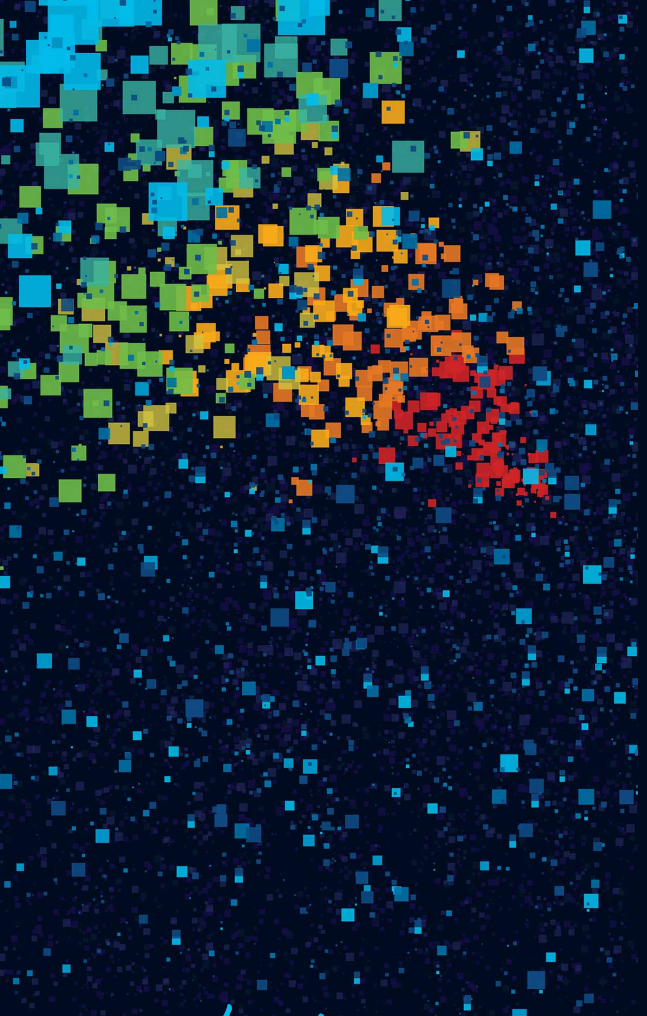
Possibilities

#CiscoLive

OSPF Deployment in Modern Networks

Nick Russo, Technical Leader
CCIE 42518 (RS/SP), CCDE 20160041
@nickrusso42518

DGTL-BRKRST-2337



Agenda

- Designing an Enterprise Network with Areas
- Interacting with BGP at the Internet Edge
- Scaling OSPF in Hub/Spoke Networks
- Integrating Partners using Extranets
- Optimizing OSPF for Service Providers
- Conclusion

What this session is all about

- Relatable scenarios
- Expand your thinking
- Mostly analysis/design
- A little config/validation

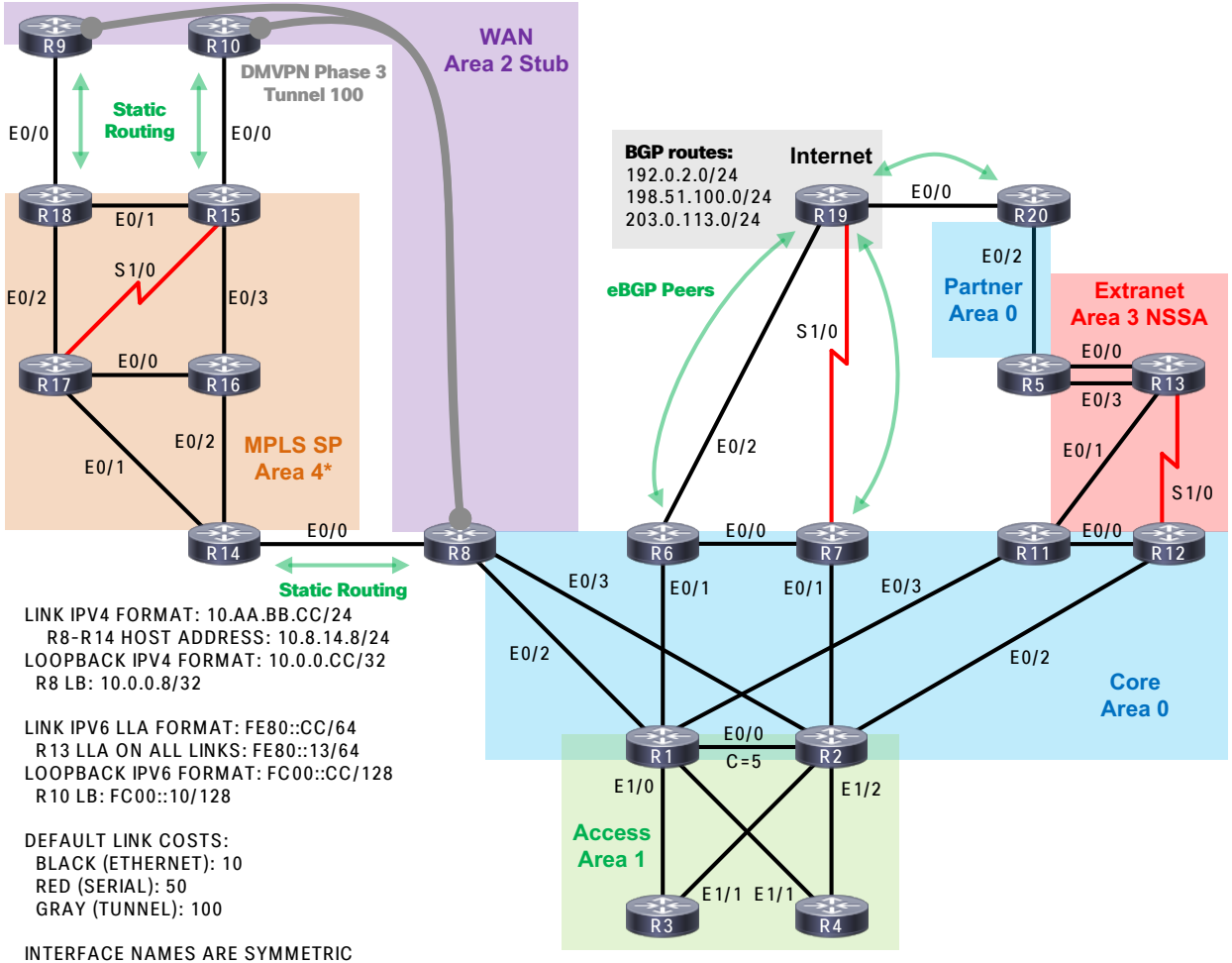
What this session is not about

- A fact recitation
- An exploration into every possible design choice
- An exhaustive list of every OSPF detail
  - Bits, flags, codes, and RFCs
- Reference icon in upper right hand corner

Things you should know beforehand

- Intermediate OSPF skills
  - LSA types
  - Network types
  - Area types
  - Routing preference
  - Summarization
  - Filtering
  - Redistribution
- Basic DMVPN, NAT, EIGRP
- Basic IPv6 addressing

Topology



LINK IPV4 FORMAT: 10.AA.BB.CC/24
R8-R14 HOST ADDRESS: 10.8.14.8/24
LOOPBACK IPV4 FORMAT: 10.0.0.CC/32
R8 LB: 10.0.0.8/32

LINK IPV6 LLA FORMAT: FE80::CC/64
R13 LLA ON ALL LINKS: FE80::13/64
LOOPBACK IPV6 FORMAT: FC00::CC/128
R10 LB: FC00::10/128

DEFAULT LINK COSTS:
BLACK (ETHERNET): 10
RED (SERIAL): 50
GRAY (TUNNEL): 100

INTERFACE NAMES ARE SYMMETRIC

Business Scenario

Our Client

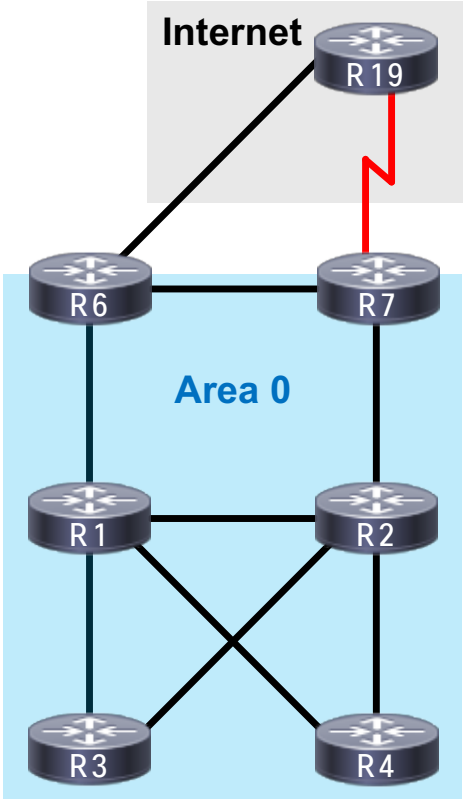
- Consultant hired by Smart Building Central Inc. (SBC)
- Designs smart lighting systems
- Occupies large campus
- Pursuing a long-term vertical integration strategy
  - Manufacturing
  - Retail
  - Distribution

OSPF Area Design

Challenge: Campus Access

- SBC employees experience slow network
- High CPU utilization due to OSPF
- Area 0 includes hundreds of old devices

- Solution must:
  - Be scalable and standards-based
  - Provide optimal routing to/from the Internet
  - Tolerate an access switch uplink failure



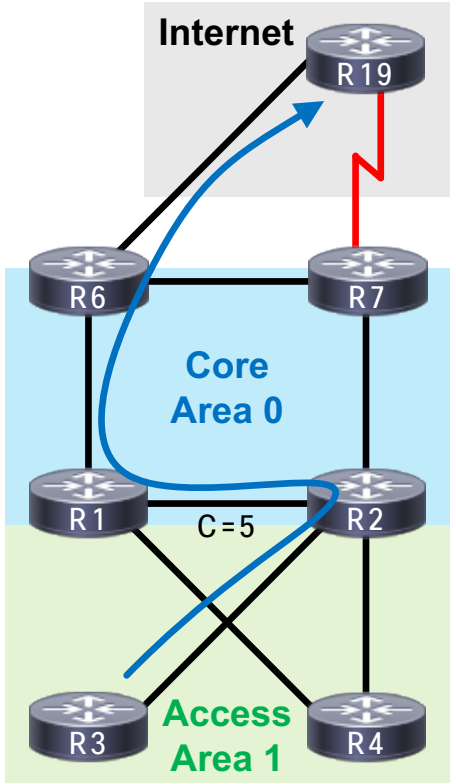
Area 0 Upstream

- R1/R3 link has failed
- R1/R2 link in area 0

- R3 must choose R2

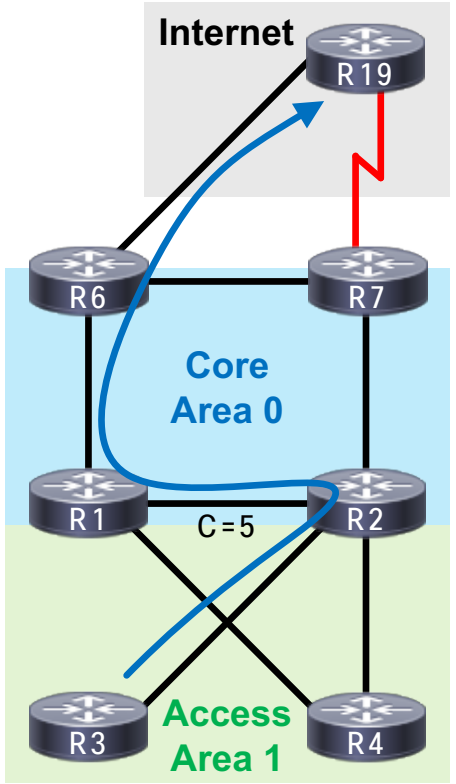
- R2 paths to R6:
  - via R1, cost 15
  - via R7, cost 20

- R1 must choose R6



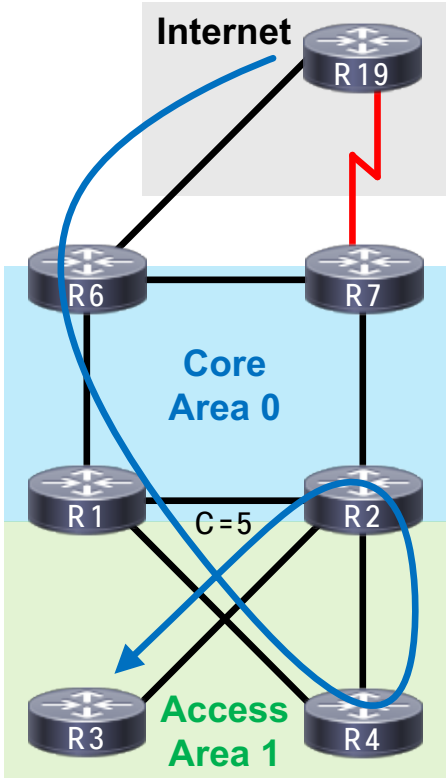
Area 0 Upstream

```
R3#traceroute 192.0.2.19
Tracing the route to 192.0.2.19
  1 10.2.3.2 1 msec 0 msec 0 msec
  2 10.1.2.1 1 msec 1 msec 0 msec
  3 10.1.6.6 1 msec 0 msec 1 msec
  4 10.6.19.19 1 msec 1 msec 1 msec

R3#show ip route | include 0.0.0.0/0
O*E2  0.0.0.0/0 [110/10] via 10.2.3.2, 00:01:31, Ethernet1/1

R2#show ip route | include 0.0.0.0/0
O*E2  0.0.0.0/0 [110/10] via 10.1.2.1, 00:02:54, Ethernet0/0

R1#show ip route | include 0.0.0.0/0
O*E2  0.0.0.0/0 [110/10] via 10.1.6.6, 00:03:06, Ethernet0/1

R6#show ip route | include 192.0.2.0/24
B     192.0.2.0/24 [20/0] via 10.6.19.19, 00:03:31
```



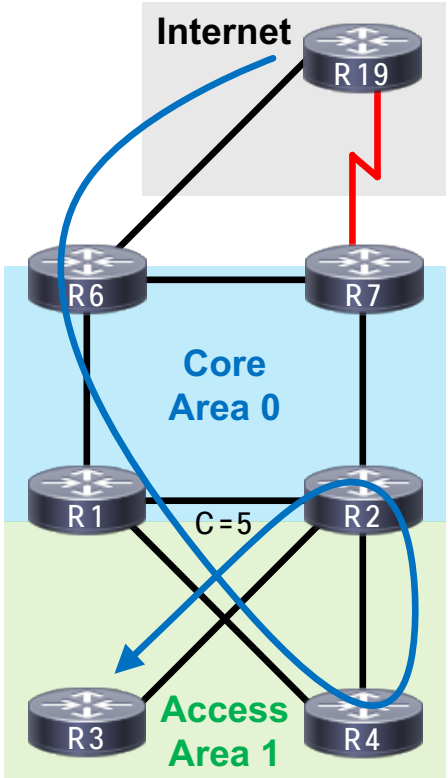
Area 0 Downstream

- Assume R19 always picks R6 (BGP)
- R6 paths to final destination (R3):
  - LSA3 from R1: cost 30
  - LSA3 from R2: cost 10
- R6 paths to ABR:
  - To R1: cost 10
  - To R2: cost 15



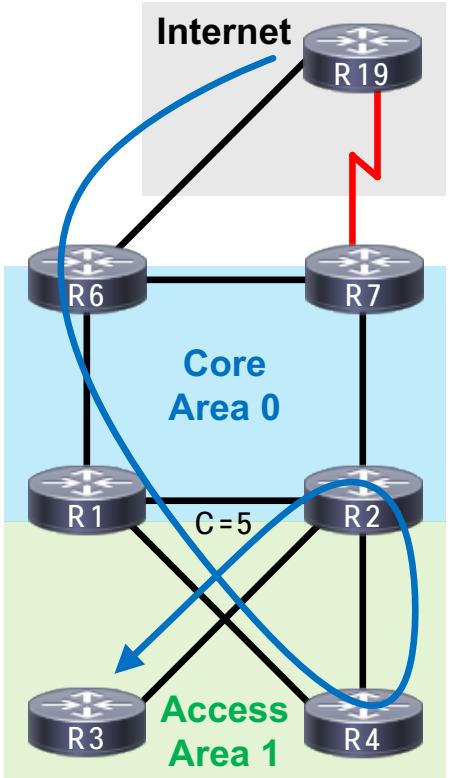
Area 0 Downstream

- Total cost via R2 (25) is less than R1 (40)
  - Plus (1) for the loopback cost itself
- Why the suboptimal routing, then?
- R1 is in the transit path to R2
  - Intercepts the packet
  - Sees intra-area route
  - OSPF prefers intra-area over inter-area



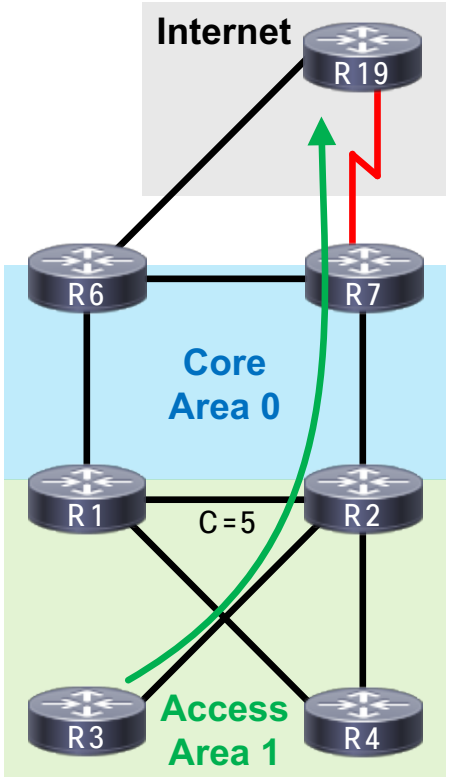
Area 0 Downstream

```
R6#traceroute 10.0.0.3
Tracing the route to 10.0.0.3
  1 10.1.6.1 5 msec 4 msec 5 msec
  2 10.1.4.4 5 msec 4 msec 4 msec
  3 10.2.4.2 6 msec 6 msec 1 msec
  4 10.2.3.3 1 msec 0 msec 1 msec

R6#show ip route | include 10.0.0.3/32
O IA 10.0.0.3/32 [110/26] via 10.1.6.1, 00:05:50, Ethernet0/1

R1#show ip route | include 10.0.0.3/32
O    10.0.0.3/32 [110/31] via 10.1.4.4, 00:06:10, Ethernet1/1

R4#show ip route | include 10.0.0.3/32
O    10.0.0.3/32 [110/21] via 10.2.4.2, 00:08:18, Ethernet1/2

R2#show ip route | include 10.0.0.3/32
O    10.0.0.3/32 [110/11] via 10.2.3.3, 00:08:29, Ethernet1/1
```



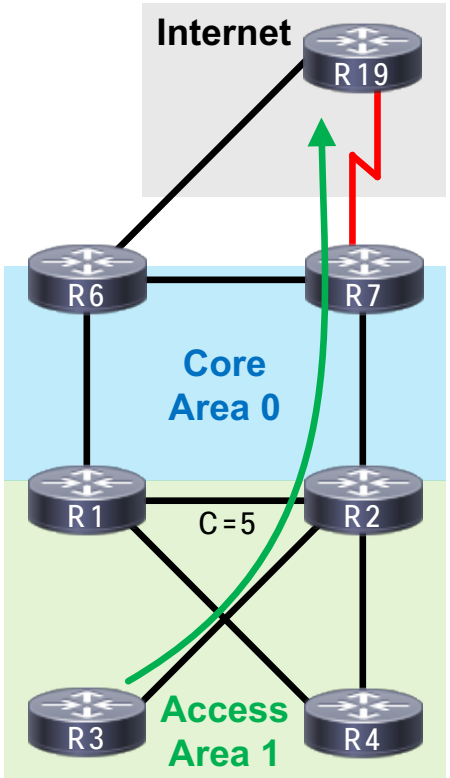
Area 1 Upstream

- Assume R6 default route is preferred
- R3 must choose R2
- R2 must choose R7
  - Another “interception”, R7 in path to R6
  - Cannot select LSA5 path via R1
    - Must prefer area 0 path to R6 through R7
    - OSPF prefers intra-area external over inter-area external
- R7 chooses R19 (assume no R6/R7 iBGP policy)



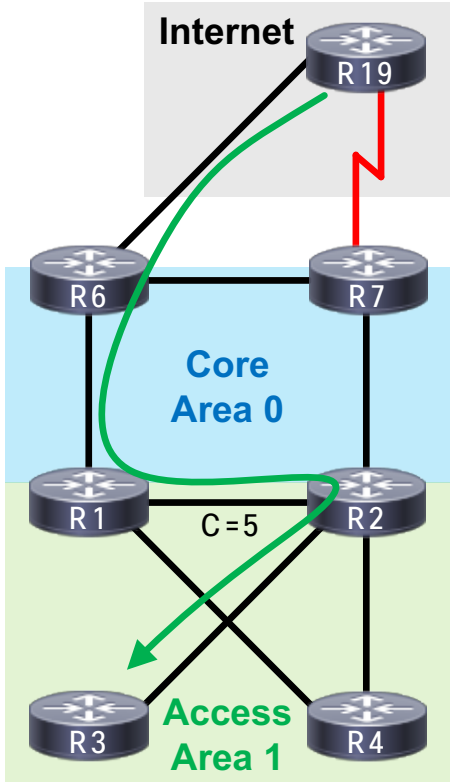
Area 1 Upstream

```
R3#traceroute 192.0.2.19
Tracing the route to 192.0.2.19
  1 10.2.3.2 0 msec 0 msec 0 msec
  2 10.2.7.7 1 msec 0 msec 0 msec
  3 10.7.19.19 7 msec 9 msec 10 msec


R3#show ip route | include 0.0.0.0/0
O*E2  0.0.0.0/0 [110/10] via 10.2.3.2, 00:12:28, Ethernet1/1

R2#show ip route | include 0.0.0.0/0
O*E2  0.0.0.0/0 [110/10] via 10.2.7.7, 00:02:36, Ethernet0/1

R7#show ip route | include 192.0.2.0/24
B     192.0.2.0/24 [20/0] via 10.7.19.19, 00:15:16
```



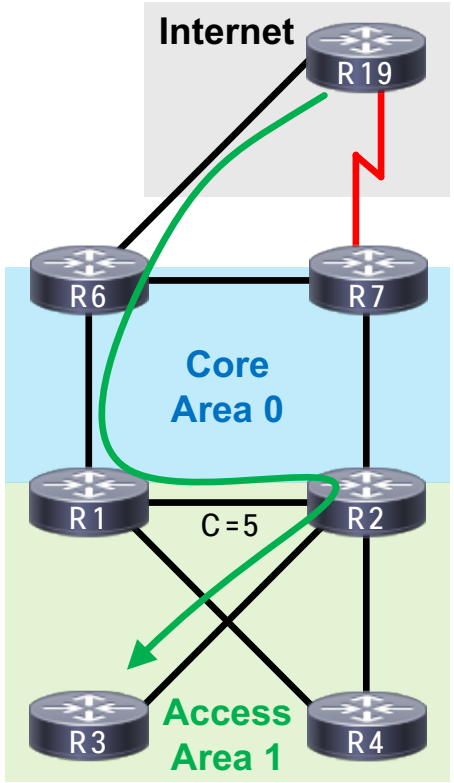
Area 1 Downstream

- Assume R19 always picks R6 (BGP)
- R6 paths to final destination (R3):
  - LSA3 from R1: cost 15
  - LSA3 from R2: cost 10
- R6 paths to ABR:
  - To R1: 10
  - To R2: 20



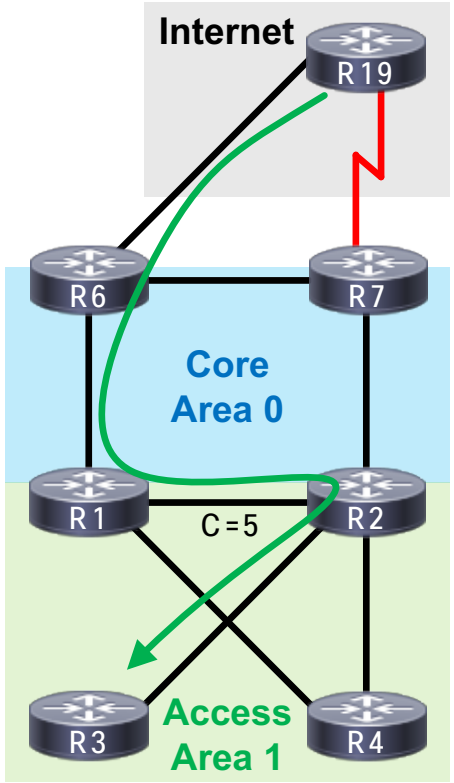
Area 1 Downstream

- Total cost via R1 (25) is less than R2 (30)
  - Plus (1) for the loopback cost itself
- Why is it suddenly better now?
- R1 is the true shortest path
  - No interception
  - R1 has intra-area path via R2



Area 1 Downstream

```
R6#traceroute 10.0.0.3
Tracing the route to 10.0.0.3
  1 10.1.6.1 1 msec 0 msec 1 msec
  2 10.1.2.2 0 msec 1 msec 1 msec
  3 10.2.3.3 0 msec 1 msec 0 msec


R6#show ip route | include 10.0.0.3/32
O IA 10.0.0.3/32 [110/26] via 10.1.6.1, 00:04:14, Ethernet0/1

R1#show ip route | include 10.0.0.3/32
O    10.0.0.3/32 [110/16] via 10.1.2.2, 00:04:26, Ethernet0/0

R2#show ip route | include 10.0.0.3/32
O    10.0.0.3/32 [110/11] via 10.2.3.3, 00:17:58, Ethernet1/1
```

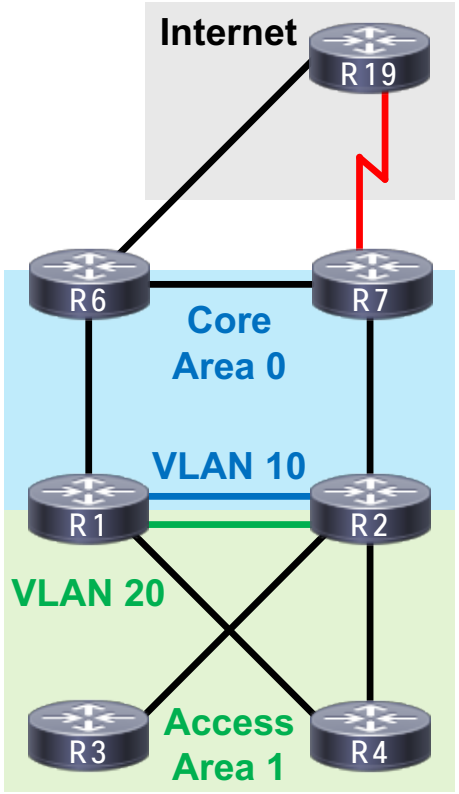


How can we fix this?

Solution 1: Link Multiplexing

- Layer-2 technology needs to support it
  - Ethernet VLAN
  - Frame-relay DLCI

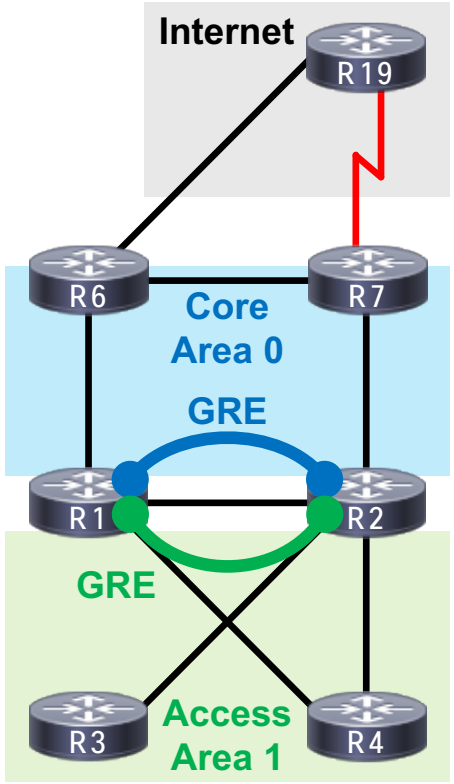
- Often is configuration intensive
  - New interfaces
  - New IPs
  - New VLAN config on intermediate switches



Solution 2: GRE Tunnels

- Multiple P2P tunnels over non-OSPF link
  - Could use IP unnumbered
  - Must use different tunnel keys
  - Works over non-multiplexed transport (PPP)

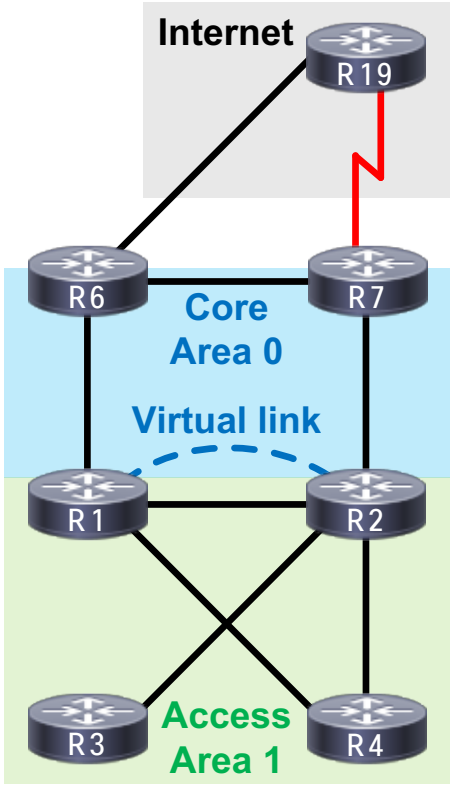
- Often is configuration intensive
  - New interfaces
  - Additional encapsulation
  - FW/IPS inspection challenges



Solution 3: OSPF Virtual-link

- No new technologies; just OSPF
  - Use non-zero area as base
  - Run VL over this link
  - No new interfaces, IPs, etc.

- Although elegant, can be tricky
  - Scales poorly; only works for one area



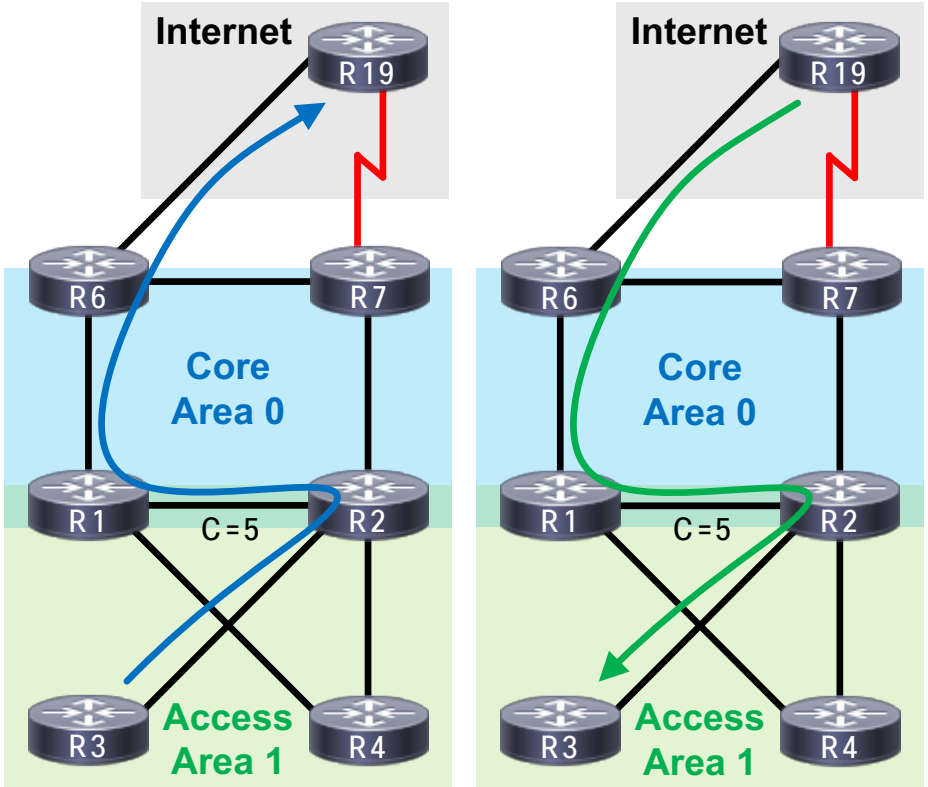
Which is best?

	Area 0	Area 1	VLANs	GRE	VL
Standardized	Yes	Yes	Yes	Yes	Yes
Scalable	Yes	No	Semi	Semi	No
Optimal	Upstream	Downstream	Yes	Yes	Yes

Better way: Multi-area Adjacency

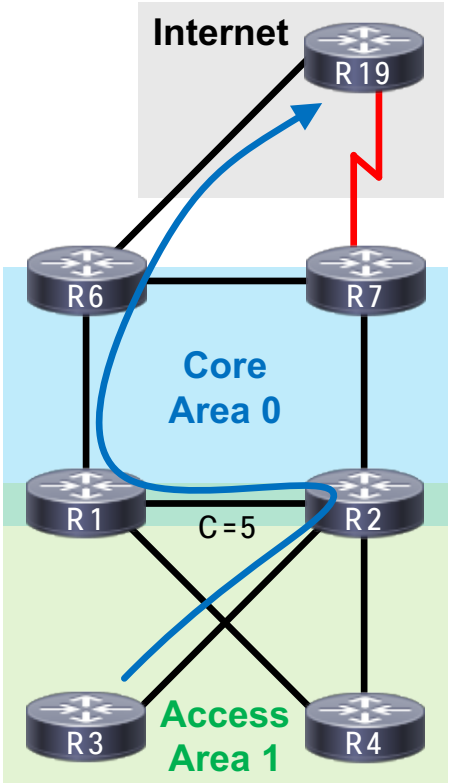
- Single link in multiple areas
  - RFC-5185
  - Identify area 0 as base
  - Add more areas to it
- Similar to IS-IS L1/L2

```
R1/R2 config:
interface Ethernet0/0
 ip ospf multi-area 1
 ip ospf 1 area 0
 ip ospf cost 5
```



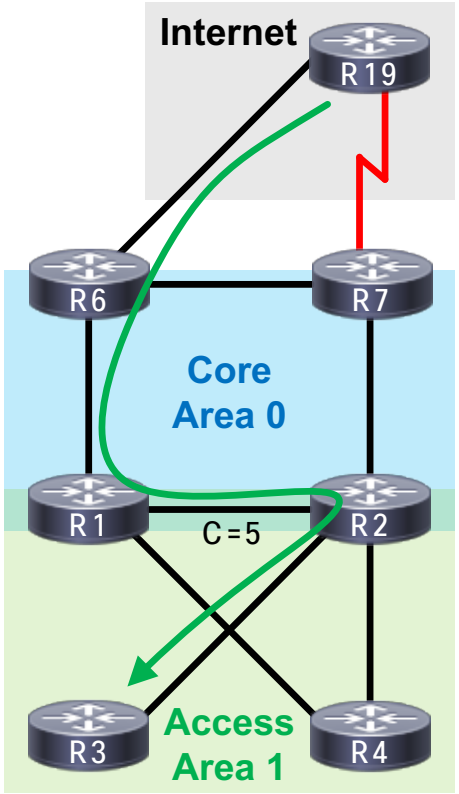
Multi-area Adjacency Upstream

```
R3#traceroute 192.0.2.19
Tracing the route to 192.0.2.19
  1 10.2.3.2 0 msec 0 msec 0 msec
  2 10.1.2.1 0 msec 1 msec 4 msec
  3 10.1.6.6 1 msec 0 msec 5 msec
  4 10.6.19.19 1 msec 1 msec 1 msec

R3#show ip route | include 0.0.0.0/0
O*E2  0.0.0.0/0 [110/10] via 10.2.3.2, 00:00:23, Ethernet1/1

R2#show ip route | include 0.0.0.0/0
O*E2  0.0.0.0/0 [110/10] via 10.1.2.1, 00:01:35, Ethernet0/0

R1#show ip route | include 0.0.0.0/0
O*E2  0.0.0.0/0 [110/10] via 10.1.6.6, 00:34:21, Ethernet0/1

R6#show ip route | include 192.0.2.0/24
B     192.0.2.0/24 [20/0] via 10.6.19.19, 00:34:30
```



Multi-area Adjacency Downstream

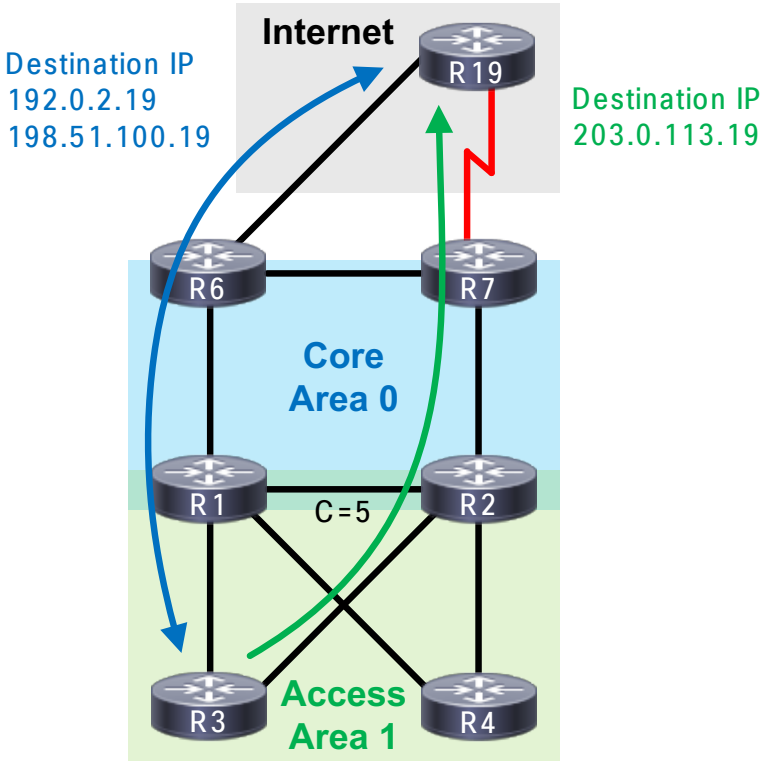
```
R6#traceroute 10.0.0.3
Tracing the route to 10.0.0.3
  1 10.1.6.1 1 msec 0 msec 0 msec
  2 10.1.2.2 1 msec 0 msec 1 msec
  3 10.2.3.3 0 msec 1 msec 1 msec

R6#show ip route | include 10.0.0.3/32
O IA 10.0.0.3/32 [110/26] via 10.1.6.1, 00:01:39, Ethernet0/1

R1#show ip route | include 10.0.0.3/32
O    10.0.0.3/32 [110/16] via 10.1.2.2, 00:01:44, Ethernet0/0

R2#show ip route | include 10.0.0.3/32
O    10.0.0.3/32 [110/11] via 10.2.3.3, 00:36:20, Ethernet1/1
```



OSPF Near the Internet Edge

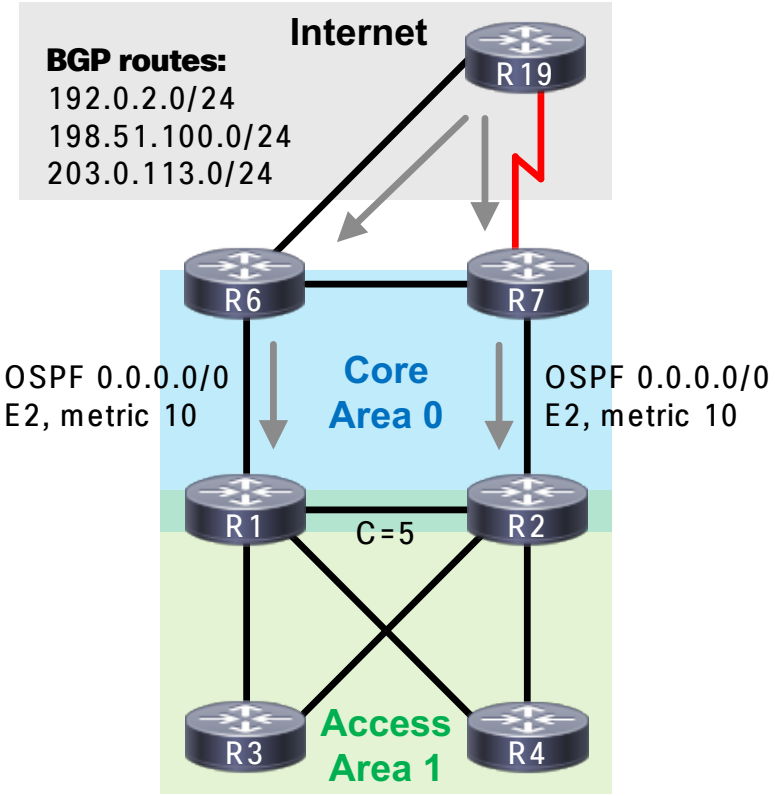
Challenge: Internet Edge

- Primary Internet link is congested
  - Slow access to cloud-hosted apps
  - Business networks:
    - CAD systems: 192.0.2.0/24
    - Team chat/collab: 198.51.100.0/24
    - File transfer/email: 203.0.113.0/24
- Solution must:
  - Use R6 for time-sensitive apps
  - Use R7 for non-transactional apps
  - Provide automatic failure



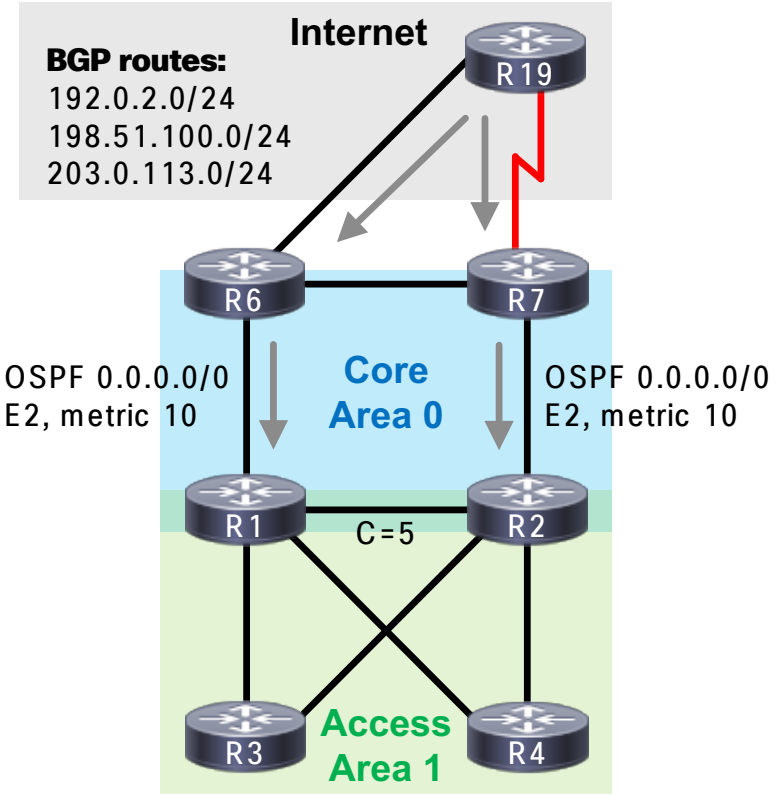
A First Step

- ISP not sending a default
- Conditionally originate one
  - If any “route of interest” exists
  - Start off with same type and cost
- Benefits
  - Better than “always” originating
  - Avoids risky redistribution



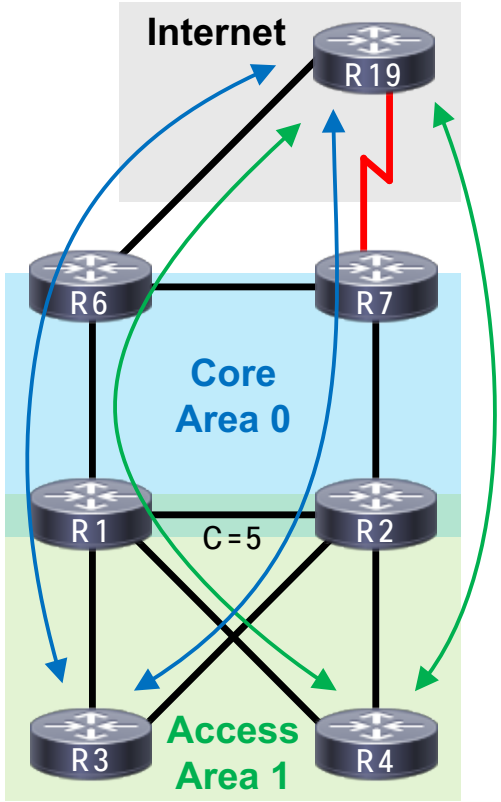
Configuration Snippet

```
Config on R6/R7:

ip prefix-list PL_BGP permit 192.0.2.0/24
ip prefix-list PL_BGP permit 198.51.100.0/24
ip prefix-list PL_BGP permit 203.0.113.0/24

route-map RM_OSPF_DEFAULT permit 10
 match ip address prefix-list PL_BGP

router ospf 1
 default-information originate
  route-map RM_OSPF_DEFAULT


R6#show ip ospf database | begin Type-5
  Type-5 AS External Link States

Link ID     ADV Router      Age
0.0.0.0     10.0.0.6        479
0.0.0.0     10.0.0.7        487
```



Load Sharing is Good, Right?

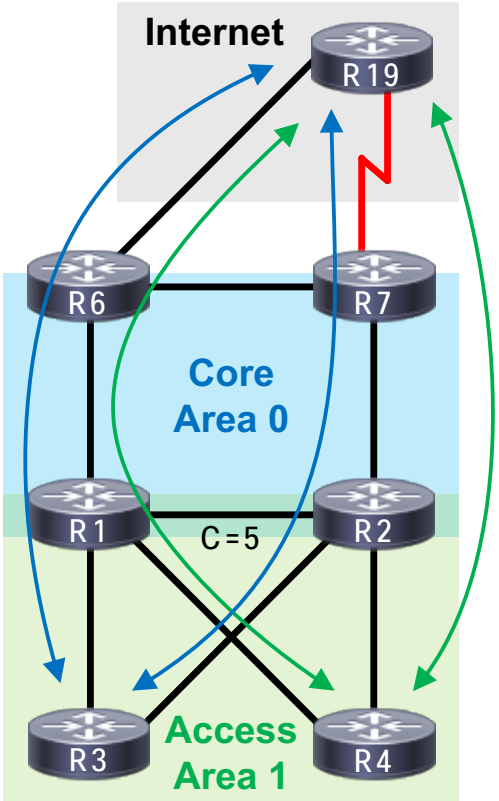
- Both routes are OSPF E2
- Both routes have seed cost of 10
- Topology is generally symmetric
- Both links will be used



ECMP Everywhere!

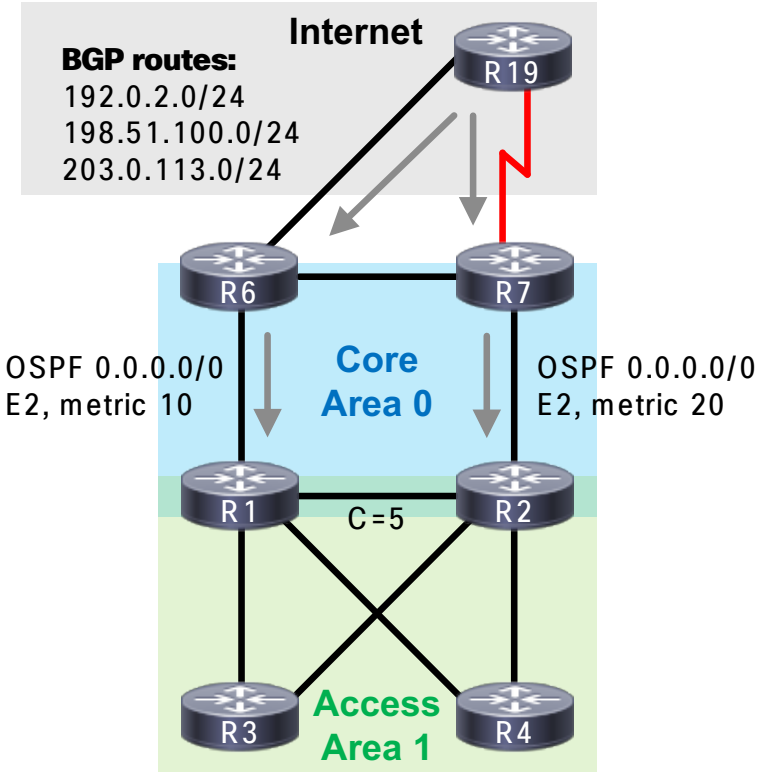
```
R3#traceroute 192.0.2.19 probe 2
Tracing the route to 192.0.2.19
  1 10.1.3.1 1 msec
    10.2.3.2 0 msec
  2 10.1.6.6 1 msec
    10.2.7.7 0 msec
  3 10.6.19.19 1 msec
    10.7.19.19 6 msec
```

```
R4#traceroute 192.0.2.19 probe 2
Tracing the route to 192.0.2.19
  1 10.1.4.1 1 msec
    10.2.4.2 0 msec
  2 10.1.6.6 1 msec
    10.2.7.7 0 msec
  3 10.6.19.19 0 msec
    10.7.19.19 11 msec
```



OK, Bad Idea, Back to Normal

- R6 should be “generally preferred”
  - Easiest: OSPF E2, low seed cost on R6
- Alternatives
  - OSPF E1 on R6, E2 on R7
    - Good for only two exits, poor scale
  - OSPF E1, low seed cost on R6
    - Poor choice for strict active/standby

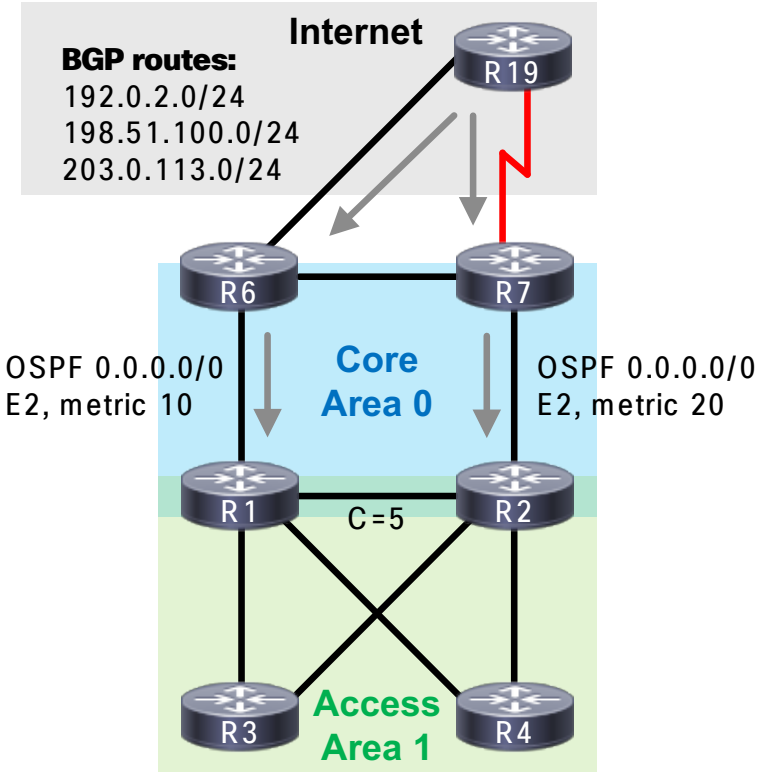


Configuration Snippet

Config update R7:

```
router ospf 1
 default-information originate metric 20
  route-map RM_OSPF_DEFAULT
```

```
R3#show ip ospf database external 0.0.0.0 |
  include Adv|Metric:

  Advertising Router: 10.0.0.6
        Metric: 10
  Advertising Router: 10.0.0.7
        Metric: 20
```

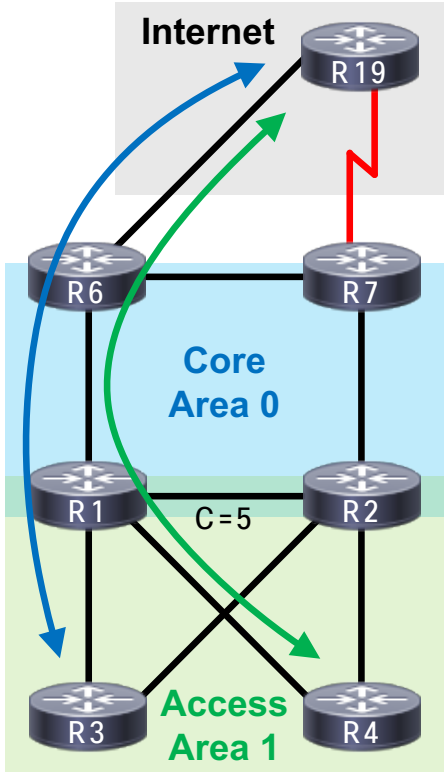


Getting Closer

- Both routes are OSPF E2

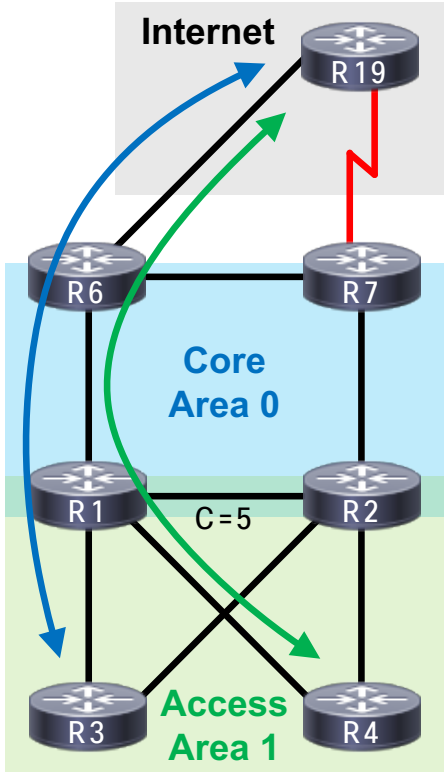
- Routes have different seed costs
  - R6: cost 10
  - R7: cost 20

- R6 is always preferred
  - Watch out for “interceptions” by R7!



Tracing the Path

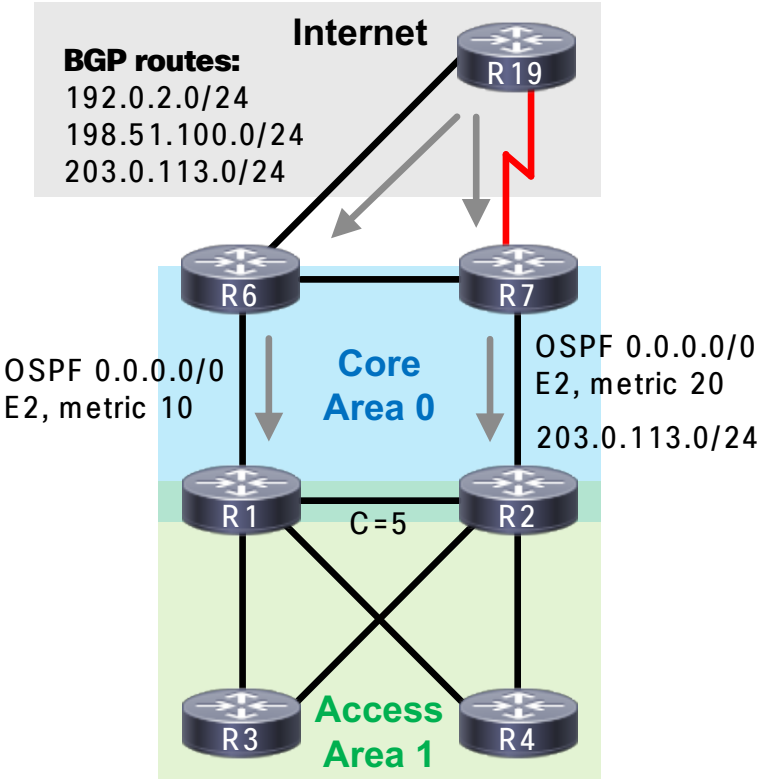
```
R3#show ip cef 192.0.2.19
0.0.0.0/0
  nexthop 10.1.3.1 Ethernet1/0

R3#traceroute 192.0.2.19
Tracing the route to 192.0.2.19
  1 10.1.3.1 1 msec 0 msec 1 msec
  2 10.1.6.6 0 msec 1 msec 1 msec
  3 10.6.19.19 1 msec 0 msec 1 msec


R4#show ip cef 192.0.2.19
0.0.0.0/0
  nexthop 10.1.4.1 Ethernet1/1

R4#traceroute 192.0.2.19
Tracing the route to 192.0.2.19
  1 10.1.4.1 0 msec 0 msec 0 msec
  2 10.1.6.6 1 msec 0 msec 0 msec
  3 10.6.19.19 1 msec 1 msec 1 msec
```



Simple OSPF Traffic Engineering

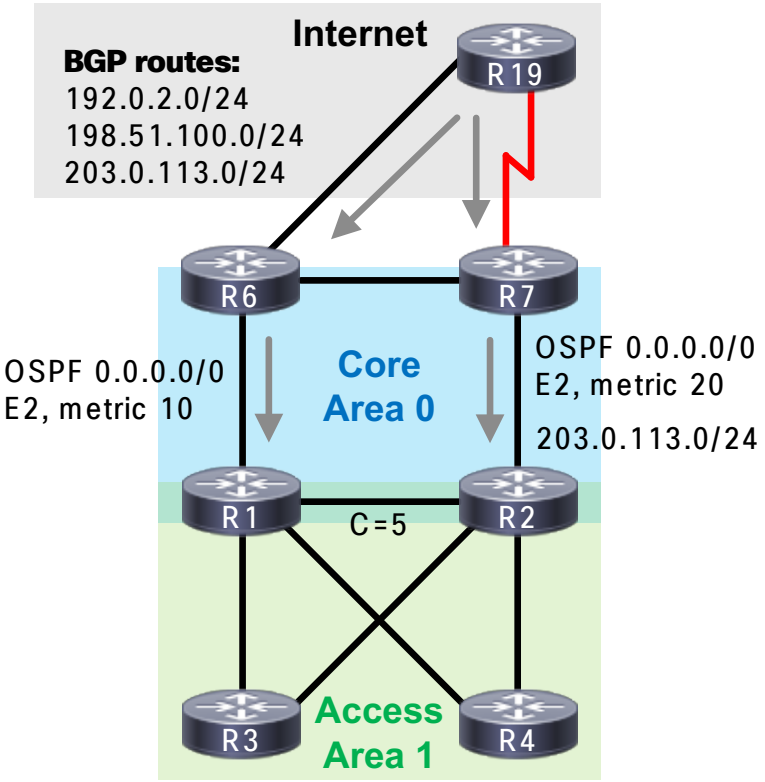
- Leak longer matches via R7
  - Match 203.0.113.0/24
  - Redistribute from BGP into OSPF
  - Type and cost are irrelevant

- What we avoided
  - BGP manipulations
  - Hardcore policy application



Configuration Snippet

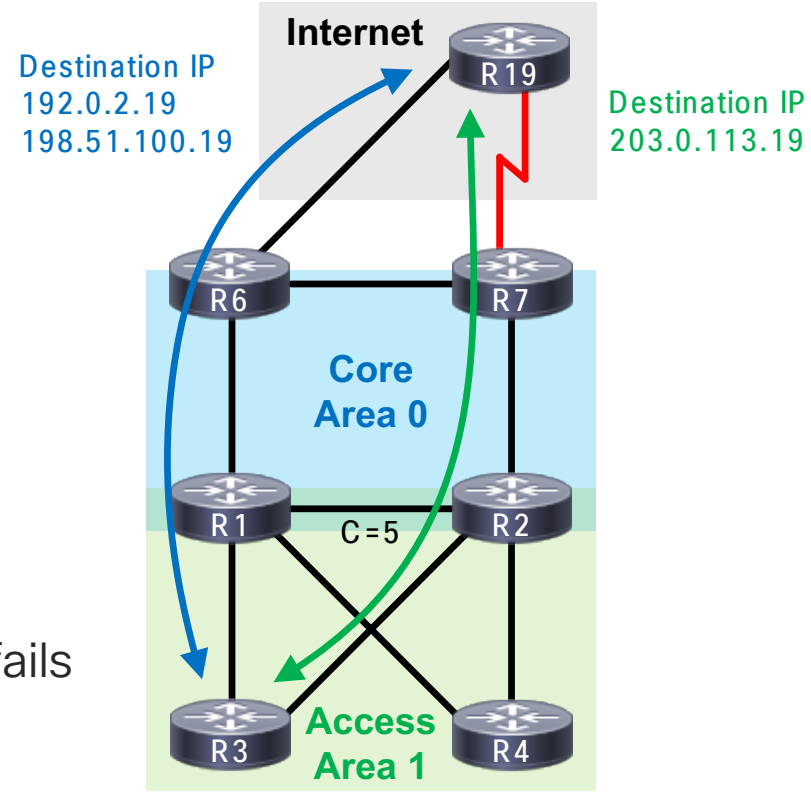
```
Config on R7:

ip prefix-list PL_LEAK_203 permit 203.0.113.0/24

route-map RM_BGP_TO_OSPF permit 10
 match ip address prefix-list PL_LEAK_203

router ospf 1
 redistribute bgp 65067 subnets
  route-map RM_BGP_TO_OSPF
```



It Works!

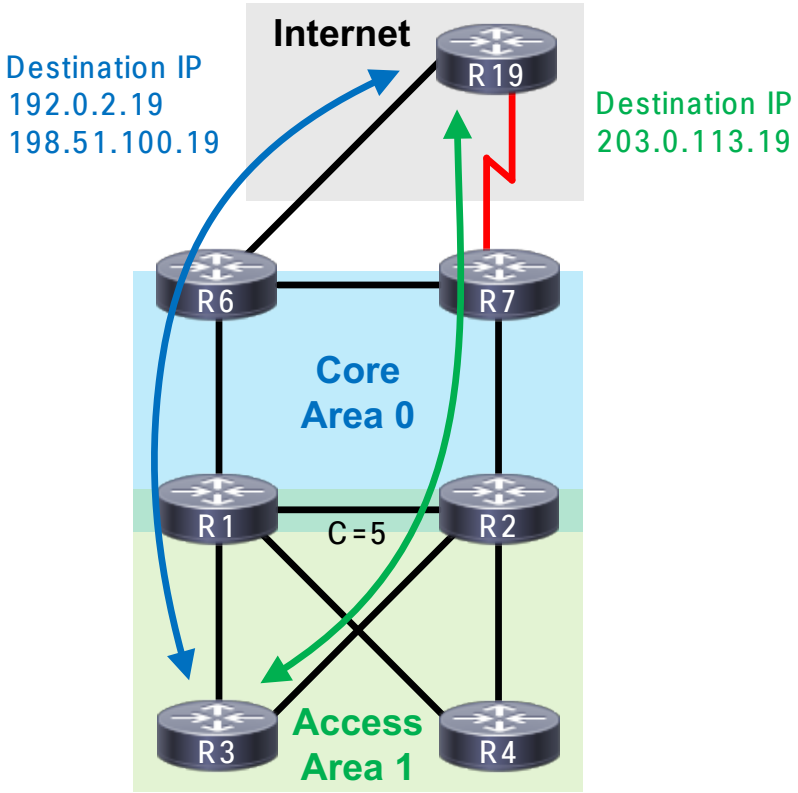
- All routes are OSPF E2
  - 0.0.0.0/0 from R6, cost 10
  - 0.0.0.0/0 from R7, cost 20
  - 203.0.113.0/24 from R7

- Optimal and resilient
  - R7 used for 203.0.113.0/24 only
  - R6 used for all other destinations
  - R6 default covers 203.0.113.0/24 if R7 fails
  - R7 default covers everything if R6 fails



Tracing the Path

```
R3#show ip ospf database | begin Type-5
  Type-5 AS External Link States

Link ID         ADV Router      Age
0.0.0.0         10.0.0.6        318
0.0.0.0         10.0.0.7        326
203.0.113.0     10.0.0.7        153

R3#show ip cef 192.0.2.19
0.0.0.0/0
  nexthop 10.1.3.1 Ethernet1/0

R3#show ip cef 203.0.113.19
203.0.113.0/24
  nexthop 10.2.3.2 Ethernet1/1

R3#traceroute 203.0.113.19
Tracing the route to 203.0.113.19
  1 10.2.3.2 1 msec 0 msec 0 msec
  2 10.2.7.7 1 msec 0 msec 1 msec
  3 10.7.19.19 9 msec 11 msec 10 msec
```

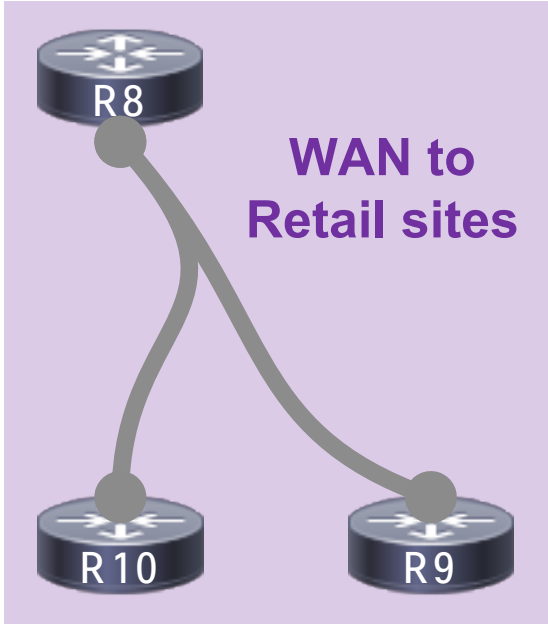


OSPF over Hub/spoke

Challenge: WAN

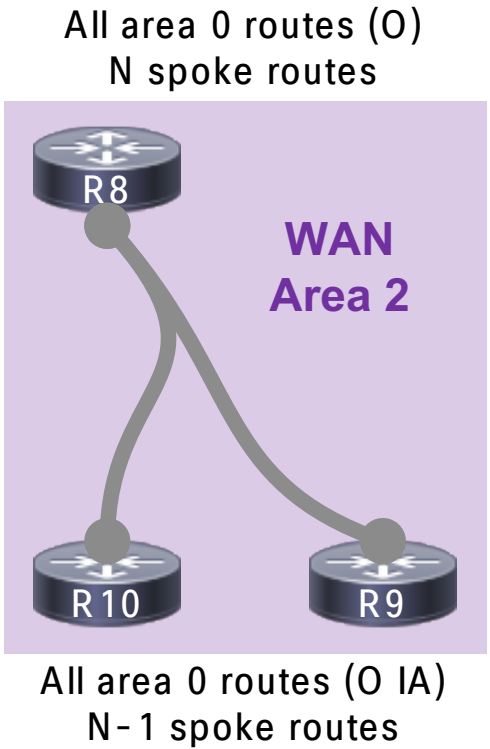
- Leased floor space in new factory
  - Started building products; need sales outlet
- Standing up new retail stores
- Contracted MPLS carrier for transport

- Solution must:
  - Use OSPF for consistency with campus
  - Scale to thousands of stores
  - Provide optimal spoke-to-spoke routing



Why is OSPF a Poor Choice?

- All routers in an area have same “view”
  - That’s how link-state protocols work
- All spokes see all other spokes
  - Unnecessary; all have same next-hop
- Linear scaling; not great

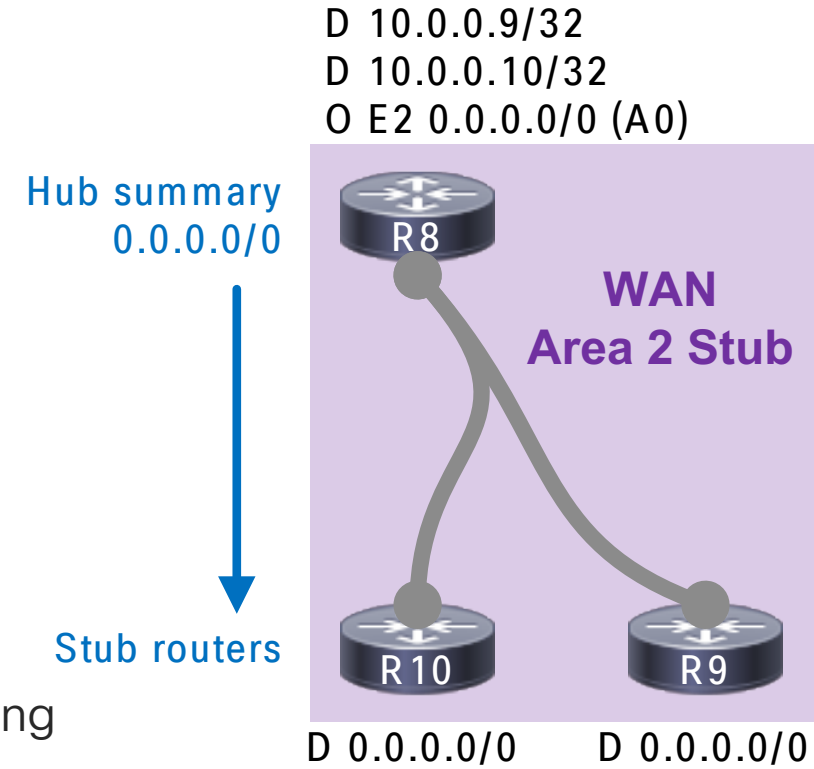


Thought Exercise: EIGRP over WAN

- Two big advantages

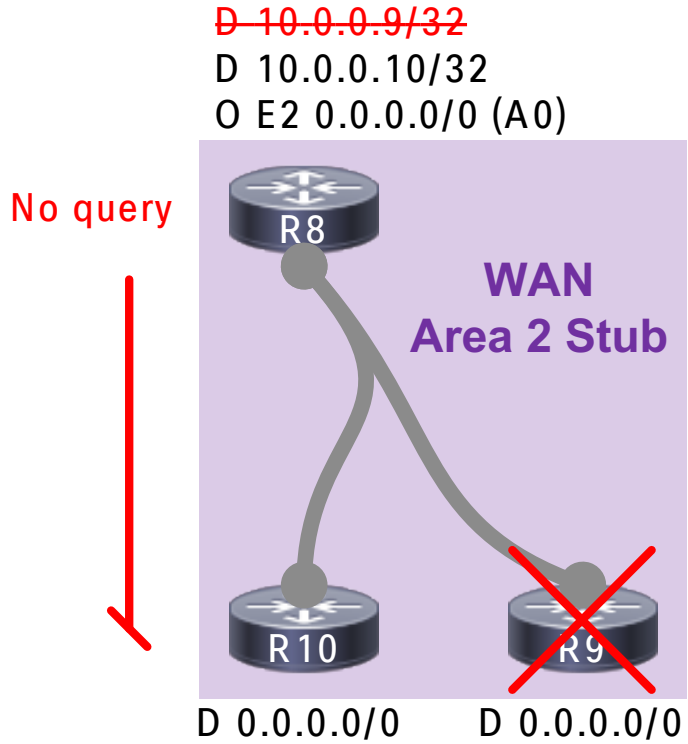
- Configure spokes as “stub”
  - Cannot be queried
  - Only advertise connected networks
  - Non-transit

- Hub sends downstream summary
  - Covers all other spokes; no linear scaling



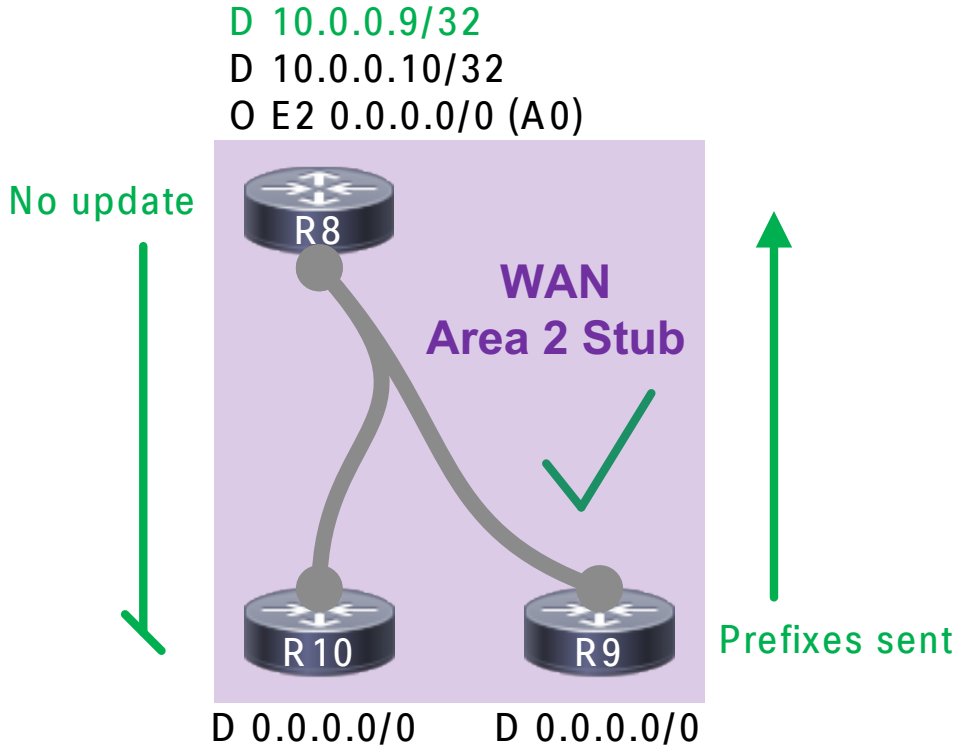
EIGRP Spoke Down Event

- Suppose R9 fails
- Once R8 finds out
  - Withdraw 10.0.0.9/32 from topology
- OSPF (presumably) removes LSA5
- No impact on other spokes



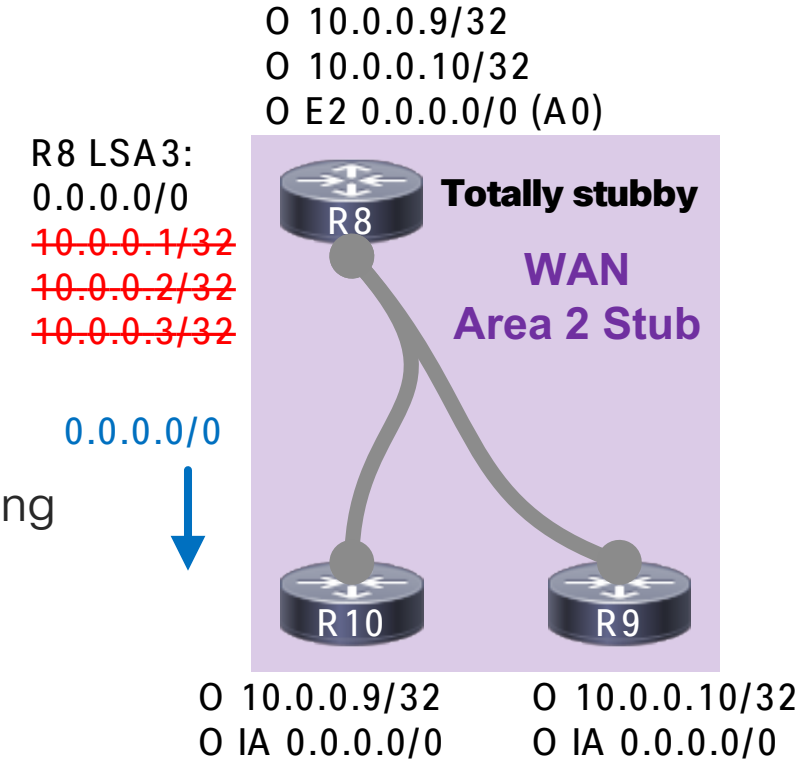
EIGRP Spoke Up Event

- Suppose R9 comes back up
- R9 sends new prefixes to R8
- OSPF originates new LSA5
- No impact on other spokes



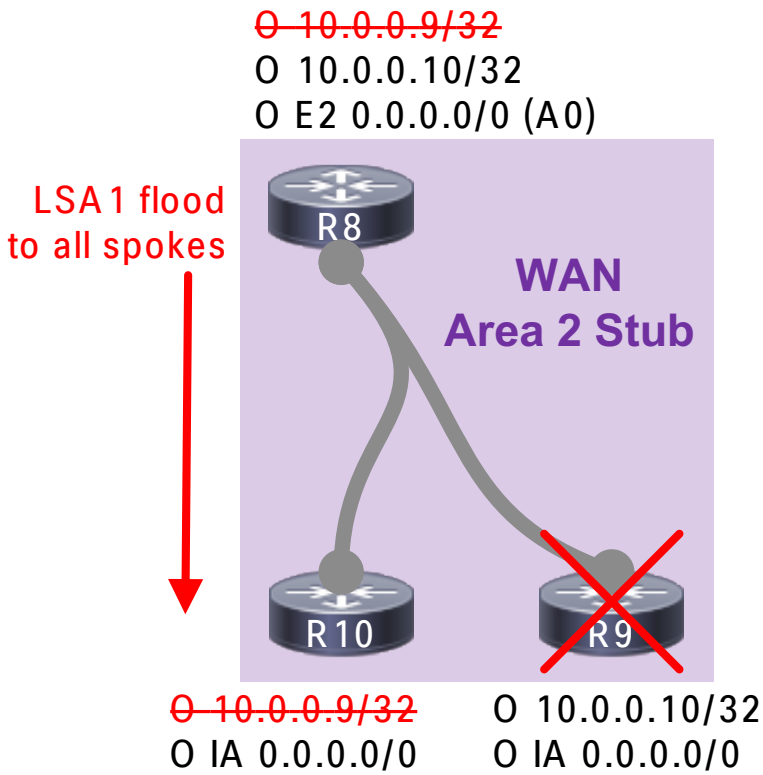
Does “Totally Stubby” Help?

- No inter-area or externals allowed
- ABR originates inter-area default
- Saves some compute on R8
  - Will not copy LSA3/4/5 into local LSDB
  - Does not address intra-area state sharing
- Short answer: yes, but insufficient



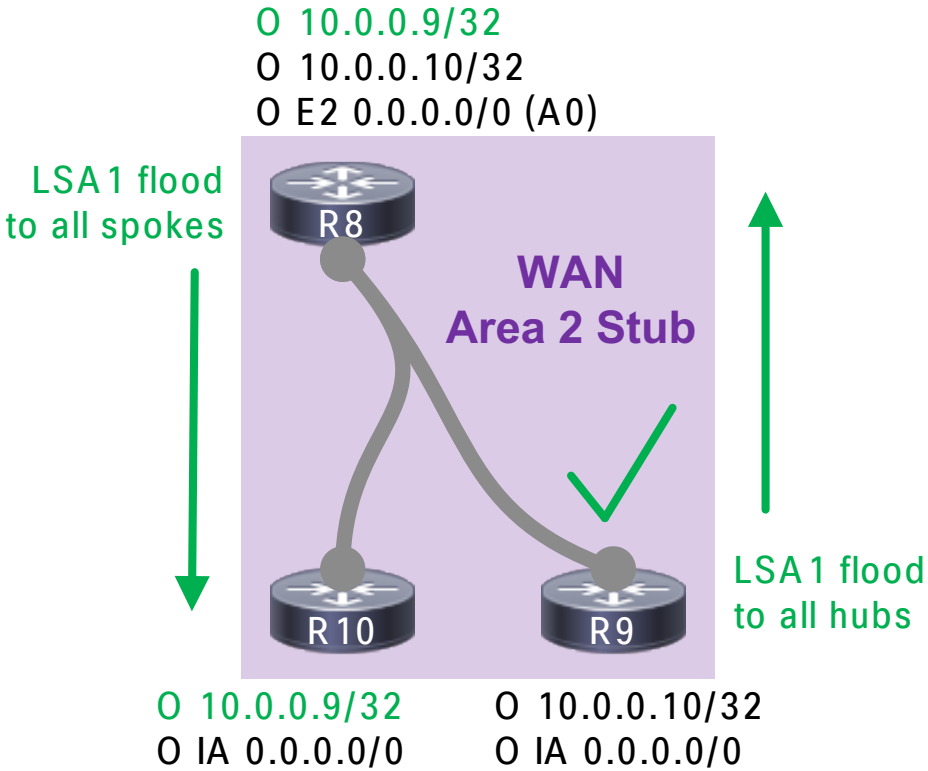
OSPF Spoke Down Event

- Suppose R9 fails
- Once R8 finds out
  - Flood new LSA1 to R10
  - Run SPF
- Then R10:
  - Receive new LSA1 from R8
  - Run SPF



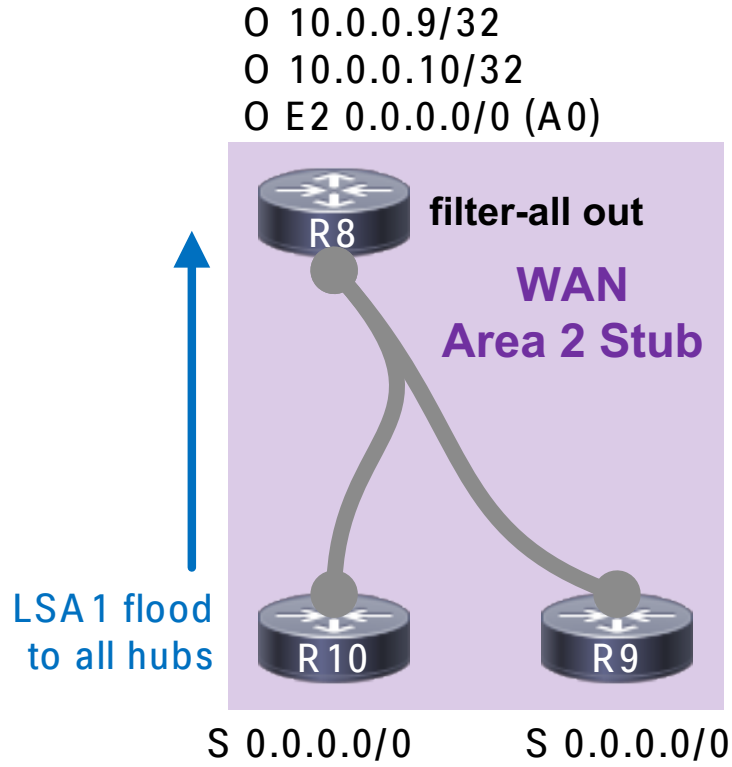
OSPF Spoke Up Event

- Suppose R9 comes back up
- Even worse
  - R9 floods LSA1 to R8, runs SPF
  - R8 floods R8 + R9 LSA1 to R10
  - R10 receives all LSA1, runs SPF
- Image these events at scale



A solution: OSPF LSA Filter

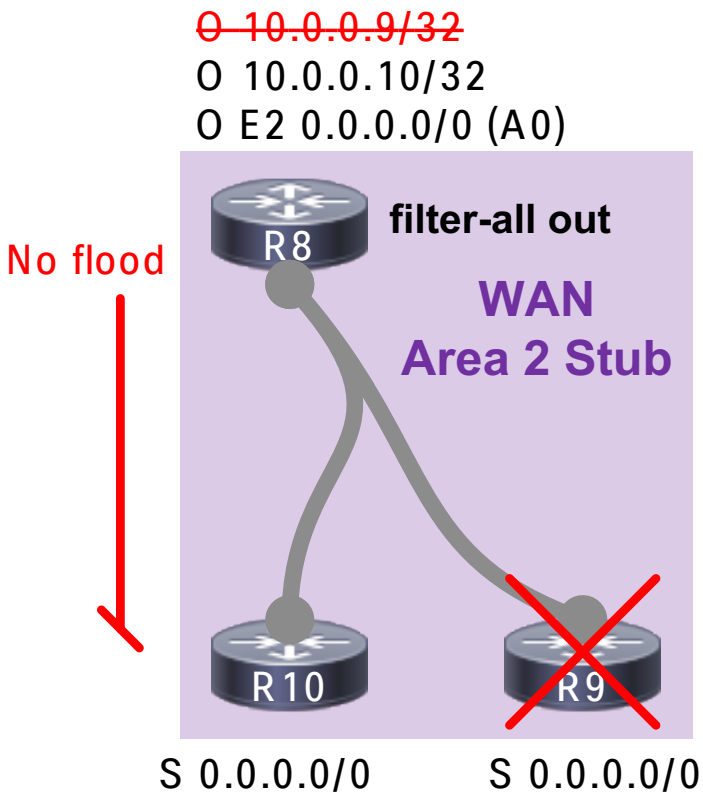
- No outbound LSAs, period
- Used to reduce flooding
  - Useful in full mesh topologies
- What about hub/spoke WANs?
  - Can use a static default route upstream
  - Imitates EIGRP “hub summary” design



OSPF LSA Filter Spoke Down Event

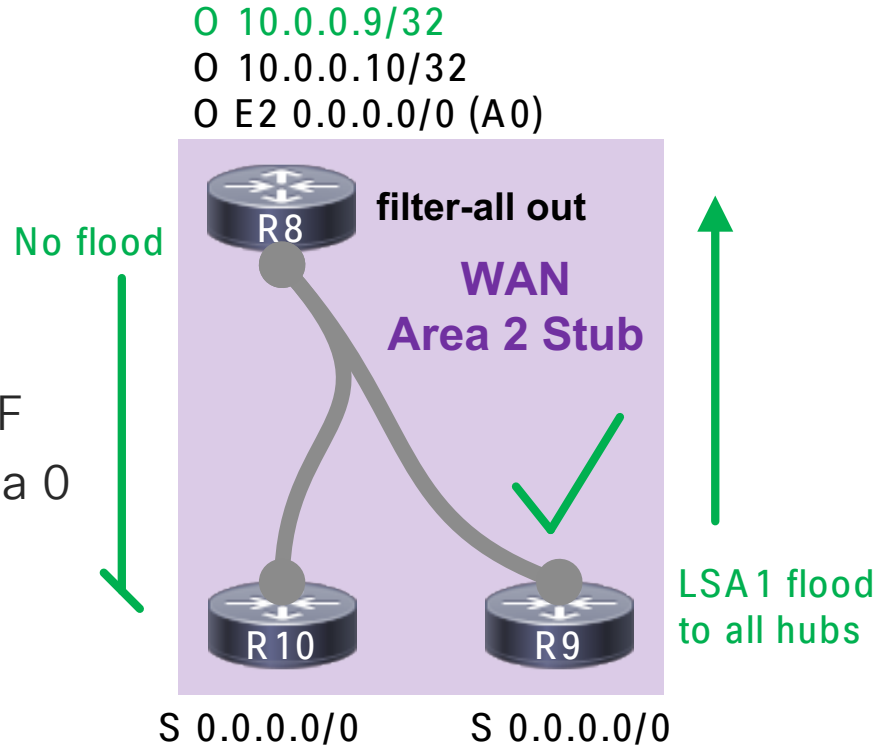
- Suppose R9 fails

- Once R8 finds out
  - Neighbor down; run SPF
  - Withdrawn OSPF LSA3 from area 0
  - That's all (for area 2 at least)



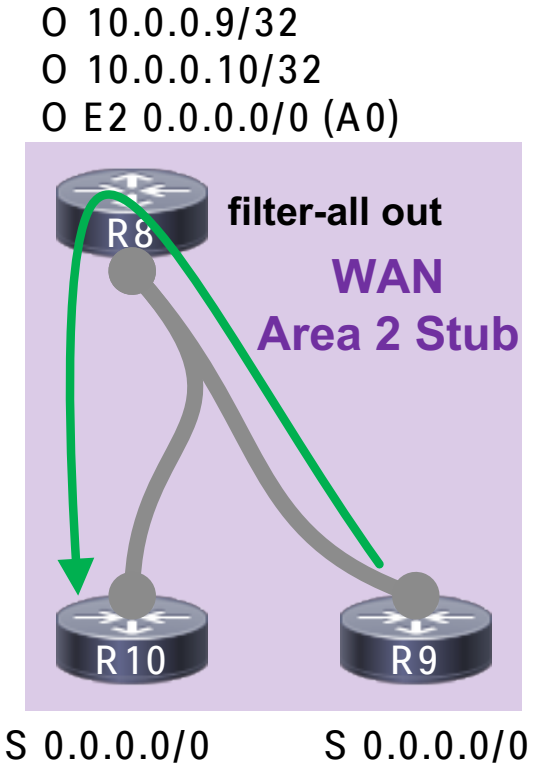
OSPF LSA Filter Spoke Up Event

- Suppose R9 comes back up
- Other spokes not informed
  - R9 floods LSA1 to R8, runs SPF
  - R8 receives LSA1 from R8, runs SPF
  - R8 re-originate OSPF LSA3 into area 0
  - That's all (for area 2 at least)
- Comparable scale to EIGRP



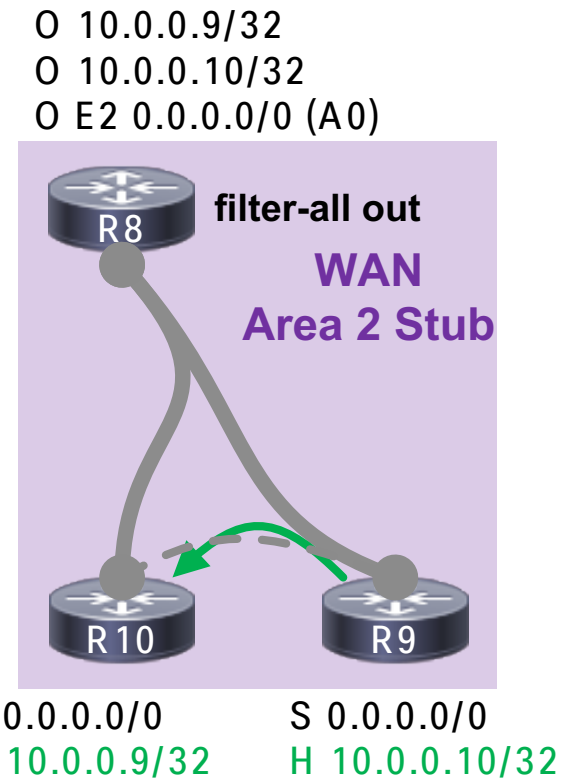
Initial Spoke-to-spoke Traffic

```
R9#show ip route 10.0.0.10
% Subnet not in table

R9#show ip route 0.0.0.0
Routing entry for 0.0.0.0/0, supernet
  Known via "static", distance 240, metric 0
  Routing Descriptor Blocks:
  * 10.0.100.8
      Route metric is 0

R9#traceroute 10.0.0.10
Tracing the route to 10.0.0.10
  1 10.0.100.8 5 msec 2 msec 0 msec
  2 10.0.100.10 7 msec 2 msec 1 msec
```



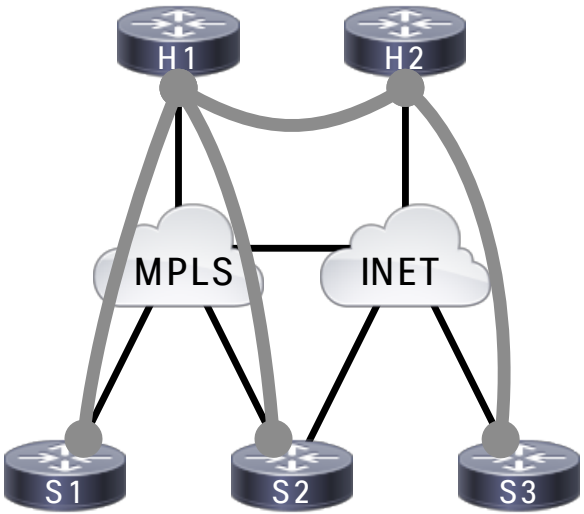
Subsequent Spoke-to-spoke traffic

```
R9#show ip route 10.0.0.10
Routing entry for 10.0.0.10/32
  Known via "nhrp", distance 250, metric 255
  Last update from 10.0.100.10 on Tunnel100, 00:00:04 ago
  Routing Descriptor Blocks:
  * 10.0.100.10, from 10.0.100.10, via Tunnel100
      Route metric is 255, traffic share count is 1

R9#traceroute 10.0.0.10
Tracing the route to 10.0.0.10
  1 10.0.100.10 7 msec 2 msec 2 msec
```



What about Multiple Hubs?

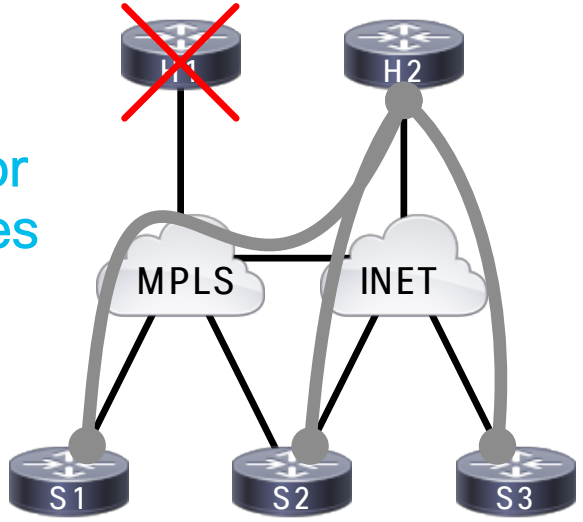
DMVPN source: 203.0.113.100



DMVPN destination: 203.0.113.100

Use anycast IP for hub tunnel sources

DMVPN source: 203.0.113.100



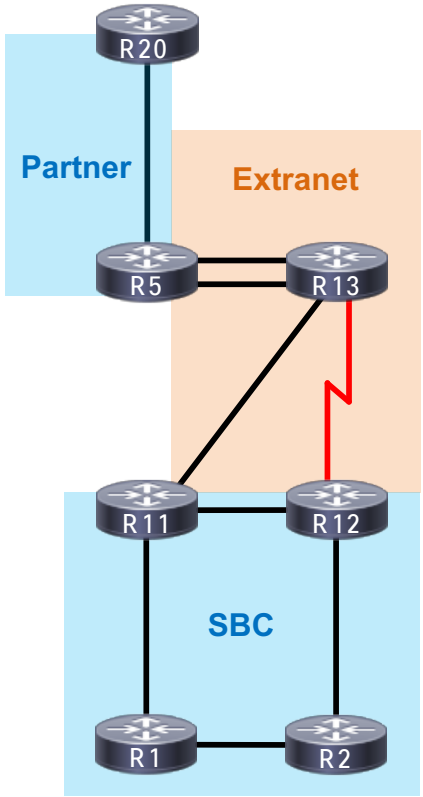
DMVPN destination: 203.0.113.100

OSPF Extranets

Challenge: Extranet

- Finally adding factory to network
- Shared floor space/LAN with a partner firm
- Both need access to common resources

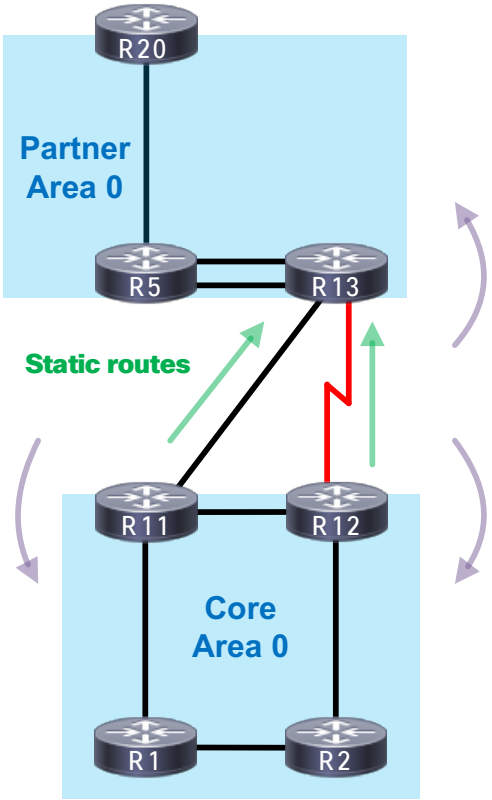
- Solution must:
  - Be accessible from campus, WAN, and partner
  - No Internet or cross-network transit
  - Provide optimal routing to/from extranet
  - Minimize OSPF knobs/new protocols



Option 1: Static Routes

- Use static routes to extranet

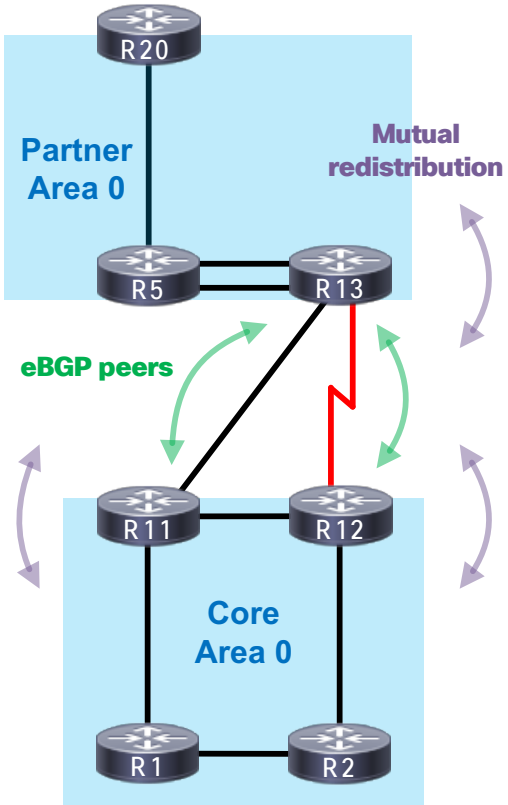
- Benefits
  - Simple and safe
  - Suboptimal routing/loops unlikely
- Drawbacks
  - Poor scale if extranet grows
  - Need outbound statics on R13 too
  - Need explicit metrics for R11/R12 redistribution



Option 2: Use BGP

- Use eBGP to peer with the partner

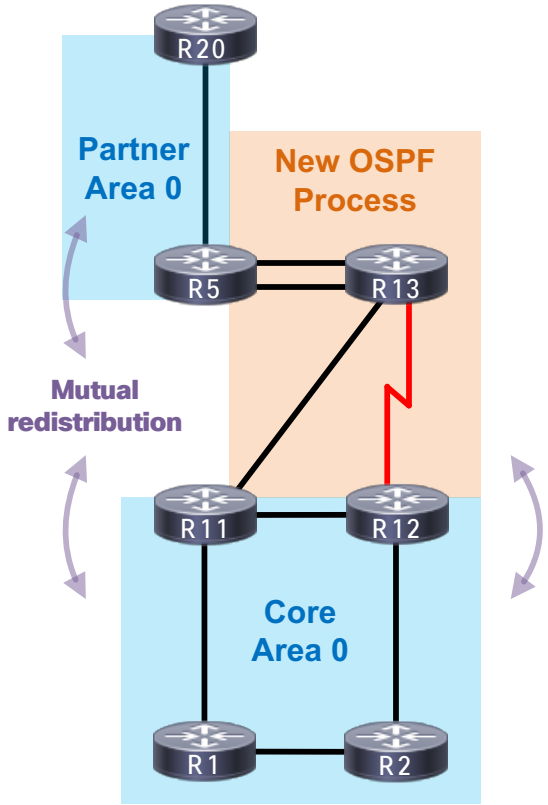
- Benefits
  - Maximum control
  - Scalable
- Drawbacks
  - Heavy up-front planning/config
  - Probably “gold-plated”
  - Need explicit metrics for R11/R12 redistribution



Option 3: Use Another OSPF Process

- Place the extranet in a new OSPF process

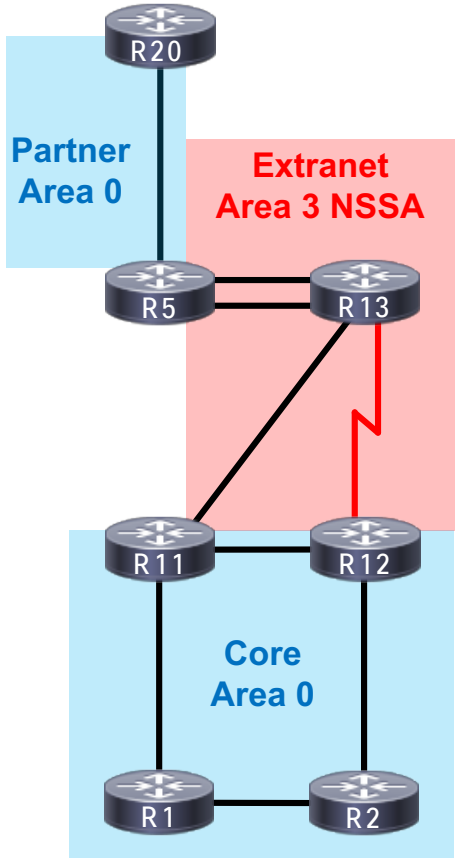
- Benefits
  - Good control
  - No new protocols
- Drawbacks
  - Higher risk of suboptimal routing/loops
  - Sloppy/uncommon
  - Misconfiguration risk (wrong PID, etc.)



Option 4: Disjoint Area 0 with NSSA

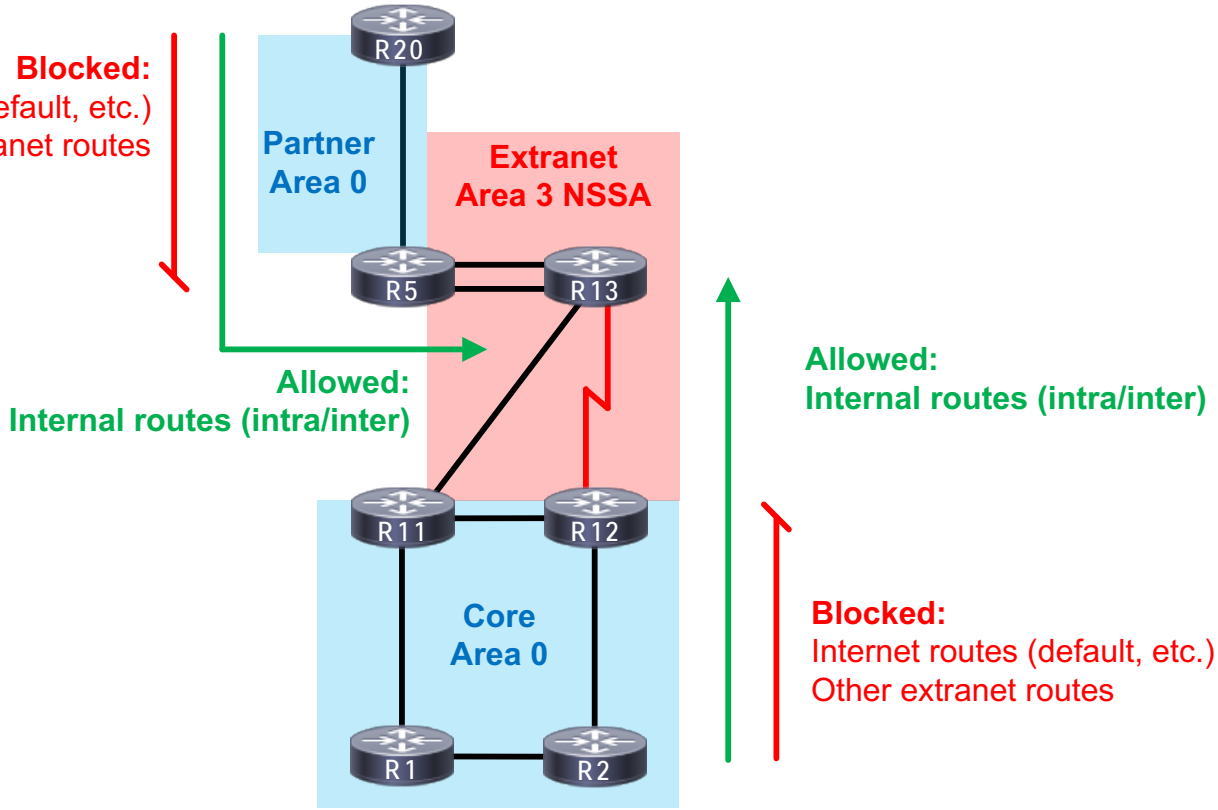
- Use an NSSA in between the two networks

- Benefits
  - Simple, safe, and scalable
  - Good control
  - Suboptimal routing/loops unlikely
  - No explicit metrics needed on R11/12
- Drawbacks
  - You must discard your OSPF preconceptions ☺



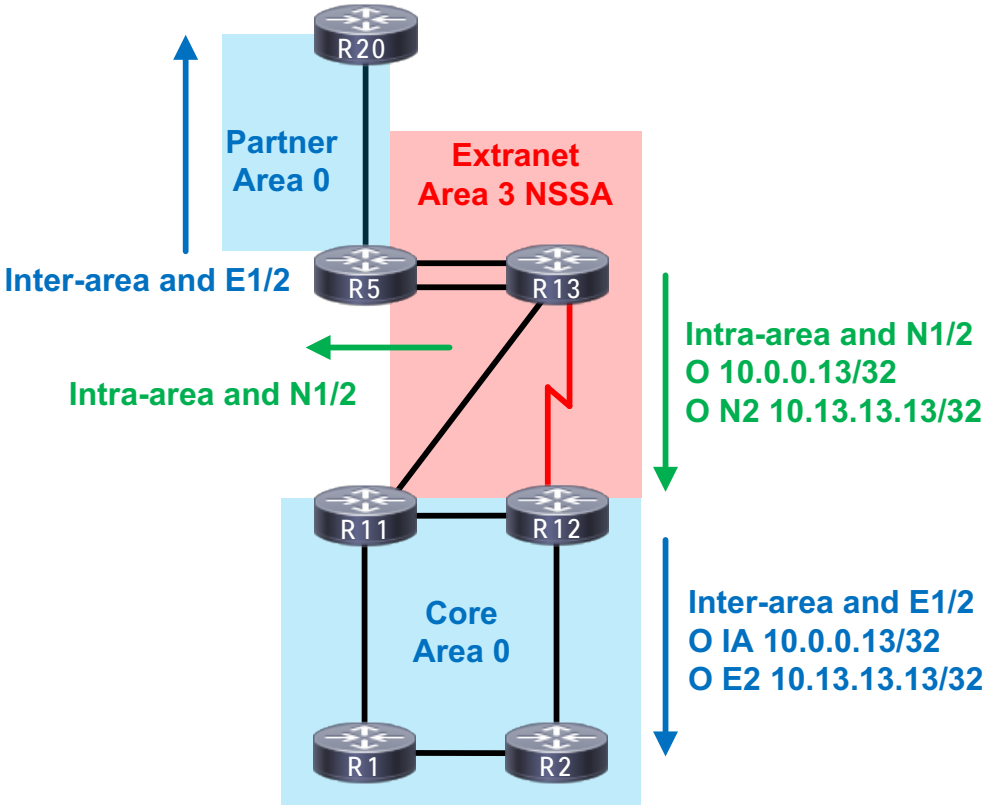
NSSA Extranet Routes Inbound

- No externals into any kind of stub area
  - No Internet transit
  - No inter-area 0 transit
- Full extranet access to internal routes in both networks



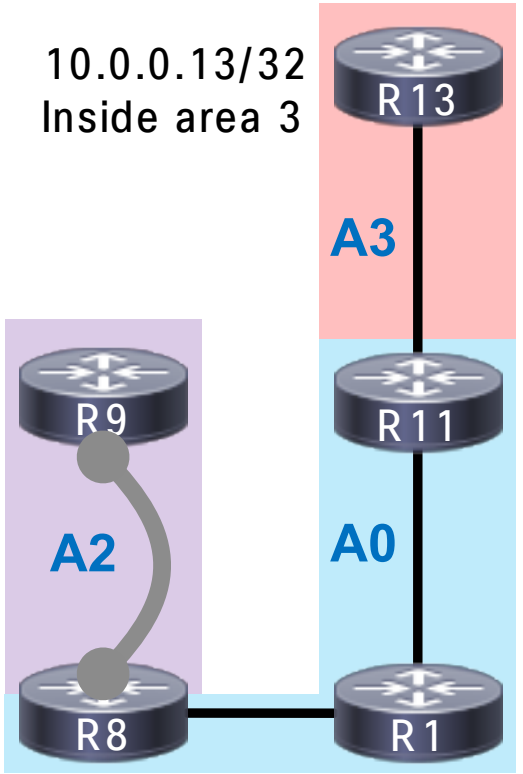
NSSA Extranet Routes Outbound

- Within the NSSA
  - Intra-area (direct area 3)
  - Redistributed (NSSA extern)
- Beyond the NSSA
  - Inter-area (standard ABR)
  - NSSA translation (external)



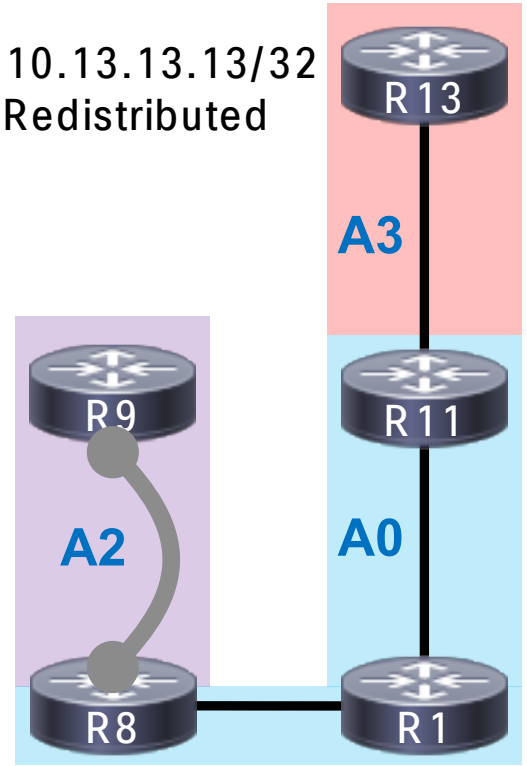
Reaching Into the Extranet – Internal Route

```
R9#traceroute 10.0.0.13
Tracing the route to 10.0.0.13
  1 10.0.100.8 1 msec 1 msec 1 msec
  2 10.1.8.1 0 msec 0 msec 1 msec
  3 10.1.11.11 1 msec 1 msec 1 msec
  4 10.11.13.13 1 msec 1 msec 2 msec

R8#show ip route | include 10.0.0.13
O IA 10.0.0.13/32 [110/31] via 10.1.8.1, Ethernet0/2

R1#show ip route | include 10.0.0.13
O IA 10.0.0.13/32 [110/21] via 10.1.11.11, Ethernet0/3

R11#show ip route | include 10.0.0.13
O    10.0.0.13/32 [110/11] via 10.11.13.13,Ethernet0/1
```



Reaching Into the Extranet – External Route

```
R9#traceroute 10.13.13.13
Tracing the route to 10.13.13.13
  1 10.0.100.8 2 msec 1 msec 1 msec
  2 10.1.8.1 1 msec 1 msec 1 msec
  3 10.1.11.11 1 msec 2 msec 2 msec
  4 10.11.13.13 2 msec 2 msec 2 msec

R8#show ip route | include 10.13.13.13
O E2 10.13.13.13/32 [110/20] via 10.1.8.1, Ethernet0/2

R1#show ip route | include 10.13.13.13
O E2 10.13.13.13/32 [110/20] via 10.1.11.11, Ethernet0/3

R11#show ip route | include 10.13.13.13
O N2 10.13.13.13/32 [110/20] via 10.11.13.13, Ethernet0/1
```

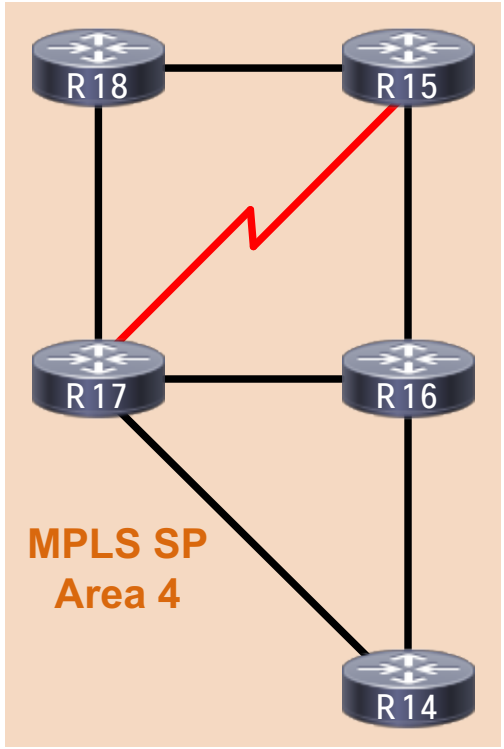


Tuning OSPF for Service Providers

Challenge: Service Provider

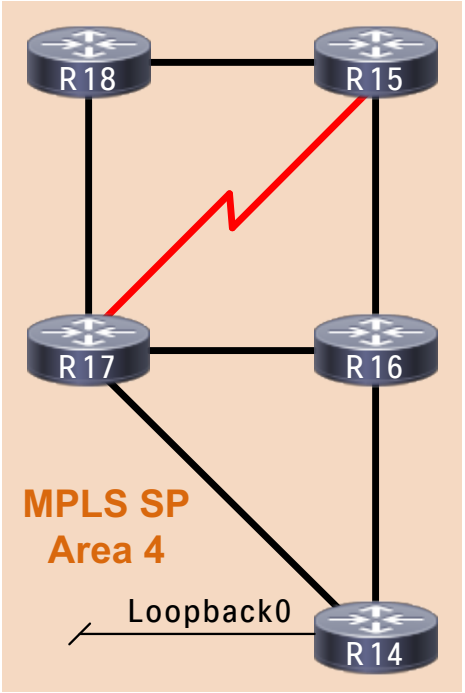
- New client (at the SP)!
- Asked to “optimize the network”
  - What does that even mean?

- No requirements, but consider:
  - Reducing OSPF LSDB/RIB size
  - Reducing OSPF convergence time



Unnecessary RIB Bloating

- LSA1 includes all IP networks
  - Loopbacks
  - P2P networks
  - Multi-access networks
- For MPLS LSPs, only loopbacks matter
- Can we filter out the rest?

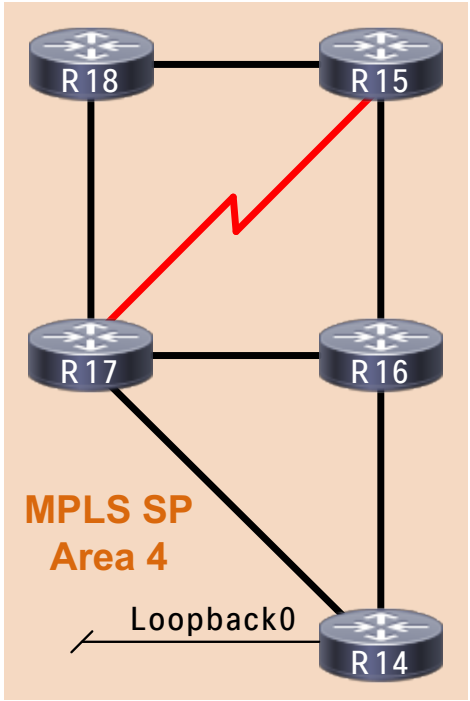


R15 routes:
O 10.0.0.14/32
O 10.0.0.16/32
O 10.0.0.17/32
O 10.0.0.18/32
O 10.14.16.0/24
O 10.14.17.0/24
O 10.16.17.0/24
O 10.17.18.0/24

R14 LSA1 num links: 5
Stub network: 10.0.0.14/32
Point-to-point: RID 10.0.0.16
Point-to-point: RID 10.0.0.17
Stub network: 10.14.16.0/24
Stub network: 10.14.17.0/24

OSPF Prefix-suppression

- Each router can suppress non-passive/transit links
- Removes “stub network” from LSA1
- Retains only true stubs
  - Loopbacks
  - Client LANs



R15 routes:
O 10.0.0.14/32
O 10.0.0.16/32
O 10.0.0.17/32
O 10.0.0.18/32

R14 LSA1 num links: 3
Stub network: 10.0.0.14/32
Point-to-point: RID 10.0.0.16
Point-to-point: RID 10.0.0.17
~~Stub network: 10.14.16.0/24~~
~~Stub network: 10.14.17.0/24~~

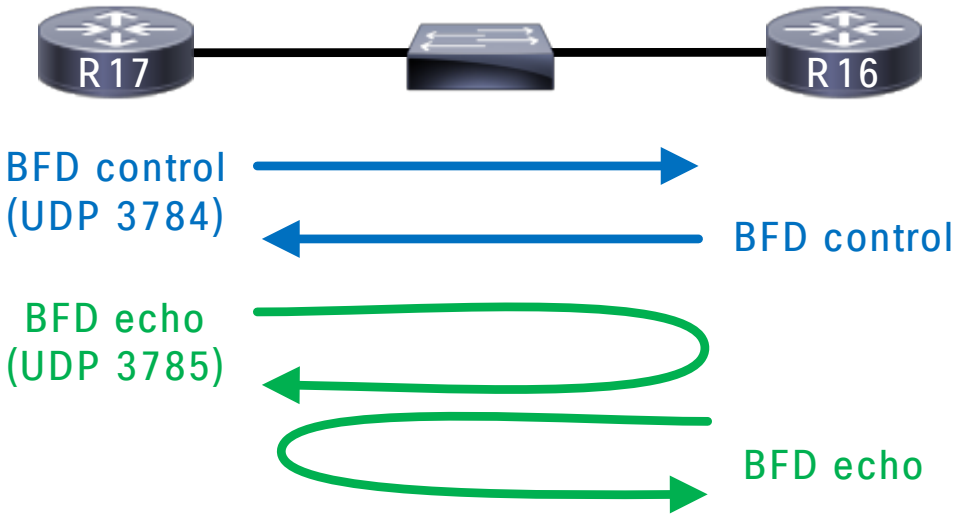
Link-change Detection: Carrier-delay

- Delays link up/down notifications for a short period
- Lower values yield faster convergence
- Higher values can filter micro-flaps (reduces churn)

interface down after carrier-delay

Link-change Detection: BFD

- Bidirectional Forwarding Detection (RFC-5880)
- Bidirectional control probes
- Looping echo probes
  - Same IP address for src/dest
  - Regular dest MAC address

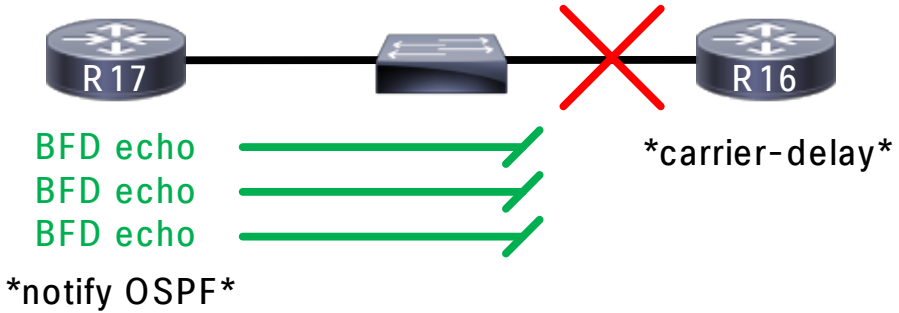


Multi-access Link-failure Example

- Carrier-delay and BFD work together

- Local failure on R16
  - R16 uses carrier-delay
  - R17 uses BFD echo timeouts

- Will impact notification timing!



LSA and SPF Throttle Timer Tuning

- Complex, niche, and often unnecessary
  - … but is critical for SLA-bound SPs, financial services, etc.

- Many timers; we'll focus on the most significant ones

```
router ospf 1
 timers throttle lsa <lsa_init> <lsa_hold> <lsa_max>
 timers throttle spf <spf_init> <spf_hold> <spf_max>
```

A note on LSA and SPF Init Timers

- "How long to wait before starting our jobs?"

- Just like "carrier-delay" in concept, except relating to
  - When to generate/advertise new LSAs after a topology change
  - When to start the first SPF run after receiving updated LSAs

- Think "batch size"
  - Depending on topology higher values may improve convergence!

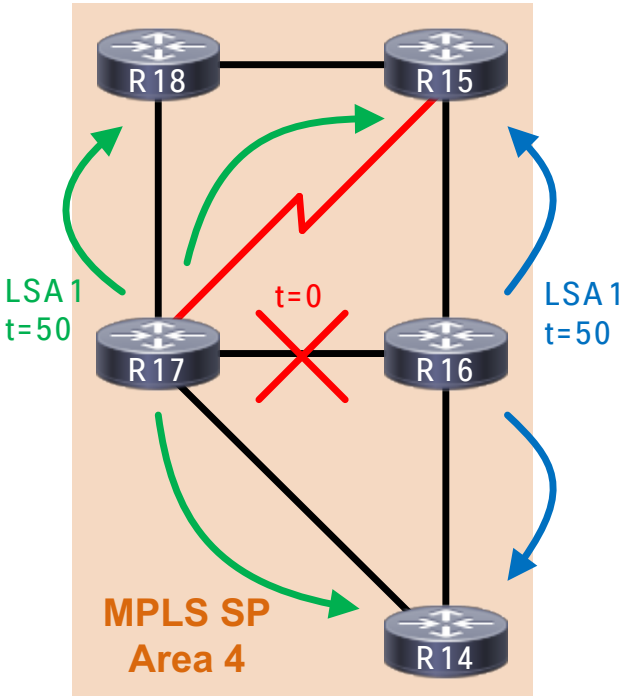
Computing the LSA Hold Timer

- "How long to wait before generating same LSA again?"
- Rule of thumb:
  - lsa_hold > lsa_init + lsa_propagate + spf_init
- Assume:
  - lsa_init = 50 ms
  - lsa_propagate = 130 ms
  - spf_init = 50 ms
  - then lsa_hold > 230 ms (maybe 250 ms)
- lsa_max is the upper bound on lsa_hold

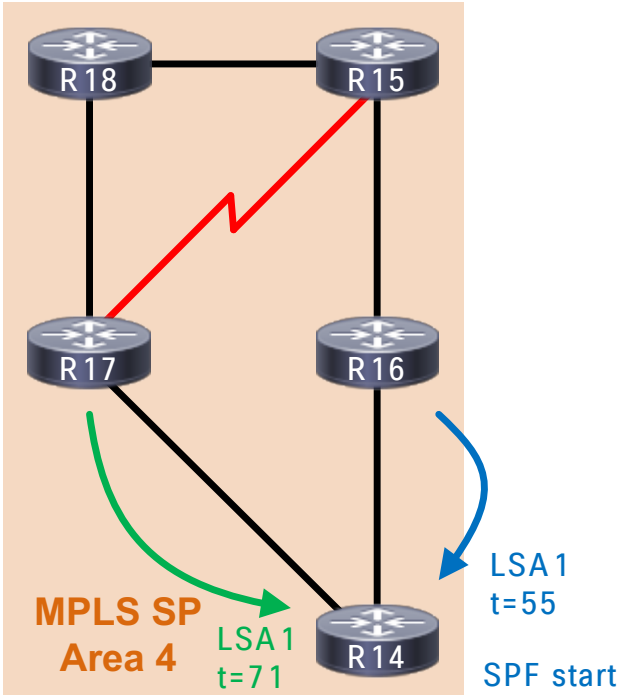
Computing the SPF Hold Timer

- “How long to wait to run SPF again after receiving updates?”
- Rule of thumb:
  - spf_hold > spf_init + spf_run + fib_update
- Assume:
  - spf_init = 50 ms
  - spf_run = 80 ms
  - fib_update = 150 ms
  - then spf_hold > 280 ms (maybe 300 ms)
- spf_max is the upper bound on spf_hold

Case 1: Updates Received Quickly on R14



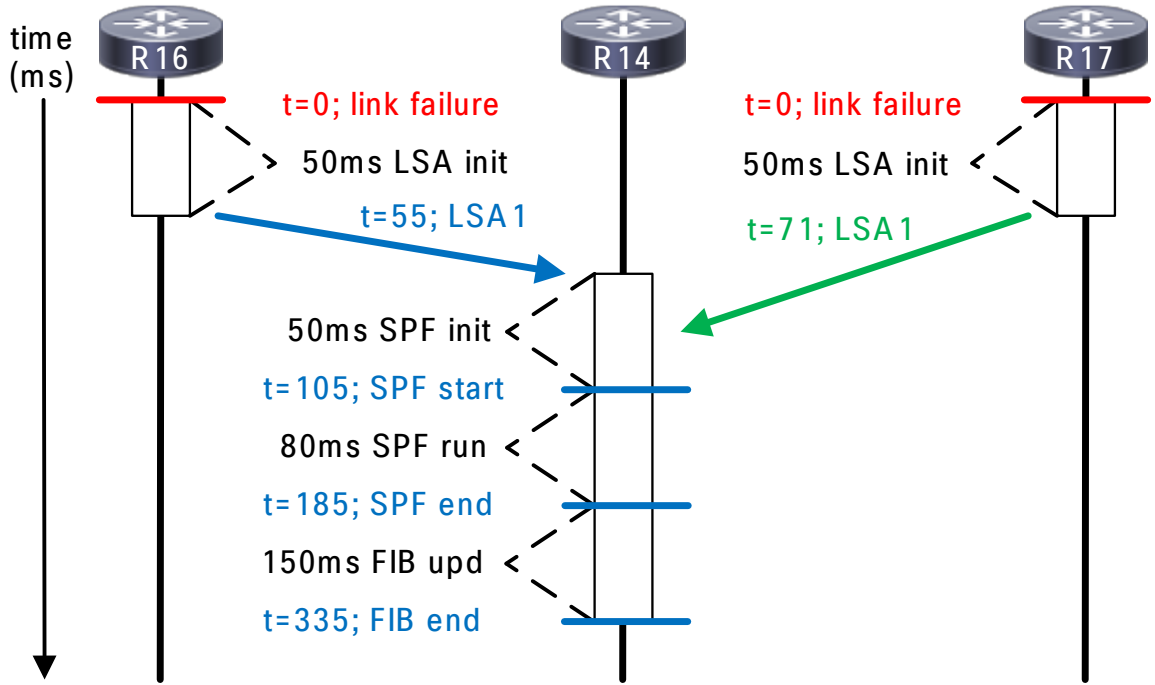
LSA init: 50



SPF init: 50

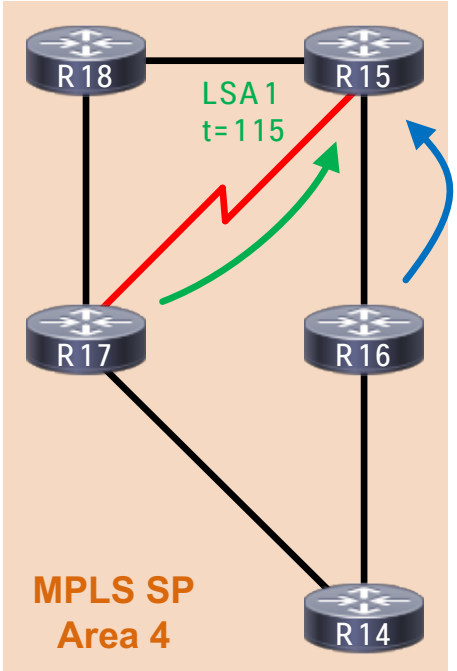
Case 1: Timeline

- Hold and max timers don't matter
- Best case scenario
- All updates received in the spf_init window



Case 2: Updates Received Slowly on R15

- Large time delta between receiving topology updates
- “hold” timers come into play
- Requires multiple SPF runs



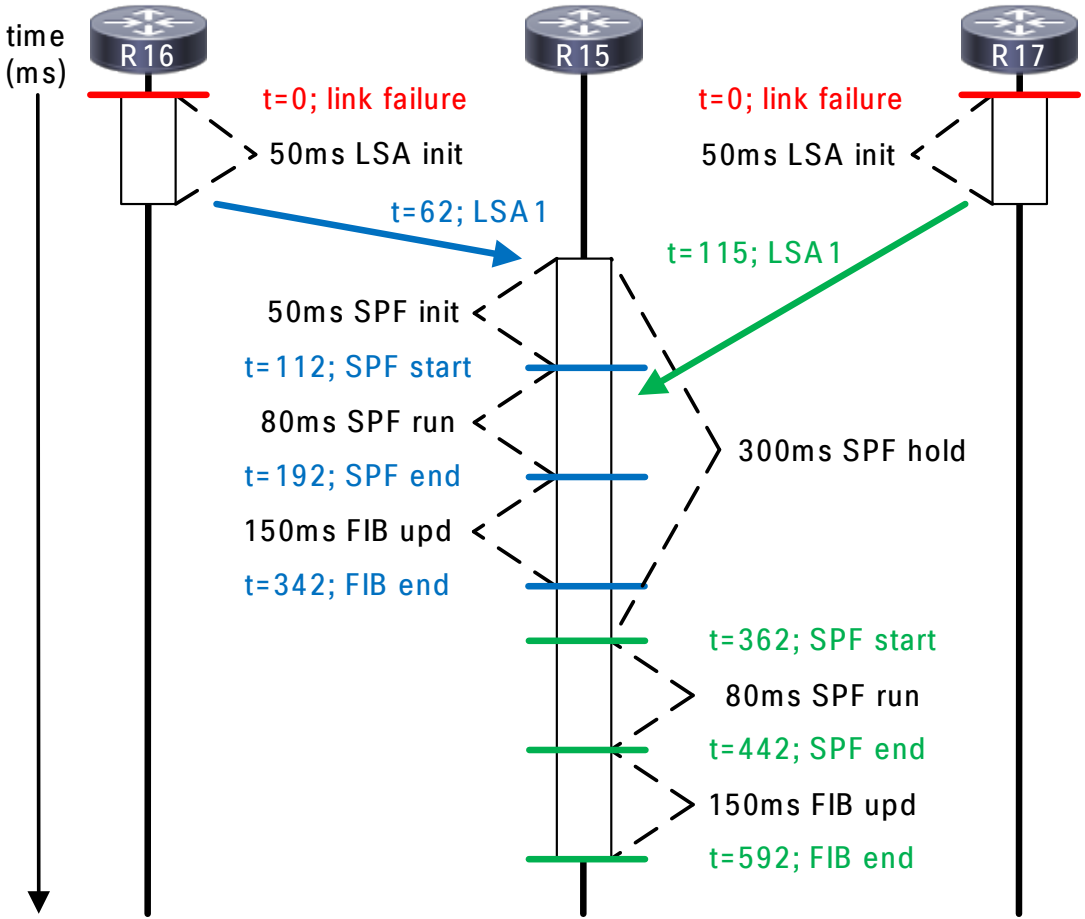
SPF init: 50
SPF hold: 300

Notified: t=62
SPF init: +50
SPF start: t=112
SPF run: +80
SPF end: t=192
FIB upd: +150
FIB end: t=342

Notified: t=62*
SPF hold: +300
SPF start: t=362
SPF run: +80
SPF end: t=442
FIB upd: +150
FIB end: t=592

Case 2: Timeline

- SPF hold time starts when first LSA is received
- Throttles next SPF run
- Increases (doubles) up to the SPF max



OSPF Timer Wrap-up

- Takes time, patience, and TESTING
  - Not a mathematical proof, only suggested starting points

- Here's our final config; try it out!

```
router ospf 1
 timers throttle lsa 50 250 1000
 timers throttle spf 50 300 1000
```

Want more?

- Session configurations on GitHub
  - https://github.com/nickrusso42518/ospf_DIGRST2337
- OSPF over Hub/spoke whitepaper
  - http://njrusmc.net/pub/ospf_dmvpn_anycast.pdf
- Other videos
  - Cisco Live 2019 BRKRST-3310: Troubleshooting OSPF
  - OSPFv3 Graph Tracing: https://www.youtube.com/watch?v=2sLqfs2JZbA
- Twitter @nickrusso42518

Thank you

Possibilities

#CiscoLive