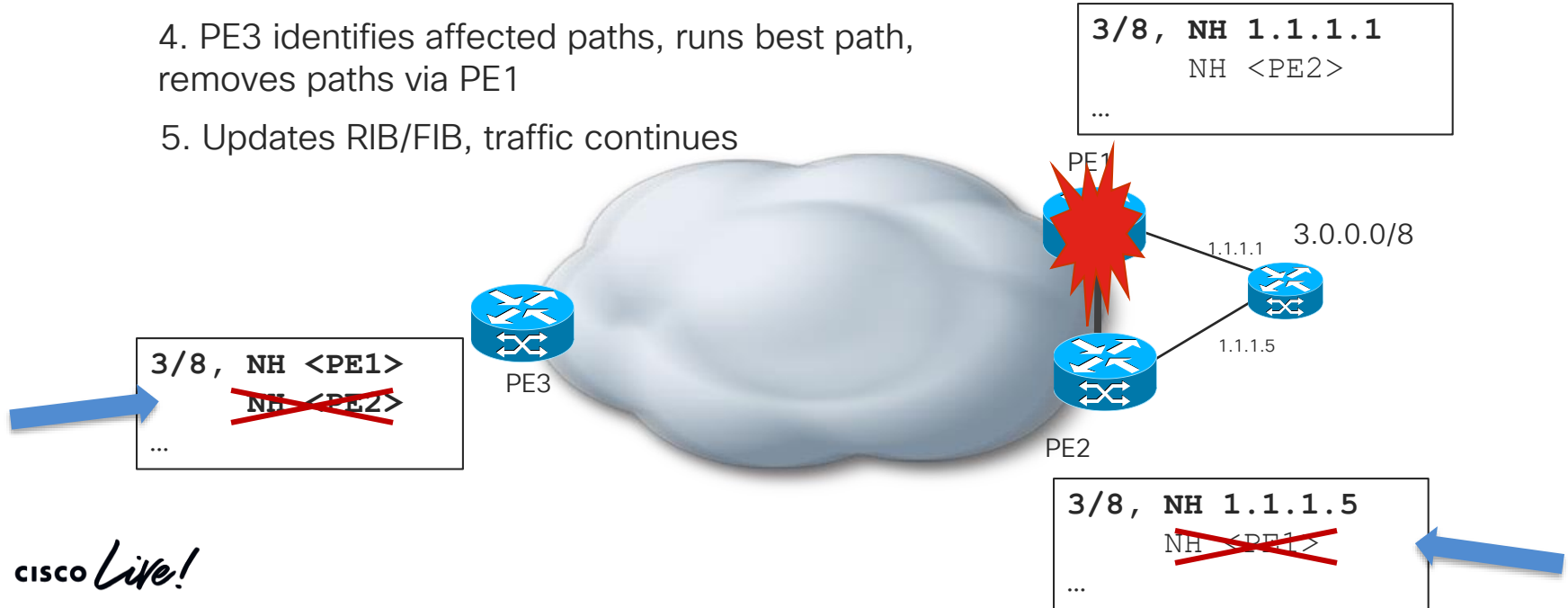
```
3/8, NH <PE1>
      NH <PE2>
...
```

```
3/8, NH 1.1.1.5
      NH <PE1>
...
```

# BGP Control-Plane Convergence Components II: Edge Node Failure

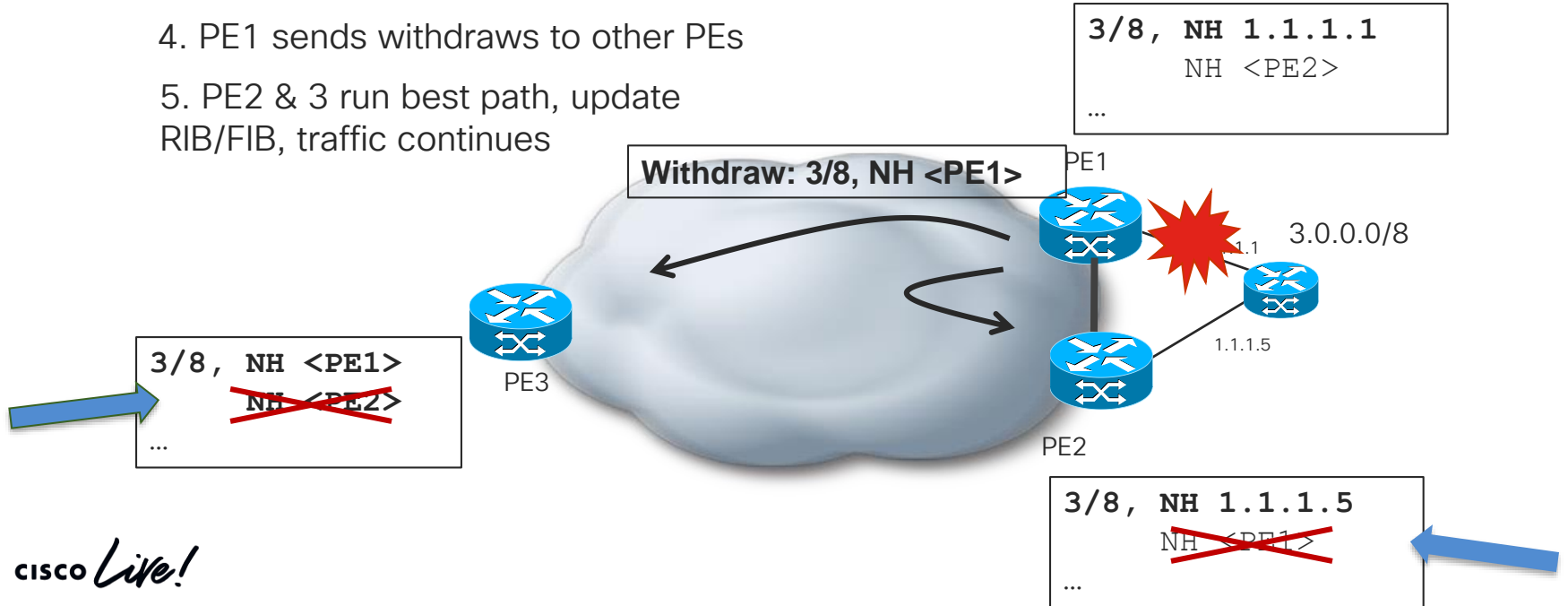
Edge Node (PE1) goes down

2. IGP notices failure, update RIB, remove path to PE1
3. IGP notifies BGP that path to PE1 is now longer valid
4. PE3 identifies affected paths, runs best path, removes paths via PE1
5. Updates RIB/FIB, traffic continues



# BGP Control-Plane Convergence Components III: Edge Neighbour Failure (with next-hop-self)

1. Edge link on PE1 goes down
2. eBGP session goes down
3. PE1 identifies affected paths, runs best path
4. PE1 sends withdraws to other PEs
5. PE2 & 3 run best path, update RIB/FIB, traffic continues



# Failure Detection (Edge Failures)

- Problem: Detect an eBGP neighbour failure
- Available Methods
  1. Fast External Fallover – monitors line protocol for directly connected neighbours (**default behaviour**)
  2. Fast Session Deactivation (FSD), monitors routing table for reachability of next-hop address (eBGP multi-hop)
  3. “Hello”-type protocols: **BFD** and BGP Hello (don’t tune BGP hello’s, use BFD instead)

```
router bgp ...  
  [no] bgp fast-external-fallover  
interface ...  
  ip bgp fast-external-fallover {permit|deny}
```

```
router bgp ...  
  neighbor x.x.x.x fall-over
```

```
router bgp ...  
  timers bgp <hello> <hold>  
  neighbor <..> timers <hello> <hold>  
  
  neighbor <..> fall-over bfd
```

# Failure Detection – Next-Hop Failure



- Goal: Detect next-hop failures (as carried in IGP)
- Methods:
  - Next-hop Tracking, enabled by default
  - BGP scanner (legacy, very slow reaction)
- Note: On most cases, we do **not** want to use iBGP hellos to detect internal/iBGP neighbor failures, and instead rely on next-hop reachability checks

# Update Propagation

- Once failed paths are identified and best-path has been run, updates/withdrawals need to be sent to peers
- Goal: maximize TCP throughput and update packing/replication
- Design Guidance:
  - Reduce minimum advertisement interval to zero (already default in many recent releases)

```
neighbor x.x.x.x advertisement-interval 0
```

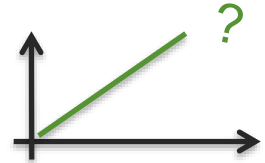
- Enable TCP PMTUD (default in recent software) and increase window-size

```
ip tcp path-mtu-discovery  
ip tcp window-size 65535
```

- Use peer-groups/peer-templates

# BGP Control-Plane Convergence – Calculating new Path

- Once BGP is notified of the change, and needs to identify affected paths by scanning the BGP table(s) and perform best path calculation to find the new best path
- We can do some optimization on scanning, but scanning effort will still grow with table size
- Hence: We need to find another approach to achieve **predictable** fast BGP convergence!



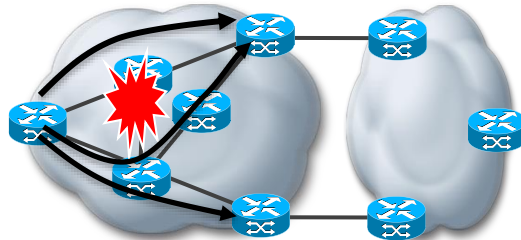
# Control vs. Data Plane Convergence – BGP PIC

## ■ Control Plane Convergence

- For the topology after the failure, the **optimal path** is known and installed in the dataplane

## ■ Data Plane Convergence

- Once IGP convergence has detected the failure, the packets are rerouted onto a **valid path** to the BGP destination
- While valid, this path may not be the most optimum one from a control plane convergence viewpoint
- BGP PIC can deliver this behaviour, in a **prefix-independent** way, no matter if BGP carries 1000 or 1,000,000 prefixes!



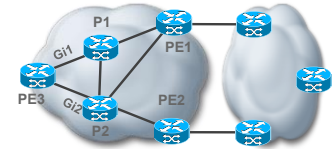
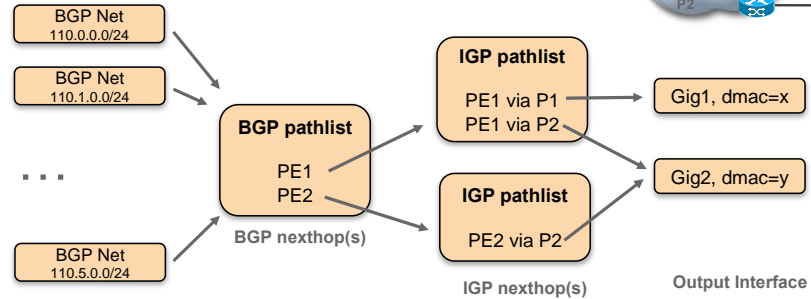


# Deploying BGP Fast Convergence / BGP PIC

Oliver Boehmer, BRKIPM-2265

Watch the recording on [ciscolive.com](https://www.ciscolive.com)

## BGP Prefix Independent Convergence (BGP PIC)



- Pointer Indirection between BGP and IGP entries allow for immediate update of the multipath BGP pathlist at IGP convergence
- Only the parts of FIB actually affected by a change needs to be touched
- Used in newer IOS and IOS-XR (all platforms), enables Prefix Independent Convergence

# Wrapping Up

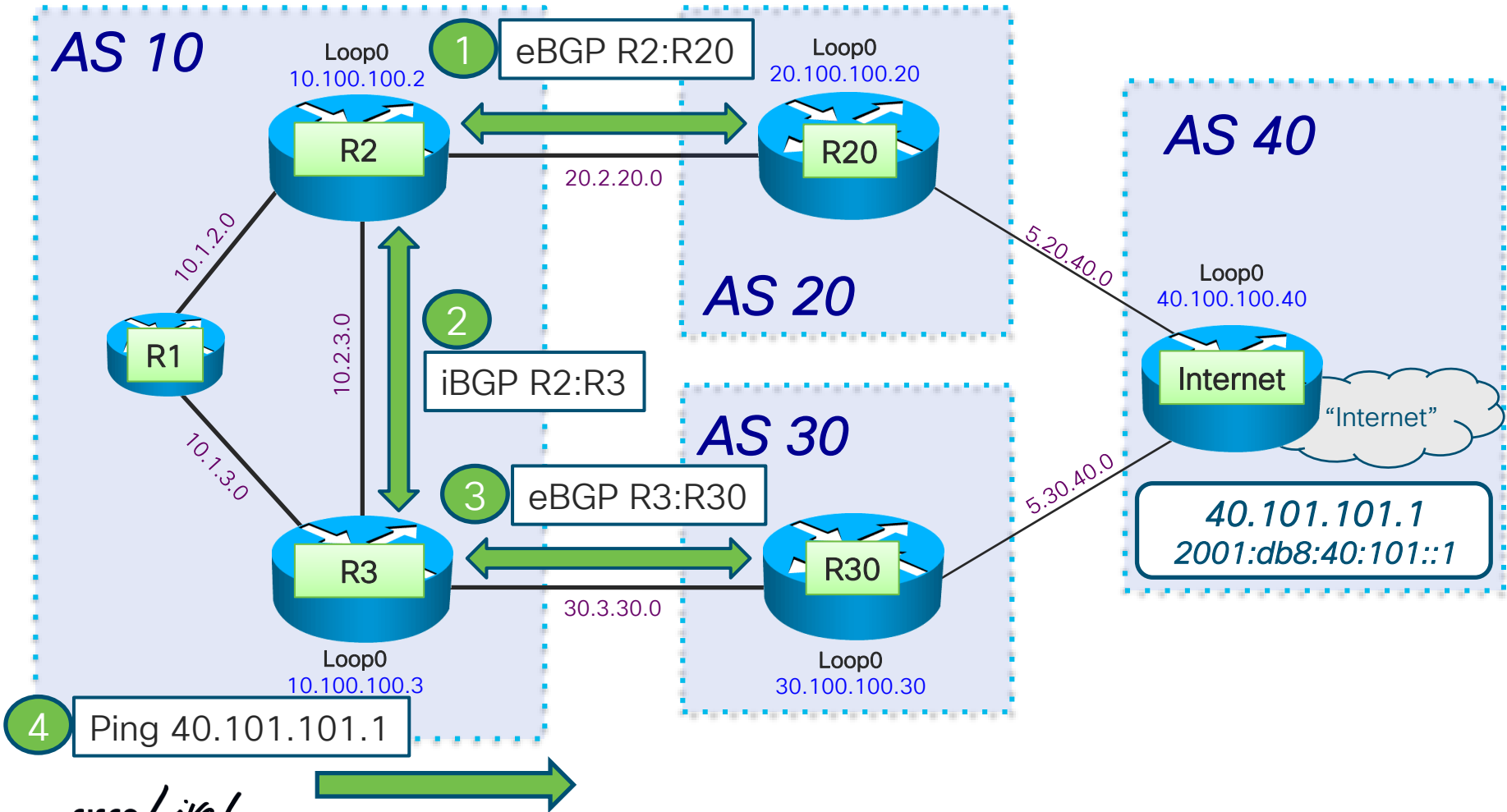
... or why we think BGP is close to the  
best thing since sliced bread

# Wrapping up...

- BGP is old, but proven! And scalable!
- BGP is THE protocol to implement complex routing policies, but comes with enough rope to hang yourself
- BGP can extend to carry much more than IP reachability information, it is used everywhere these days!
- BGP can't live alone, it usually requires an underlying IGP
- BGP can react very fast to failures
- BGP requires hands-on and practice, so lets dive right into it...

”Show and Tell”

# AS 10



# BGP Show and Tell: Beginners

<https://www.youtube.com/playlist?list=PLVuziKI5zsd6VW41III3SWC3nT1oISZBj>

The screenshot displays a video player interface. The main content is a network diagram titled "bgp\_lab\_2.pdf (1 page)". The diagram illustrates a multi-AS network topology with the following components:

- AS 10:** Contains routers R1 (IOSv), R2 (IOSv), and R3 (IOSv). R1 is connected to R2 and R3. R2 and R3 are also connected to each other. Loop0 addresses are 10.100.100.1 (R1), 10.100.100.2 (R2), and 10.100.100.3 (R3).
- AS 20:** Contains router R20 (IOS XR). Loop0 address is 20.100.100.20.
- AS 30:** Contains router R30 (IOS XR). Loop0 address is 30.100.100.30.
- AS 40:** Contains router Internet (IOSv). Loop0 address is 40.100.100.40.

Inter-AS connections and IP addresses are shown:

- AS 10 R2 (G0/0/20) to AS 20 R20 (G0/0/20): 20.2.20.0
- AS 10 R2 (G0/0/20) to AS 30 R30 (G0/0/20): 2001.db8:2:2::/2128
- AS 10 R3 (G0/0/20) to AS 30 R30 (G0/0/20): 30.3.30.0
- AS 10 R3 (G0/0/20) to AS 40 Internet (G0/0/20): 2001.db8:3:3::/128
- AS 20 R20 (G0/0/20) to AS 40 Internet (G0/0/20): 5.20.40.0
- AS 30 R30 (G0/0/20) to AS 40 Internet (G0/0/20): 5.30.40.0

The diagram also shows intra-AS connections within AS 10 and the VIRL logo at the bottom right.

On the right side of the video player, a playlist titled "BGP Show and Tell: Beginners" is visible, containing 5 videos:

- BGP Show n Tell #3: eBGP Peer (IPv4 and IPv6) R3 and R30
- BGP Show and Tell #2: iBGP Peer (IPv4 and IPv6) R2 and R3
- BGP Show n Tell #1: eBGP Peer (IPv4 and IPv6) R2 and R20
- BGP Show n Tell #4: Shut Link Between R2 and R20
- BGP Show n Tell #5: Ping the IPv4 & IPv6 Internet Addresses from R1



# Continue your education



Demos in the  
Cisco Showcase



Walk-In Labs



Meet the Engineer  
1:1 meetings



Related sessions





Thank you





You make **possible**