APRICOT 11:Multicast Workshop

Understanding & Deploying IP Multicast Networks

Srini Irigi, Stefan Olofsson, Greg Shepherd, SPG TME

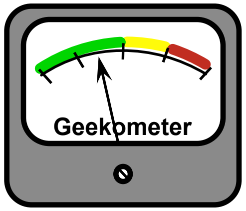
Cisco Systems

Today’s Agenda

- Multicast Fundamentals
- Multicast Service Models, Distribution Trees, Forwarding
- Multicast Protocol Basics
- Layer2 Multicast
- PIM Mechanics
- SSM
- MBGP
- MSDP

Fundamentals of IP Multicast

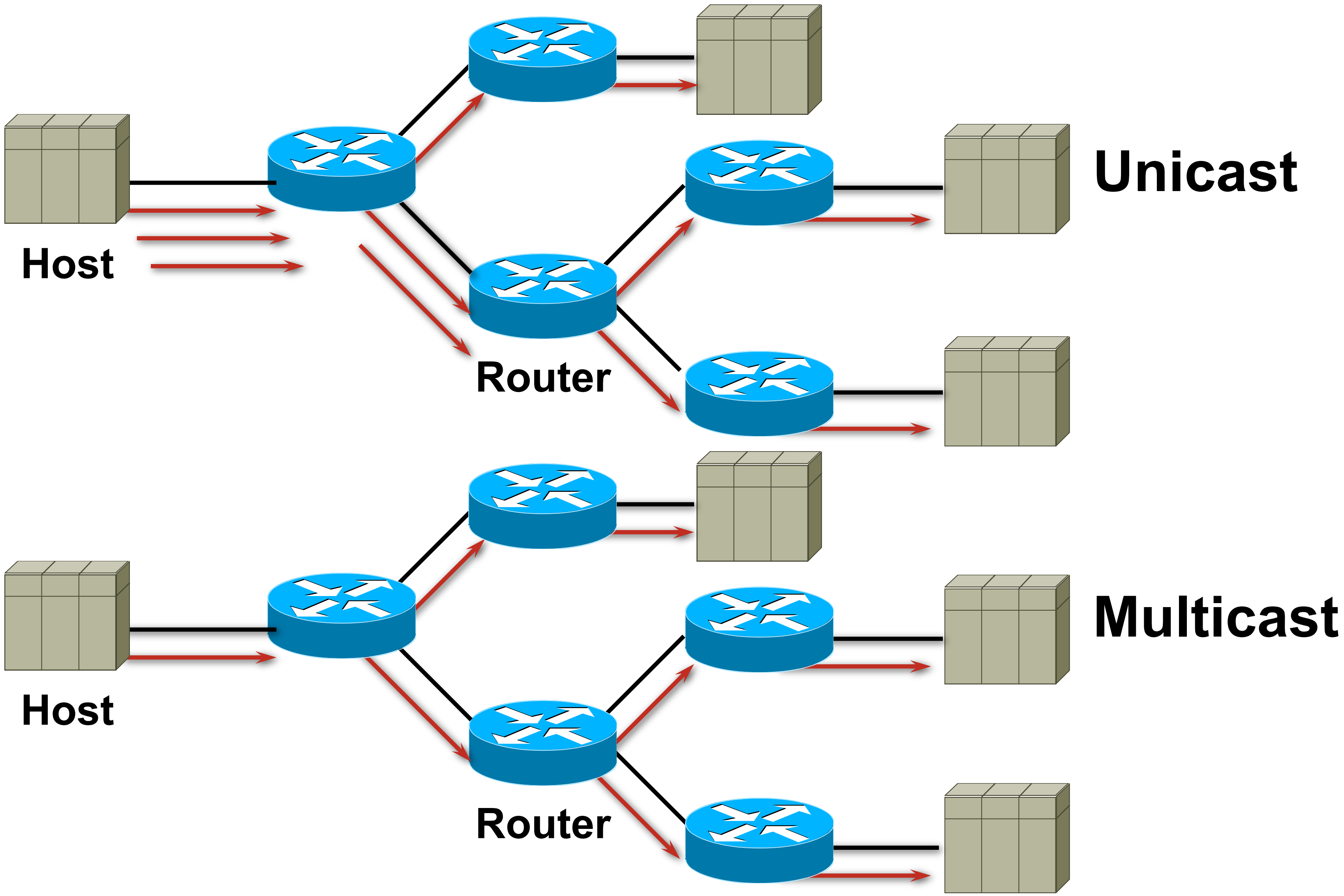
Agenda



- Why Multicast
- Multicast Applications
- Multicast Service Model
- Multicast Distribution Trees
- Multicast Forwarding
- Multicast Protocol Basics

Why Multicast?

Multicast Advantages



Multicast Disadvantages

Multicast Is UDP Based!!!

- Best-effort delivery: Drops are to be expected. Multicast applications should not expect reliable delivery of data and should be designed accordingly. Reliable Multicast is still an area for much research. Expect to see more developments in this area.
- No congestion avoidance: Lack of TCP windowing and “slow-start” mechanisms can result in network congestion. If possible, Multicast applications should attempt to detect and avoid congestion conditions.
- Duplicates: Some multicast protocol mechanisms (e.g. Asserts, Registers and Shortest-Path Tree Transitions) result in the occasional generation of duplicate packets. Multicast applications should be designed to expect occasional duplicate packets.
- Out-of-sequence packets: Various network events can result in packets arriving out of sequence. Multicast applications should be designed to handle packets that arrive in some other sequence than they were sent by the source.

Multicast Service Model

IP Multicast Service Model

- RFC 1112 (Host Ext. for Multicast Support)
- Each multicast group identified by a class-D IP address
- Members of the group could be present anywhere in the Internet
- Members join and leave the group and indicate this to the routers
- Senders and receivers are distinct:

  i.e., a sender need not be a member

- Routers listen to all multicast addresses and use multicast routing protocols to manage groups

IP Multicast Packet

- Source address
  - Unique unicast IP address of the packet source
- Destination address
  - ClassD address range
  - Does NOT represent a unique unicast destination address
  - Used to represent a unique group of receivers

IP Multicast Addressing

- Multicast Group Addresses (224.0.0.0/4)

  Range: 224.0.0.0–239.255.255.255

  Old Class D address range.

  High-order 4 bits are 1110

Multicast Address Ranges

- Link-Local Address Range

  224.0.0.0–224.0.0.255

- Global Address Range

  224.0.1.0–238.255.255.255

- Administratively Scoped Address Range

  239.0.0.0–239.255.255.25

- Scope Relative Address Range

  Top 256 addresses of a Scoped Address Range

Link-Local Address Range

- Assigned by IANA
  - 224.0.0.0–224.0.0.255
  - Local wire multicast
    - TTL = 1
  - Examples:
    - 224.0.0.5 = OSPF_DR’s
    - 224.0.0.10 = EIGRP Hello’s
    - 224.0.0.13 = All_PIM_Routers
    - 224.0.0.22 = All_IGMPv3_Routers

Global Address Range

- Assigned by IANA

  Address Range: 224.0.1.0–238.255.255.255

  Generally intended for "global" Internet scope multicast

  Sometimes assigned to specific protocols

    Example: Auto-RP (224.0.1.39 and 224.0.1.40)

  Problem:

    IANA is coming under increasing pressure from companies to assign them blocks of addresses for their applications or content services

    This was never the intent of this block!

      GLOP Addressing or SSM should be used instead!

Global Multicast Address Assignment

- Dynamic Group Address Assignment
  - Historically accomplished using SDR application
    - Sessions announced over well-known group(s)
      - Address collisions detected and resolved at session creation time
      - Has problems scaling
  - Other techniques considered
    - Multicast Address Set-Claim (MASC)
      - Hierarchical, dynamic address allocation scheme
      - Unlikely to be deployed
  - No really good dynamic assignment method available for Global multicast
  - But is dynamic assignment really necessary with GLOP and SSM available?

Global Multicast Address Assignment

- Static Group Address Assignment
  - RFC 3180—GLOP Addressing in 233/8
    - Group range: 233.0.0.0–233.255.255.255
      - Your AS number is inserted in middle two octets
      - Remaining low-order octet used for group assignment
    - EGLOP Addresses
      - Make use of private AS numbers
      - Assigned by a Registration Authority

Global Multicast Address Assignment

- Static Group Address Assignment
  - Source Specific Multicast
    - Address range: 232.0.0.0/8
    - Flows based on both Group **and** Source address
      - Two different content flows can share the same Group address without interfering with each other
    - **Provides virtually unlimited address space!**
      - Preferred method for global one-to-many multicast

Private Multicast Address Assignment

- Assigned from the private 239.0.0.0/8 range

  May be subdivided into geographic scopes ranges

  Administration responsibility can be by scope range

- Question:

  "What technology is most often used to manage private multicast assignment?"

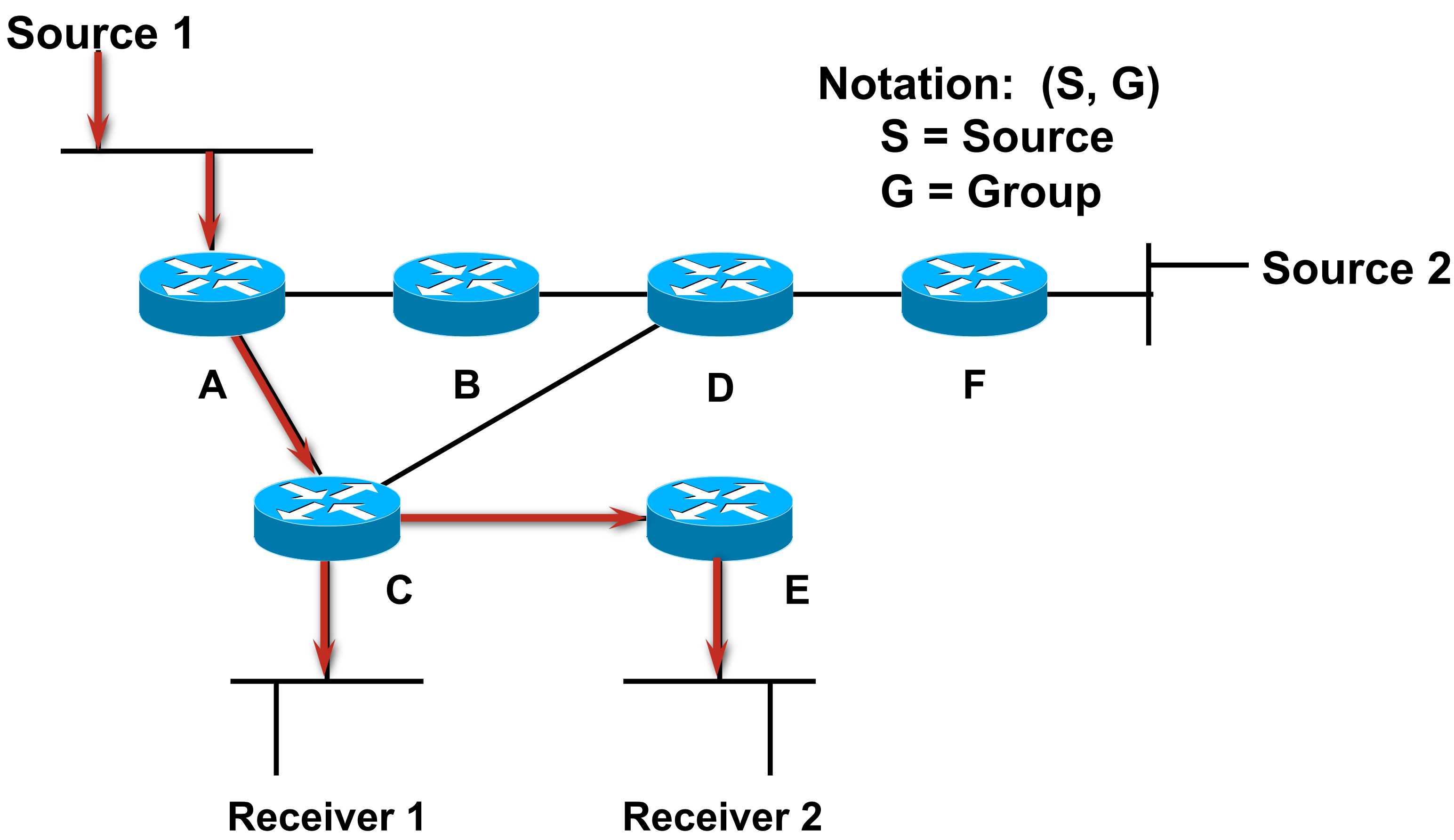
- Answer:

  A spreadsheet

Multicast Distribution Trees

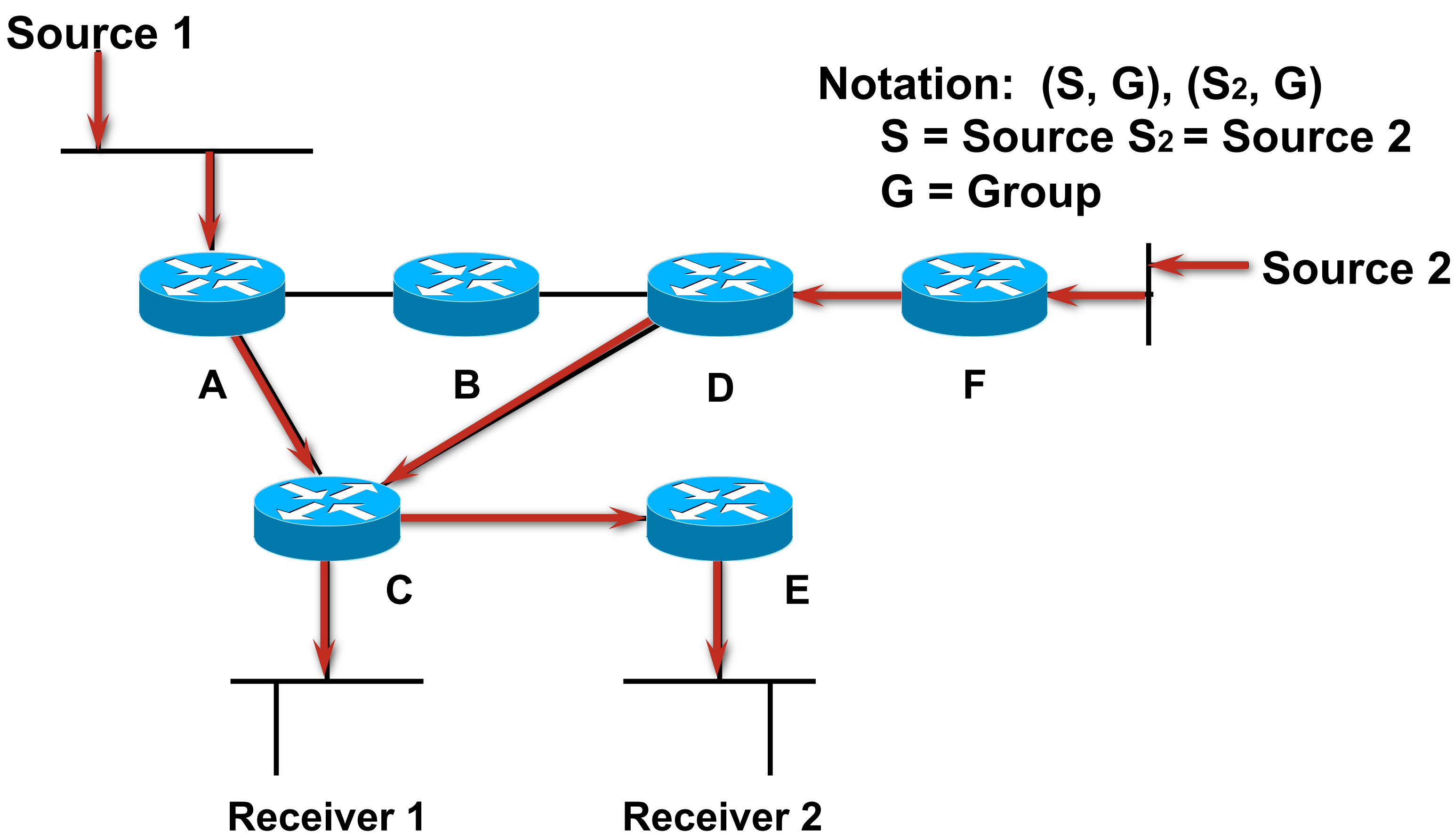
Multicast Distribution Trees

Shortest Path or Source Tree



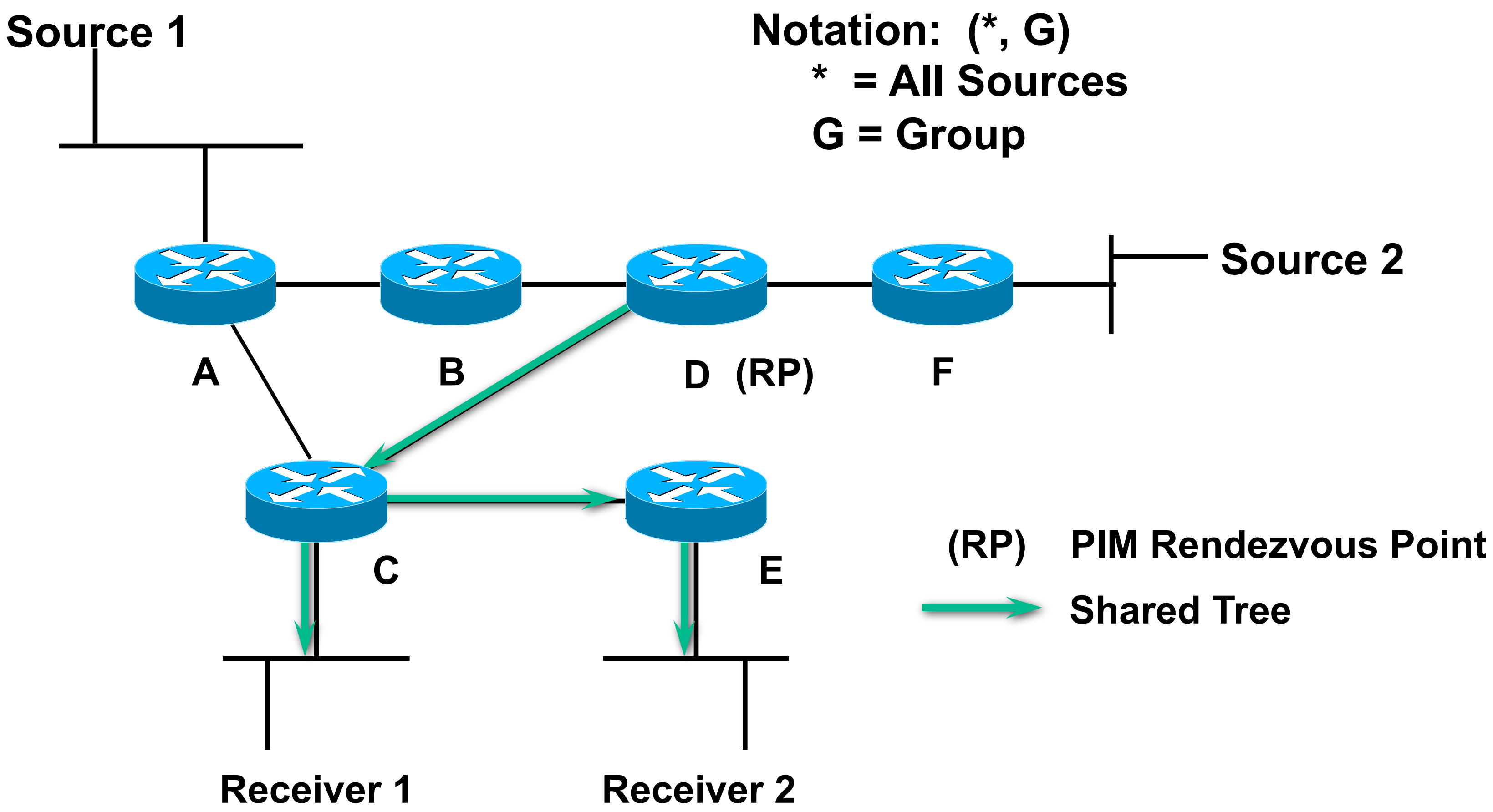
Multicast Distribution Trees

Shortest Path or Source Tree



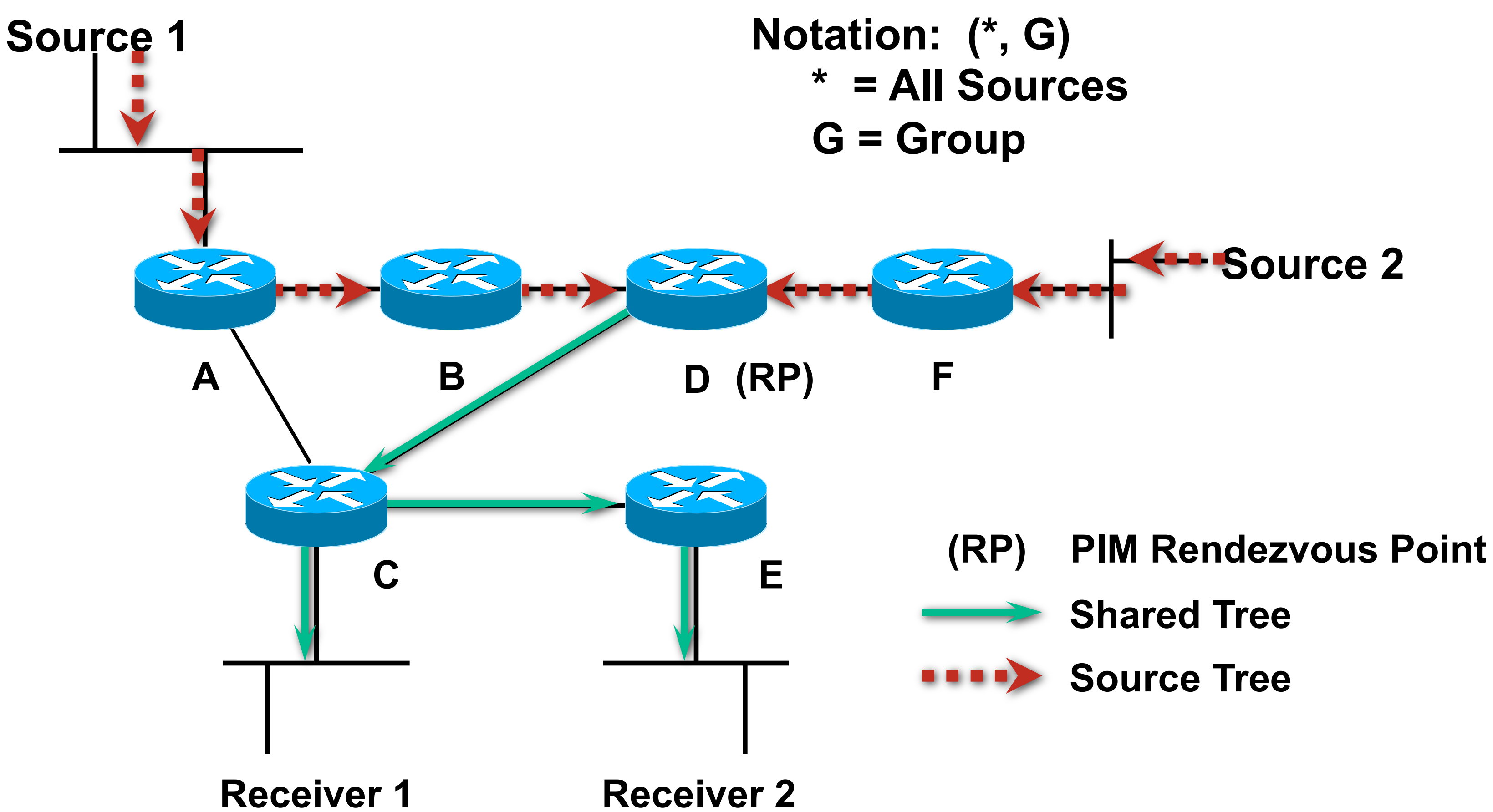
Multicast Distribution Trees

Shared Tree



Multicast Distribution Trees

Shared Tree



Multicast Distribution Trees

Characteristics of Distribution Trees

- Shortest Path trees

  Uses more memory n(S x G) but you get optimal paths from source to all receivers; minimizes delay

- Shared trees

  Uses less memory n(G) but you may get sub-optimal paths from source to all receivers; may introduce extra delay

Multicast Forwarding

Unicast vs. Multicast Forwarding

- Unicast Forwarding
  - Destination IP address directly indicates where to forward packet
  - Forwarding is hop-by-hop
    - Unicast routing table determines interface and next-hop router to forward packet

Unicast vs. Multicast Forwarding

- Multicast Forwarding
  - Destination IP address (group) doesn't directly indicate where to forward packet
  - Forwarding is connection-oriented
    - Receivers must first be "connected" to the source before traffic begins to flow
      - Connection messages (PIM Joins) follow unicast routing table toward multicast source
      - Build Multicast Distribution Trees that determine where to forward packets
      - Distribution Trees rebuilt dynamically in case of network topology changes

Reverse Path Forwarding (RPF)

- The RPF Calculation

  The multicast source address is checked against the unicast routing table

  This determines the interface and upstream router in the direction of the source to which PIM Joins are sent

  This interface becomes the "Incoming" or RPF interface

    A router forwards a multicast datagram only if received on the RPF interface

Reverse Path Forwarding (RPF)

- RPF Calculation

  Based on Address of tree root

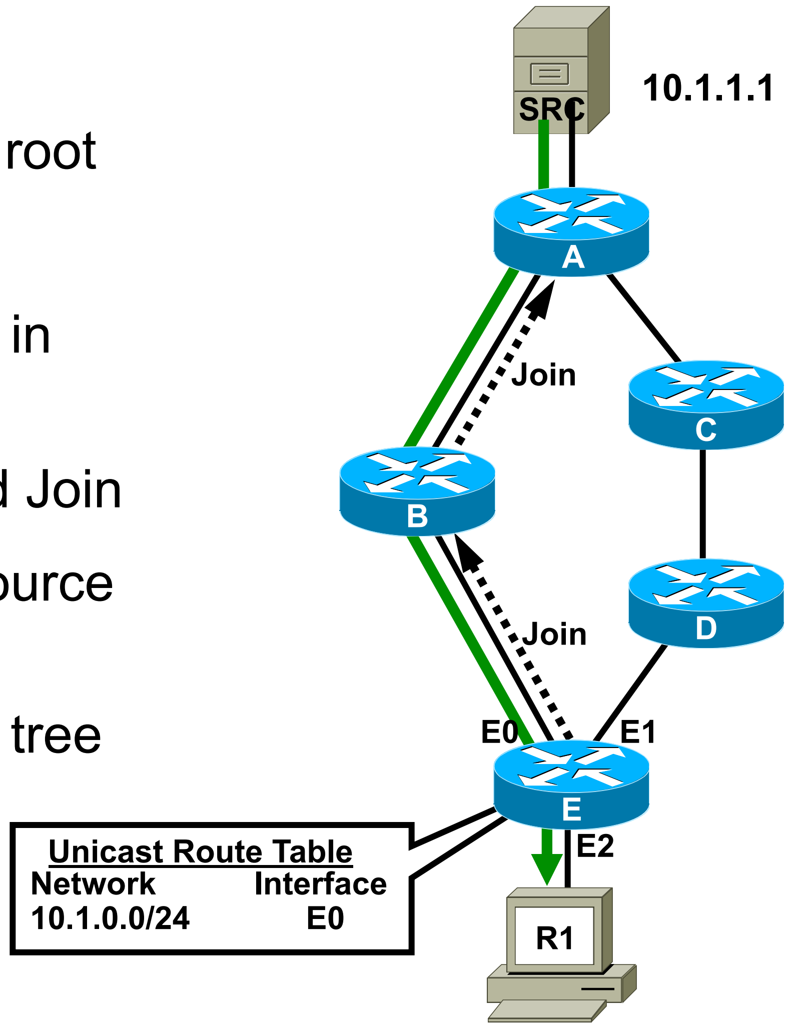
    Source or RP

  Best path to source found in Unicast Route Table

  Determines where to send Join

  Joins continue towards Source to build multicast tree

  Multicast data flows down tree



Reverse Path Forwarding (RPF)

- RPF Calculation

  Based on Address of tree root

    Source or RP

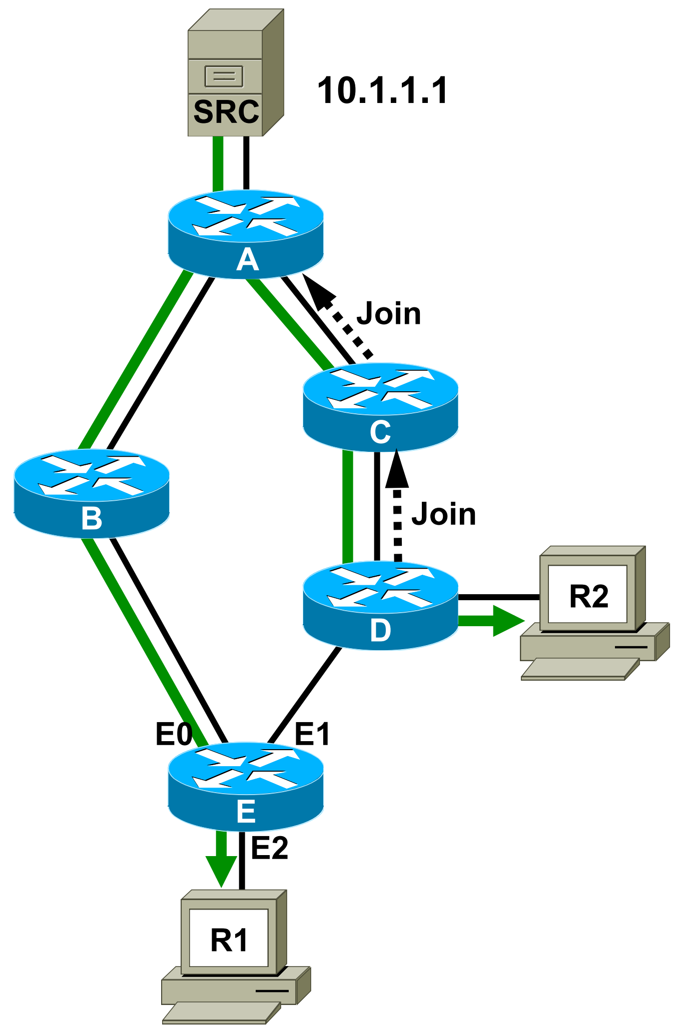
  Best path to source found in Unicast Route Table

  Determines where to send Join

  Joins continue towards Source to build multicast tree

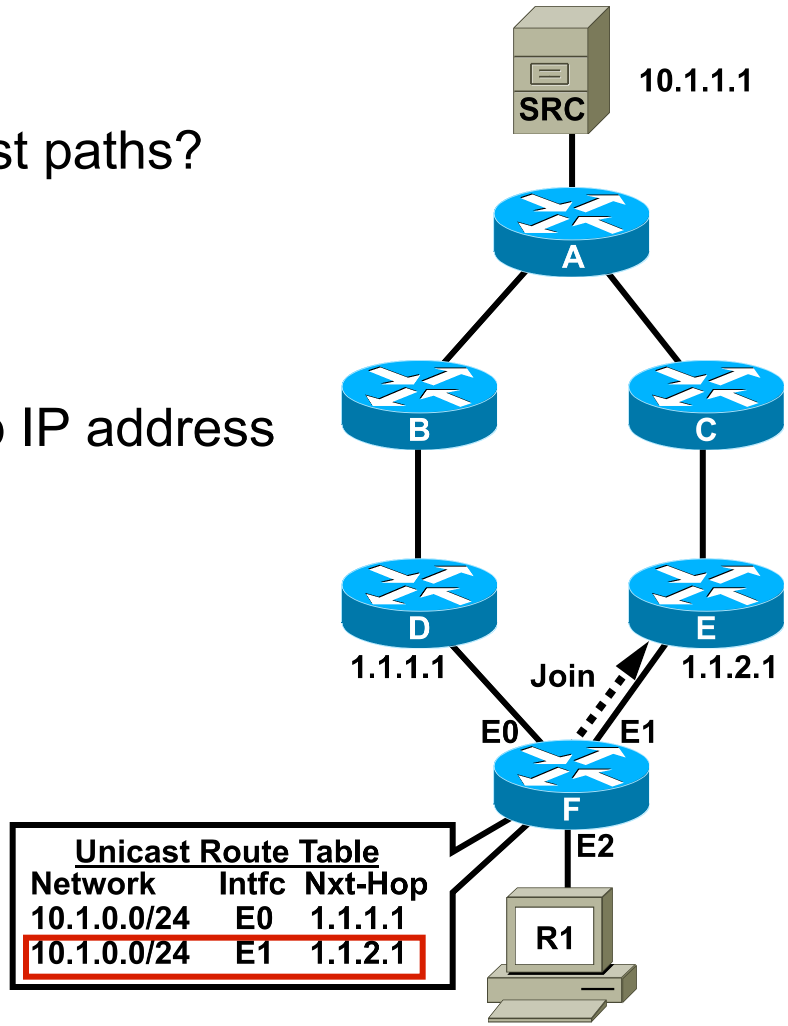
  Multicast data flows down tree

  Repeat for other receivers

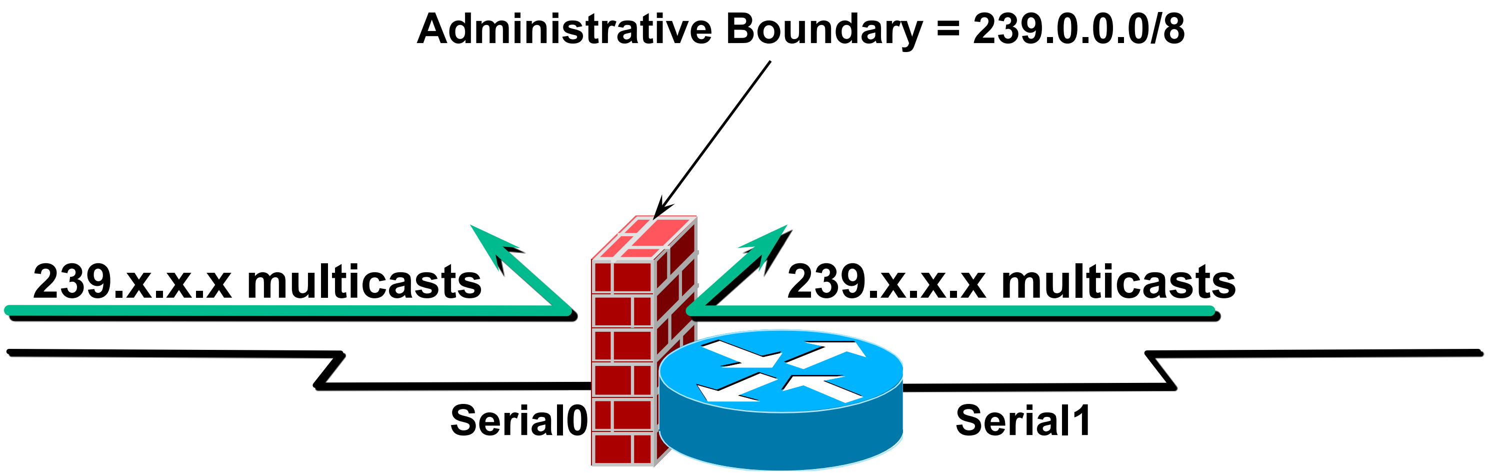


Reverse Path Forwarding (RPF)

- RPF Calculation
  - What if we have equal-cost paths?
    - We can’t use both
  - Tie-Breaker
    - Use highest Next-Hop IP address

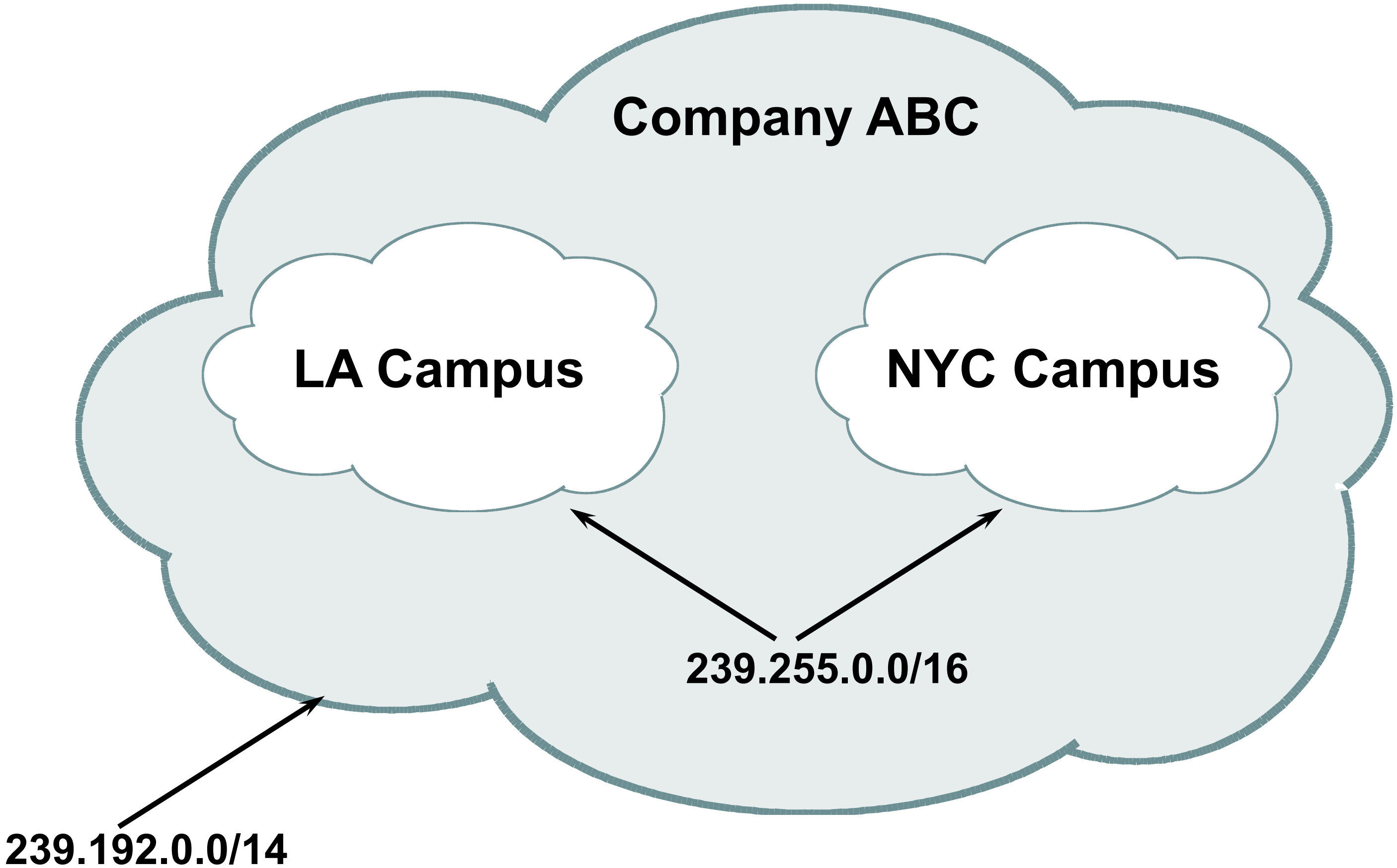


Administrative Boundaries



- Configured using the **ip multicast boundary <acl>** interface command

Administrative Boundaries



Multicast Protocol Basics

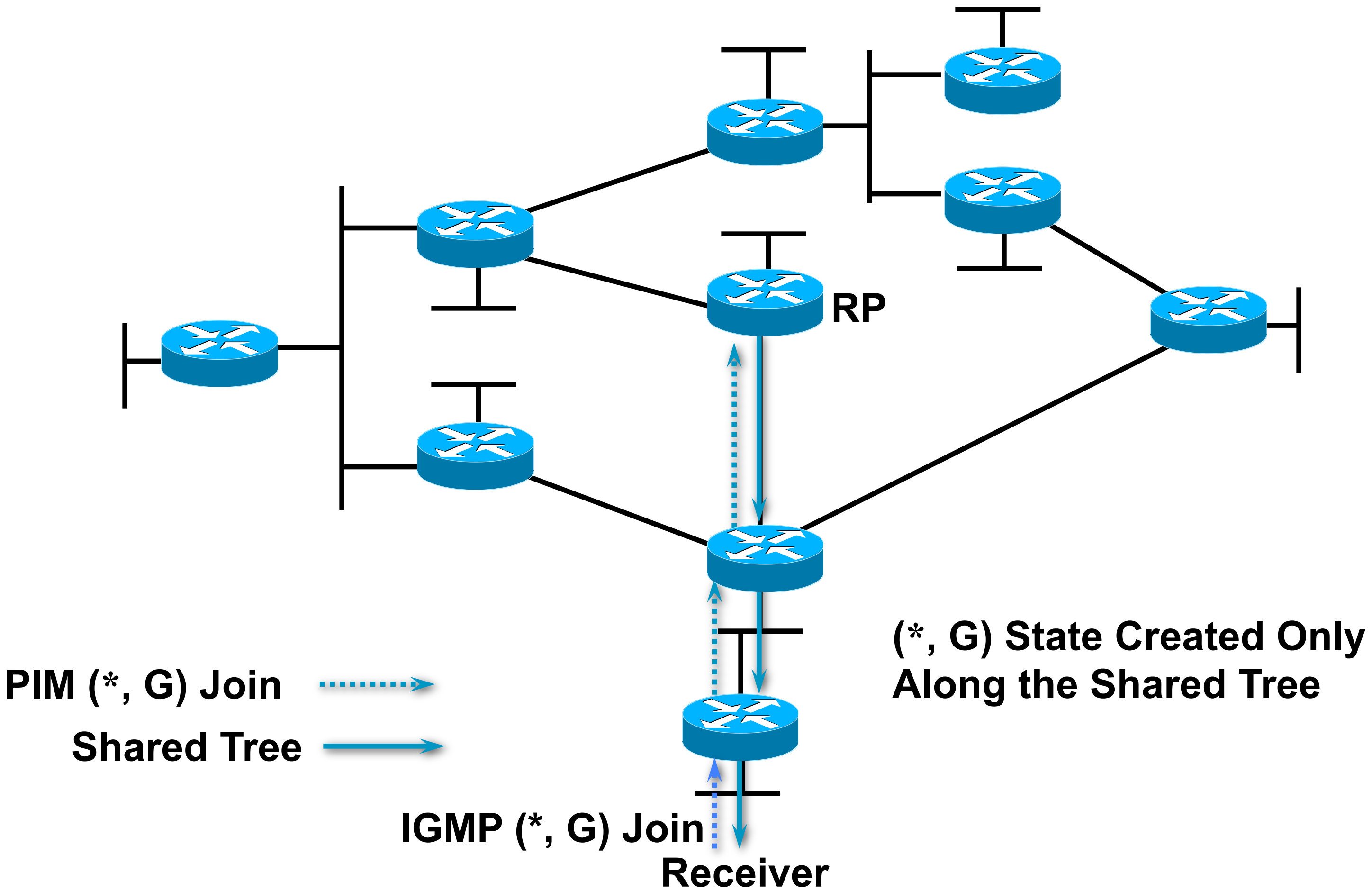
Types of Multicast Protocols

- Dense-mode
  - Uses “Push” model
  - Traffic flooded throughout network
  - Pruned back where it is unwanted
  - Flood and prune behavior (typically every three minutes)
- Sparse-mode
  - Uses “Pull” model
  - Traffic sent only to where it is requested
  - Explicit Join behavior

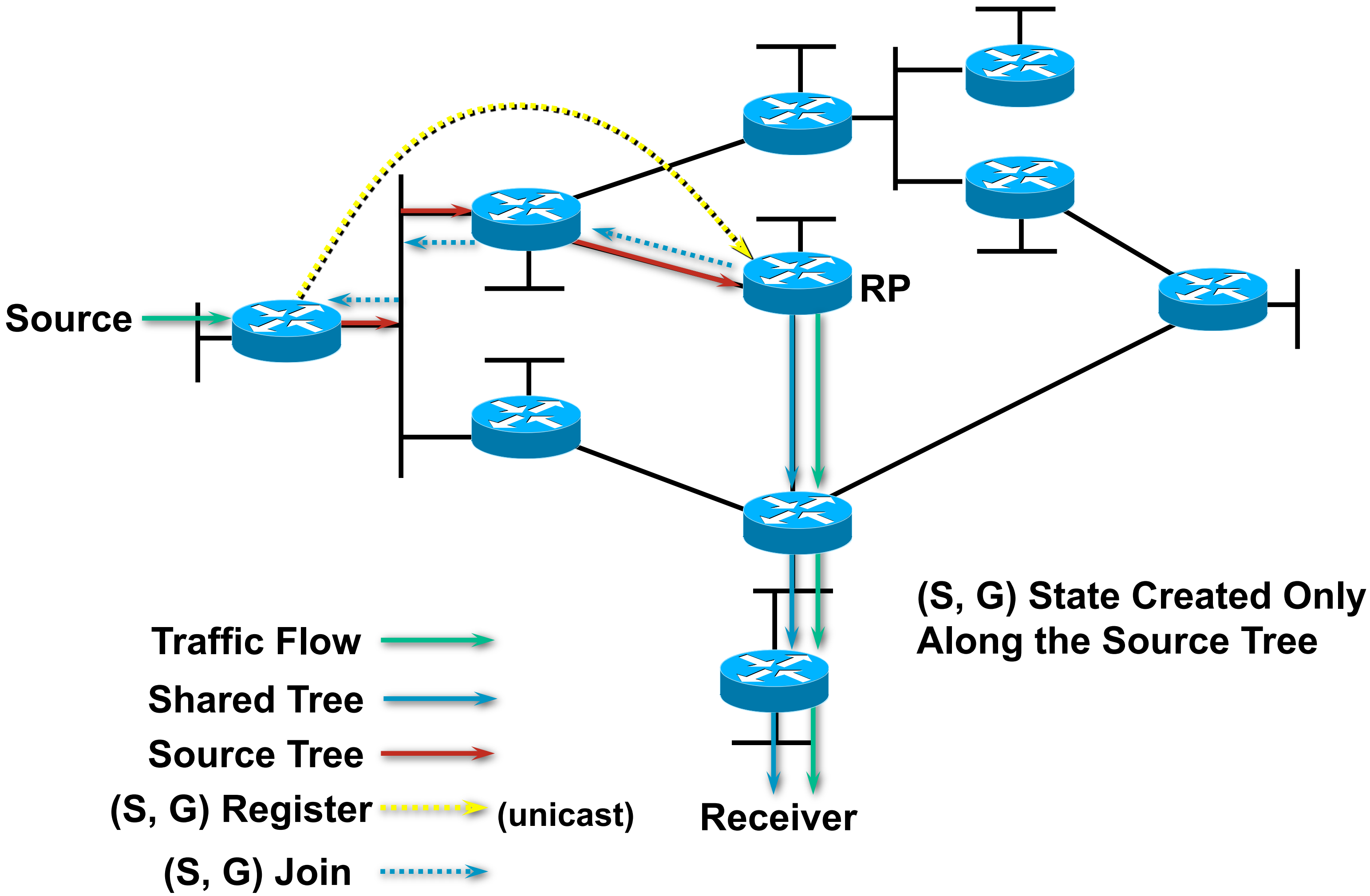
PIM-SM (RFC 4601)

- Supports both source and shared trees
  - Assumes no hosts want multicast traffic unless they specifically ask for it
- Uses a Rendezvous Point (RP)
  - Senders and Receivers "rendezvous" at this point to learn of each others existence
    - Senders are "registered" with RP by their first-hop router
    - Receivers are "joined" to the Shared Tree (rooted at the RP) by their local Designated Router (DR)
- Appropriate for …
  - Wide scale deployment for **both** densely and sparsely populated groups in the enterprise
  - Optimal choice for all production networks regardless of size and membership density

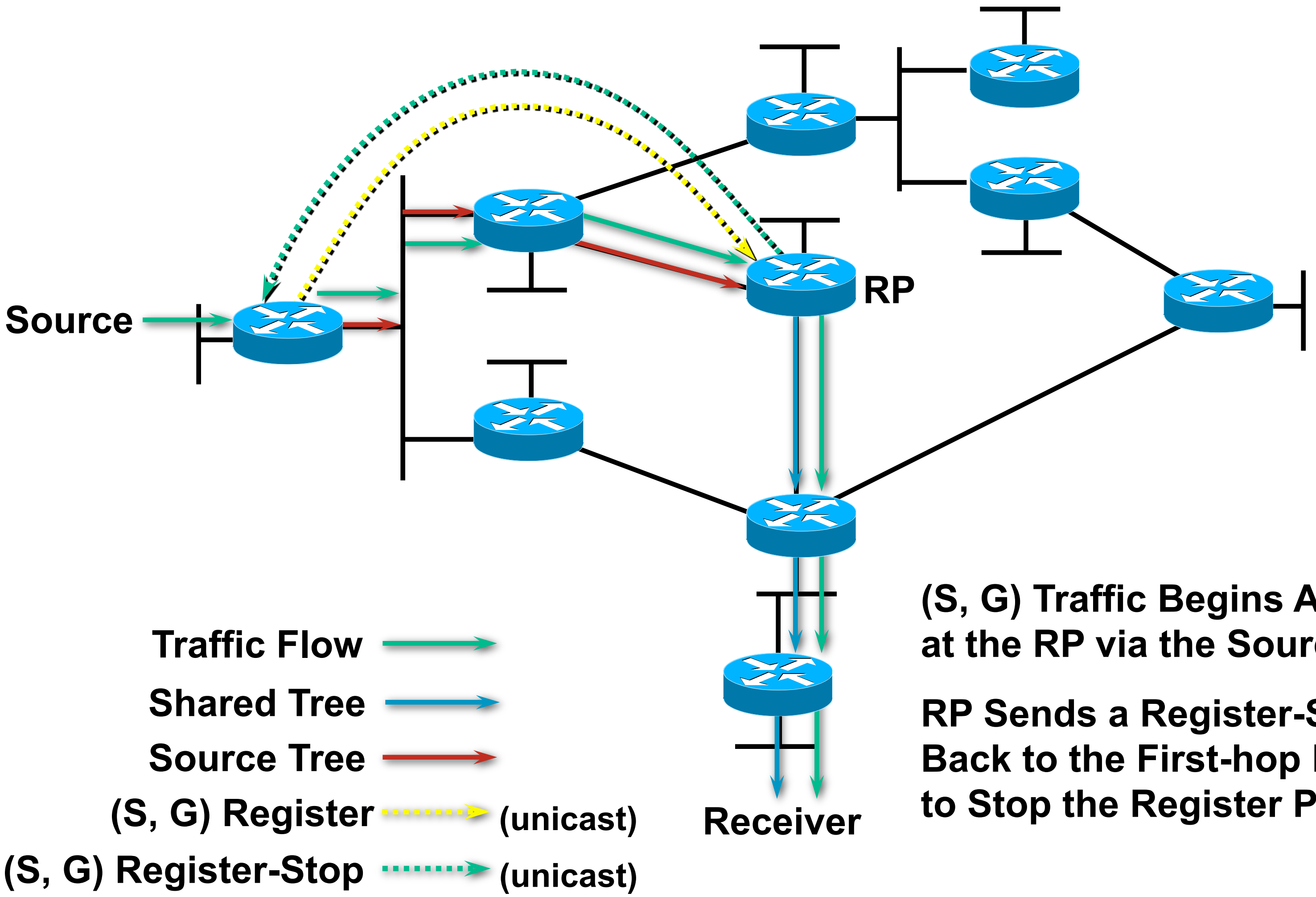
PIM-SM Shared Tree Join



PIM-SM Sender Registration



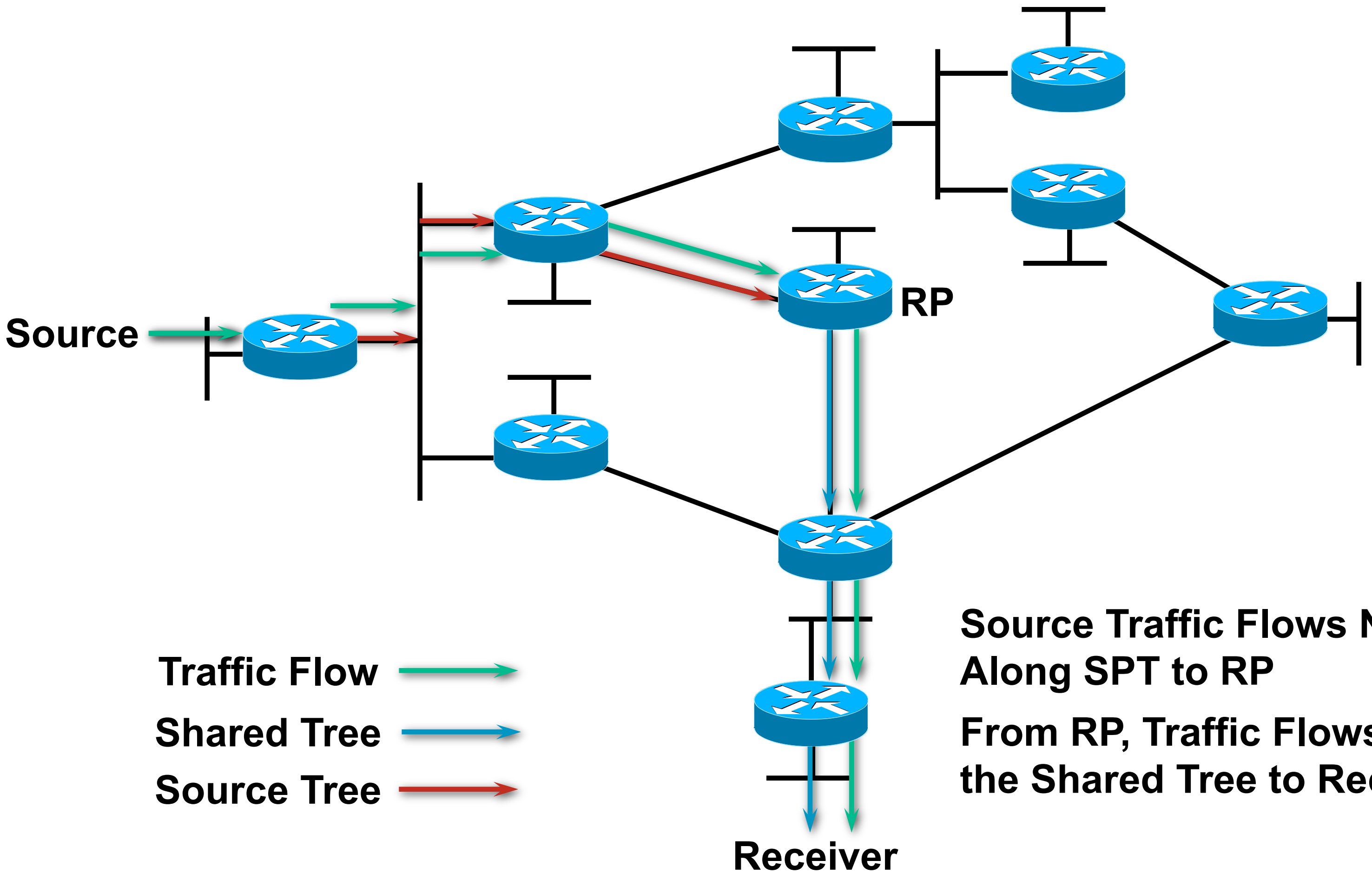
PIM-SM Sender Registration



(S, G) Traffic Begins Arriving at the RP via the Source Tree

RP Sends a Register-Stop Back to the First-hop Router to Stop the Register Process

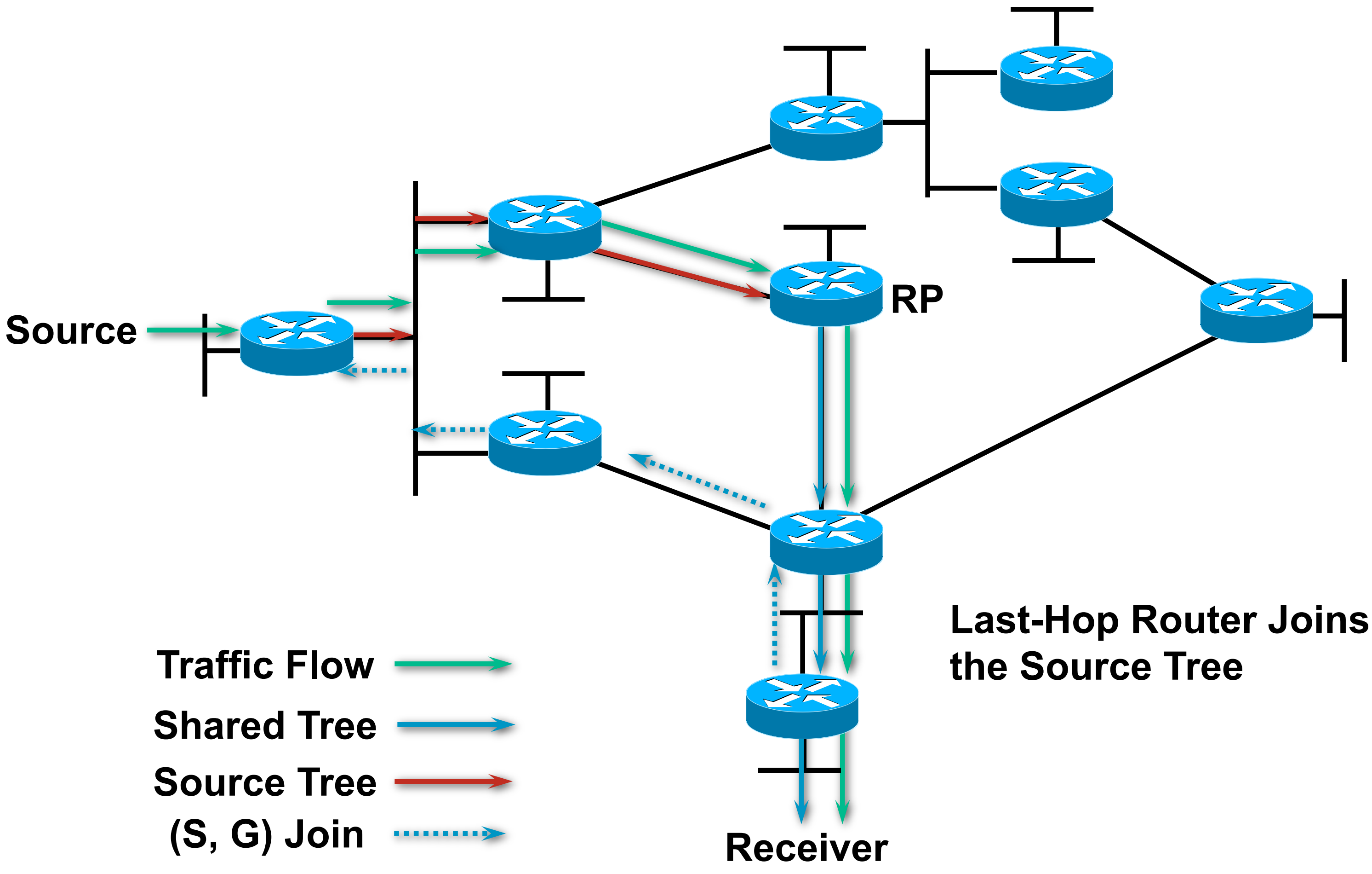
PIM-SM Sender Registration



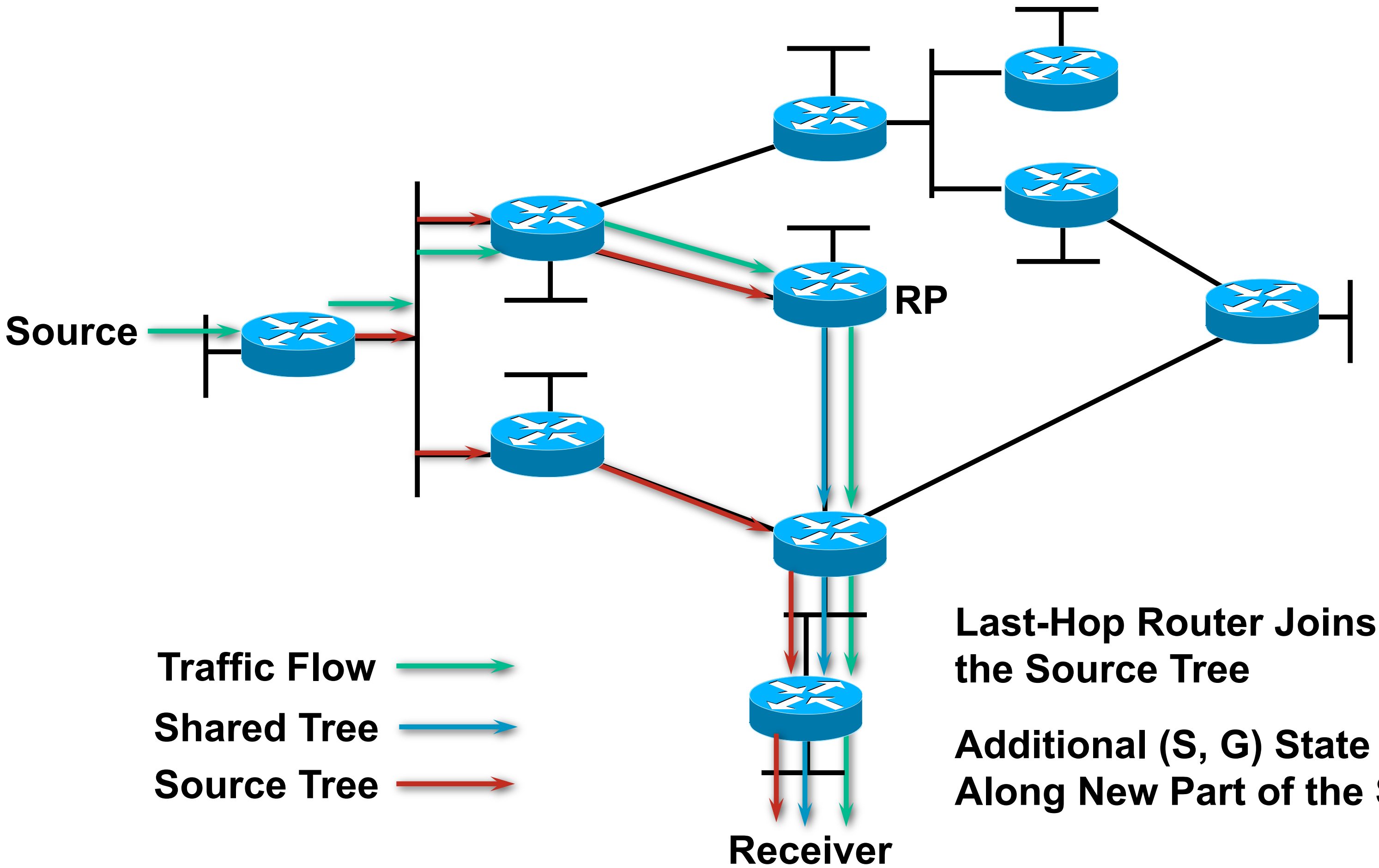
Source Traffic Flows Natively Along SPT to RP

From RP, Traffic Flows Down the Shared Tree to Receivers

PIM-SM SPT Switchover



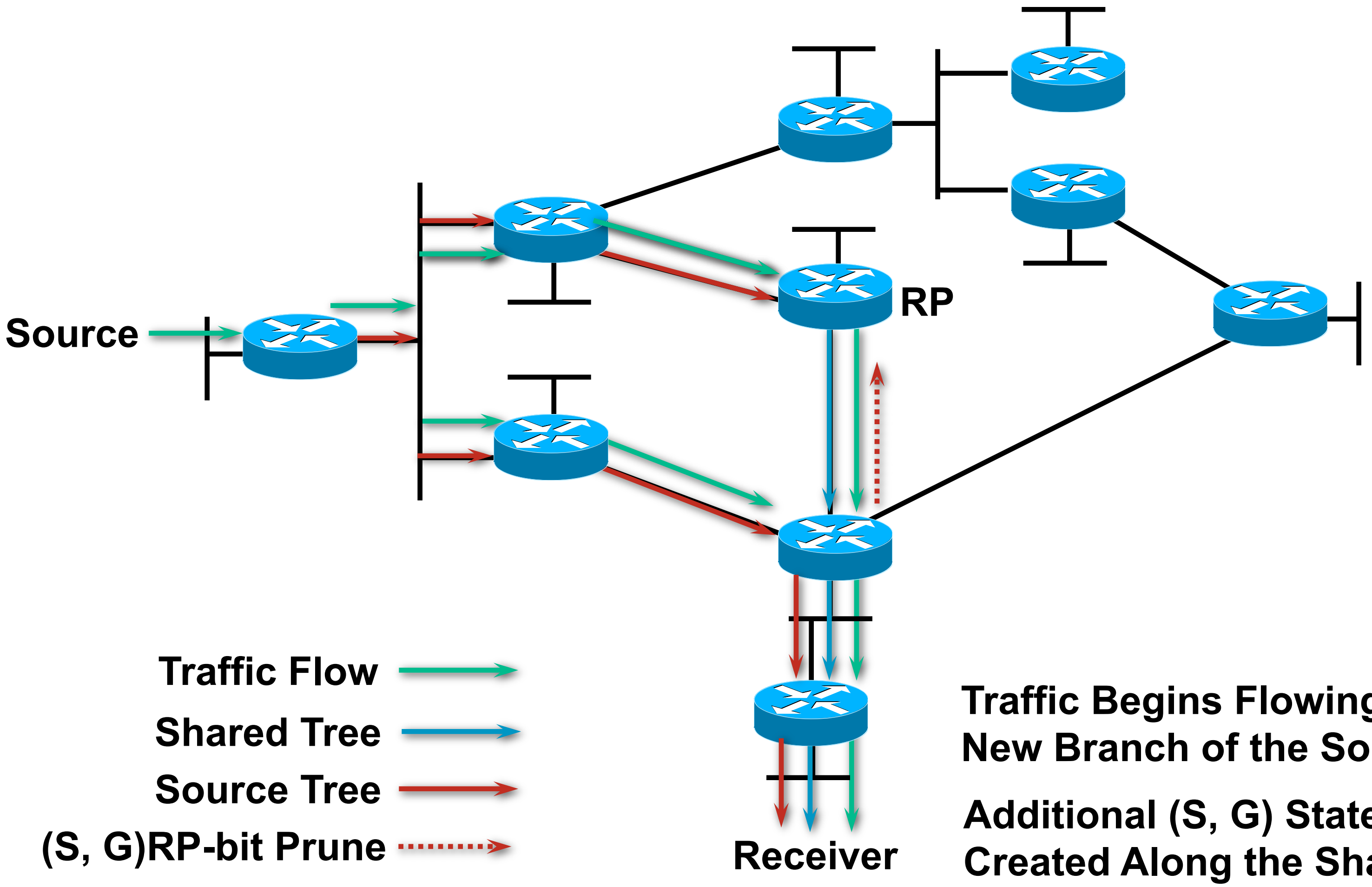
PIM-SM SPT Switchover



Last-Hop Router Joins the Source Tree

Additional (S, G) State Is Created Along New Part of the Source Tree

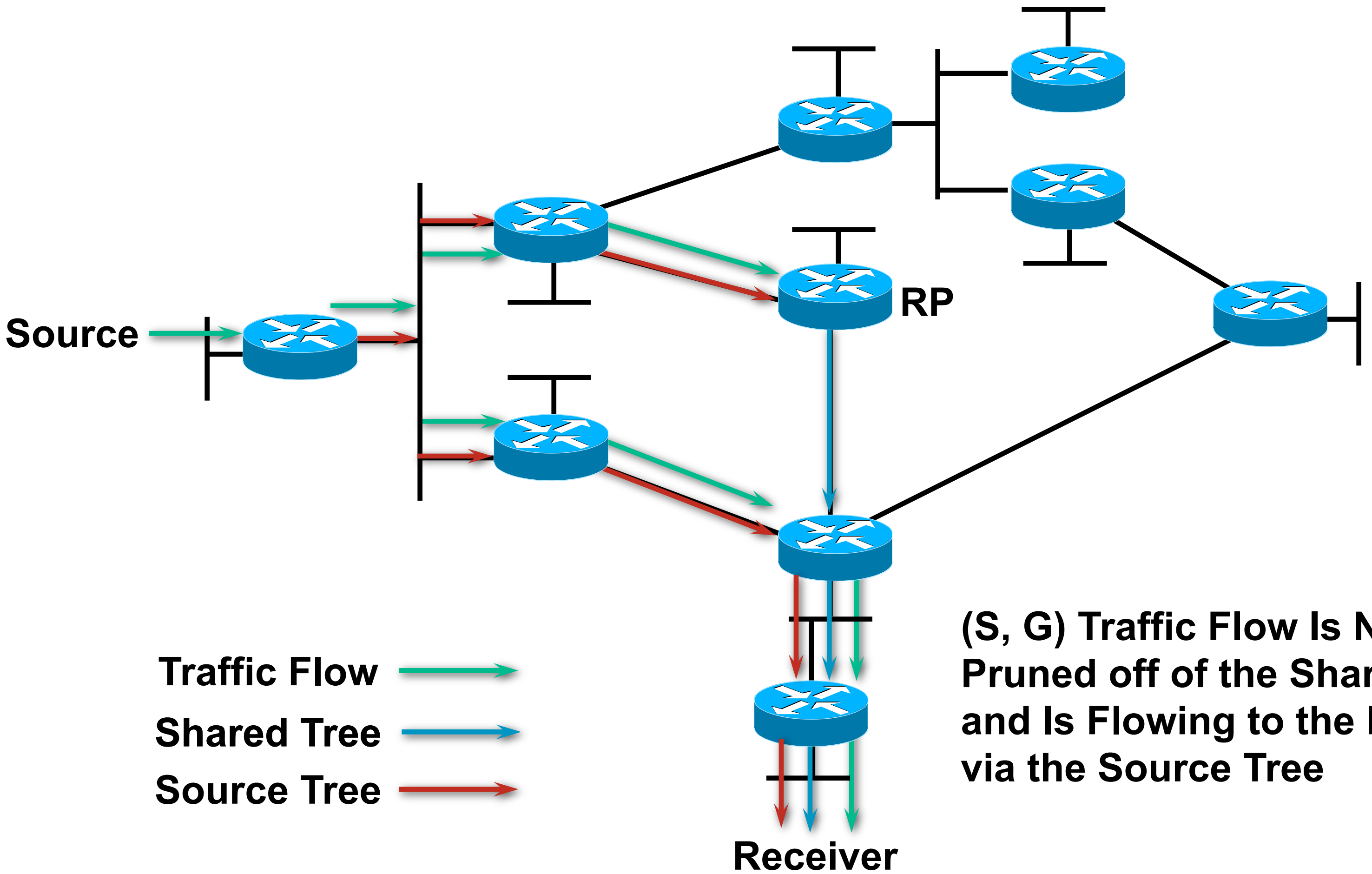
PIM-SM SPT Switchover



Traffic Begins Flowing Down the New Branch of the Source Tree

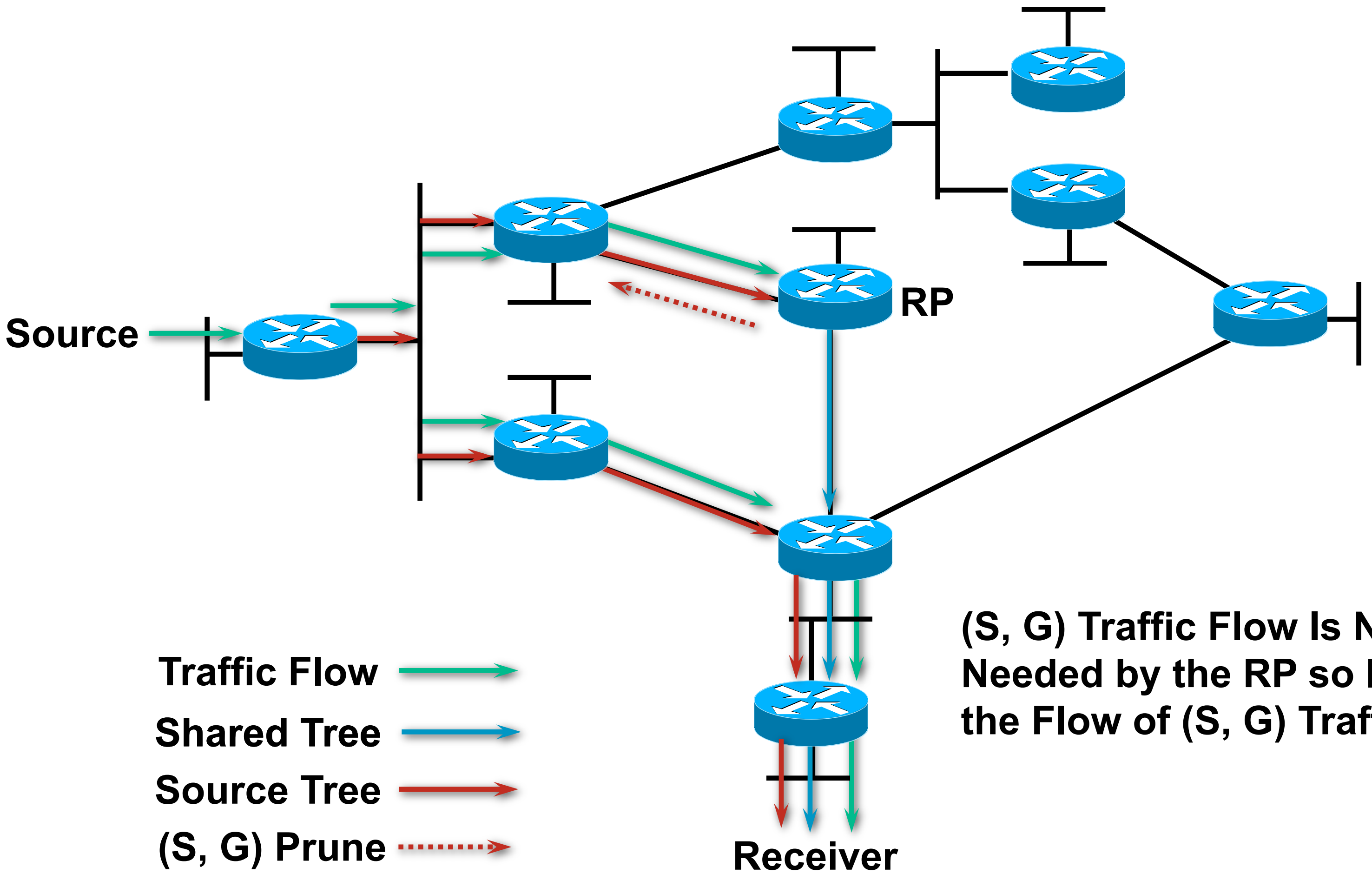
Additional (S, G) State Is Created Along the Shared Tree to Prune off (S, G) Traffic

PIM-SM SPT Switchover



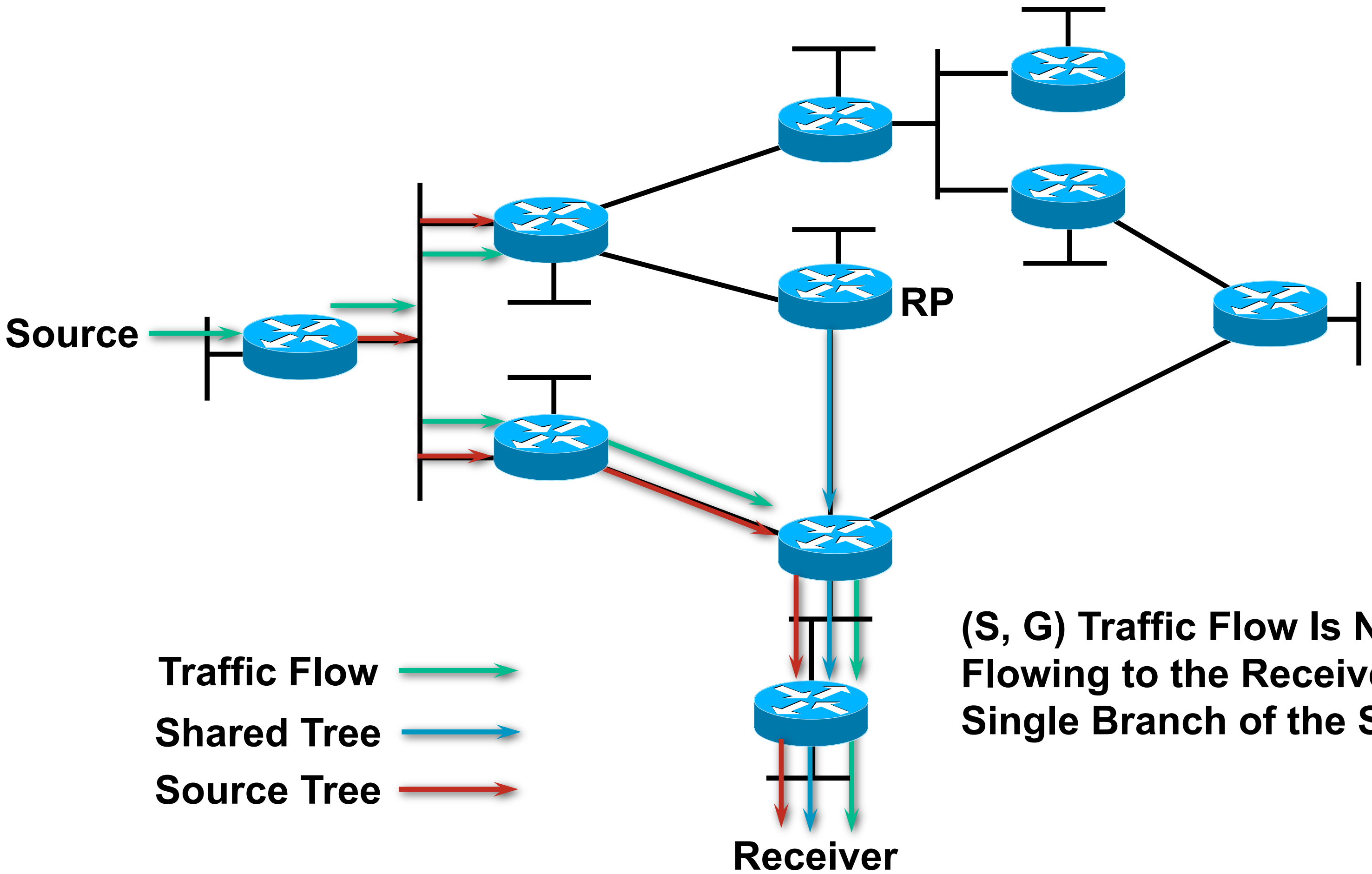
(S, G) Traffic Flow Is Now Pruned off of the Shared Tree and Is Flowing to the Receiver via the Source Tree

PIM-SM SPT Switchover



(S, G) Traffic Flow Is No Longer Needed by the RP so It Prunes the Flow of (S, G) Traffic

PIM-SM SPT Switchover



(S, G) Traffic Flow Is Now Only Flowing to the Receiver via a Single Branch of the Source Tree

PIM-SM FFF

PIM-SM Frequently Forgotten Fact

"The default behavior of PIM-SM is that routers with directly connected members will join the Shortest Path Tree as soon as they detect a new multicast source."

PIM-SM—Evaluation

- Advantages:
  - Traffic only sent down "joined" branches
  - Can switch to optimal source-trees for high traffic sources dynamically
  - Unicast routing protocol-independent
  - Basis for inter-domain multicast routing
    - When used with MBGP and MSDP
- Disadvantages
  - Few if any
- Primary application
  - All production multicast networks with sparse or dense distribution of receivers

Protocol Summary

Conclusion

"Sparse mode good, dense mode bad!"

R. Davis
"The Caveman's Guide to IP Multicast", 2000

IP Multicast at Layer 2

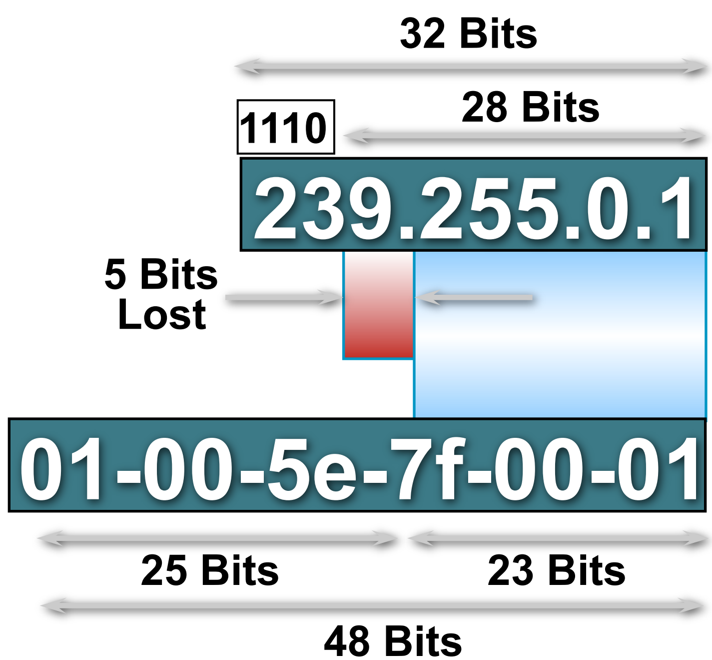
Module Agenda

- MAC Layer Multicast Addresses
- IGMPv2
- IGMPv3
- L2 Multicast Frame Switching
  - IGMP Snooping
  - PIM Snooping

MAC Layer Multicast Addresses

Layer 2 Multicast Addressing

IP Multicast MAC Address Mapping



Layer 2 Multicast Addressing

IP Multicast MAC Address Mapping

Be aware of the 32:1 address overlap

32—IP Multicast Addresses

224.1.1.1
224.129.1.1
225.1.1.1
225.129.1.1
.
.
.
238.1.1.1
238.129.1.1
239.1.1.1
239.129.1.1

1—Multicast MAC Address (Ethernet)

0x0100.5E01.0101

IGMPv2

IGMP

- How hosts tell routers about group membership
- Routers solicit group membership from directly connected hosts
- RFC 1112 specifies first version of IGMP
- RFC 2236 specifies IGMPv2
  - Most widely deployed and supported
- RFC 3376 specifies IGMPv3
  - Growing support (required for SSM)

IGMPv2

RFC 2236

- Membership queries

  Queries sent to 224.0.0.1 with ttl = 1

  One router on LAN is elected to send queries

  Query interval 60–120 seconds

- Membership reports

  IGMP report sent by one host suppresses sending by others

  Restrict to one report per group per LAN

  Unsolicited reports sent by host, when it first joins the group

IGMPv2

RFC 2236

- Group-specific query

  Router sends Group-specific queries to make sure there are no members present before stopping to forward data for the group for that subnet

- Leave Group message

  Host sends leave message if it leaves the group and is the last member (reduces leave latency in comparison to v1)

IGMPv2

RFC 2236

- Querier election mechanism

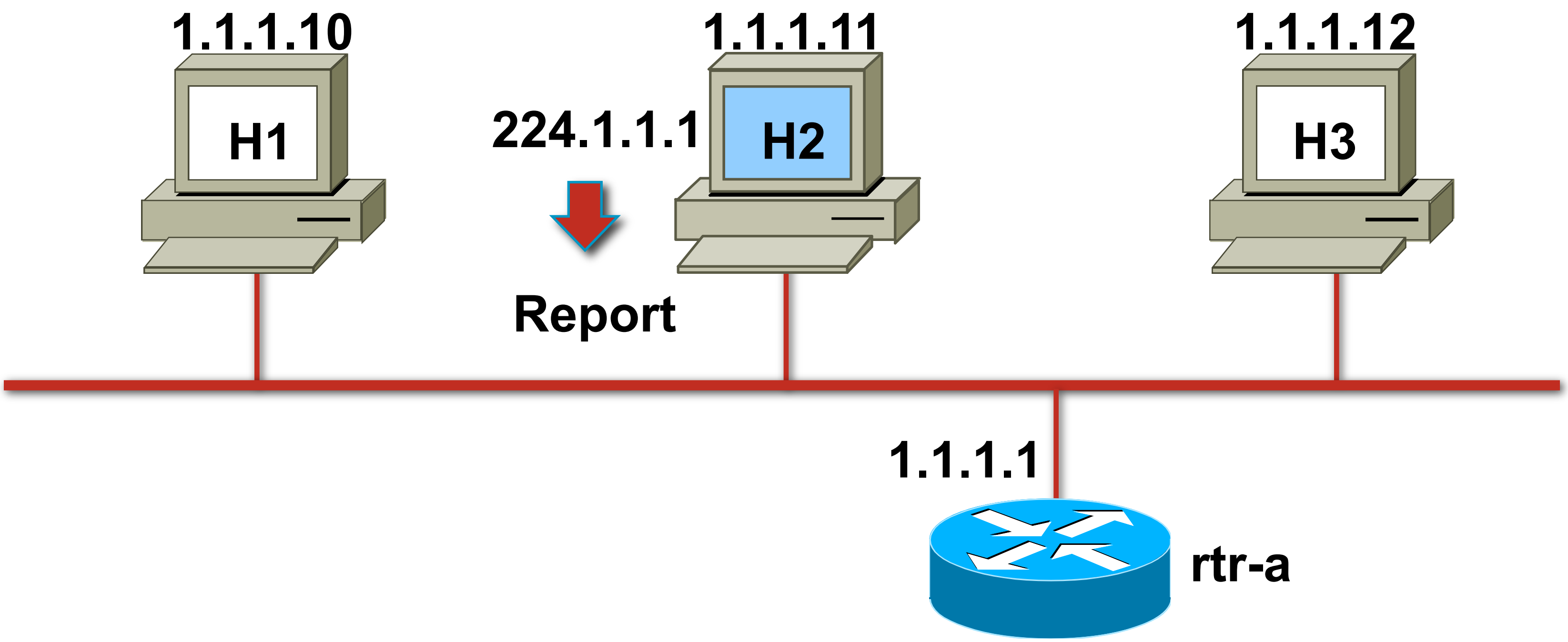
  On multi-access networks, an IGMP querier router is elected based on lowest IP address. Only the querier router sends queries.

- Query-Interval Response Time

  General queries specify "Max. Response Time" which inform hosts of the maximum time within which a host must respond to any query. (improves burstiness of the responses)

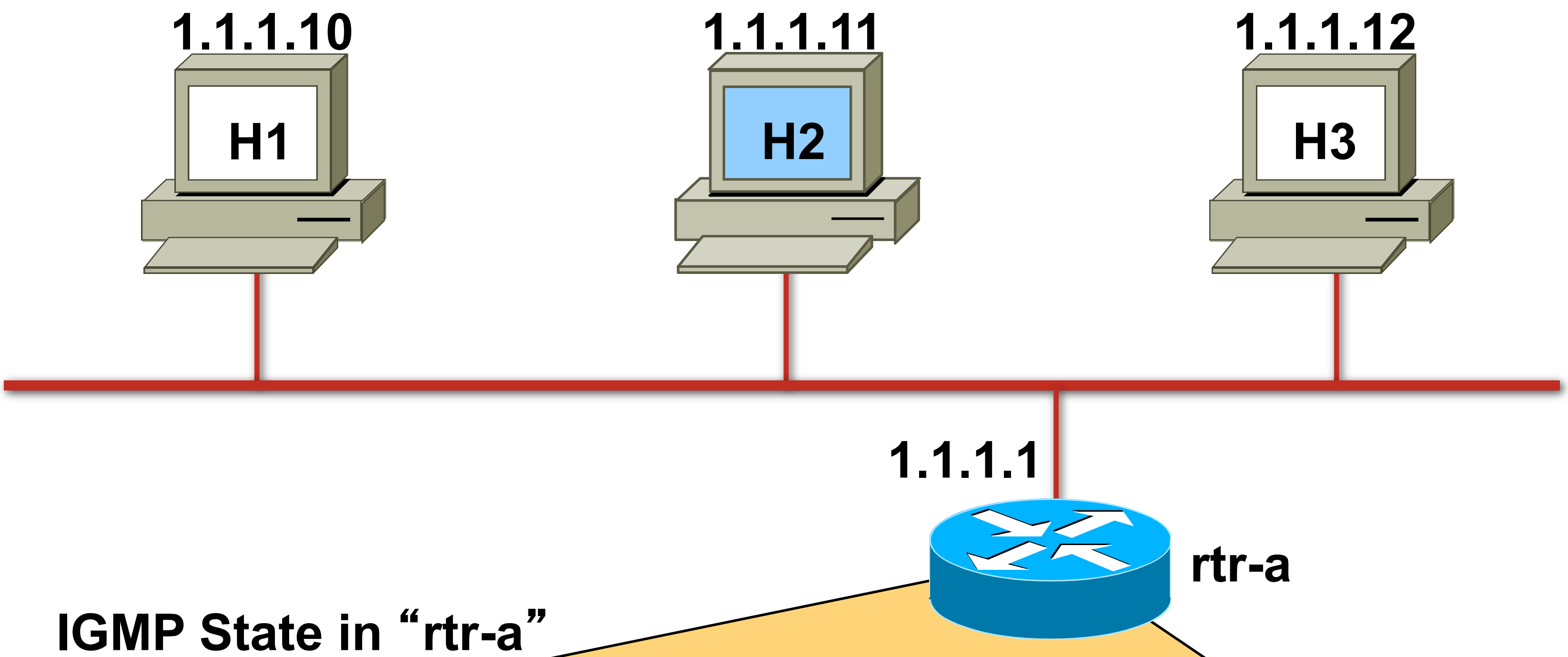
- Backward compatible with IGMPv1

IGMPv2—Joining a Group



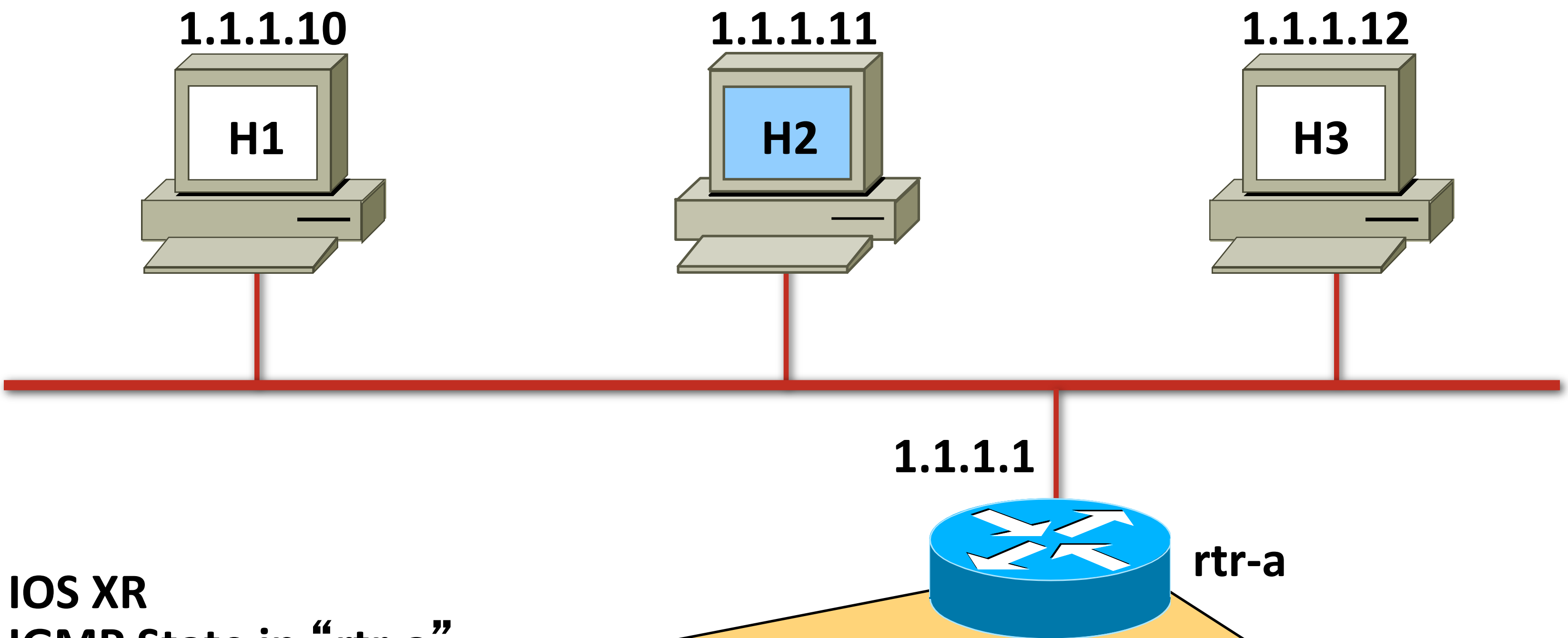
- Joining member sends report to 224.1.1.1 immediately upon joining (same as IGMPv1)

IGMPv2—Joining a Group




```
rtr-a>show ip igmp group
IGMP Connected Group Membership
Group Address      Interface      Uptime     Expires    Last Reporter
224.1.1.1          Ethernet0      6d17h      00:02:31   1.1.1.11
```

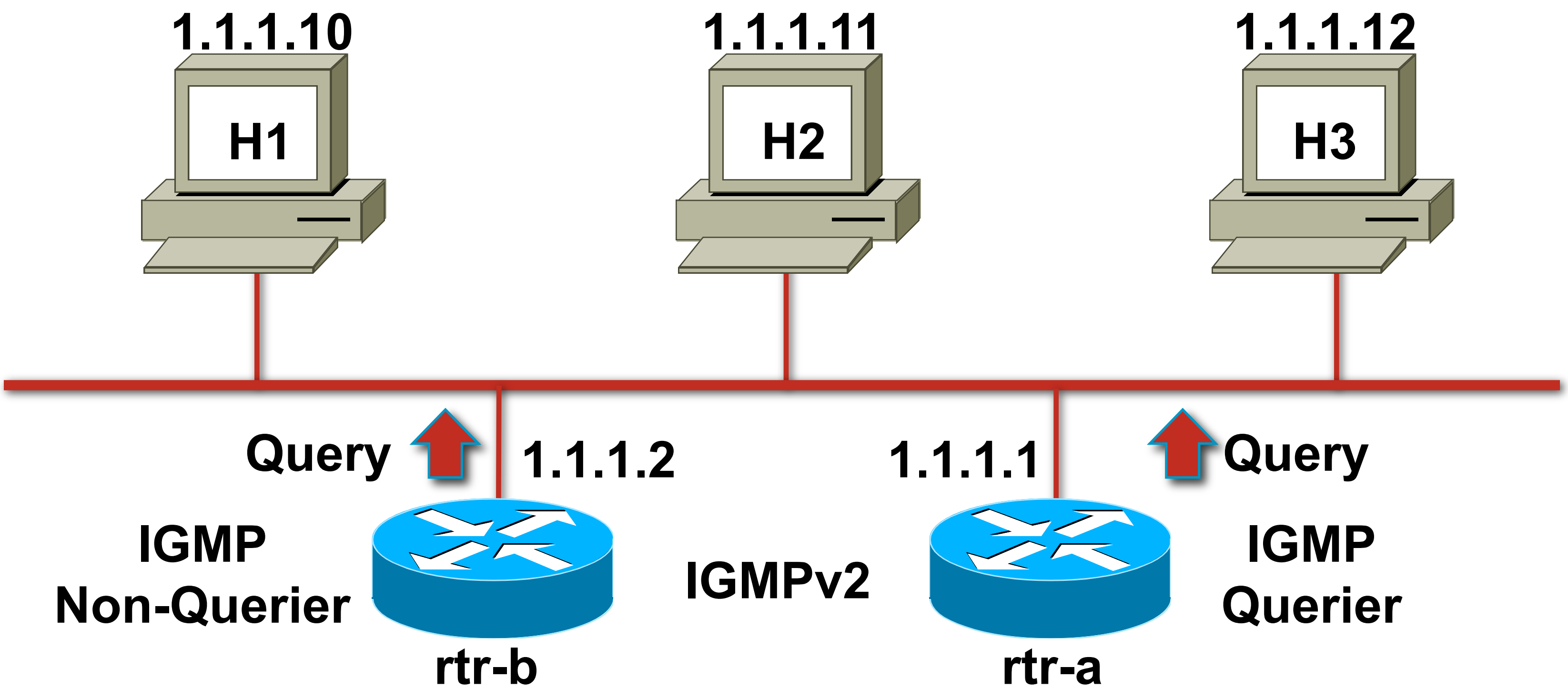
IGMPv2—Joining a Group



IOS XR
IGMP State in "rtr-a"

```
RP/0/RP0/CPU0:rtr-a#show igmp group
IGMP Connected Group Membership
Group Address    Interface                    Uptime     Expires    Last Reporter
224.1.1.1        Ethernet0                    00:00:35   00:01:34   1.1.1.11
```


IGMPv2—Querier Election



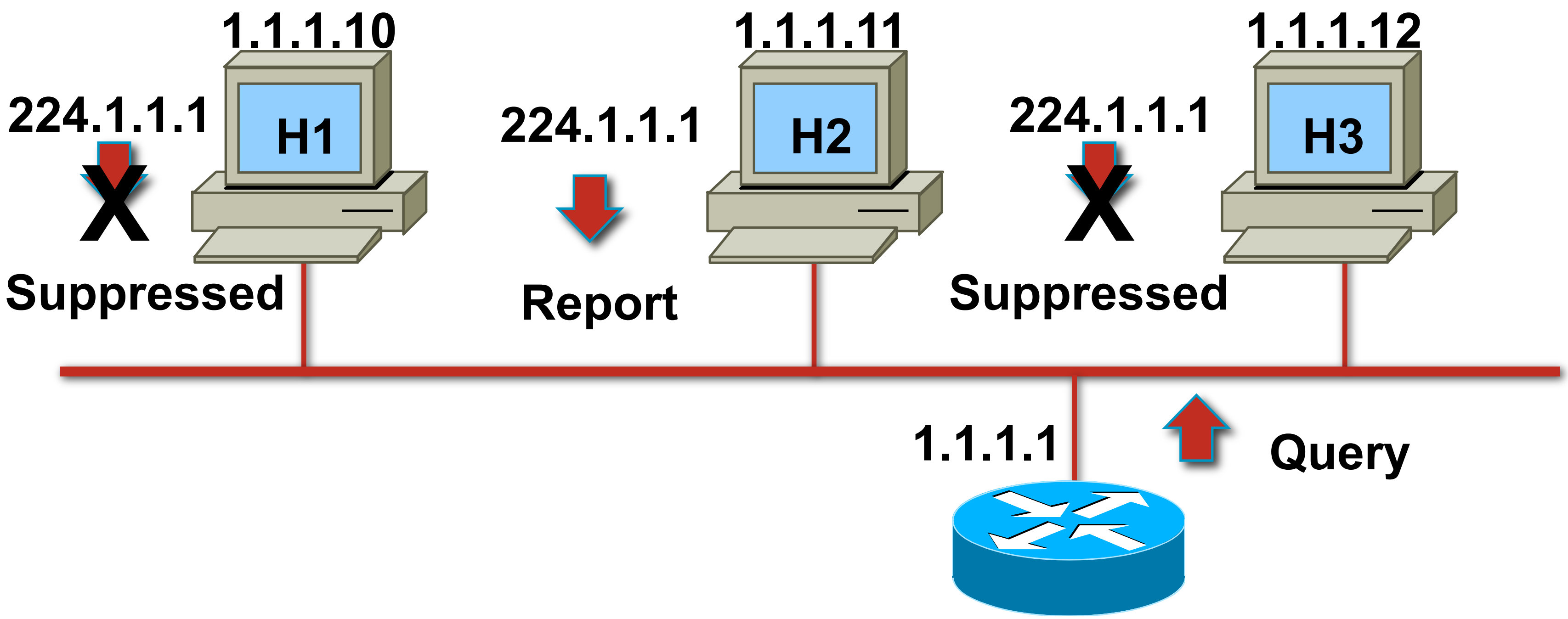
- Initially all routers send out a query
- Router with lowest IP address “elected” querier
- Other routers become “non-queriers”

IGMPv2—Querier Election

Determining Which Router Is the IGMP Querier

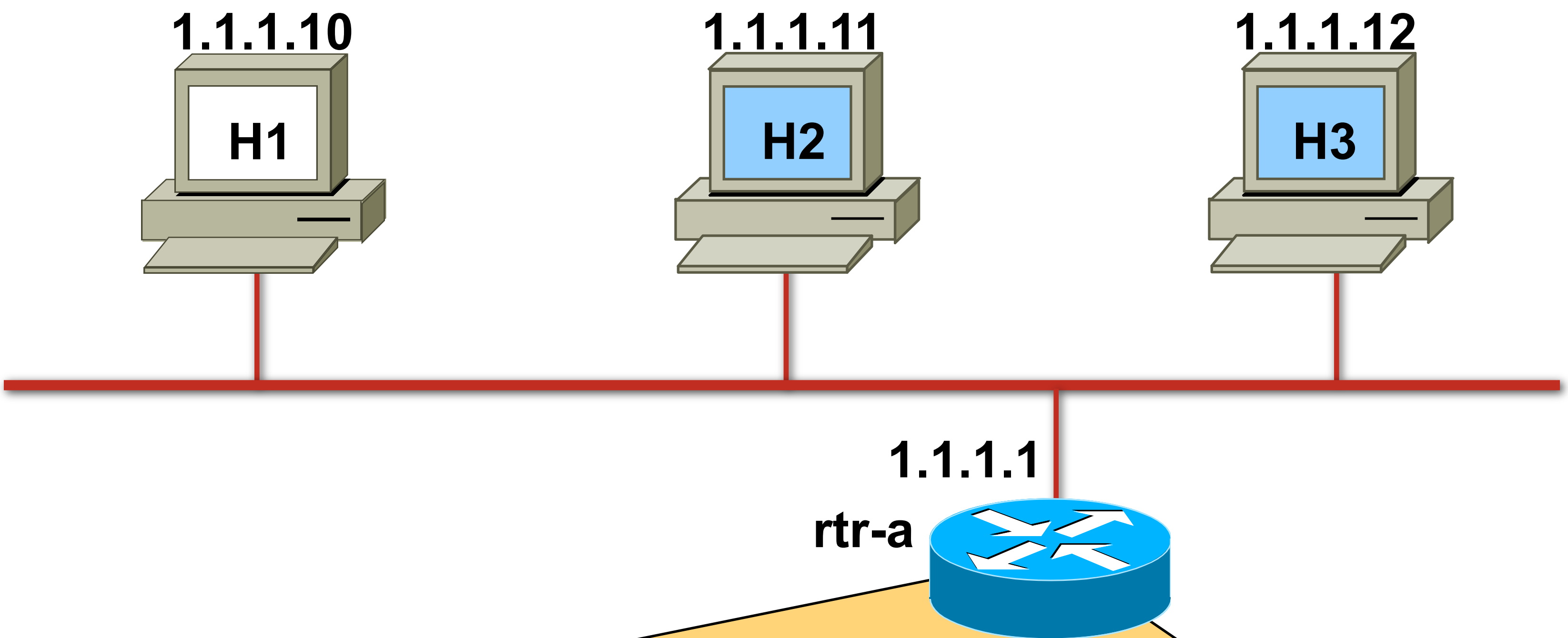
```
rtr-a>show ip igmp interface e0
Ethernet0 is up, line protocol is up
  Internet address is 1.1.1.1, subnet mask is 255.255.255.0
  IGMP is enabled on interface
  Current IGMP version is 2
  CGMP is disabled on interface
  IGMP query interval is 60 seconds
  IGMP querier timeout is 120 seconds
  IGMP max query response time is 10 seconds
  Inbound IGMP access group is not set
  Multicast routing is enabled on interface
  Multicast TTL threshold is 0
  Multicast designated router (DR) is 1.1.1.1 (this system)
  IGMP querying router is 1.1.1.1 (this system)
  Multicast groups joined: 224.0.1.40 224.2.127.254
```

IGMPv2—Maintaining a Group



- Router sends periodic queries
- One member per group per subnet reports
- Other members suppress reports

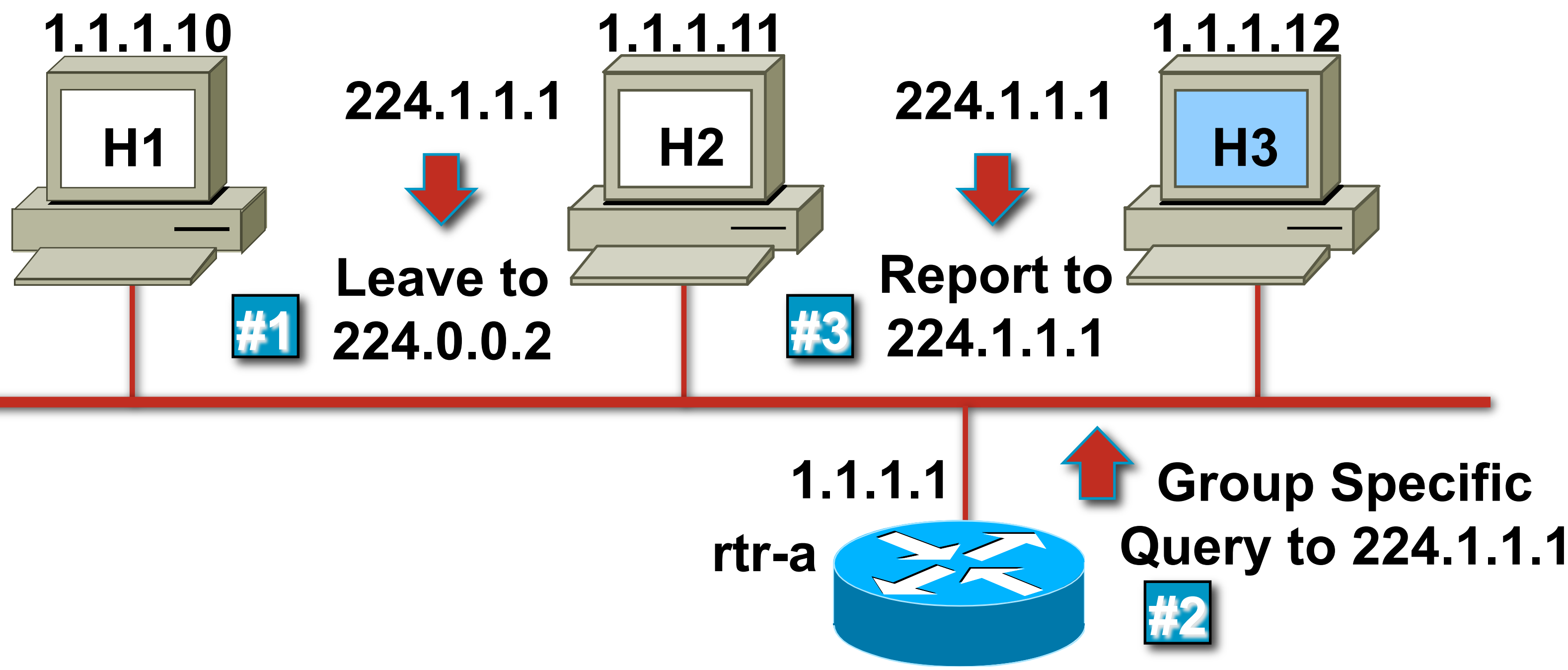
IGMPv2—Leaving a Group



IGMP State in "rtr-a" before Leave

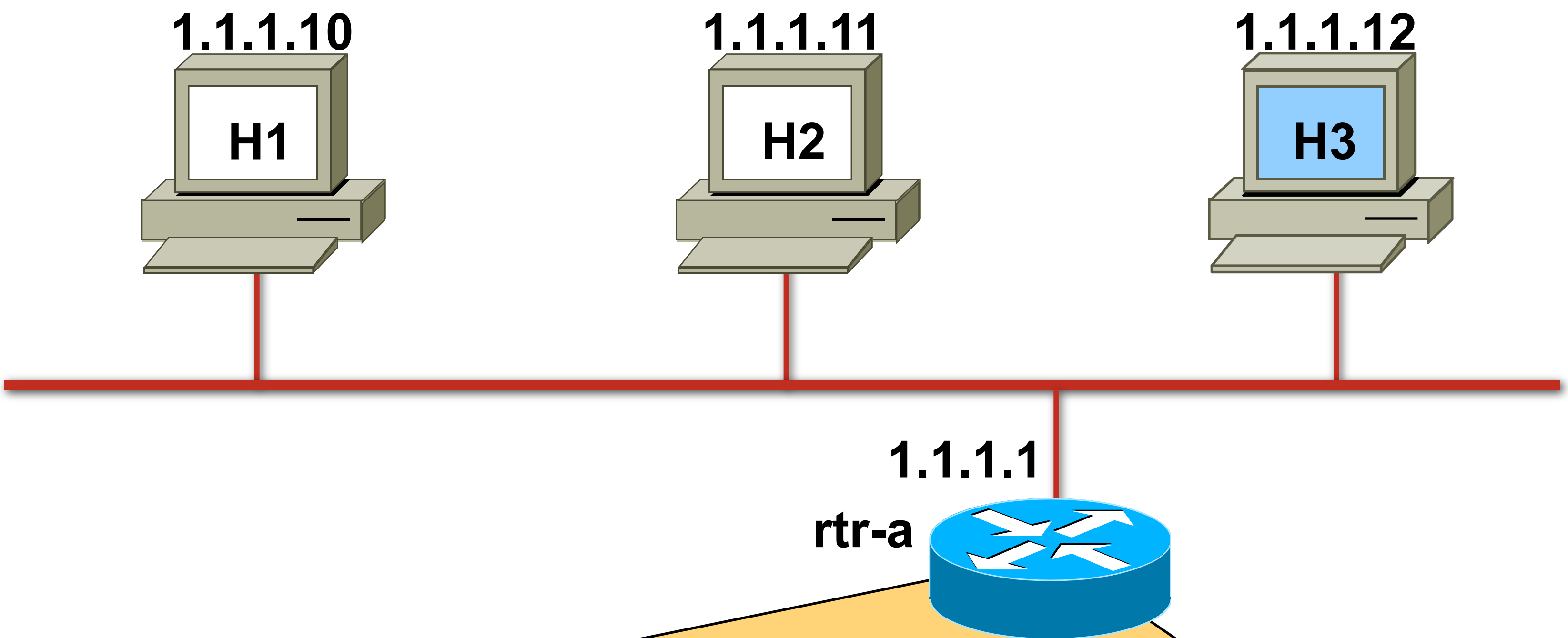
```
rtr-a>sh ip igmp group
IGMP Connected Group Membership
Group Address     Interface      Uptime     Expires    Last Reporter
224.1.1.1         Ethernet0      6d17h      00:02:31   1.1.1.11
```


IGMPv2—Leaving a Group



- H2 leaves group; sends Leave message
- Router sends Group specific query
- A remaining member host sends report
- Group remains active

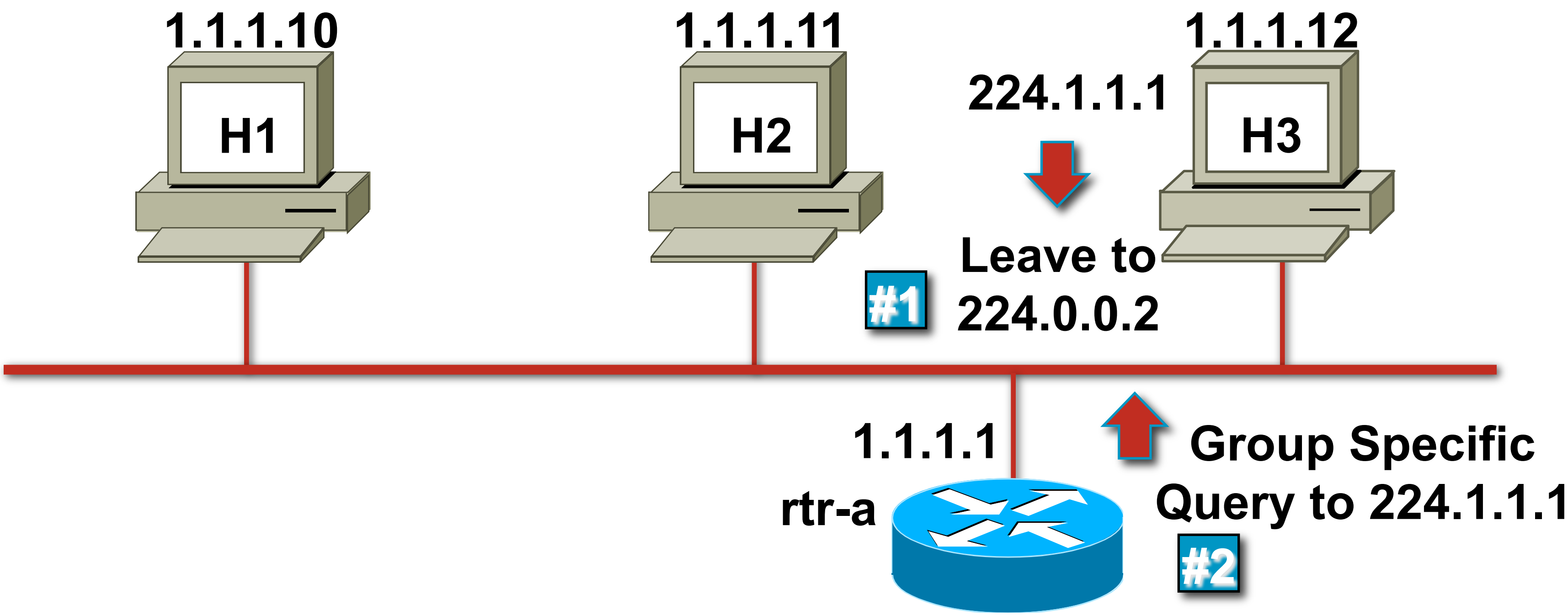
IGMPv2—Leaving a Group



IGMP State in "rtr-a" after H2 Leaves

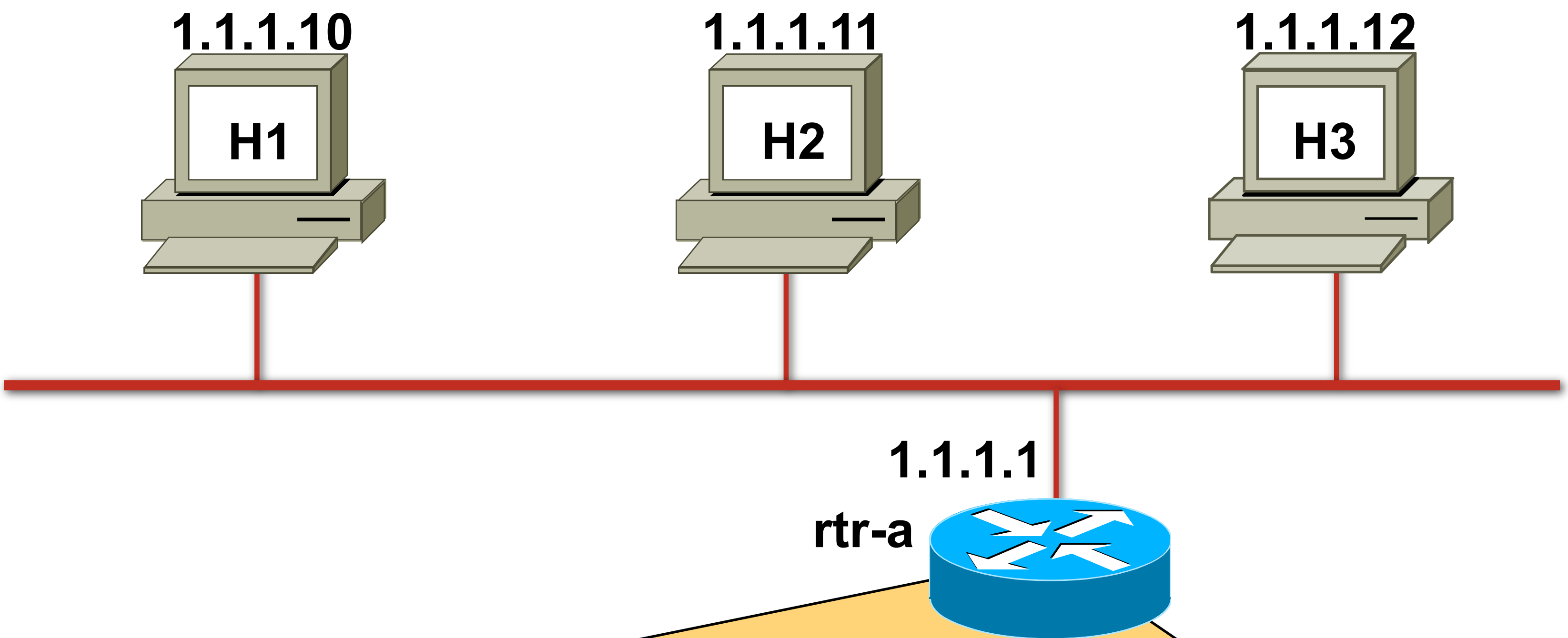
```
rtr-a>sh ip igmp group
IGMP Connected Group Membership
Group Address      Interface        Uptime     Expires    Last Reporter
224.1.1.1          Ethernet0        6d17h      00:01:47   1.1.1.12
```


IGMPv2—Leaving a Group



- Last host leaves group; sends Leave message
- Router sends Group specific query
- No report is received
- Group times out

IGMPv2—Leaving a Group



IGMP State in "rtr-a" after H3 Leaves

```
rtr-a>show ip igmp group
IGMP Connected Group Membership
Group Address     Interface       Uptime     Expires    Last Reporter
```

IGMPv3

IGMPv3

RFC 3376

- Adds Include/Exclude Source Lists

  Enables hosts to listen only to a specified subset of the hosts sending to the group

  Requires new ‘IPMulticastListen’ API

    New IGMPv3 stack required in the O/S

  Apps must be rewritten to use IGMPv3 Include/Exclude features

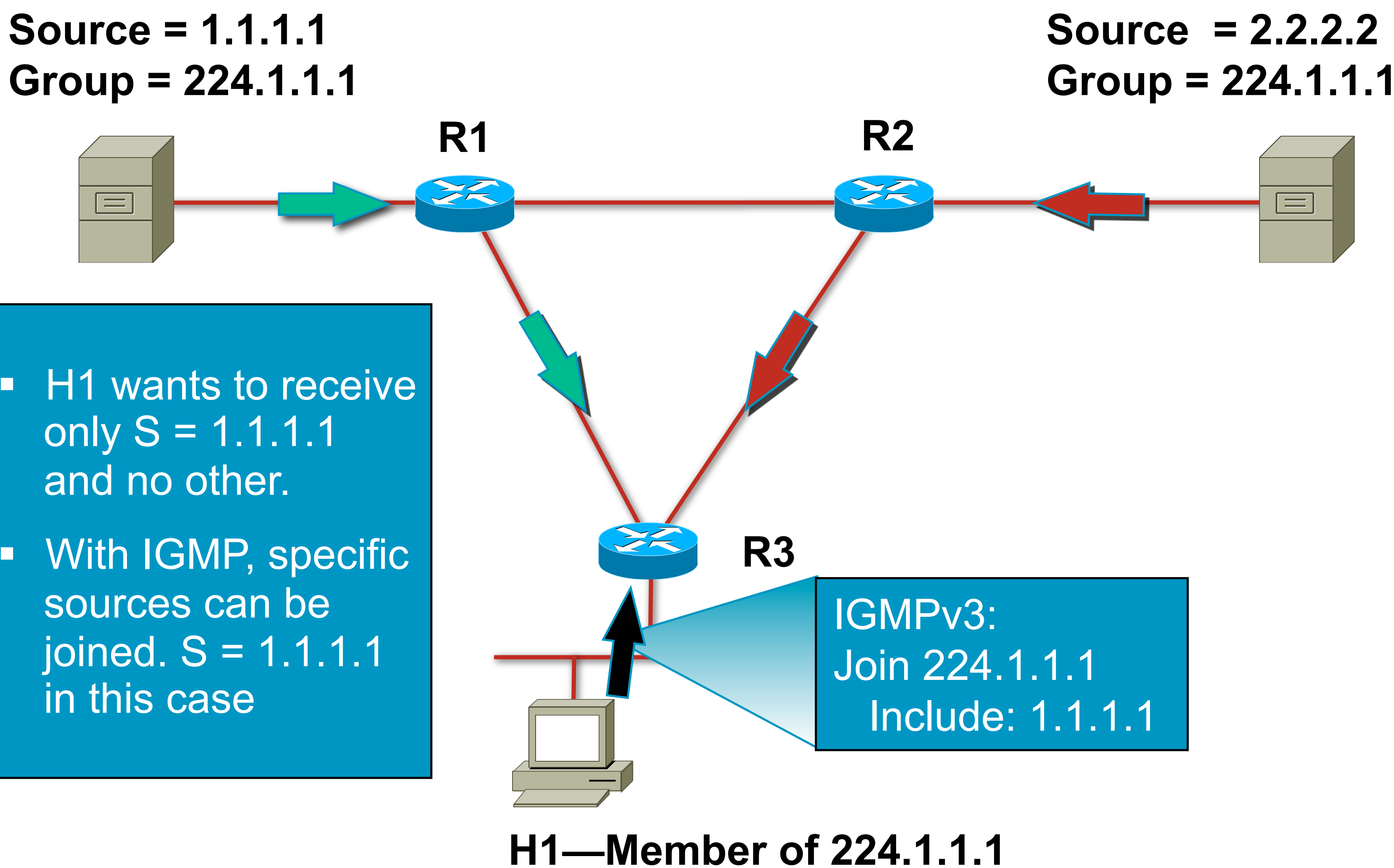
  Available in IOS 12.2, 12.1(3)T and 12.0(15)S

IGMPv3

RFC 3376

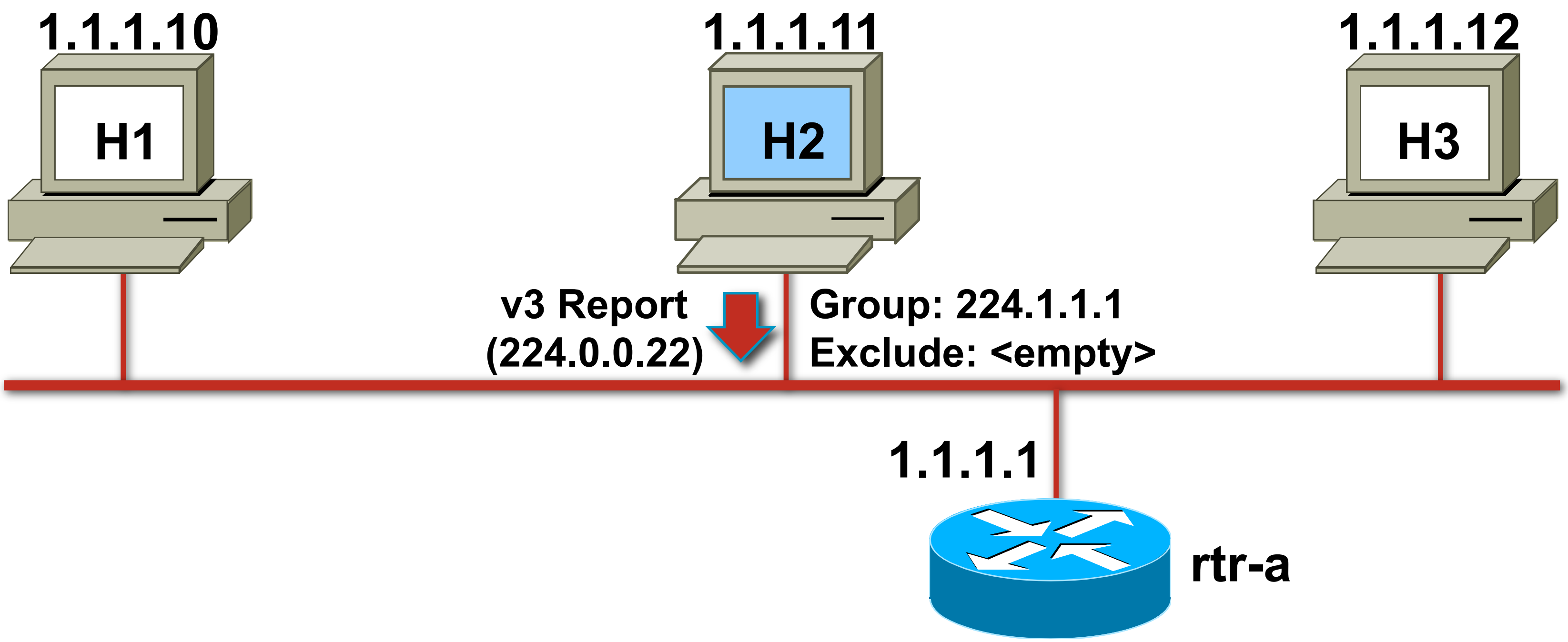
- New membership report address
  - 224.0.0.22 (All-IGMPv3-Routers)
    - All IGMPv3 hosts send reports to this address
      - Instead of the target group address as in IGMPv1/v2
    - All IGMPv3 routers listen to this address
    - Hosts do not listen or respond to this address
  - No report suppression
    - All hosts on wire respond to queries
    - Response Interval may be tuned over broad range
      - Useful when large numbers of hosts reside on subnet

IGMPv3 Example



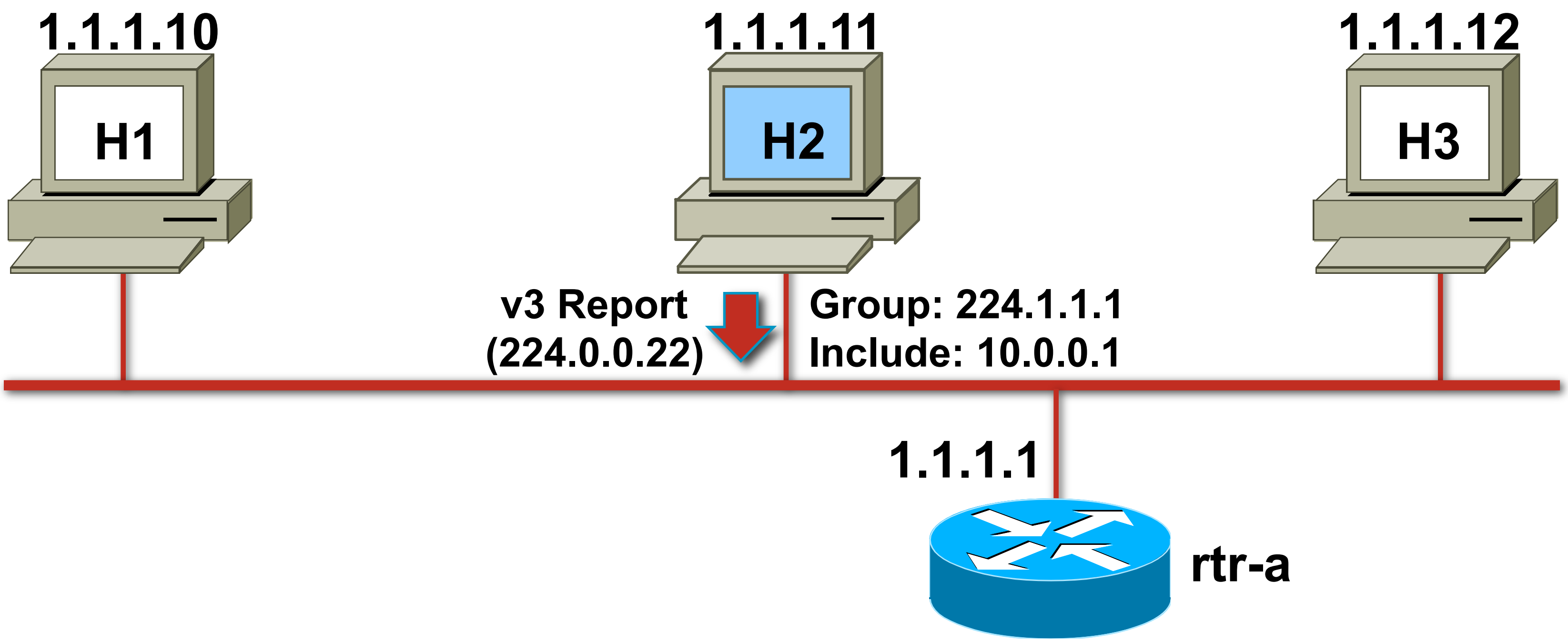
- H1 wants to receive only S = 1.1.1.1 and no other.
- With IGMP, specific sources can be joined. S = 1.1.1.1 in this case

IGMPv3—Joining a Group



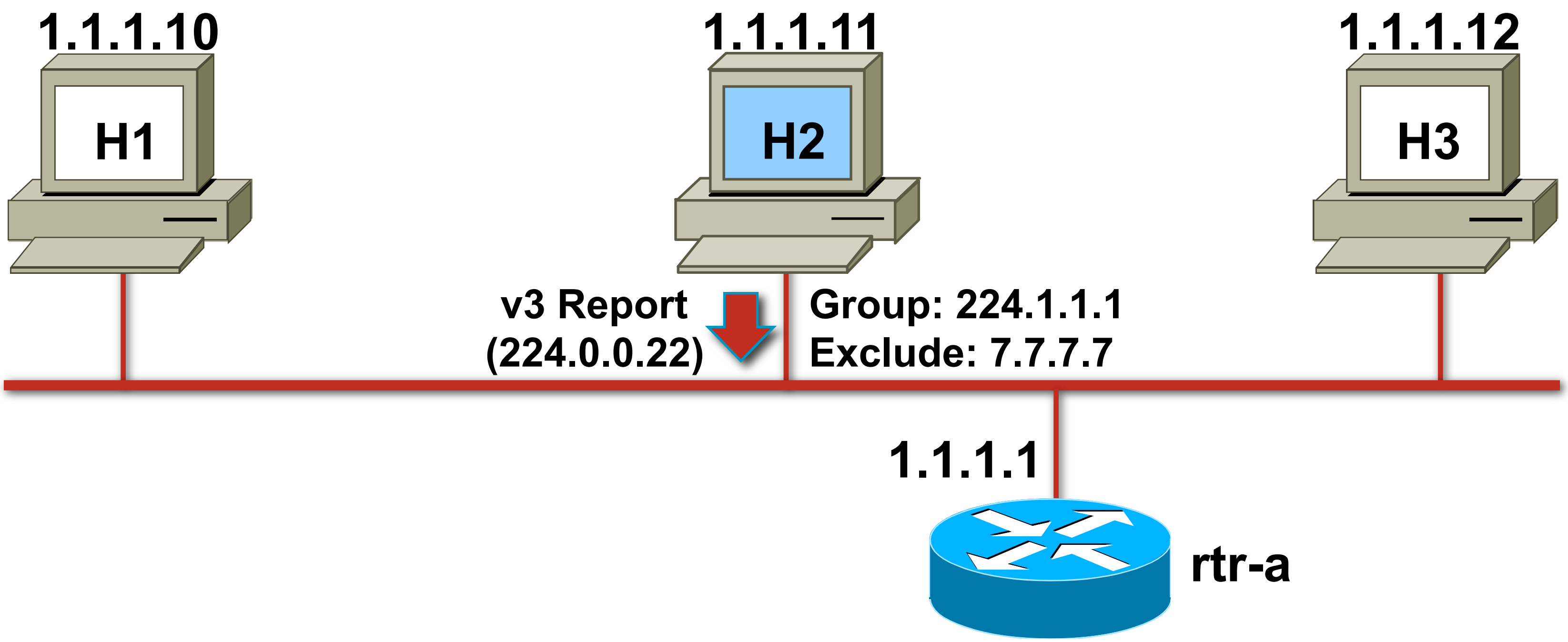
- Joining member sends IGMPv3 Report to 224.0.0.22 immediately upon joining

IGMPv3—Joining Specific Source(s)



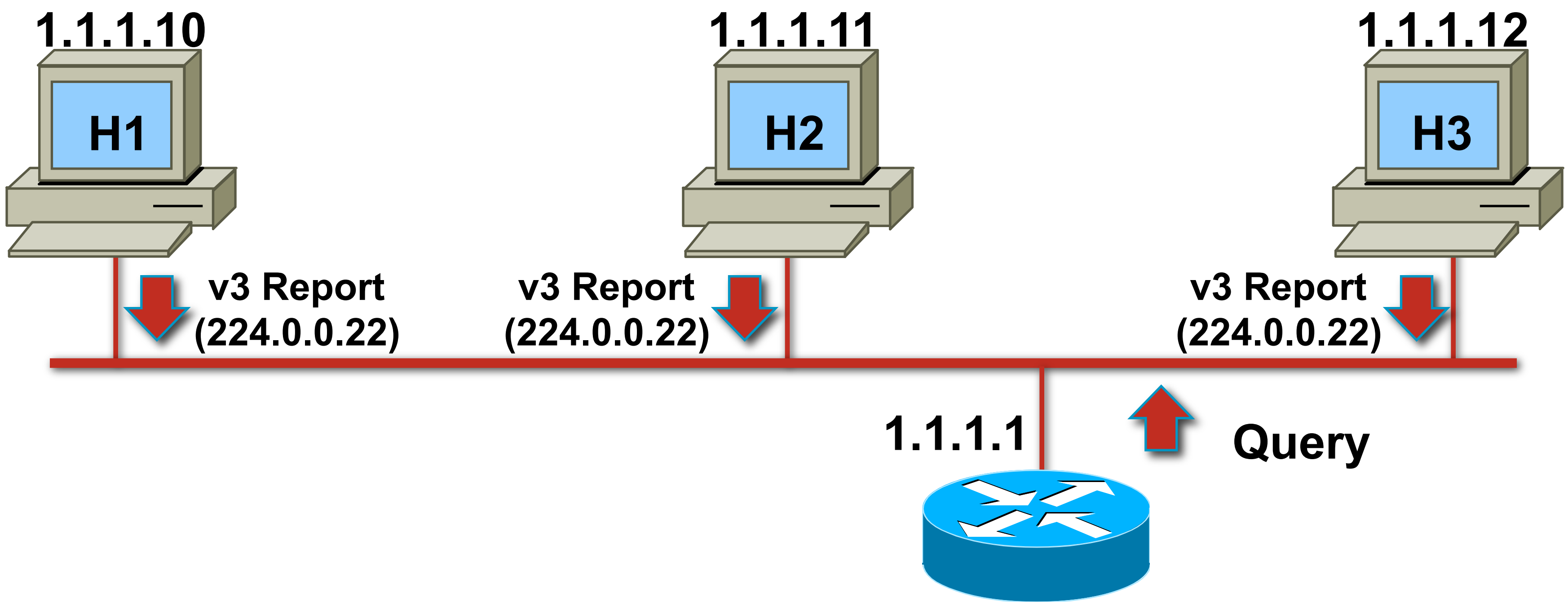
- IGMPv3 report contains desired source(s) in the Include list
- Only "Included" source(s) are joined

IGMPv3—Excluding Specific Source(s)



- IGMPv3 report contains undesired source(s) in the Exclude list
- All sources except "Excluded" source(s) are joined

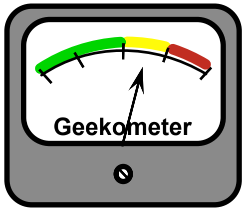
IGMPv3—Maintaining State



- Router sends periodic queries
- All IGMPv3 members respond
    - Reports contain multiple Group state records

PIM Sparse Mode

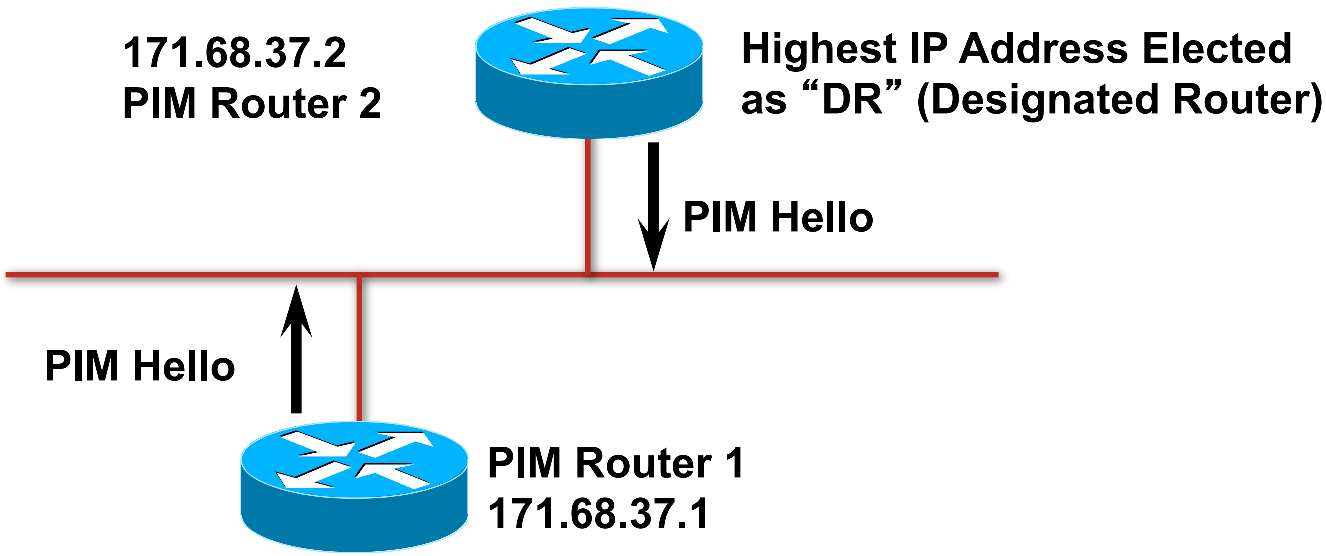
Module Agenda



- PIM Neighbor Discovery
- PIM State
- PIM SM Joining
- PIM SM Registering
- PIM SM SPT-Switchover

PIM Neighbor Discovery

PIM Neighbor Discovery

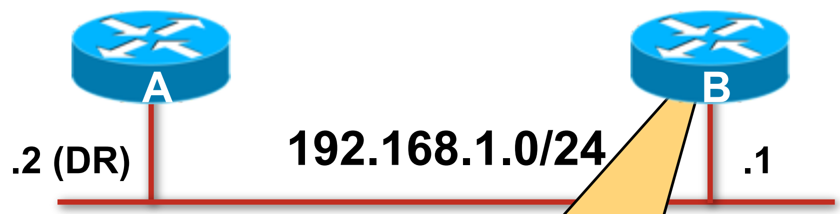


- PIMv2 Hellos are periodically multicast to the "All-PIM-Routers" (224.0.0.13) group address (default = 30 seconds)
- If the "DR" times-out, a new "DR" is elected
- The "DR" is responsible for sending all Joins and Register messages for any receivers or senders on the network

PIM Neighbor Discovery—IOS

```
wan-gw8>show ip pim neighbor
PIM Neighbor Table
Neighbor          Interface           Uptime/Expires       Ver    Mode
Address                                                           Prio/Mode
171.68.0.70       FastEthernet0/0     2w1d/00:01:24        v2     1 / B S
171.68.0.91       FastEthernet0/0     2w6d/00:01:01        v2     1 / B S
171.68.0.82       FastEthernet0/0     7w0d/00:01:14        v2     5 / DR B S
171.68.0.86       FastEthernet0/0     7w0d/00:01:13        v2     1 / B S
171.68.0.80       FastEthernet0/0     7w0d/00:01:02        v2     1 / B S
171.68.28.70      Serial2.31          22:47:11/00:01:16    v2     1 / B S
171.68.28.50      Serial2.33          22:47:22/00:01:08    v2     1 / B S
171.68.27.74      Serial2.36          22:47:07/00:01:21    v2     N /
171.68.28.170     Serial0.70          1d4h/00:01:06        v2     N /
171.68.27.2       Serial1.51          1w4d/00:01:25        v2     1 / B S
171.68.28.110     Serial3.56          1d4h/00:01:20        v2     1 / B S
171.68.28.58      Serial3.102         12:53:25/00:01:03    v2     1 / B S
```

DR Failover




```
Rtr-B>show ip pim neighbor
PIM Neighbor Table
Neighbor Address   Interface   Uptime  Expires   Mode
192.168.1.2        Ethernet0   4d22h   00:01:18  Sparse-Dense (DR)
```

- Depends on neighbor expiration time
- Expiration time sent in PIM query messages

  Expiration time = 3 x <query-interval>

  Default <query-interval> = 30 seconds

  DR failover ~ 90 seconds (worst case) by default

Tuning DR Failover

- Tune PIM query interval
  - Use interface configuration command
    - `ip pim query-interval <period> [msec]`
      - Default <period> = seconds
      - "msec" keyword available beginning with 12.1(11b)E
  - Permits DR failover to be adjusted
    - Sub-second DR failover possible
    - Smaller intervals increase PIM query traffic
      - Increase is usually insignificant

PIM State

PIM State

- Describes the "state" of the multicast distribution trees as understood by the router at this point in the network
- Represented by entries in the multicast routing (mroute) table
  - Used to make multicast traffic forwarding decisions
  - Composed of (*, G) and (S, G) entries
  - Each entry contains RPF information
    - Incoming (i.e. RPF) interface
    - RPF Neighbor (upstream)
  - Each entry contains an Outgoing Interface List (OIL)
    - OIL may be NULL

PIM-SM State Example—IOS

```
sj-mbone> show ip mroute
Flags: D - Dense, S - Sparse, B - Bidir Group, s - SSM Group, C - Connected,
       L - Local, P - Pruned, R - RP-bit set, F - Register flag,
       T - SPT-bit set, J - Join SPT, M - MSDP created entry,
       X - Proxy Join Timer Running, A - Candidate for MSDP Advertisement,
       U - URD, I - Received Source Specific Host Report
Outgoing interface flags: H - Hardware switched
Timers: Uptime/Expires
Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.1.1.1), 2w1d/00:00:00, RP 172.16.25.1, flags: SJC
  Incoming interface: Serial0/1, RPF nbr 172.16.4.1
  Outgoing interface list:
    Ethernet0/1, Forward/Sparse-Dense, 2w1d/00:01:40
    Serial0/0, Forward/Sparse-Dense, 00:4:52/00:02:08

(172.16.8.2, 224.1.1.1), 00:00:10/00:02:59, flags: CJT
  Incoming interface: Serial0/1, RPF nbr 172.16.4.1
  Outgoing interface list:
    Ethernet0/1, Forward/Sparse-Dense, 00:00:10/00:02:49
    Serial0/0, Forward/Sparse-Dense, 00:4:52/00:02:08
```


PIM-SM (*,G) State Rules

- (*,G) creation
  - Receipt of a (*,G) Join or IGMP Report
  - Automatically if (S,G) must be created
- (*,G) reflects default group forwarding
  - IIF = RPF interface toward RP
  - OIL = interfaces
    - That received a (*,G) Join or
    - With directly connected members or
    - Manually configured
- (*,G) deletion
  - When OIL = NULL and
  - No child (S,G) state exists

PIM-SM (S,G) State Rules

- (S,G) creation

  By receipt of (S,G) Join or Prune or

  By "Register" process

  Parent (*,G) created (if doesn't exist)

- (S,G) reflects forwarding of "S" to "G"

  IIF = RPF Interface normally toward source

  RPF toward RP if "RP-bit" set

  OIL = Initially, copy of (*,G) OIL minus IIF

- (S,G) deletion

  By normal (S,G) entry timeout

PIM-SM OIL Rules

- Interfaces in OIL added
  - By receipt of Join message
    - Interfaces added to (*,G) are added to all (S,G)s
- Interfaces in OIL removed
  - By receipt of Prune message
    - Interfaces removed from (*,G) are removed from all (S,G)s
  - Interface expire timer counts down to zero
    - Timer reset (to 3 min.) by receipt of periodic Join

      or

      By IGMP membership report

PIM-SM Triggered Join/Prune Rules

- Triggering Join/Prune Messages
  - (*,G) Joins are triggered when:
    - The (*,G) OIL transitions from Null to non-Null
  - (*,G) Prunes are triggered when:
    - The (*,G) OIL transitions from non-Null to Null
  - (S,G) Joins are triggered when:
    - The (S,G) OIL transitions from Null to non-Null
  - (S,G) Prunes are triggered when:
    - The (S,G) OIL transitions from non-Null to Null
  - (S,G)RP-bit Prunes are triggered when:
    - The (S,G) RPF info != the (*,G) RPF info

PIM-SM State Flags

- S = Sparse
- C = Directly Connected Host
- L = Local (Router is member)
- P = Pruned (All intfcs in OIL = Prune)
- T = Forwarding via SPT

  Indicates at least one packet was forwarded

PIM-SM State Flags (Cont.)

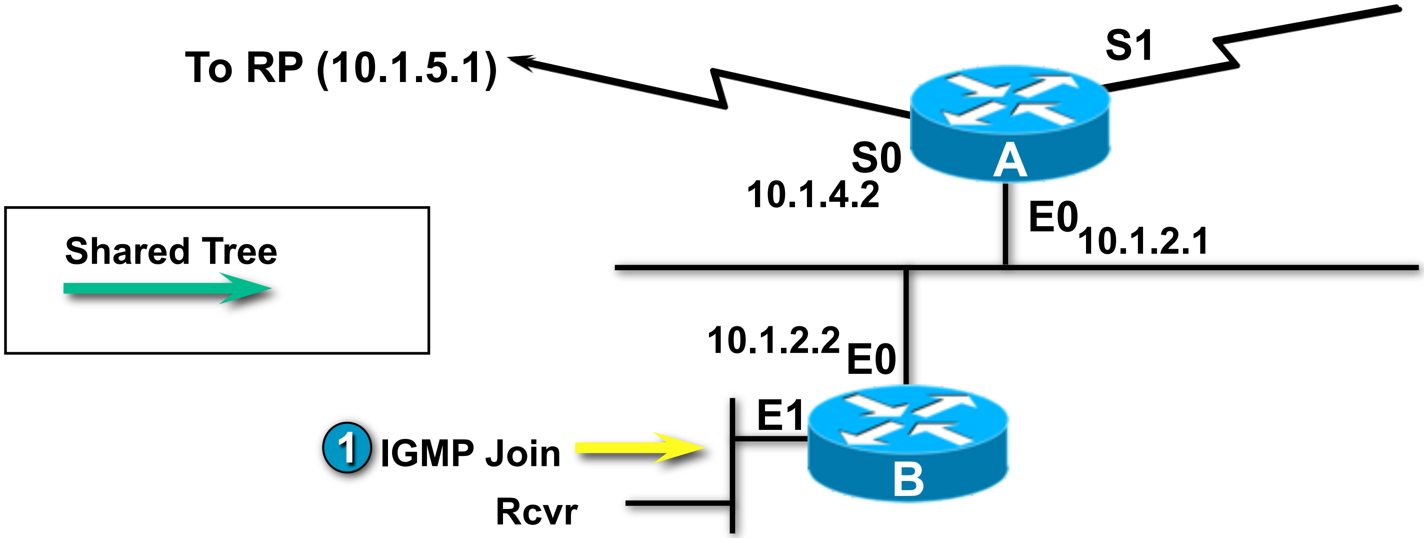
- J = Join SPT
  - In (*, G) entry
    - Indicates SPT-Threshold is being exceeded
    - Next (S,G) received will trigger join of SPT
  - In (S, G) entry
    - Indicates SPT joined due to SPT-Threshold
    - If rate < SPT-Threshold, switch back to Shared Tree
- F = Register/First-Hop
  - In (S,G) entry
    - "S" is a directly connected source
    - Triggers the Register Process
  - In (*, G) entry
    - Set when "F" set in at least one child (S,G)

PIM-SM State Flags (Cont.)

- R = RP bit
  - (S, G) entries only
  - Set by (S,G)RP-bit Prune
  - Indicates info is applicable to Shared Tree
  - Used to prune (S,G) traffic from Shared Tree
    - Initiated by Last-hop router after switch to SPT
  - Modifies (S,G) forwarding behavior
    - IIF = RPF toward RP (I.e. up the Shared Tree)
    - OIL = Pruned accordingly

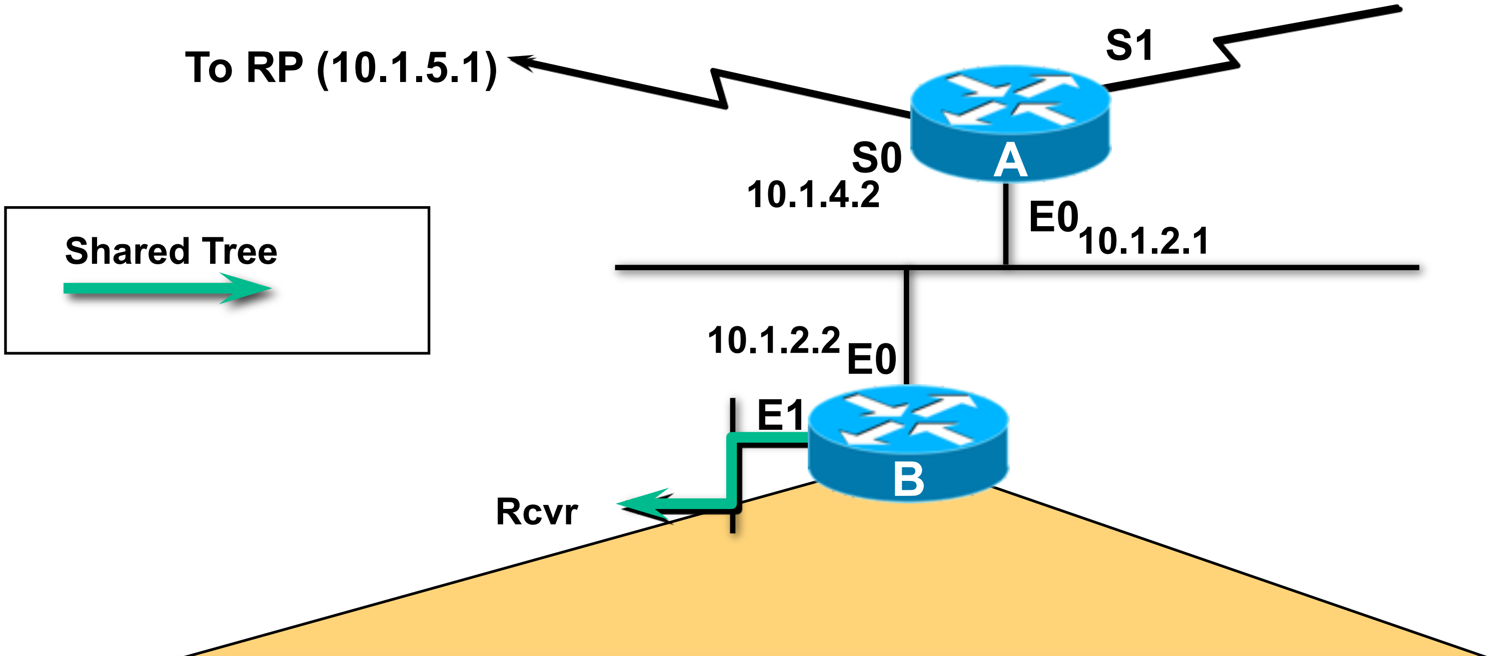
PIM SM Joining

PIM SM Joining



1. Rcvr wishes to receive group G traffic. Sends IGMP Join for G.

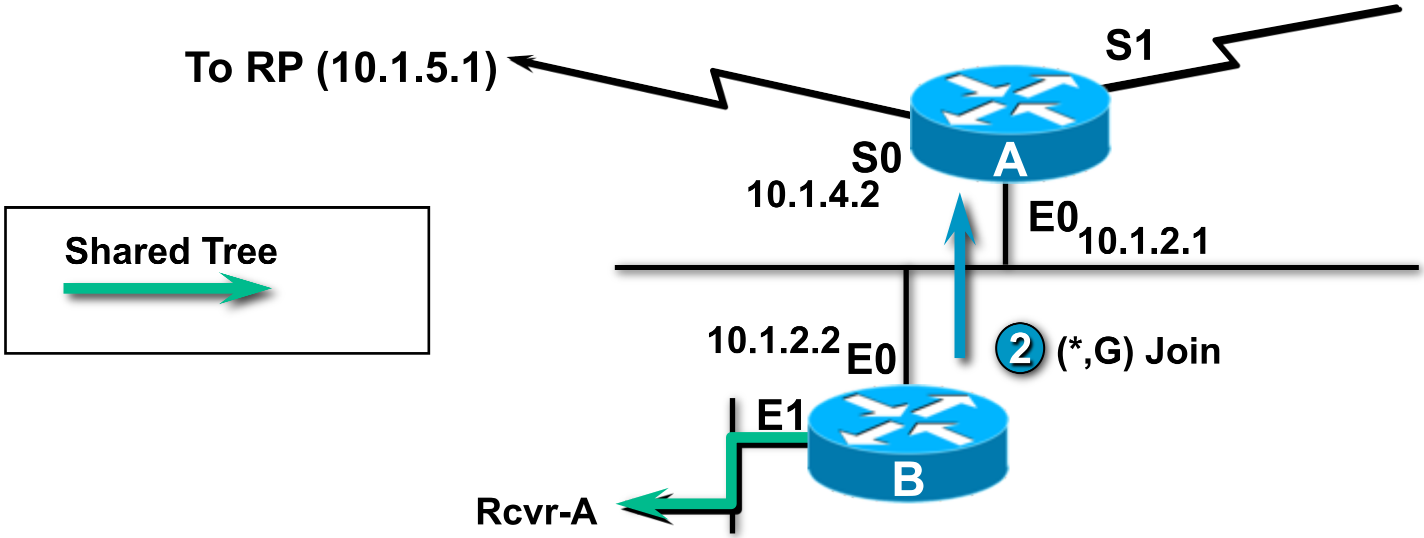
PIM SM Joining




```
(*, 224.1.1.1), 00:00:05/00:00:00, RP 10.1.5.1, flags: SC
  Incoming interface: Ethernet0, RPF nbr 10.1.2.1
  Outgoing interface list:
    Ethernet1, Forward/Sparse-Dense, 00:00:05/00:02:54
```

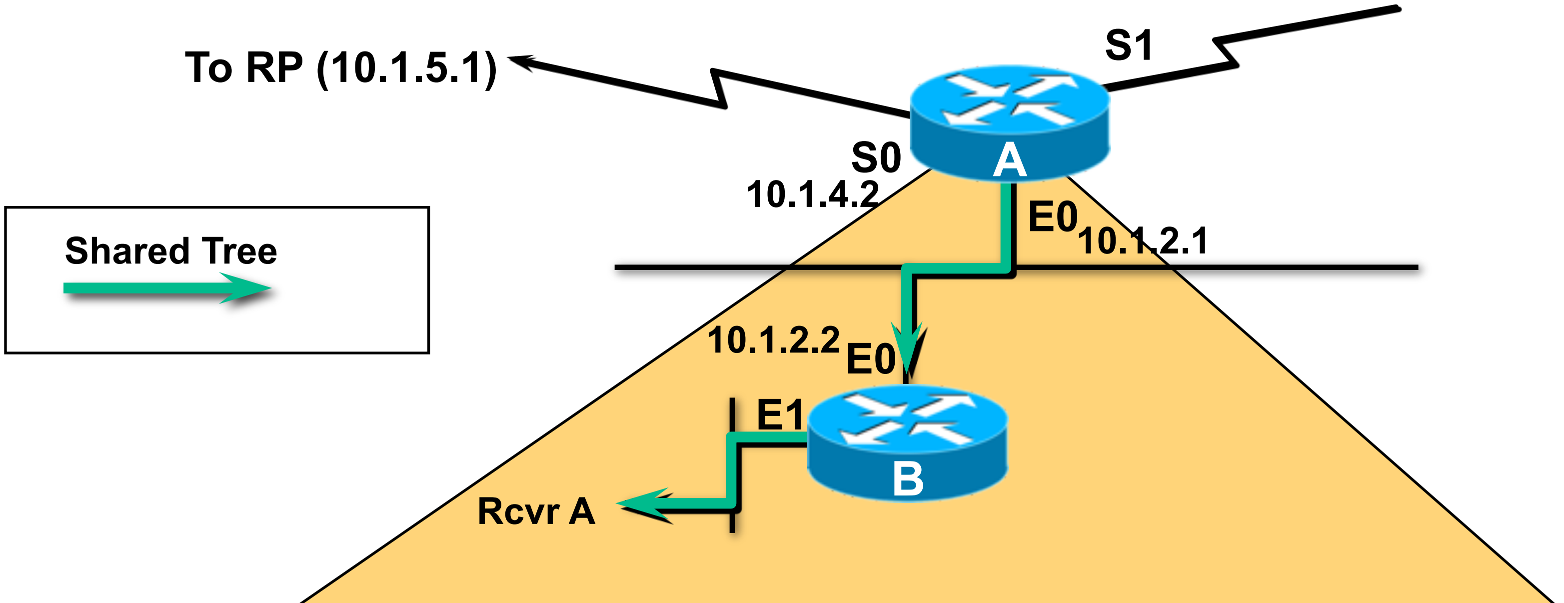
B Creates (*, 224.1.1.1) State

PIM SM Joining



1. Rcvr wishes to receive group G traffic. Sends IGMP Join for G.
2. B sends (*,G) Join towards RP.

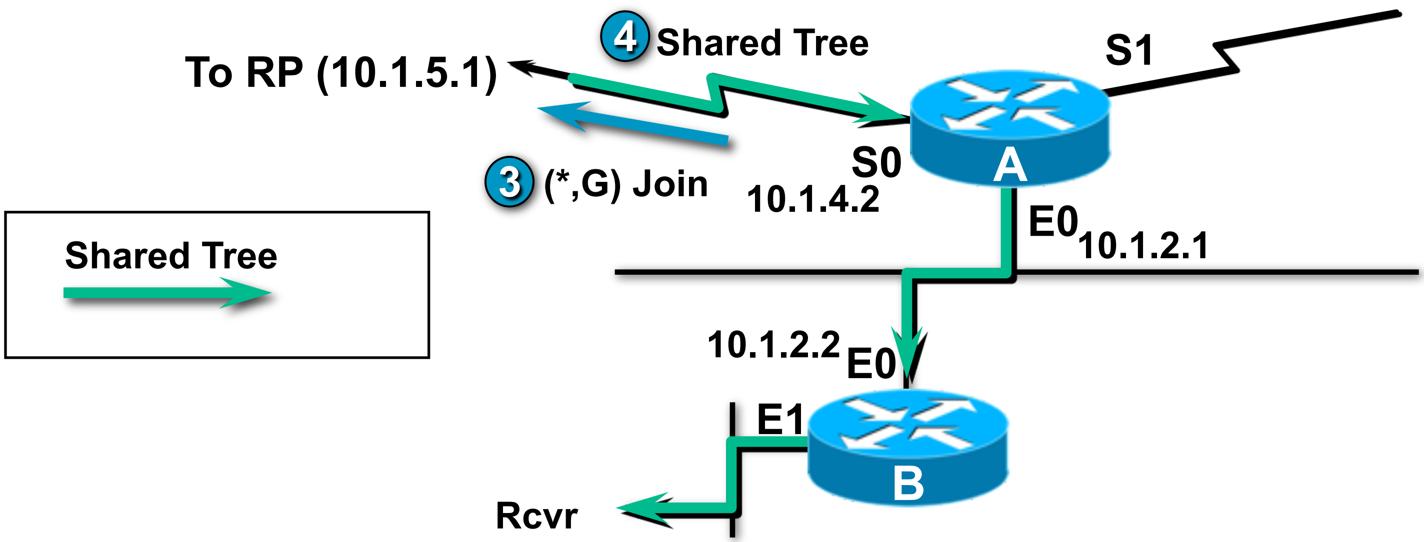
PIM SM Joining




```
(*, 224.1.1.1), 00:00:05/00:03:24, RP 10.1.5.1, flags: S
  Incoming interface: Serial0, RPF nbr 10.1.4.1
  Outgoing interface list:
    Ethernet0, Forward/Sparse-Dense, 00:00:05/00:02:54
```

A Creates (*, 224.1.1.1) State

PIM SM Joining



1. Rcvr wishes to receive group G traffic. Sends IGMP Join for G.
2. B sends (*,G) Join towards RP.
3. A sends (*,G) Join towards RP.
4. Shared tree is built all the way back to the RP.

PIM SM Registering

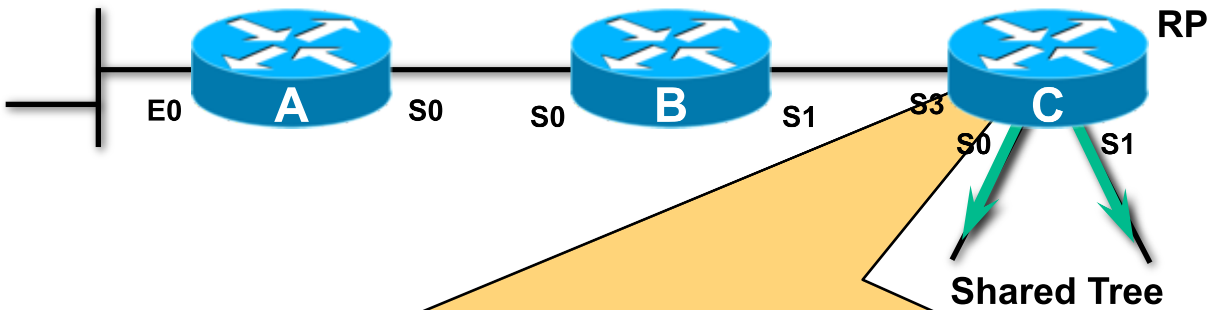
PIM SM Register Scenarios

- Receivers Join Group First
- Source Registers First
- Receivers along the SPT

PIM SM Registering: Receiver Joins First

PIM SM Registering

Receiver Joins Group First

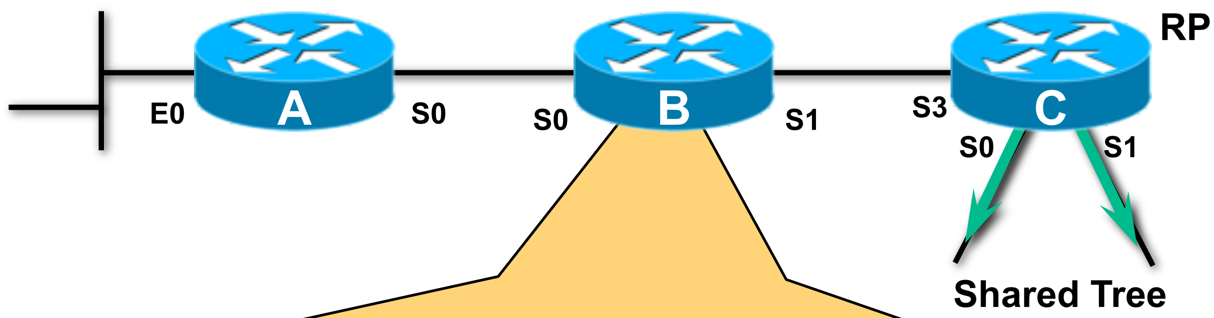



```
(*, 224.1.1.1), 00:03:14/00:02:59, RP 171.68.28.140, flags:S
  Incoming interface: Null, RPF nbr 0.0.0.0,
  Outgoing interface list:
    Serial0, Forward/Sparse-Dense, 00:03:14/00:03:15
    Serial1, Forward/Sparse-Dense, 00:03:14/00:03:15
```

State in “RP” Before Any Source Registers
(With Receivers on Shared Tree)

PIM SM Registering

Receiver Joins Group First



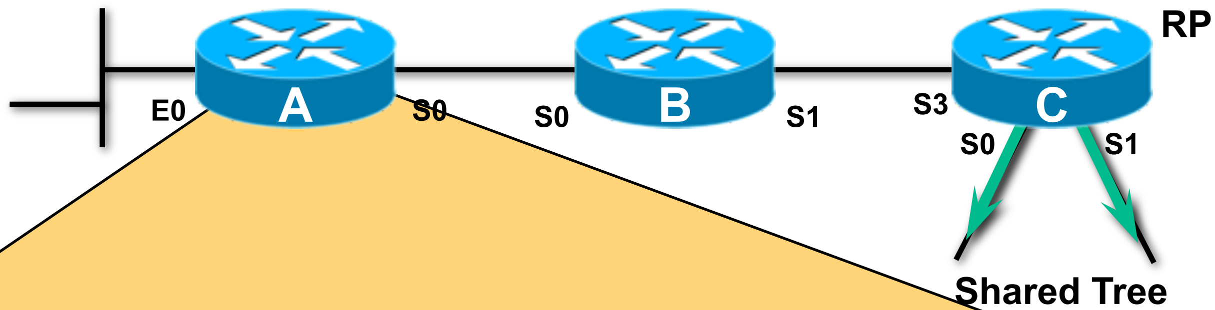

```
rtr-b>sh ip mroute 224.1.1.1

No such group
```

State in B Before Any Source Registers

(With Receivers on Shared Tree)

PIM SM Registering

Receiver Joins Group First

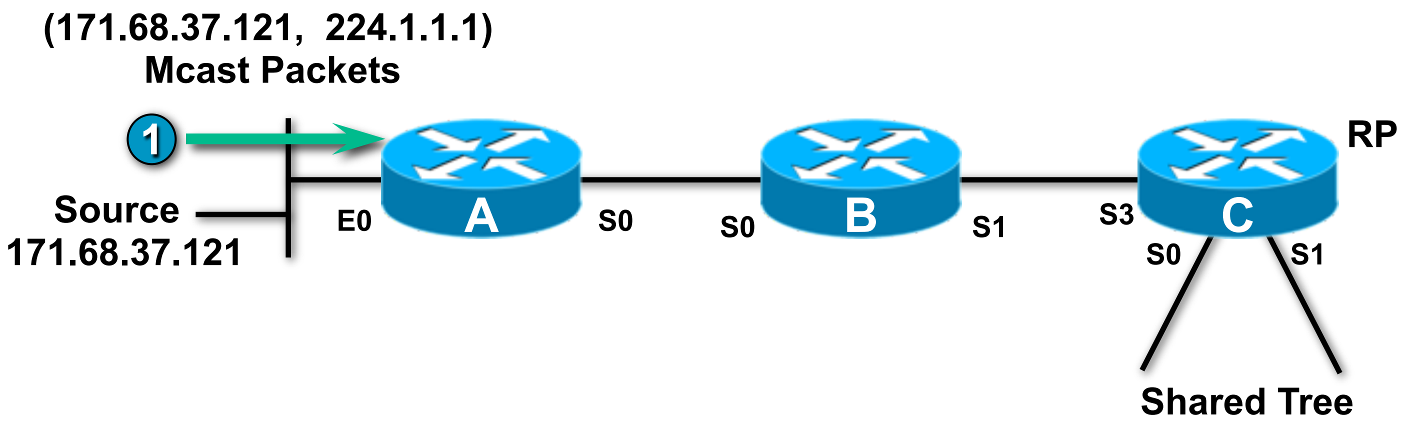



```
rtr-a>sh ip mroute 224.1.1.1

No such group.
```

State in A Before Any Source Registers
(With Receivers on Shared Tree)

PIM SM Registering

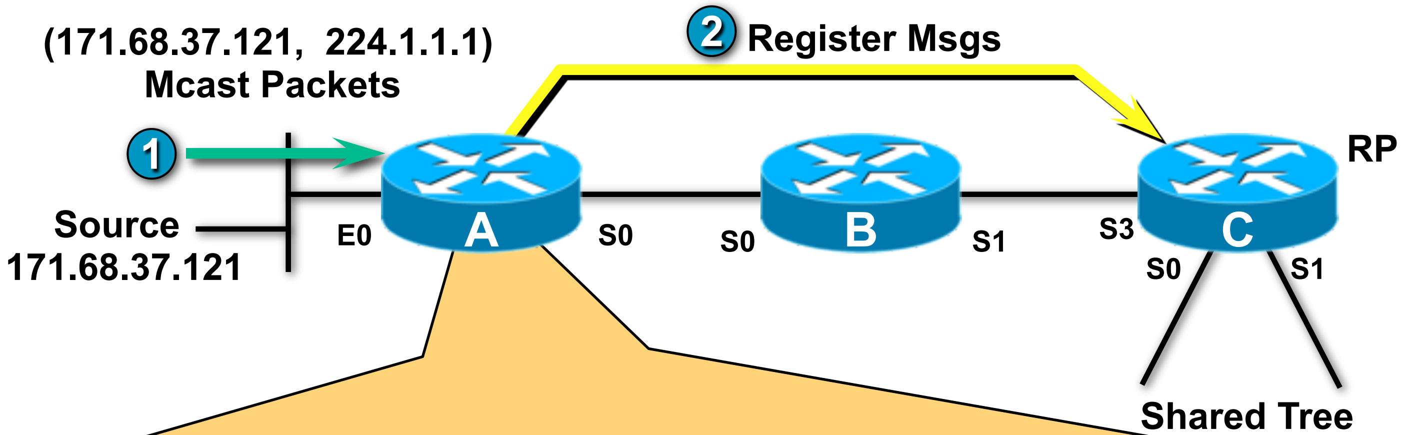
Receiver Joins Group First



1 **Source begins sending group G traffic.**

PIM SM Registering

Receiver Joins Group First




```
(*, 224.1.1.1), 00:00:03/00:00:00, RP 171.68.28.140, flags: SP
  Incoming interface: Serial0, RPF nbr 171.68.28.191,
  Outgoing interface list: Null

(171.68.37.121, 224.1.1.1), 00:00:03/00:02:56, flags: FPT
  Incoming interface: Ethernet0, RPF nbr 0.0.0.0, Registering
  Outgoing interface list: Null
```

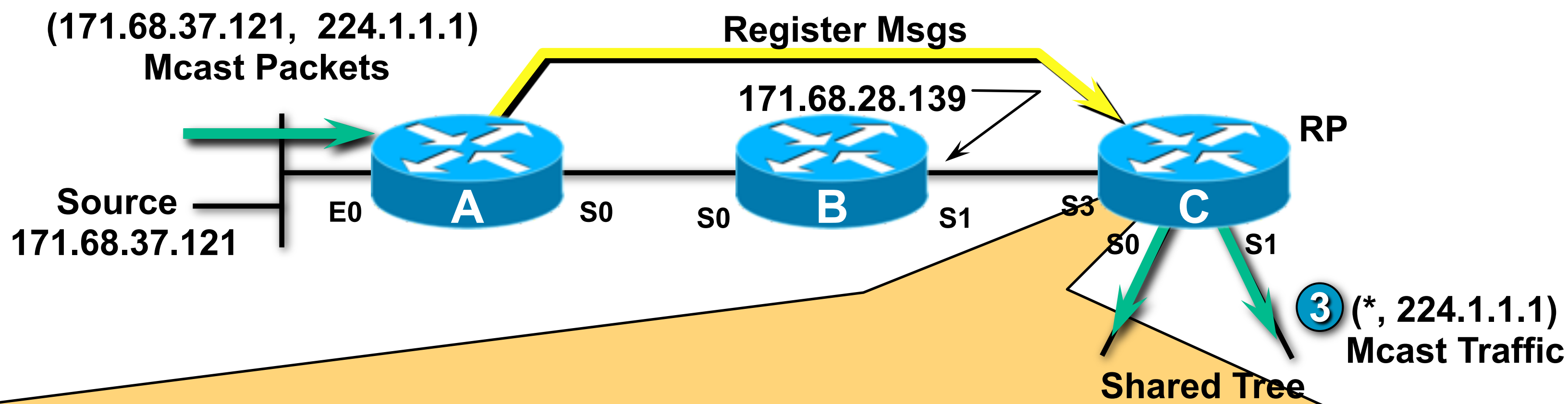
A Creates (S, G) State for Source

(After Automatically Creating a (*, G) entry)

1. **Source begins sending group G traffic.**
2. **A encapsulates packets in Registers; unicasts to RP.**

PIM SM Registering

Receiver Joins Group First

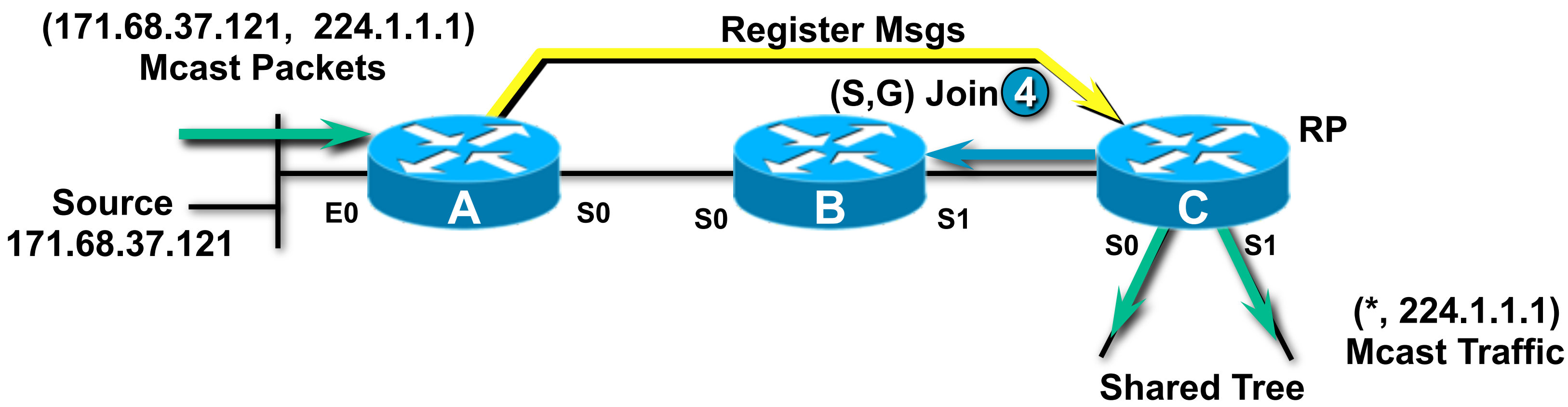



```
(*, 224.1.1.1), 00:09:21/00:00:00, RP 171.68.28.140, flags: S
  Incoming interface: Null, RPF nbr 0.0.0.0,
  Outgoing interface list:
    Serial0, Forward/Sparse-Dense, 00:09:21/00:02:38
    Serial1, Forward/Sparse-Dense, 00:03:14/00:02:46

(171.68.37.121, 224.1.1.1, 00:01:15/00:02:46, flags:
  Incoming interface: Serial3, RPF nbr 171.68.28.139,
  Outgoing interface list:
    Serial0, Forward/Sparse-Dense, 00:00:49/00:02:11
    Serial1, Forward/Sparse-Dense, 00:00:49/00:02:11
```

“RP” Processes Register; Creates (S, G) State

③ RP (C) de-encapsulates packets; forwards down Shared tree.

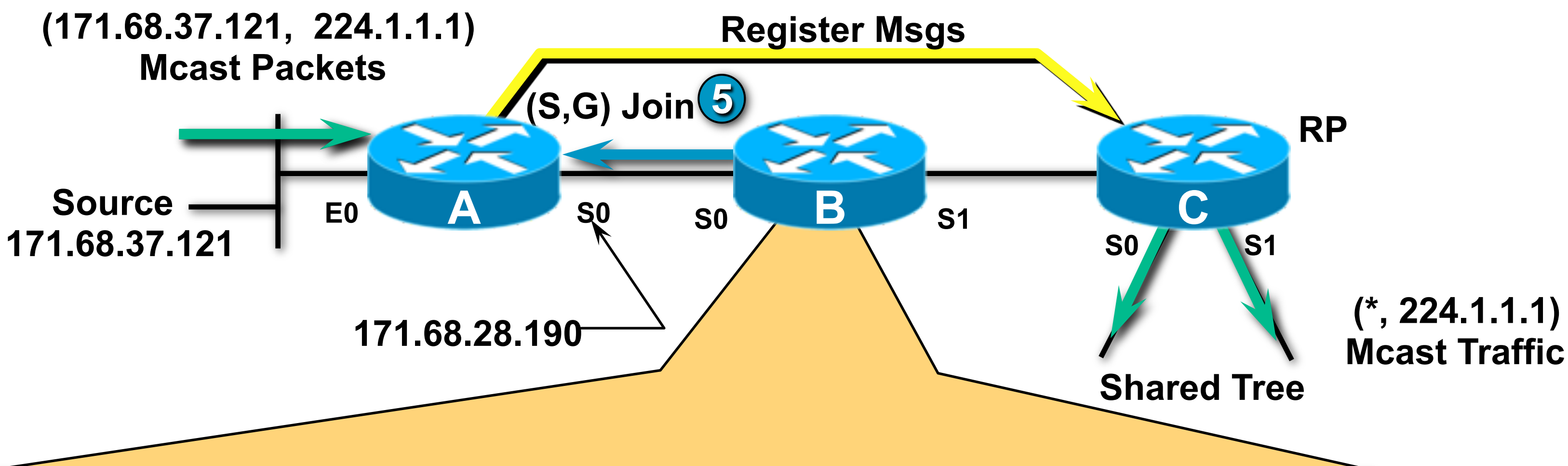
PIM SM Registering
Receiver Joins Group First



4 **RP sends (S,G) Join toward Source to build SPT.**

PIM SM Registering

Receiver Joins Group First




```
(*, 224.1.1.1), 00:04:28/00:00:00, RP 171.68.28.140, flags: SP
  Incoming interface: Serial1, RPF nbr 171.68.28.140,
  Outgoing interface list: Null

(171.68.37.121, 224.1.1.1), 00:04:28/00:01:32, flags:
  Incoming interface: Serial0, RPF nbr 171.68.28.190
  Outgoing interface list:
    Serial1, Forward/Sparse-Dense, 00:04:28/00:01:32
```

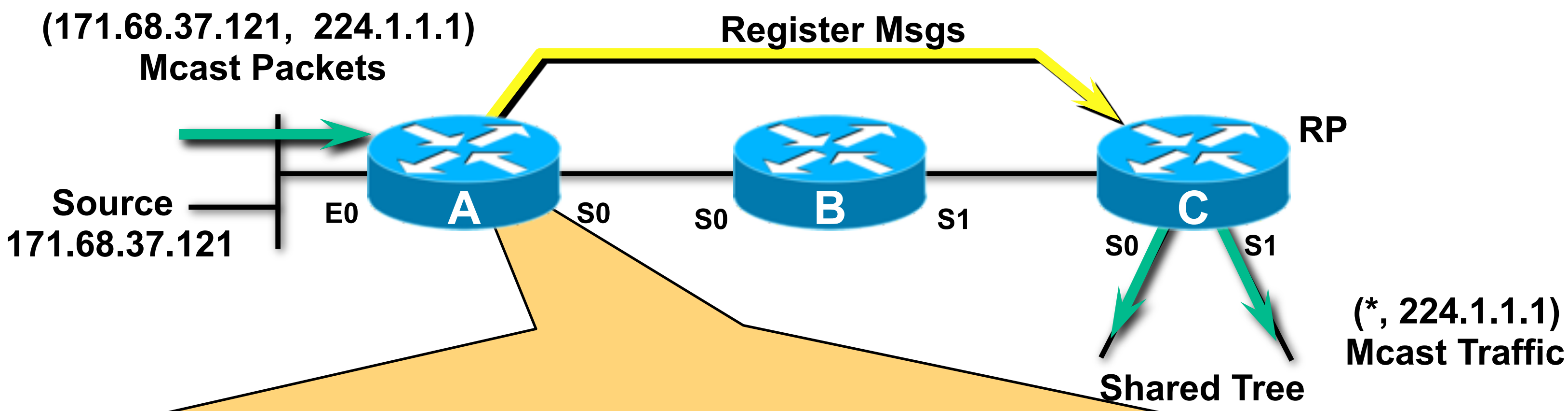
B Processes Join, Creates (S, G) State

(After Automatically Creating the (*, G) Entry)

5 **B sends (S,G) Join toward Source to continue building SPT.**

PIM SM Registering

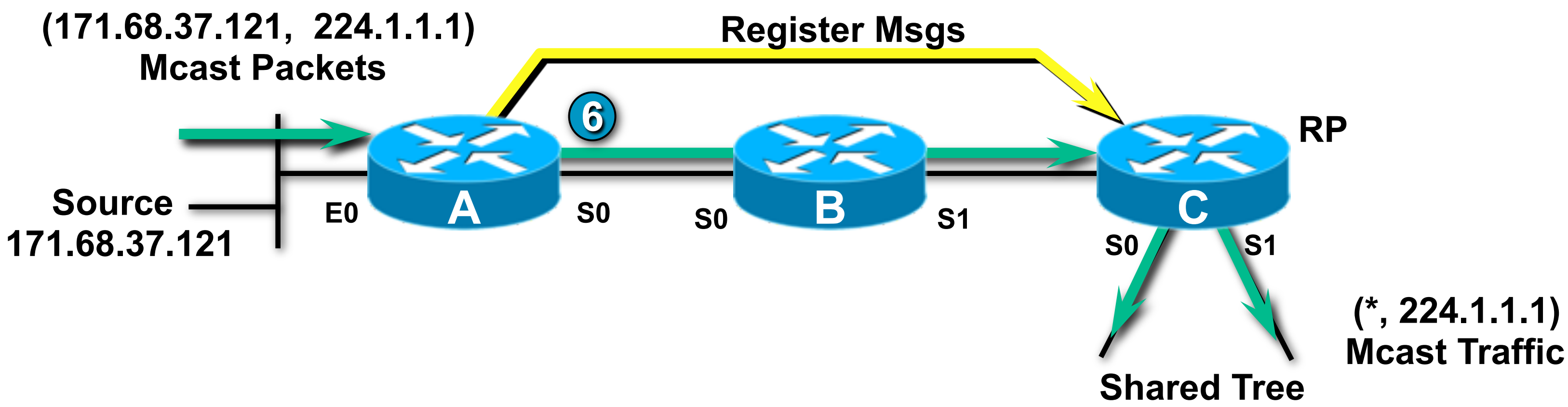
Receiver Joins Group First




```
(*, 224.1.1.1), 00:04:28/00:00:00, RP 171.68.28.140, flags: SP
  Incoming interface: Serial0, RPF nbr 171.68.28.191,
  Outgoing interface list: Null

(171.68.37.121, 224.1.1.1), 00:04:28/00:01:32, flags: FT
  Incoming interface: Ethernet0, RPF nbr 0.0.0.0, Registering
  Outgoing interface list:
    Serial0, Forward/Sparse-Dense, 00:04:28/00:01:32
```

A Processes the (S, G) Join; Adds Serial0 to OIL

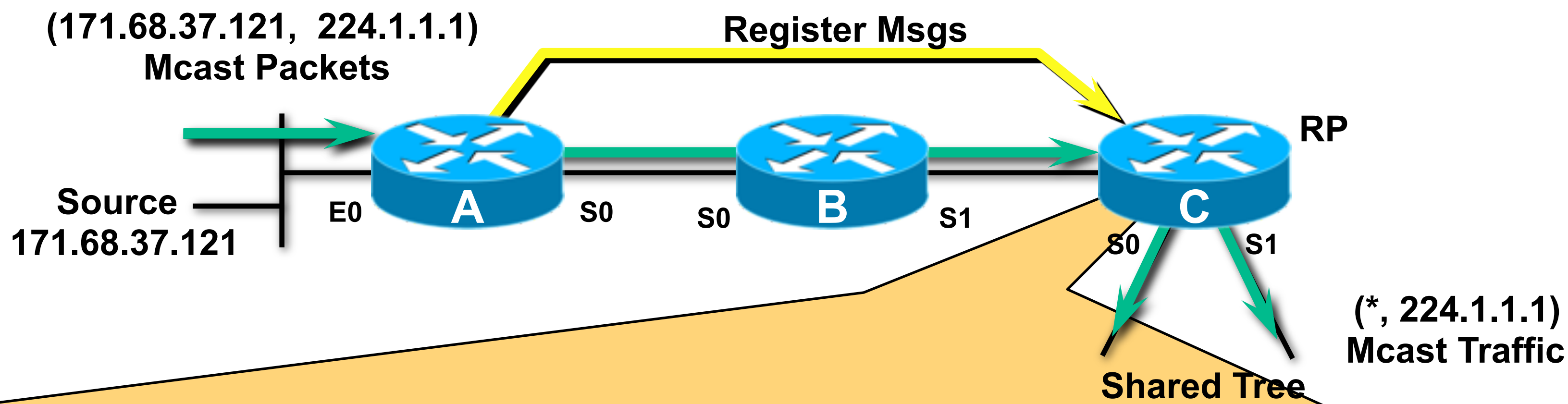
PIM SM Registering
Receiver Joins Group First



6 **RP begins receiving (S,G) traffic down SPT.**

PIM SM Registering

Receiver Joins Group First

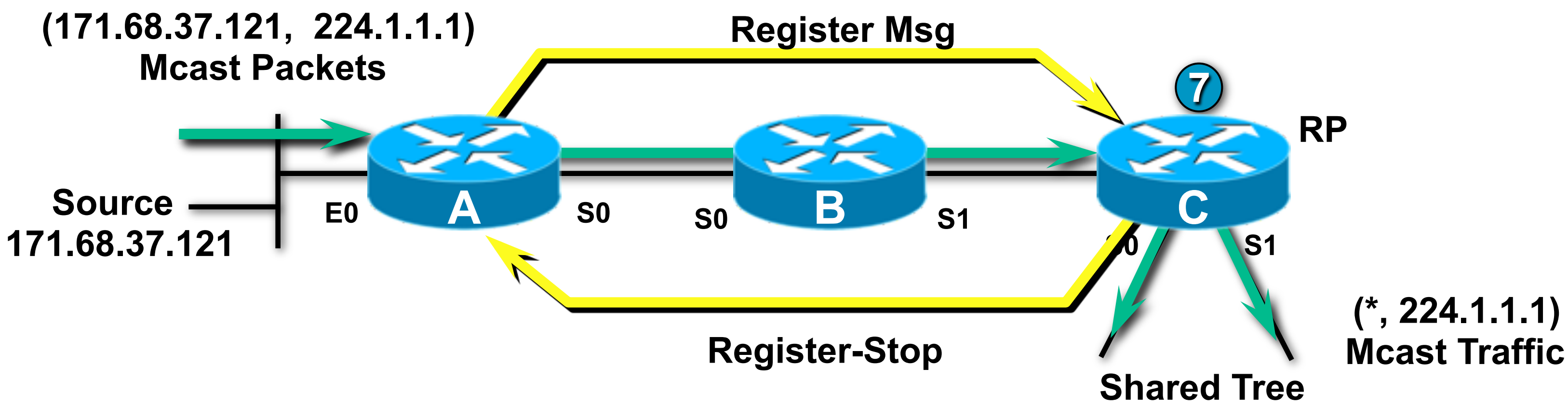



```
(*, 224.1.1.1), 00:09:21/00:00:00, RP 171.68.28.140, flags: S
  Incoming interface: Null, RPF nbr 0.0.0.0,
  Outgoing interface list:
    Serial0, Forward/Sparse-Dense, 00:09:21/00:02:38
    Serial1, Forward/Sparse-Dense, 00:03:14/00:02:46

(171.68.37.121, 224.1.1.1, 00:01:15/00:02:46, flags:T
  Incoming interface: Serial3, RPF nbr 171.68.28.139,
  Outgoing interface list:
    Serial0, Forward/Sparse-Dense, 00:00:49/00:02:11
    Serial1, Forward/Sparse-Dense, 00:00:49/00:02:11
```

Note “T”Flag Is Now Set

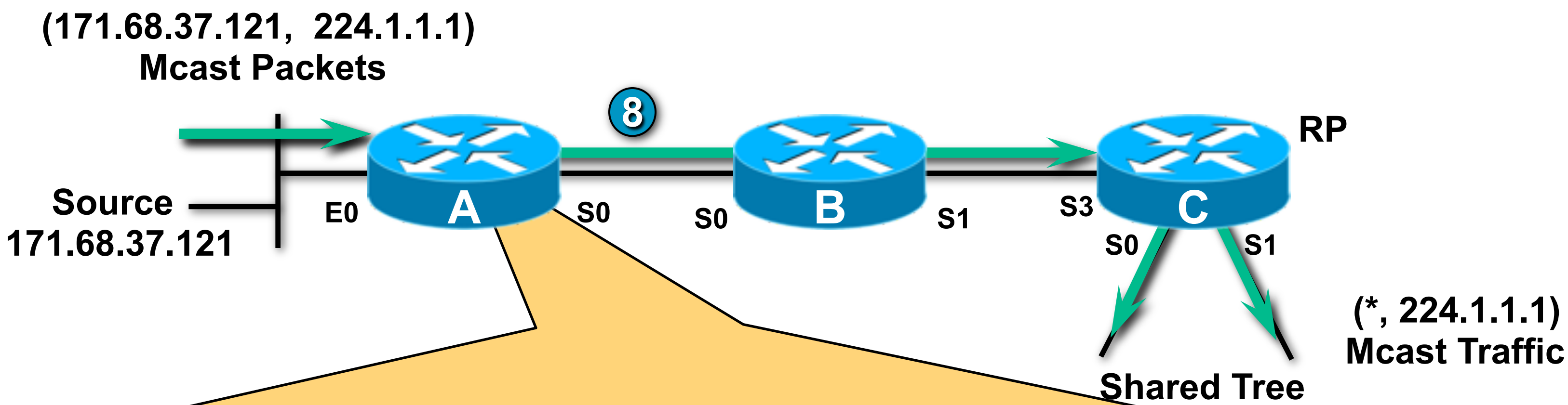
Traffic Arriving via SPT Is Forwarded Down Shared Tree (This Causes the “T” Flag to Be Set)

PIM SM Registering
Receiver Joins Group First



7 **Once "T" Flag is set, next "Register" causes RP to send back a "Register-Stop" to A**

PIM SM Registering
Receiver Joins Group First



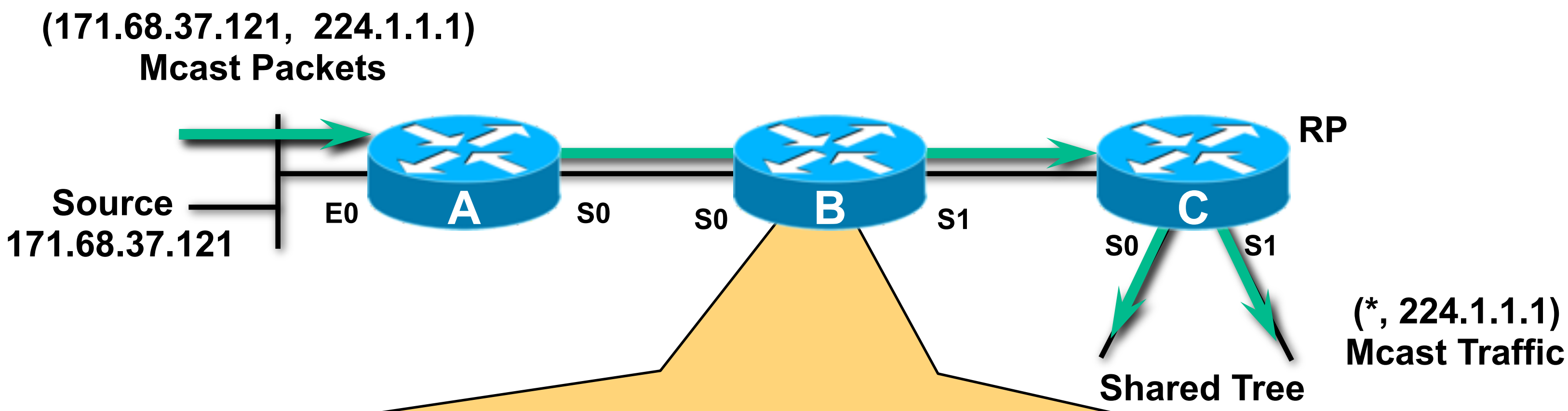

```
(*, 224.1.1.1), 00:04:28/00:00:00, RP 171.68.28.140, flags: SP
  Incoming interface: Serial0, RPF nbr 171.68.28.191,
  Outgoing interface list: Null

(171.68.37.121, 224.1.1.1), 00:04:28/00:01:32, flags: FT
  Incoming interface: Ethernet0, RPF nbr 0.0.0.0,
  Outgoing interface list:
     Serial0, Forward/Sparse-Dense, 00:04:28/00:01:32
```

A Stops Sending Register Messages

(Final State in A)

8 **(S,G) Traffic now flowing down a single path (SPT) to RP.**

PIM SM Registering
Receiver Joins Group First

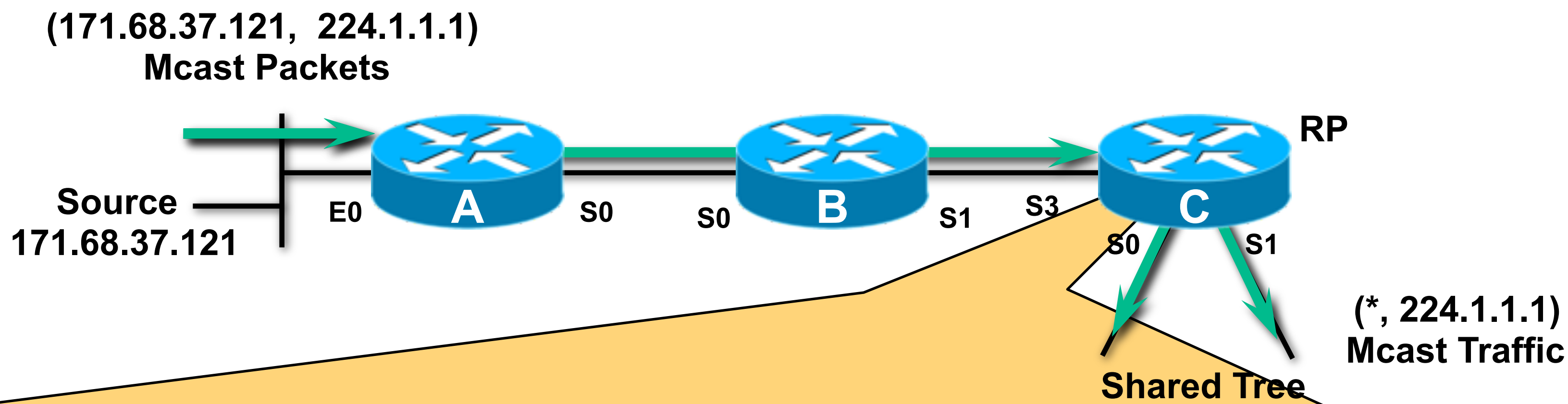



```
(*, 224.1.1.1), 00:04:28/00:00:00, RP 171.68.28.140, flags: SP
  Incoming interface: Serial1, RPF nbr 171.68.28.140,
  Outgoing interface list: Null

(171.68.37.121, 224.1.1.1), 00:04:28/00:01:32, flags: T
  Incoming interface: Serial0, RPF nbr 171.68.28.190
  Outgoing interface list:
    Serial1, Forward/Sparse-Dense, 00:04:28/00:01:32
```

Final State in B

PIM SM Registering

Receiver Joins Group First




```
(*, 224.1.1.1), 00:09:21/00:00:00, RP 171.68.28.140, flags: S
  Incoming interface: Null, RPF nbr 0.0.0.0,
  Outgoing interface list:
    Serial0, Forward/Sparse-Dense, 00:09:21/00:02:38
    Serial1, Forward/Sparse-Dense, 00:03:14/00:02:46

(171.68.37.121, 224.1.1.1, 00:01:15/00:02:46, flags: T
  Incoming interface: Serial3, RPF nbr 171.68.28.139,
  Outgoing interface list:
    Serial0, Forward/Sparse-Dense, 00:00:49/00:02:11
    Serial1, Forward/Sparse-Dense, 00:00:49/00:02:11
```

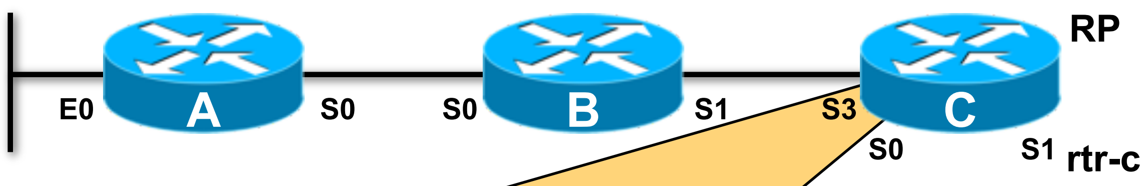
Final State in the “RP”

(With Receivers on Shared Tree)

PIM SM Registering: Source Registers First

PIM SM Registering

Source Registers First



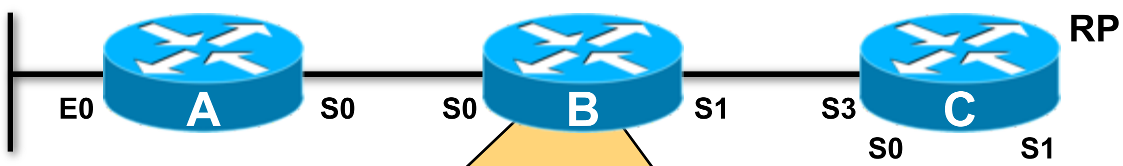

```
rtr-c>show ip mroute 224.1.1.1

Group 224.1.1.1 not found.
```

```
RP/0/5/CPU0:rtr-c#show mrib route 224.1.1.1
No matching routes in MRIB route-DB
```

State in “RP” Before Registering
(Without Receivers on Shared Tree)

PIM SM Registering

Source Registers First




```
rtr-b>show ip mroute 224.1.1.1

Group 224.1.1.1 not found.
```

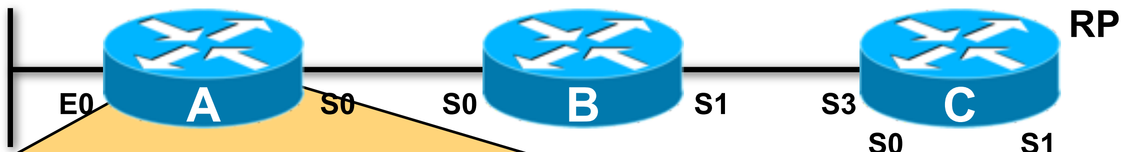
```
RP/0/5/CPU0:rtr-b#show mrib route 224.1.1.1
No matching routes in MRIB route-DB
```

State in B Before Any Source Registers

(With Receivers on Shared Tree)

PIM SM Registering

Source Registers First



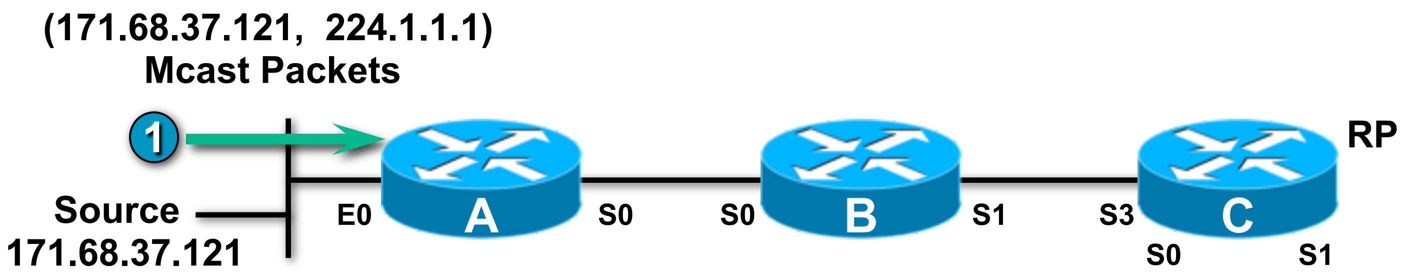

```
rtr-a>show ip mroute 224.1.1.1

Group 224.1.1.1 not found.
```

```
RP/0/5/CPU0:rtr-a#show mrib route 224.1.1.1
No matching routes in MRIB route-DB
```

State in A Before Any Source Registers

(With Receivers on Shared Tree)

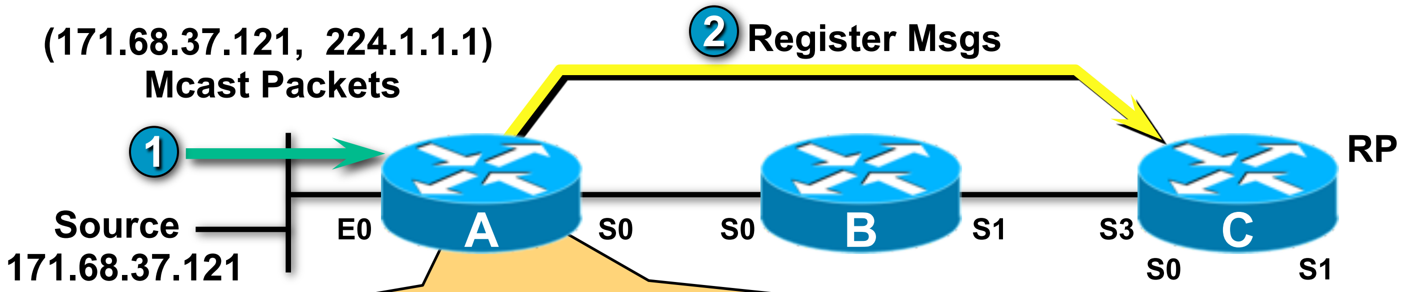
PIM SM Registering
Source Registers First



1 **Source begins sending group G traffic.**

PIM SM Registering

Source Registers First




```
(*, 224.1.1.1), 00:00:03/00:00:00, RP 171.68.28.140, flags: SP
  Incoming interface: Serial0, RPF nbr 171.68.28.191,
  Outgoing interface list: Null

(171.68.37.121, 224.1.1.1), 00:00:03/00:02:56, flags: FPT
  Incoming interface: Ethernet0, RPF nbr 0.0.0.0, Registering
  Outgoing interface list: Null
```

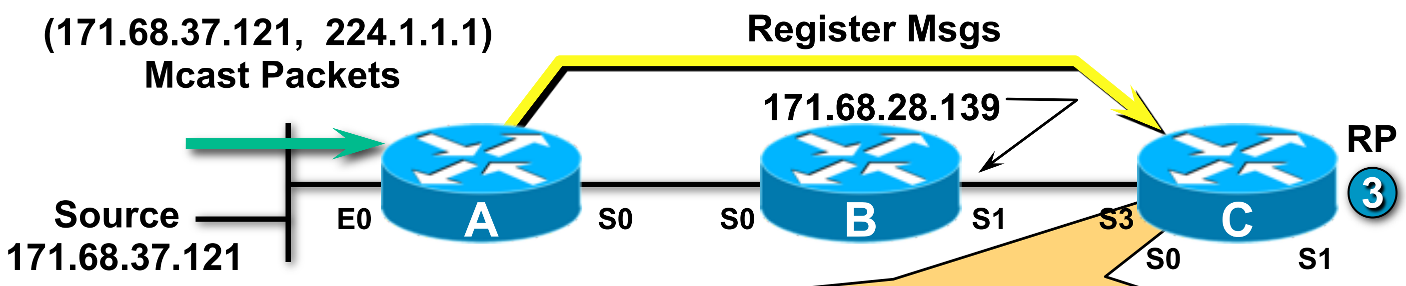
A Creates (S, G) State for Source

(After Automatically Creating a (*, G) Entry)

1. **Source begins sending group G traffic.**
2. **A encapsulates packets in Registers; unicasts to RP.**

PIM SM Registering

Source Registers First




```
(*, 224.1.1.1), 00:01:15/00:00:00, RP 171.68.28.140, flags: SP
  Incoming interface: Null, RPF nbr 0.0.0.0,
  Outgoing interface list: Null

(171.68.37.121, 224.1.1.1), 00:01:15/00:01:45, flags: P
  Incoming interface: Serial3, RPF nbr 171.68.28.139,
  Outgoing interface list: Null
```

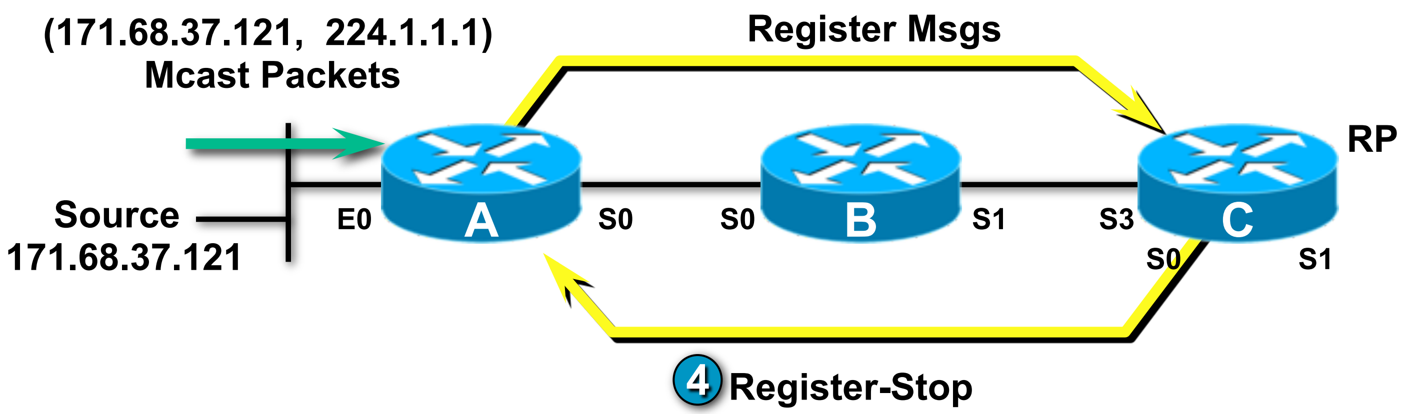
"RP" Processes Register; Creates (S, G) State

(After Automatically Creating the (*, G) Entry)**

3 RP (C) has no receivers on Shared Tree; discards packet.

PIM SM Registering

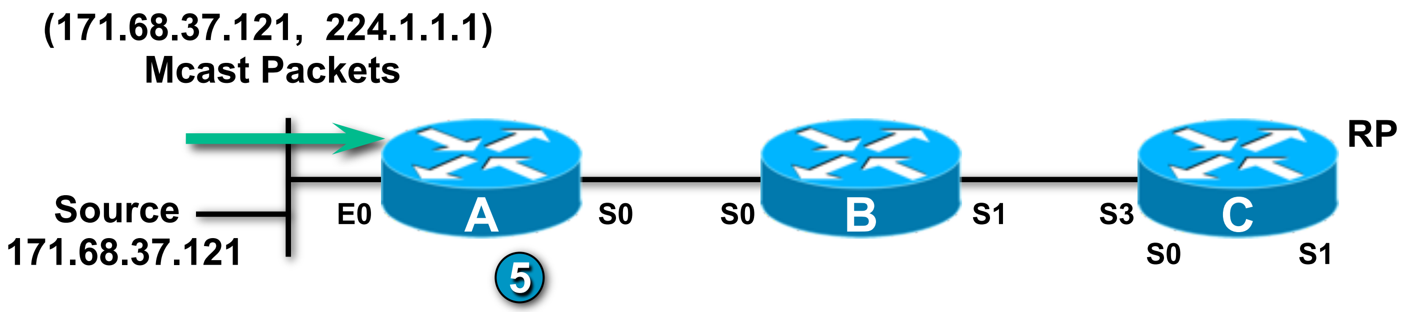
Source Registers First



4 **RP sends "Register-Stop" to A.**

PIM SM Registering

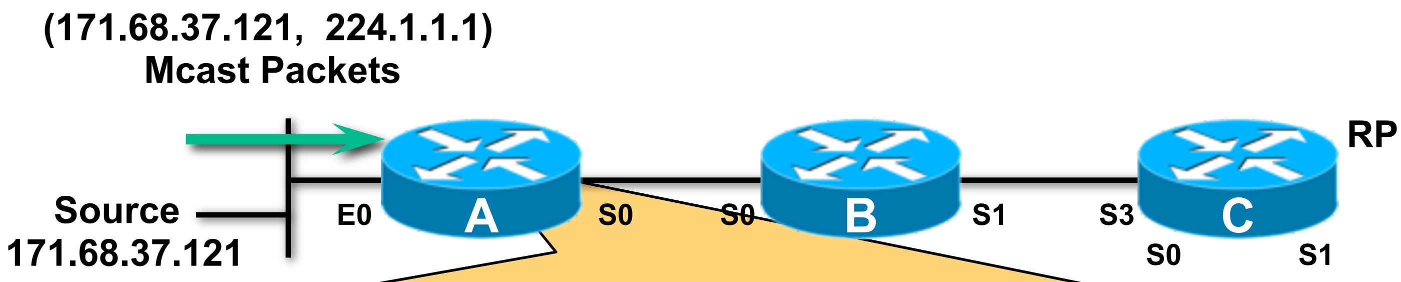
Source Registers First



5 A stops encapsulating traffic in Register Messages; drops packets from Source.

PIM SM Registering

Source Registers First



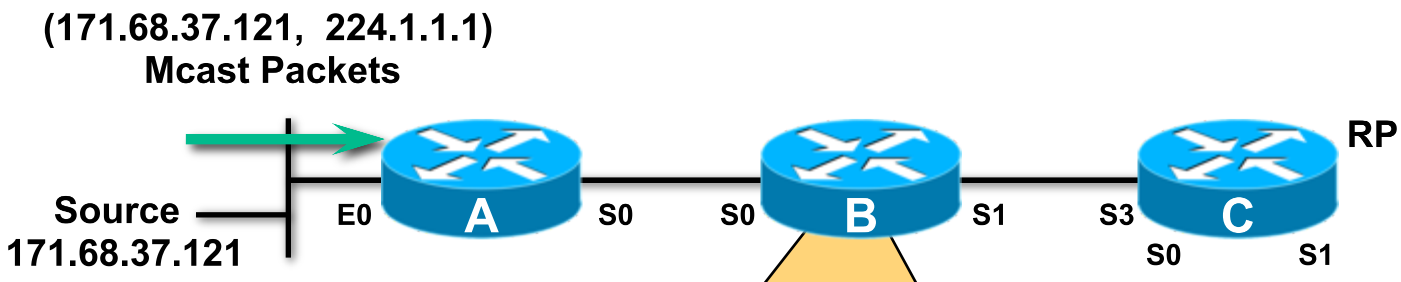

```
(*, 224.1.1.1), 00:01:28/00:00:00, RP 171.68.28.140, flags: SP
  Incoming interface: Serial0, RPF nbr 171.68.28.191,
  Outgoing interface list: Null

(171.68.37.121, 224.1.1.1), 00:01:28/00:01:32, flags: FPT
  Incoming interface: Ethernet0, RPF nbr 0.0.0.0
  Outgoing interface list: Null
```

State in A After Registering

(Without Receivers on Shared Tree)

PIM SM Registering

Source Registers First




```
rtr-b>show ip mroute 224.1.1.1

Group 224.1.1.1 not found.
```

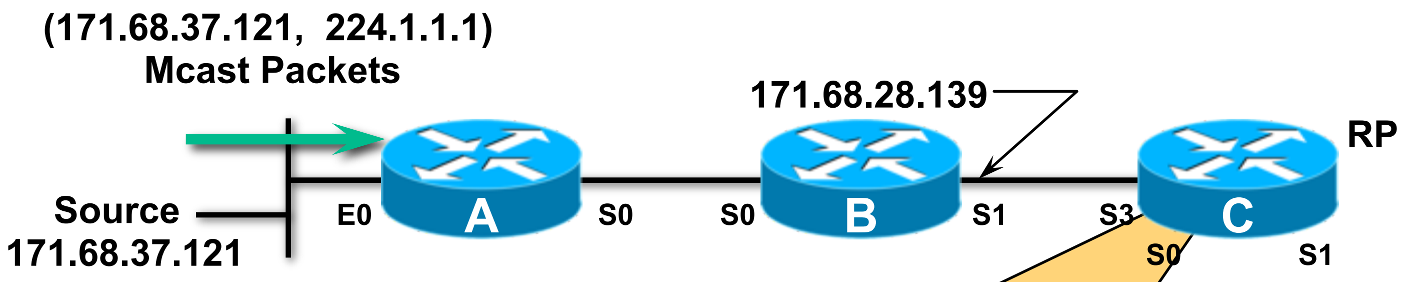
```
RP/0/5/CPU0:rtr-b#show mrib route 224.1.1.1
No matching routes in MRIB route-DB
```

State in B After A Registers

(Without Receivers on Shared Tree)

PIM SM Registering

Source Registers First

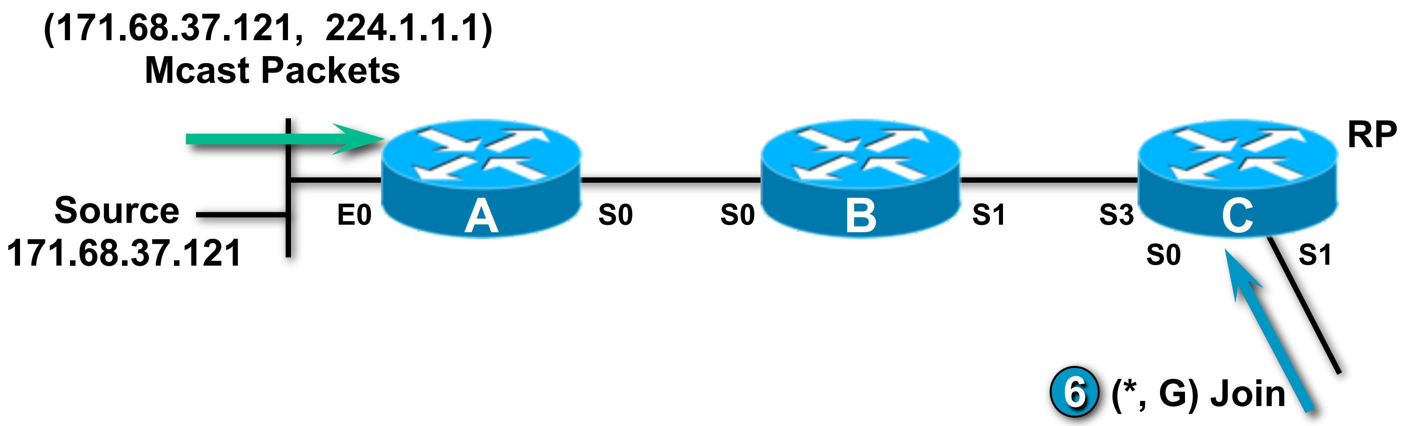



```
(*, 224.1.1.1), 00:01:15/00:00:00, RP 171.68.28.140, flags: SP
  Incoming interface: Null, RPF nbr 0.0.0.0,
  Outgoing interface list: Null

(171.68.37.121, 224.1.1.1), 00:01:15/00:01:45, flags: P
  Incoming interface: Serial3, RPF nbr 171.68.28.139,
  Outgoing interface list: Null
```

State in RP After A Registers
(Without Receivers on Shared Tree)

PIM SM Registering

Source Registers First

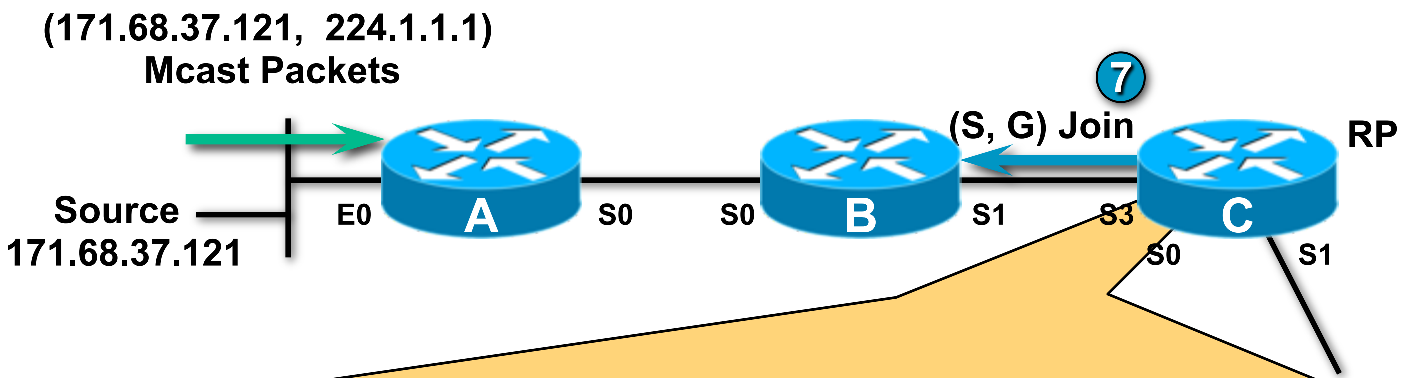


Receivers Begin Joining the Shared Tree

6 **RP (C) receives (*, G) Join from a receiver on Shared Tree.**

PIM SM Registering

Source Registers First




```
(*, 224.1.1.1), 00:09:21/00:00:00, RP 171.68.28.140, flags: S
  Incoming interface: Null, RPF nbr 0.0.0.0,
  Outgoing interface list:
     Serial1, Forward/Sparse-Dense, 00:00:14/00:02:46

(171.68.37.121, 224.1.1.1, 00:01:15/00:02:46, flags: T
  Incoming interface: Serial3, RPF nbr 171.68.28.139,
  Outgoing interface list:
     Serial1, Forward/Sparse-Dense, 00:00:14/00:02:46
```

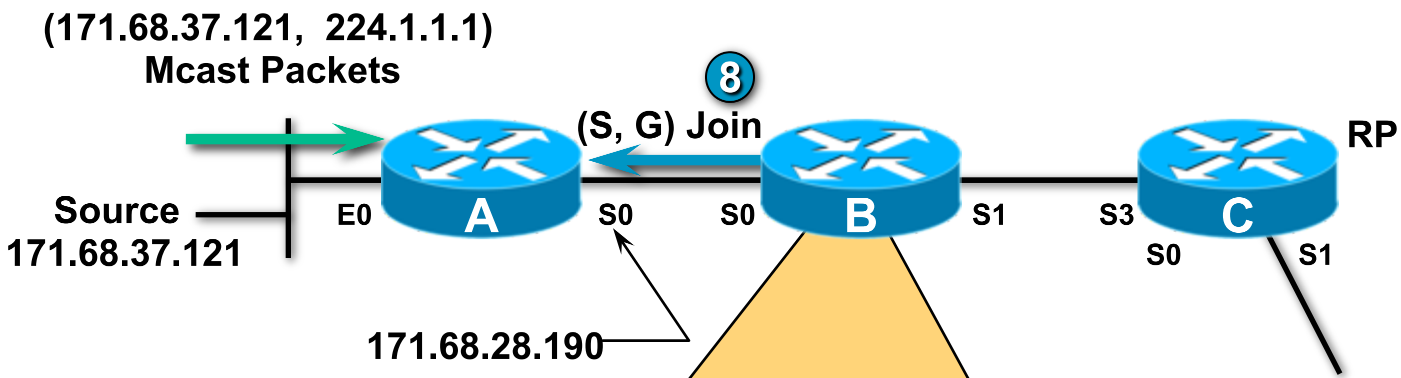
RP Processes (*,G) Join

(Adds Serial1 to Outgoing Interface Lists)

7 **RP sends (S,G) Joins for all known Sources in Group.**

PIM SM Registering

Source Registers First




```
(*, 224.1.1.1), 00:04:28/00:00:00, RP 171.68.28.140, flags: SP
  Incoming interface: Serial1, RPF nbr 171.68.28.140,
  Outgoing interface list: Null

(171.68.37.121, 224.1.1.1), 00:04:28/00:01:32, flags:
  Incoming interface: Serial0, RPF nbr 171.68.28.190
  Outgoing interface list:
    Serial1, Forward/Sparse-Dense, 00:04:28/00:01:32
```

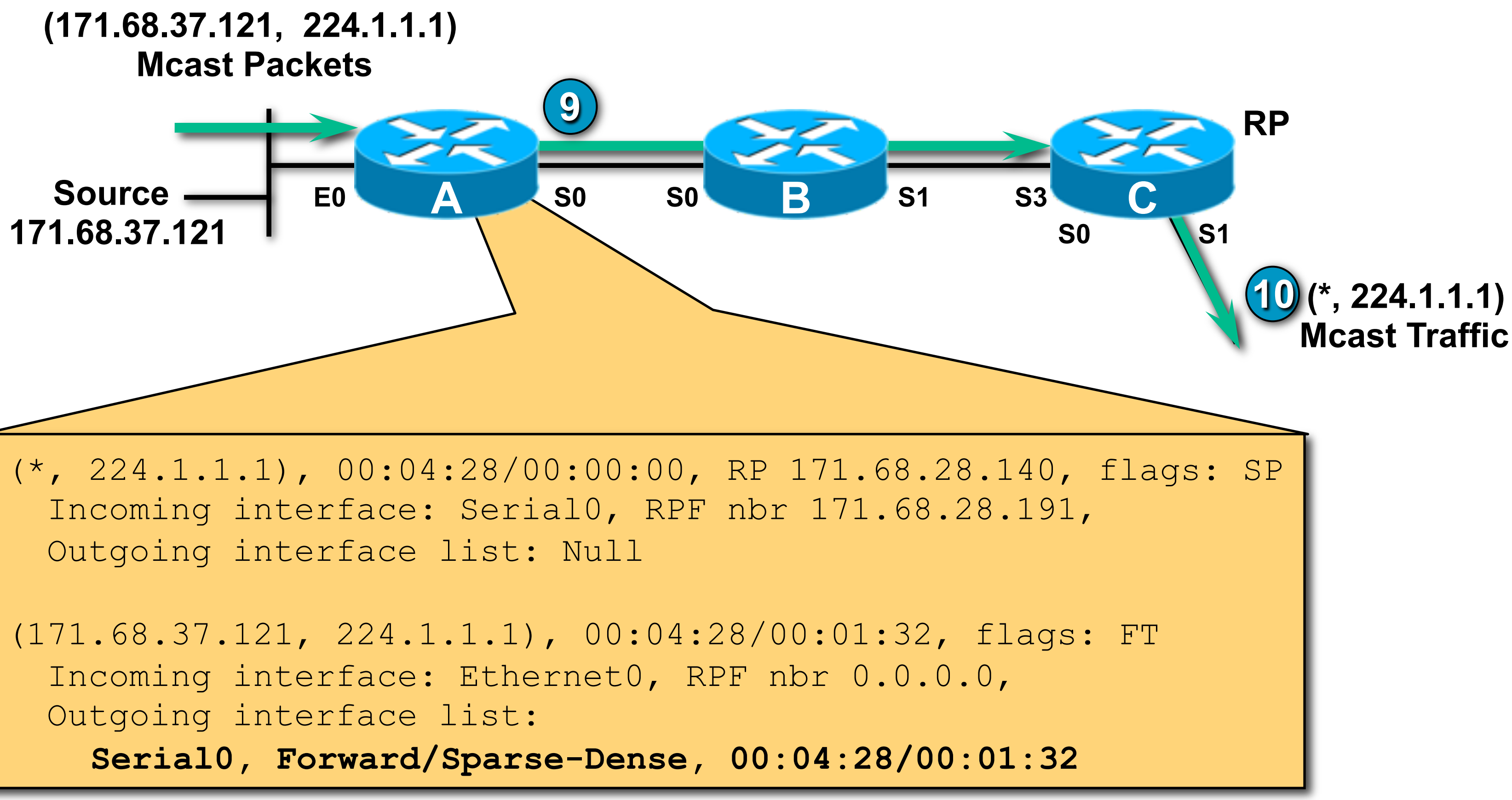
B Processes Join, Creates (S, G) State

(After Automatically Creating the (*, G) Entry)

8 B sends (S,G) Join toward Source to continue building SPT.

PIM SM Registering

Source Registers First

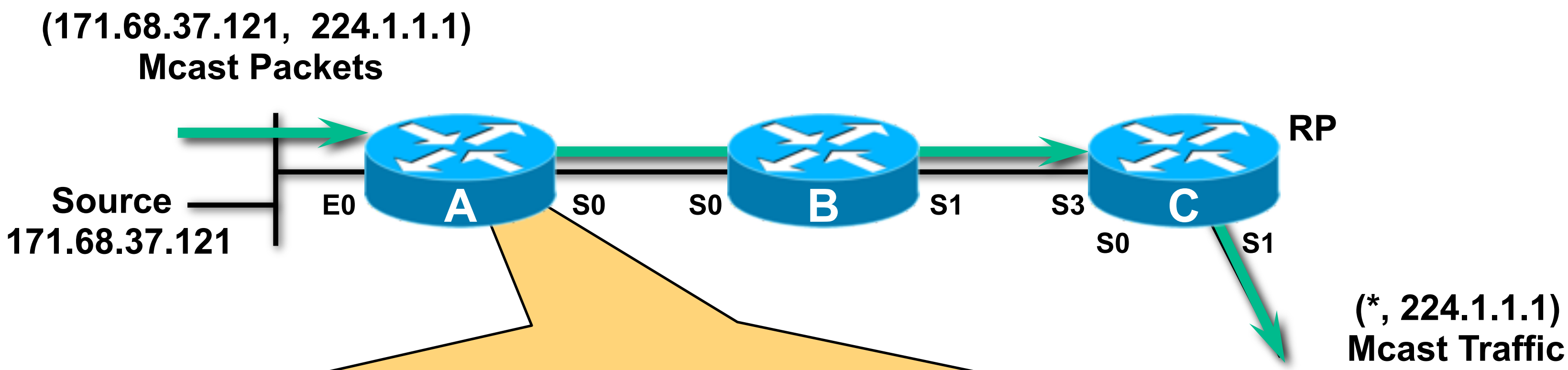


A Processes the (S, G) Join; Adds Serial0 to OIL

9. **RP begins receiving (S,G) traffic down SPT.**
10. **RP forwards (S,G) traffic down Shared Tree to receivers.**

PIM SM Registering

Source Registers First

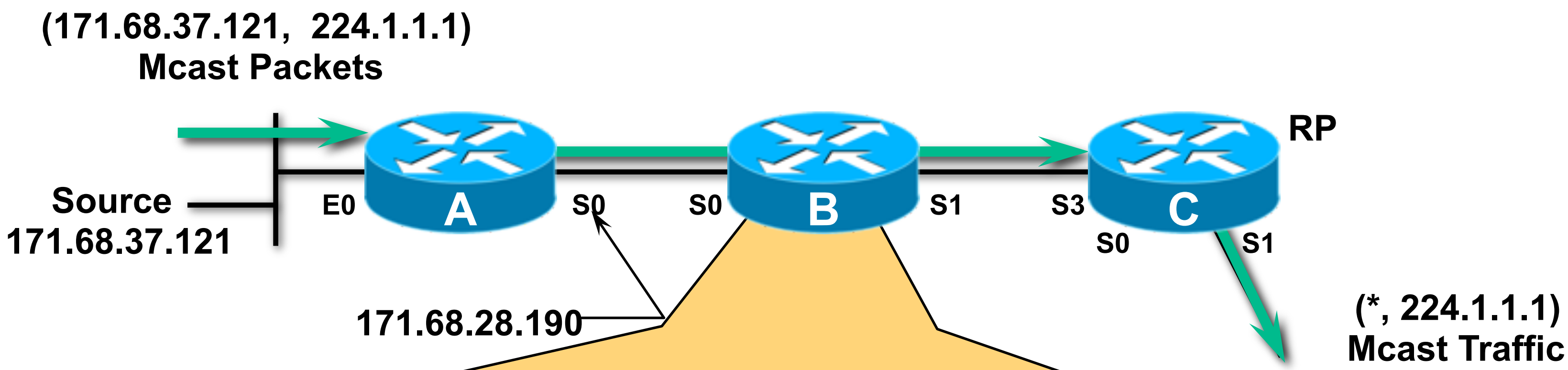



```
(*, 224.1.1.1), 00:04:28/00:00:00, RP 171.68.28.140, flags: SP
  Incoming interface: Serial0, RPF nbr 171.68.28.191,
  Outgoing interface list: Null

(171.68.37.121, 224.1.1.1), 00:04:28/00:01:32, flags: FT
  Incoming interface: Ethernet0, RPF nbr 0.0.0.0,
  Outgoing interface list:
    Serial1, Forward/Sparse-Dense, 00:04:28/00:01:32
```

Final State in Router A (IOS)

PIM SM Registering

Source Registers First

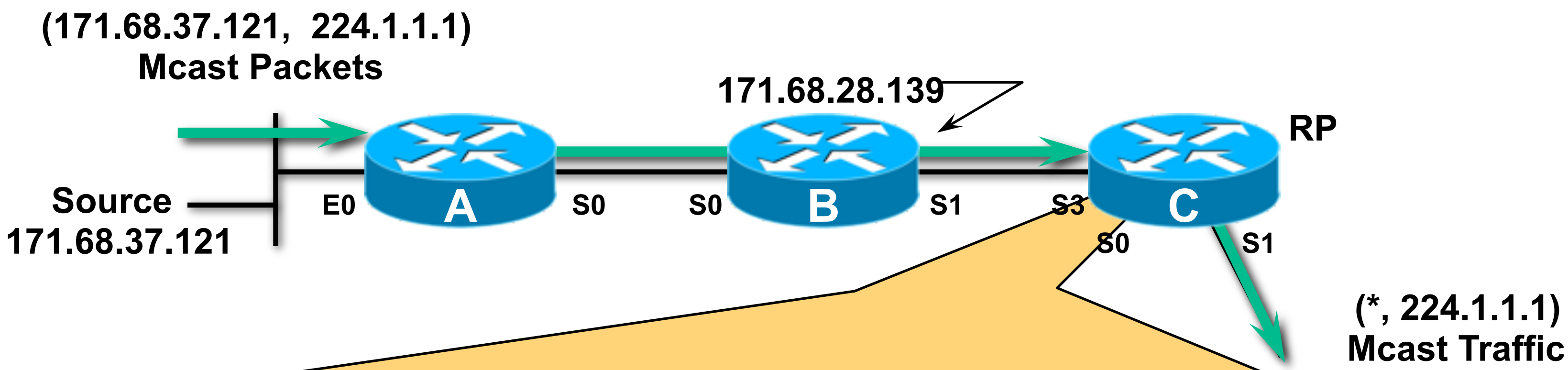



```
(*, 224.1.1.1), 00:04:28/00:00:00, RP 171.68.28.140, flags: SP
  Incoming interface: Serial1, RPF nbr 171.68.28.140,
  Outgoing interface list: Null

(171.68.37.121, 224.1.1.1), 00:04:28/00:01:32, flags: T
  Incoming interface: Serial0, RPF nbr 171.68.28.190
  Outgoing interface list:
    Serial1, Forward/Sparse-Dense, 00:04:28/00:01:32
```

Final State in B After Receivers Join

PIM SM Registering

Source Registers First



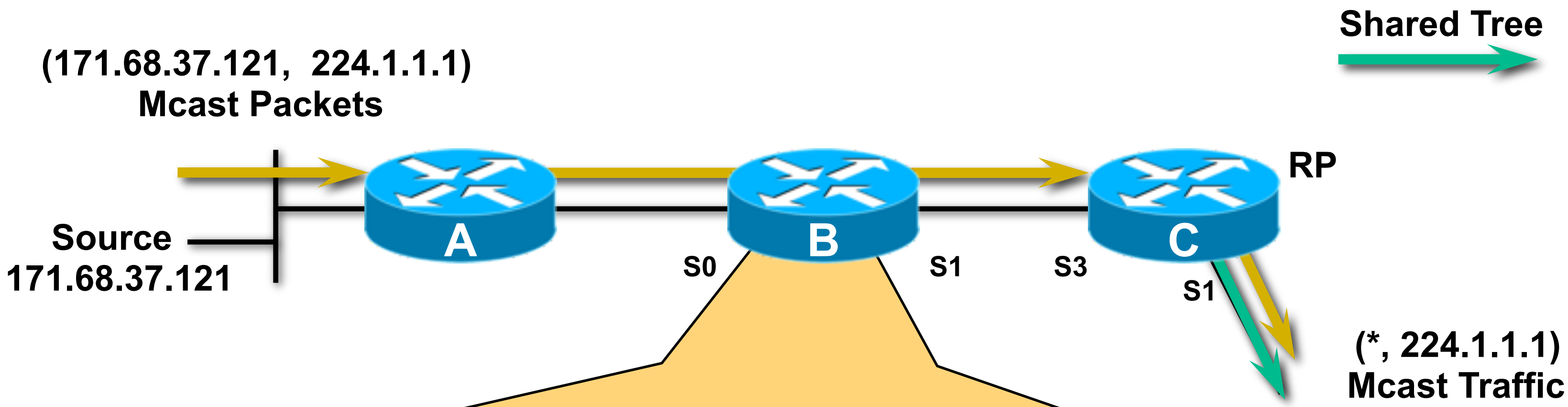

```
(*, 224.1.1.1), 00:09:21/00:00:00, RP 171.68.28.140, flags: S
  Incoming interface: Null, RPF nbr 0.0.0.0,
  Outgoing interface list:
    Serial1, Forward/Sparse-Dense, 00:03:14/00:02:46

(171.68.37.121, 224.1.1.1, 00:01:15/00:02:46, flags: T
  Incoming interface: Serial3, RPF nbr 171.68.28.139,
  Outgoing interface list:
    Serial1, Forward/Sparse-Dense, 00:00:49/00:02:11
```

Final State in RP After Receivers Join (IOS)

PIM SM Registering: Receiver Along the SPT

PIM SM Registering

Receivers Along the SPT

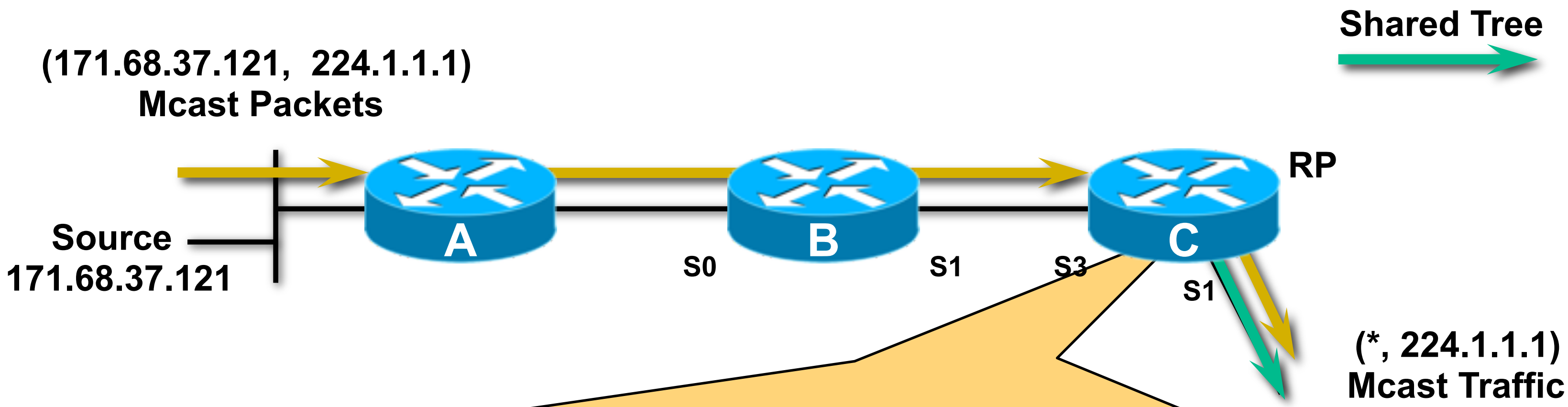



```
(*, 224.1.1.1), 00:04:28/00:00:00, RP 171.68.28.140, flags: SP
  Incoming interface: Serial1, RPF nbr 171.68.28.140,
  Outgoing interface list: Null

(171.68.37.121, 224.1.1.1), 00:04:28/00:01:32, flags: T
  Incoming interface: Serial0, RPF nbr 171.68.28.190
  Outgoing interface list:
    Serial1, Forward/Sparse-Dense, 00:04:28/00:01:32
```

Current State in B

PIM SM Registering

Receivers Along the SPT

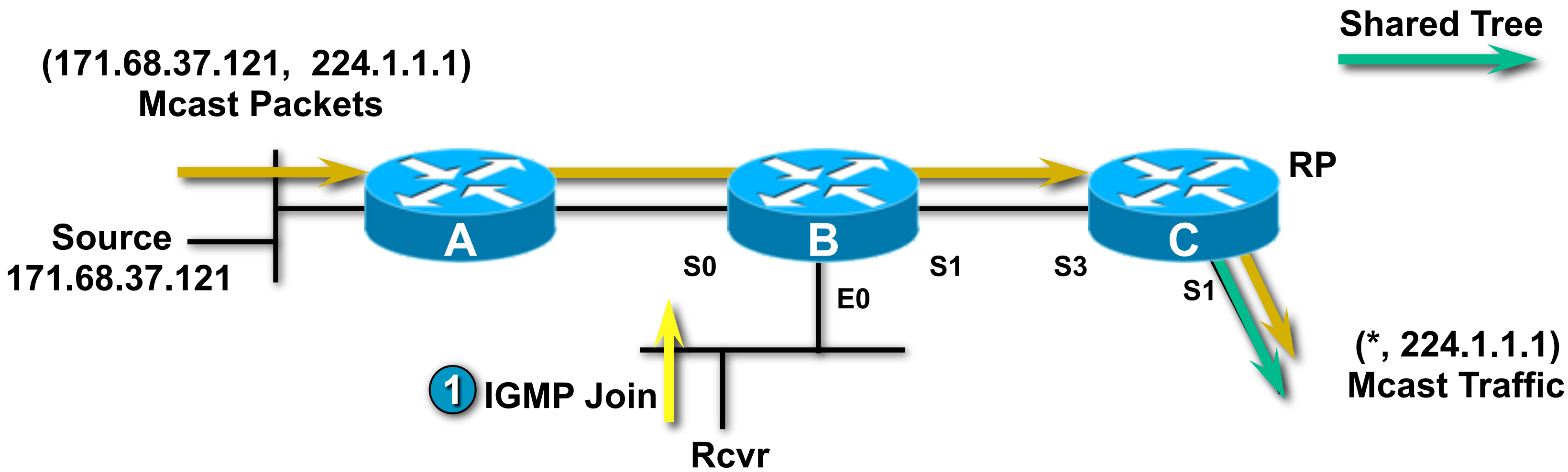



```
(*, 224.1.1.1), 00:09:21/00:00:00, RP 171.68.28.140, flags: S
  Incoming interface: Null, RPF nbr 0.0.0.0,
  Outgoing interface list:
    Serial1, Forward/Sparse-Dense, 00:03:14/00:02:46

(171.68.37.121, 224.1.1.1, 00:01:15/00:02:46, flags: T
  Incoming interface: Serial3, RPF nbr 171.68.28.139,
  Outgoing interface list:
    Serial1, Forward/Sparse-Dense, 00:00:49/00:02:11
```

Current State in the RP

PIM SM Registering

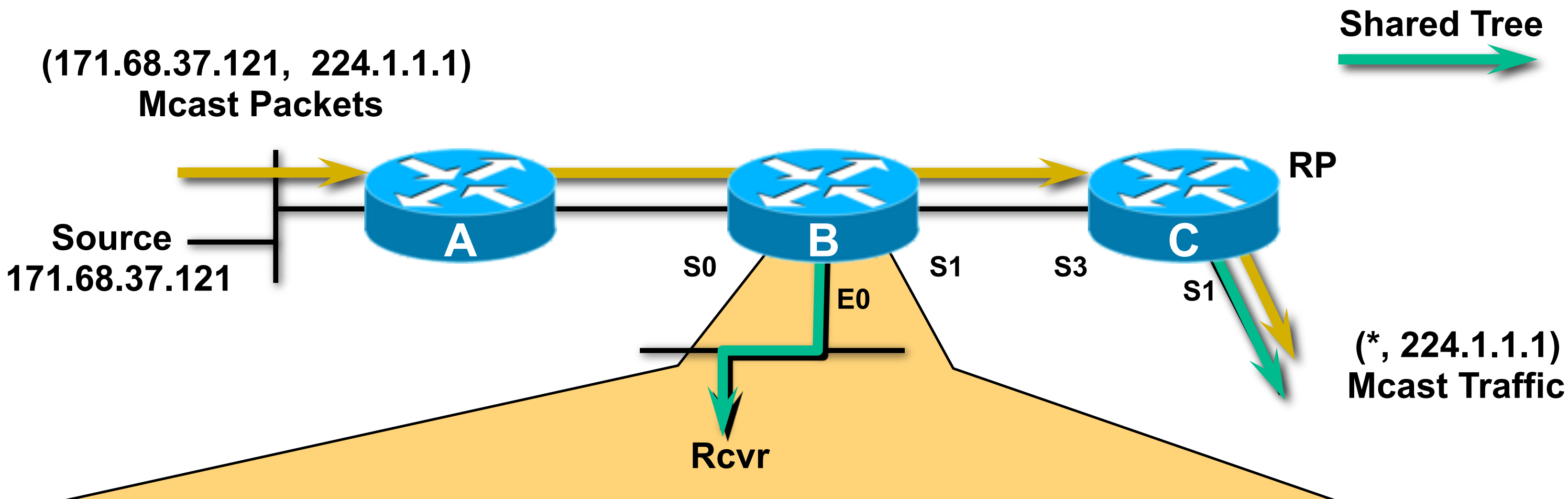
Receivers Along the SPT



1 Rcvr wishes to receive group G traffic. Sends IGMP Join for G.

PIM SM Registering

Receivers Along the SPT

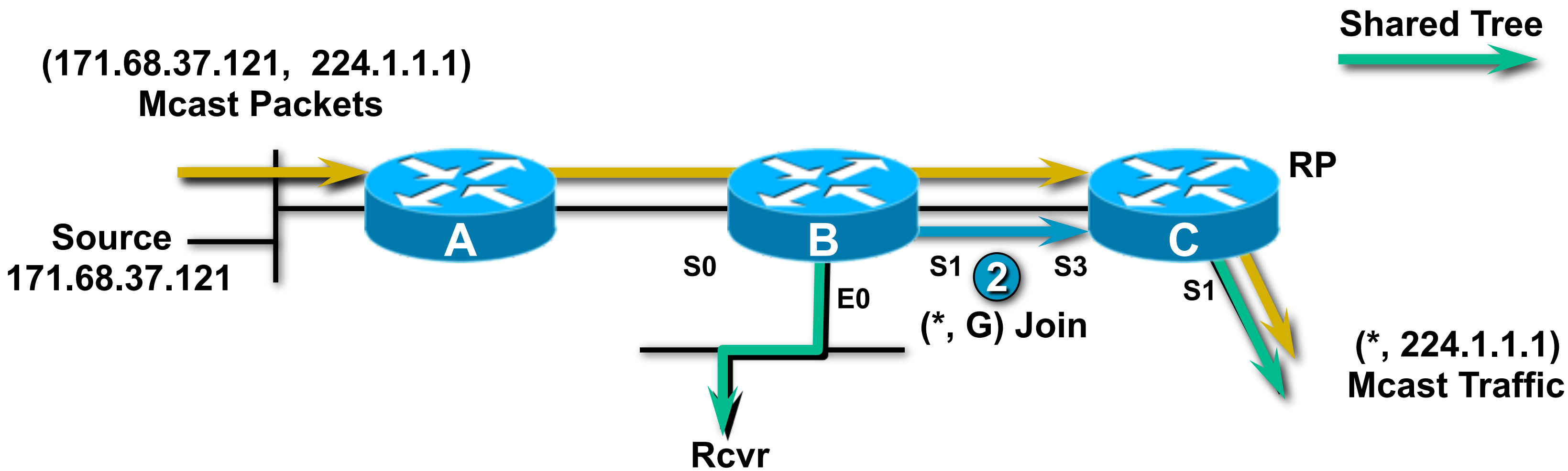



```
(*, 224.1.1.1), 00:04:28/00:00:00, RP 171.68.28.140, flags: SC
  Incoming interface: Serial1, RPF nbr 171.68.28.140,
  Outgoing interface list:
    Ethernet0, Forward/Sparse-Dense, 00:00:30/00:02:30

(171.68.37.121, 224.1.1.1), 00:04:28/00:01:32, flags: CT
  Incoming interface: Serial0, RPF nbr 171.68.28.190
  Outgoing interface list:
    Serial1, Forward/Sparse-Dense, 00:04:28/00:01:32
    Ethernet0, Forward/Sparse-Dense, 00:00:30/00:02:30
```

State in B After Rcvr Joins Group

PIM SM Registering

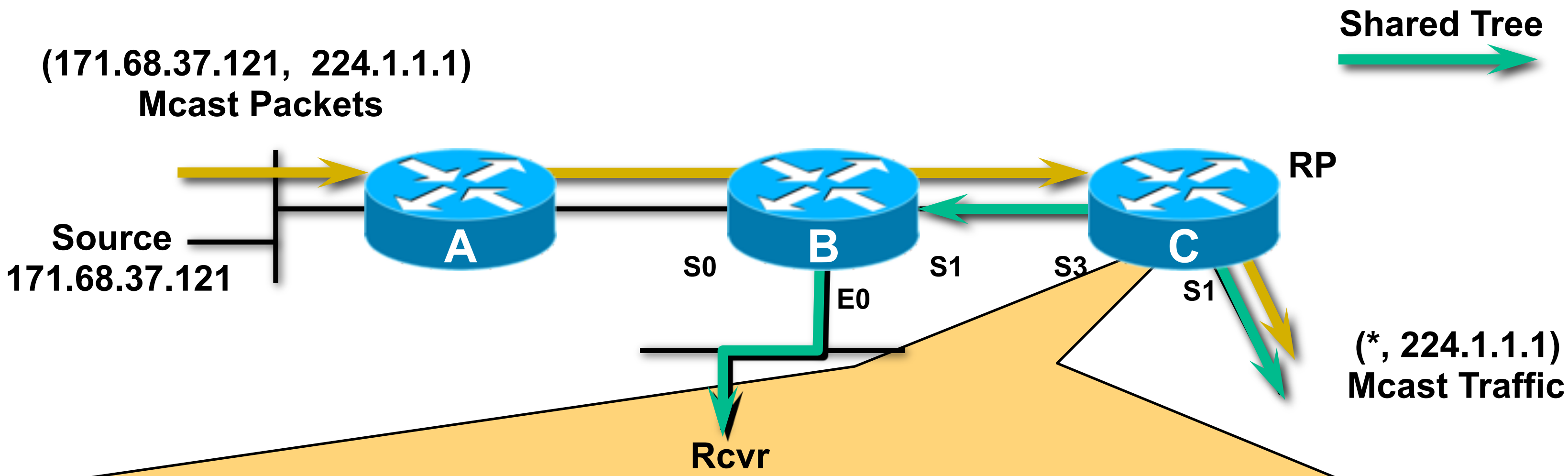
Receivers Along the SPT



2 B triggers a (*,G) Join to join the Shared Tree

PIM SM Registering

Receivers Along the SPT

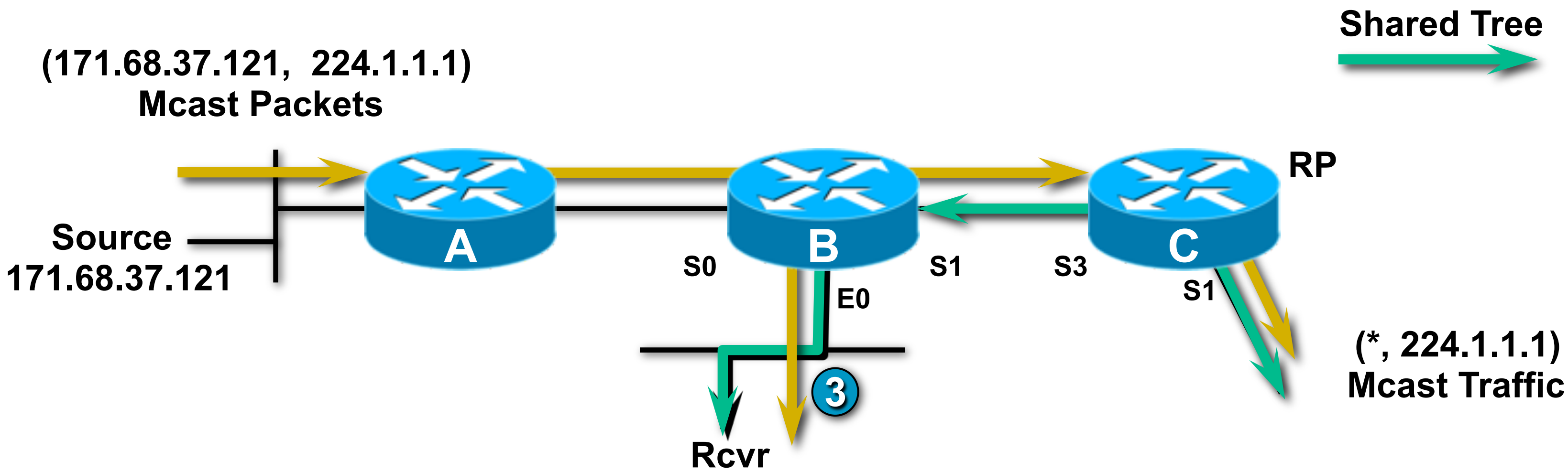



```
(*, 224.1.1.1), 00:09:21/00:00:00, RP 171.68.28.140, flags: S
  Incoming interface: Null, RPF nbr 0.0.0.0,
  Outgoing interface list:
    Serial1, Forward/Sparse-Dense, 00:03:14/00:02:46
    Serial3, Forward/Sparse-Dense, 00:00:10/00:02:50

(171.68.37.121, 224.1.1.1, 00:01:15/00:02:46, flags: T
  Incoming interface: Serial3, RPF nbr 171.68.28.139,
  Outgoing interface list:
    Serial1, Forward/Sparse-Dense, 00:00:49/00:02:11
```

State in RP After B Joins Shared Tree

PIM SM Registering

Receivers Along the SPT



3 **Group G traffic begins to flow to Rcvr.**

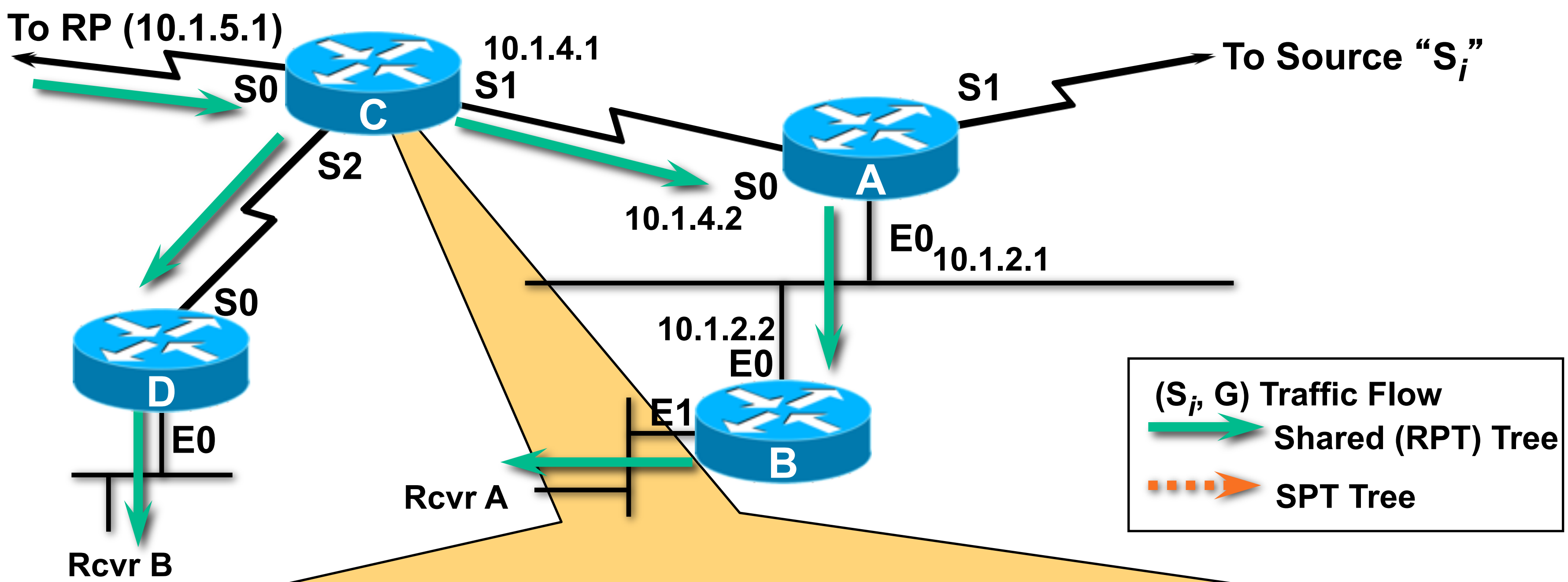
(Note: 171.68.37.121 traffic doesn’t flow to RP then back down to B)

PIM SM SPT-Switchover

PIM SM SPT-Switchover

- SPT Thresholds may be set for any Group
  - Access Lists may be used to specify which Groups
  - Default Threshold = 0kbps (I.e. immediately join SPT)
  - Threshold = "infinity" means "never join SPT"
  - **Don't use values in between "0" and "infinity"**
    - **(In IOS XR, "0" and "infinity" are the only options)**
- Threshold triggers Join of Source Tree
  - Sends an (S,G) Join up SPT for next "S" in "G" packet received

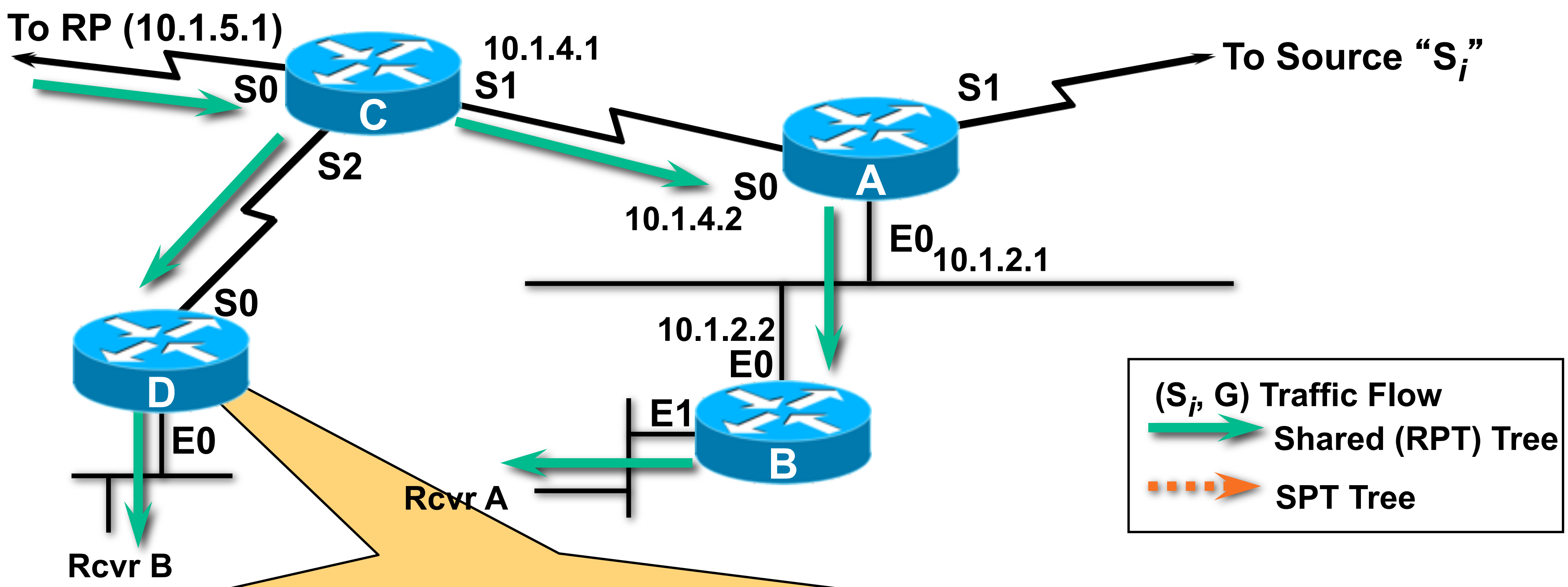
PIM SM SPT-Switchover




```
(*, 224.1.1.1), 00:01:43/00:02:13, RP 10.1.5.1, flags: S
  Incoming interface: Serial0, RPF nbr 10.1.5.1,
  Outgoing interface list:
    Serial1, Forward/Sparse-Dense, 00:01:43/00:02:11
    Serial2, Forward/Sparse-Dense, 00:00:32/00:02:28
```

State in C Before Switch

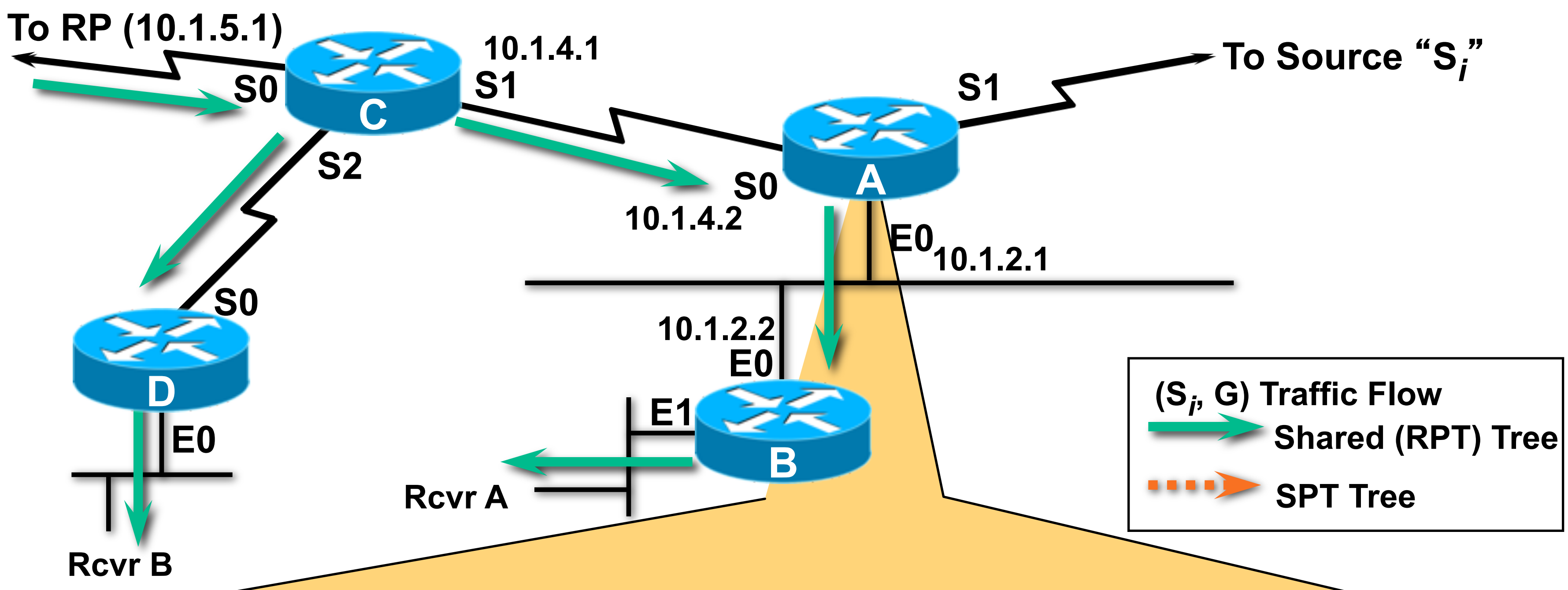
PIM SM SPT-Switchover




```
(*, 224.1.1.1), 00:01:43/00:02:13, RP 10.1.5.1, flags: SC
  Incoming interface: Serial0, RPF nbr 10.1.4.9,
  Outgoing interface list:
    Ethernet0, Forward/Sparse-Dense, 00:01:43/00:02:11
```

State in D Before Switch

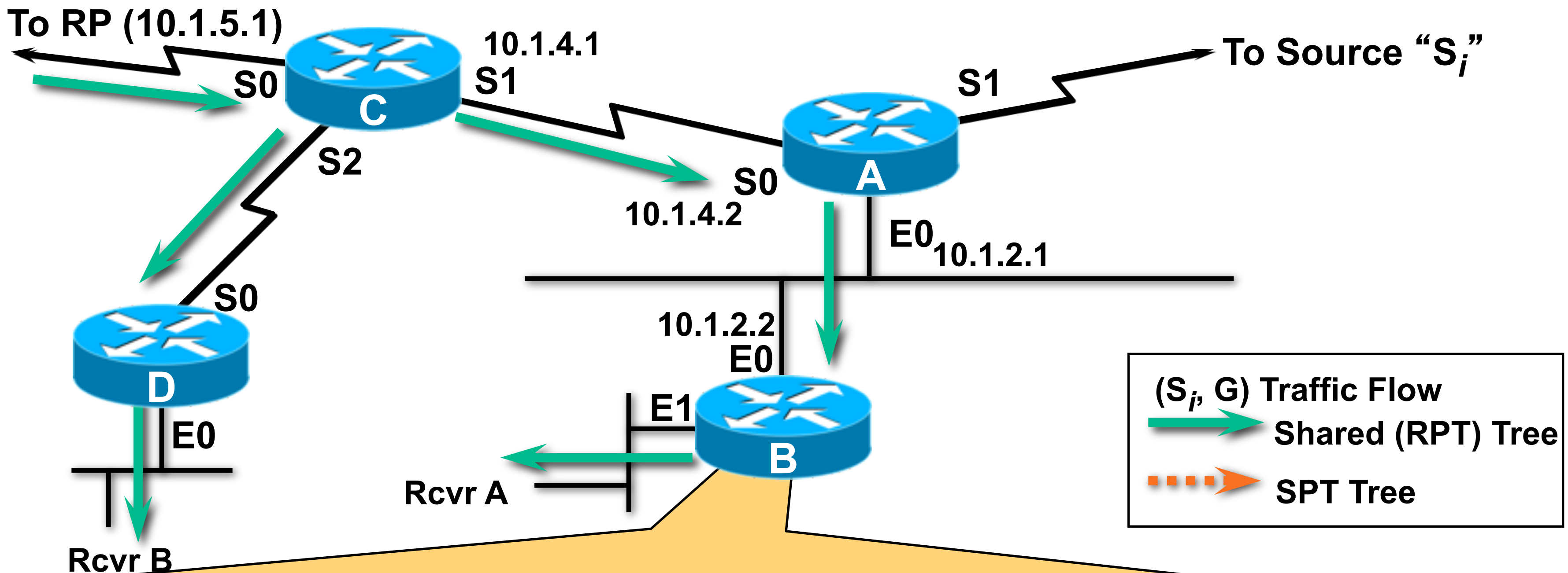
PIM SM SPT-Switchover




```
(*, 224.1.1.1), 00:01:43/00:02:13, RP 10.1.5.1, flags: S
  Incoming interface: Serial0, RPF nbr 10.1.4.1,
  Outgoing interface list:
    Ethernet0, Forward/Sparse-Dense, 00:01:43/00:02:11
```

State in A Before Switch

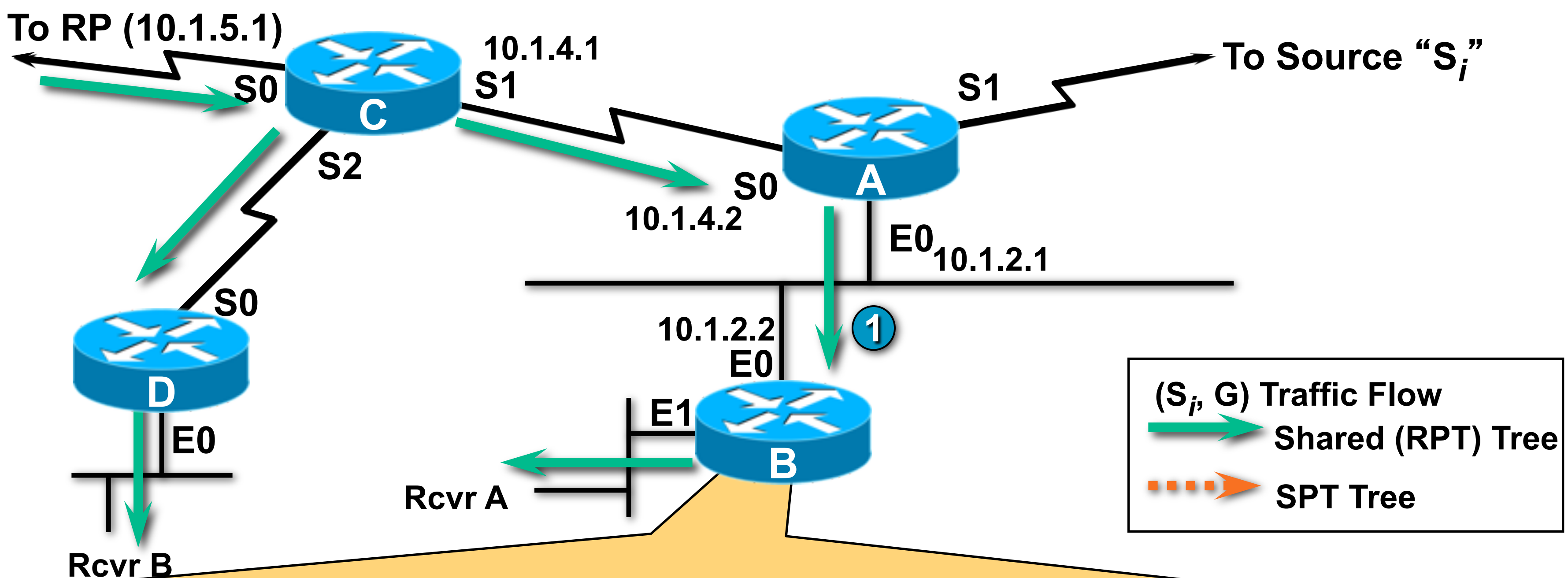
PIM SM SPT-Switchover




```
(*, 224.1.1.1), 00:01:43/00:02:13, RP 10.1.5.1, flags: SCJ
  Incoming interface: Ethernet0, RPF nbr 10.1.2.1,
  Outgoing interface list:
    Ethernet1, Forward/Sparse-Dense, 00:01:43/00:02:11
```

State in B Before Switch

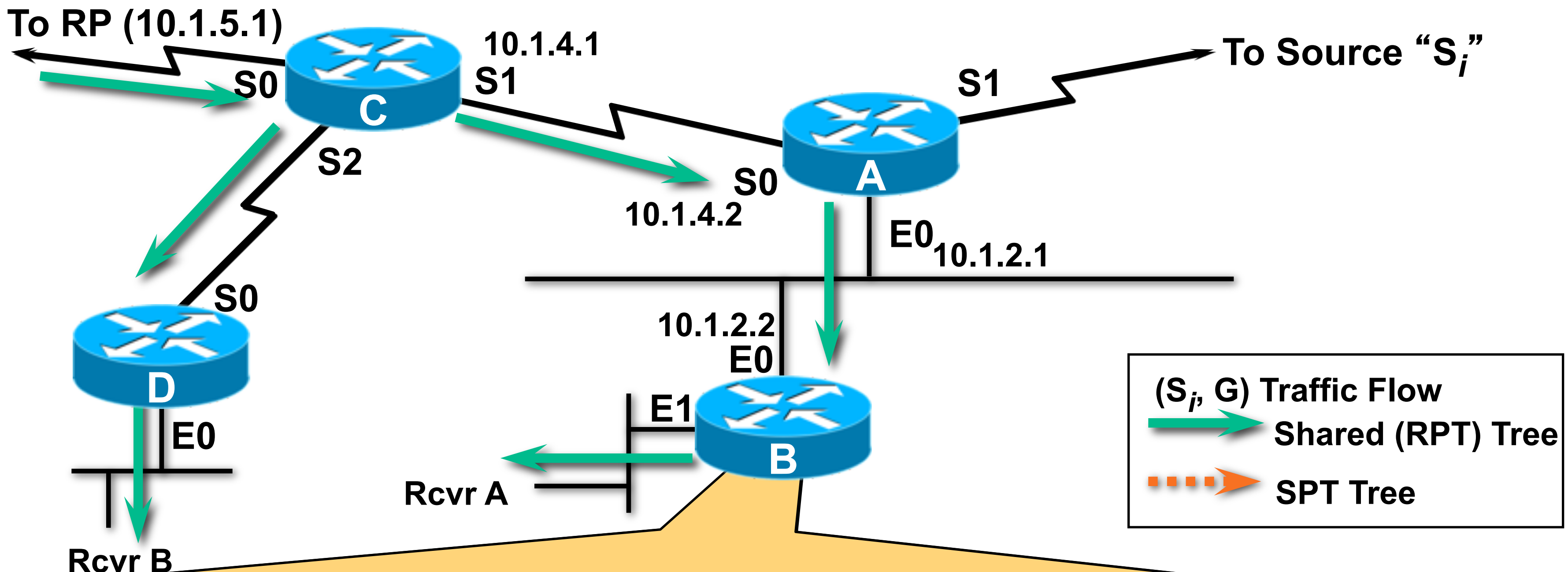
PIM SM SPT-Switchover




```
(*, 224.1.1.1), 00:01:43/00:02:13, RP 10.1.5.1, flags: SCJ
  Incoming interface: Ethernet0, RPF nbr 10.1.2.1,
  Outgoing interface list:
    Ethernet1, Forward/Sparse-Dense, 00:01:43/00:02:11
```

1. **New source (S_i,G) packet arrives down Shared tree.**

PIM SM SPT-Switchover

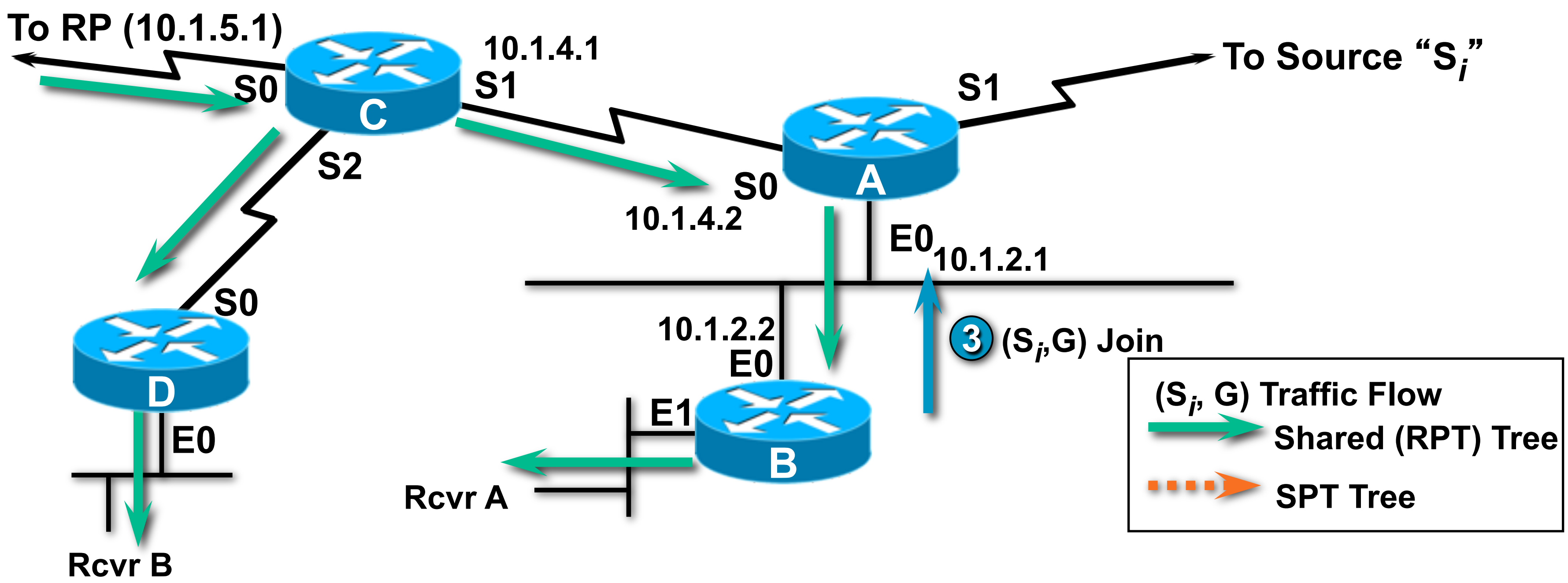



```
(*, 224.1.1.1), 00:01:43/00:00:00, RP 10.1.5.1, flags: SCJ
  Incoming interface: Ethernet0, RPF nbr 10.1.2.1,
  Outgoing interface list:
    Ethernet1, Forward/Sparse-Dense, 00:01:43/00:02:11

(171.68.37.121, 224.1.1.1), 00:00:28/00:02:51, flags: CJ
  Incoming interface: Ethernet0, RPF nbr 10.1.2.1
  Outgoing interface list:
    Ethernet1, Forward/Sparse-Dense, 00:00:28/00:02:32
```

← 2

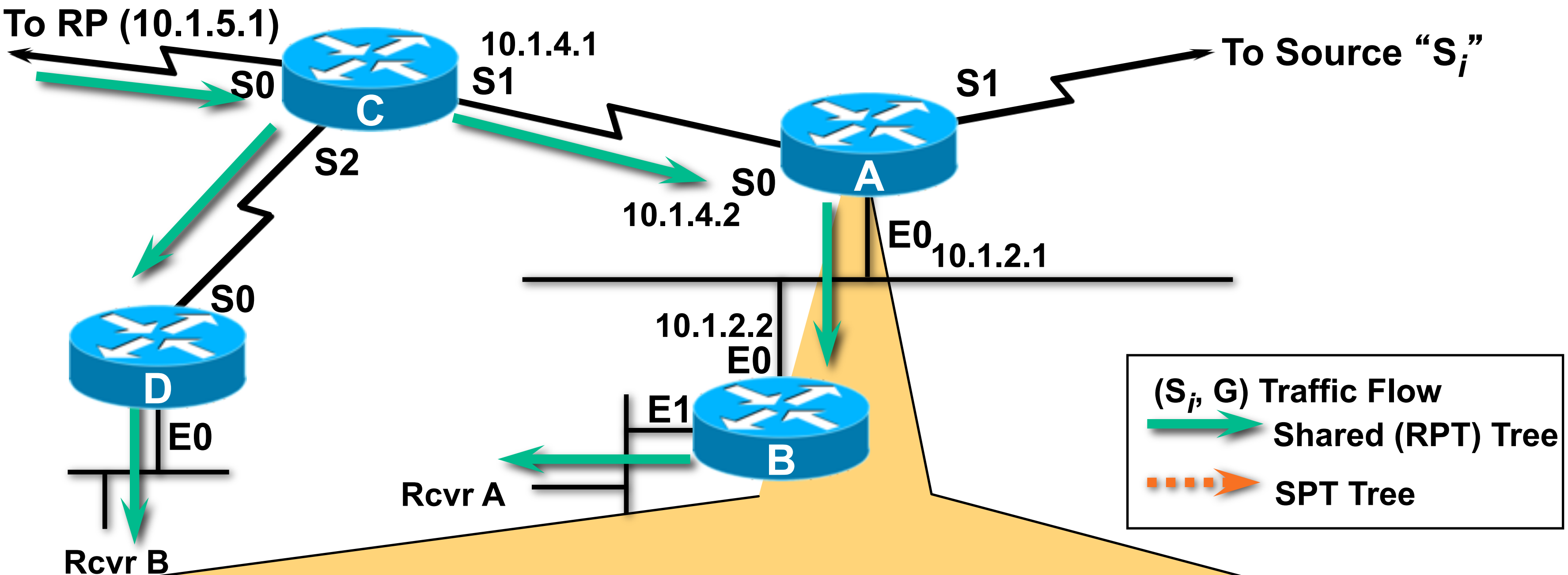
2 B creates (S_i,G) state.

PIM SM SPT-Switchover



3 B sends (S_i,G) Join towards S_i .

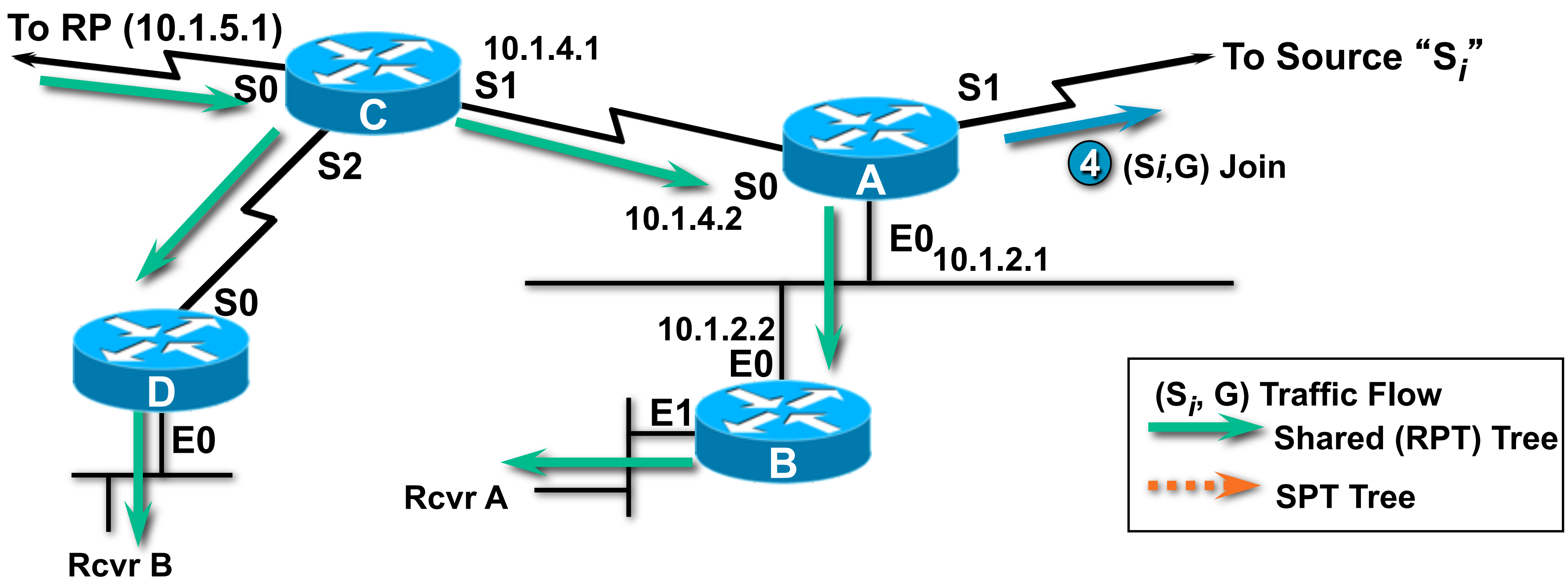
PIM SM SPT-Switchover




```
(*, 224.1.1.1), 00:01:43/00:00:00, RP 10.1.5.1, flags: S
  Incoming interface: Serial0, RPF nbr 10.1.4.1,
  Outgoing interface list:
    Ethernet0, Forward/Sparse-Dense, 00:01:43/00:02:11

(171.68.37.121, 224.1.1.1), 00:13:28/00:02:53, flags:
  Incoming interface: Serial1, RPF nbr 10.1.9.2
  Outgoing interface list:
    Ethernet0, Forward/Sparse-Dense, 00:13:25/00:02:30
```

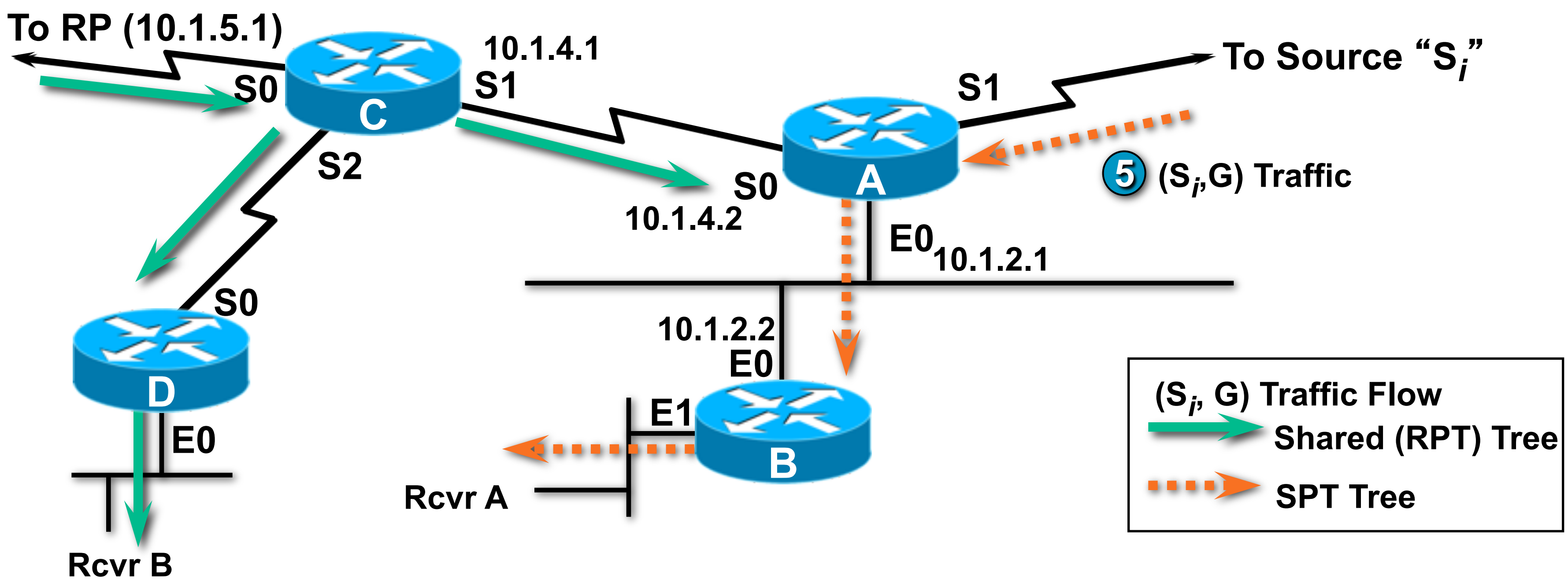
New State in A

PIM SM SPT-Switchover



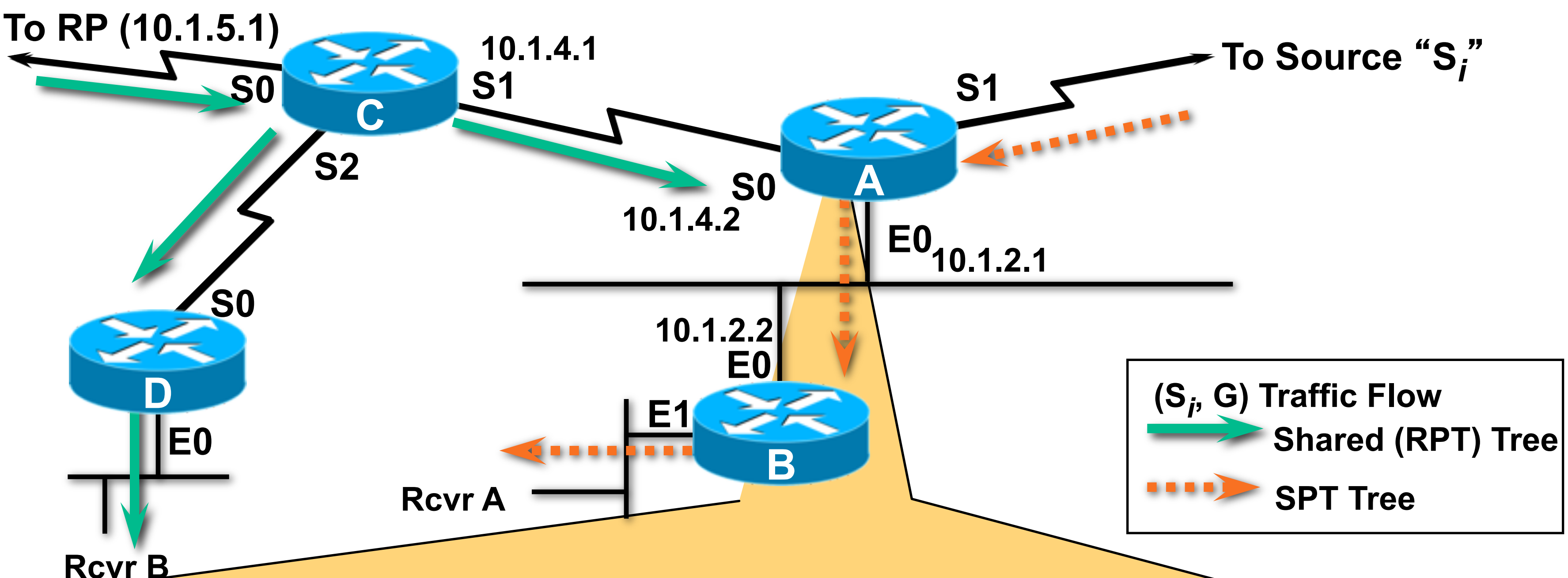
4 A triggers (S_i,G) Join toward S_i.

PIM SM SPT-Switchover



4. A triggers (S_i,G) Join toward S_i.
5. (S_i, G) traffic begins flowing down SPT tree.

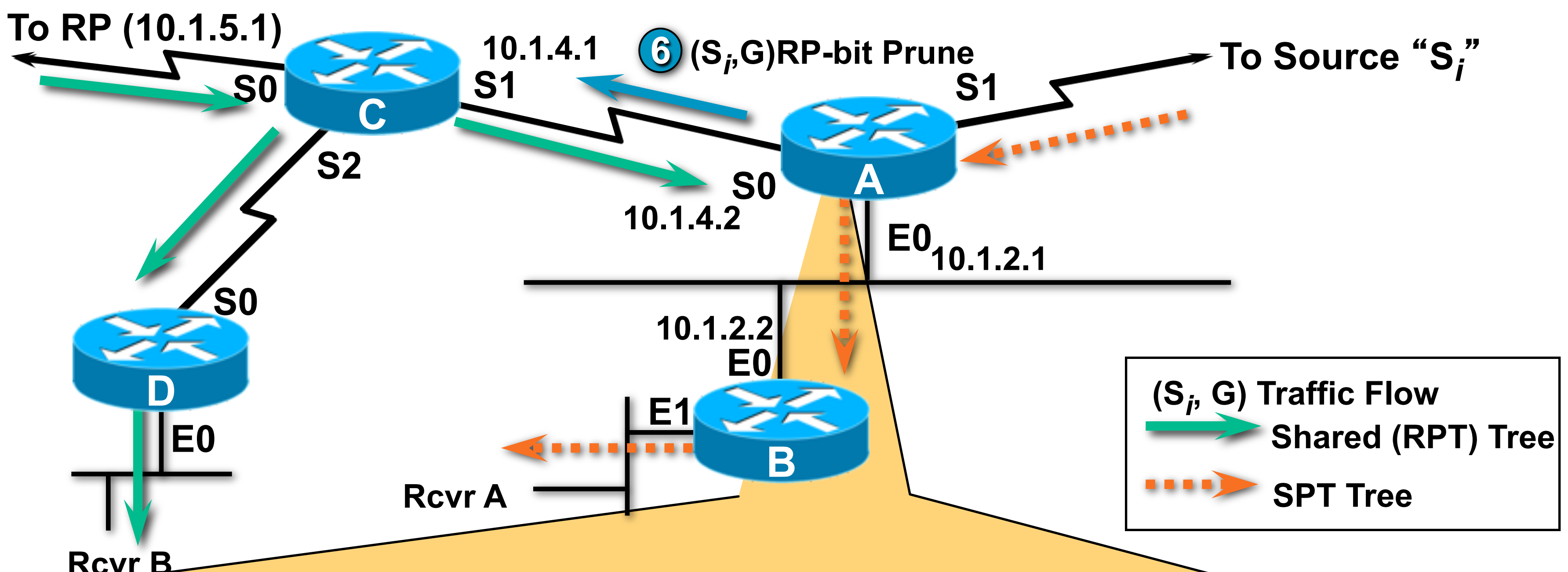
PIM SM SPT-Switchover




```
(*, 224.1.1.1), 00:01:43/00:00:00, RP 10.1.5.1, flags: S
  Incoming interface: Serial0, RPF nbr 10.1.4.1,
  Outgoing interface list:
    Ethernet0, Forward/Sparse-Dense, 00:01:43/00:02:11

(171.68.37.121, 224.1.1.1), 00:13:28/00:02:53, flags: T
  Incoming interface: Serial1, RPF nbr 10.1.9.2
  Outgoing interface list:
    Ethernet0, Forward/Sparse-Dense, 00:13:25/00:02:30
```

"T" Flag Set by Arriving Traffic on SPT

PIM SM SPT-Switchover

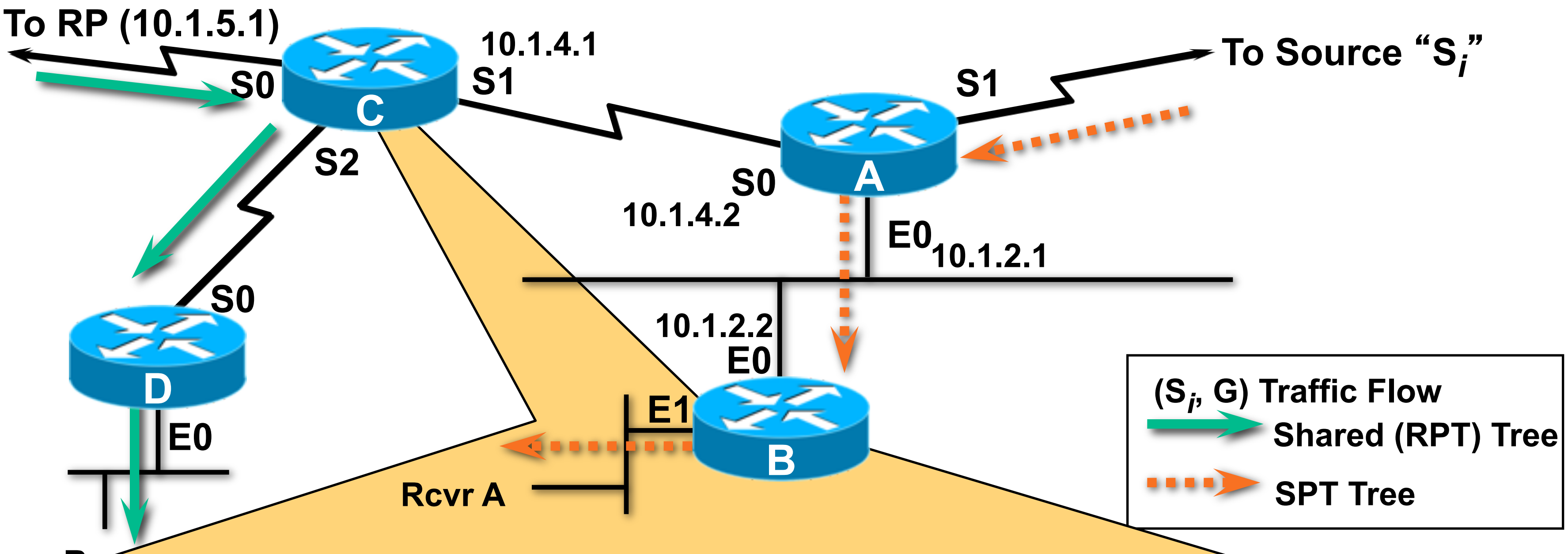



```
(*, 224.1.1.1), 00:01:43/00:00:00, RP 10.1.5.1, flags: S
  Incoming interface: Serial0, RPF nbr 10.1.4.1,
  Outgoing interface list:
    Ethernet0, Forward/Sparse-Dense, 00:01:43/00:02:11

(171.68.37.121, 224.1.1.1), 00:13:28/00:02:53, flags:T
  Incoming interface: Serial1, RPF nbr 10.1.9.2
  Outgoing interface list:
    Ethernet0, Forward/Sparse-Dense, 00:13:25/00:02:30
```

Note RPF Info Does Not Match. This Indicates SPT and RPT Diverge.

6 Once “T” flag is set, A triggers (S_i,G)RP-bit Prunes toward RP.

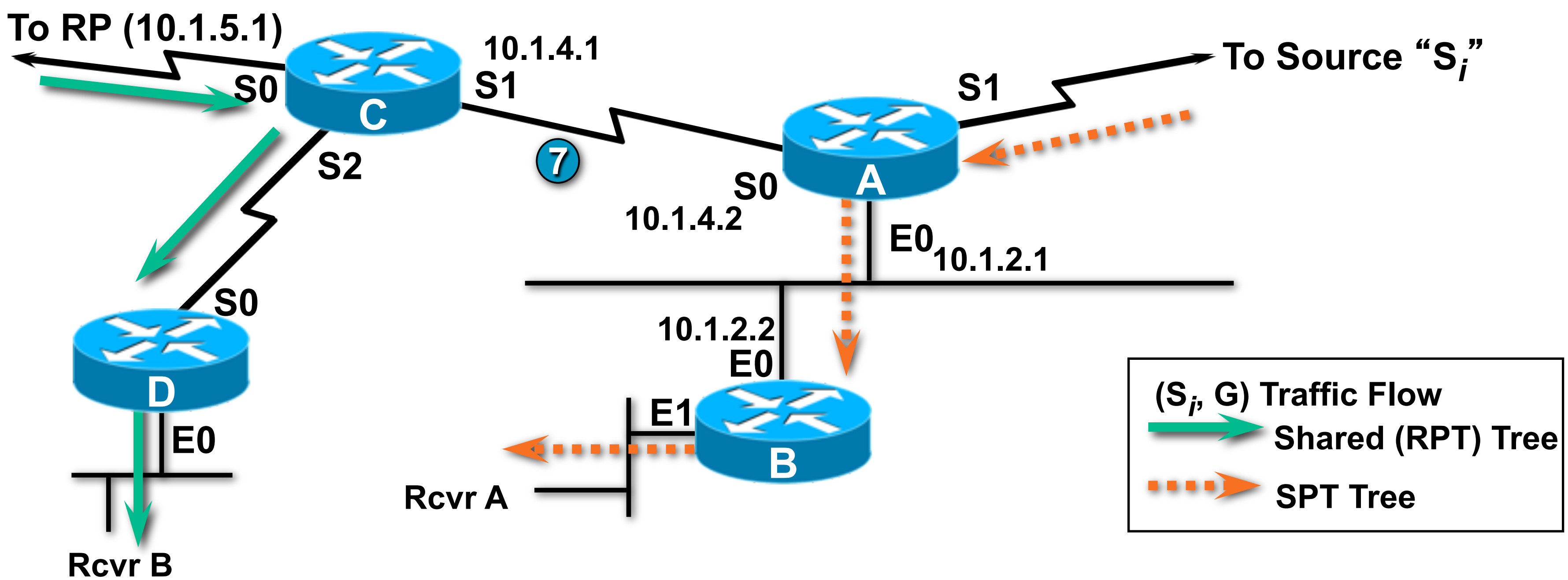
PIM SM SPT-Switchover




```
(*, 224.1.1.1), 00:01:43/00:00:00, RP 10.1.5.1, flags: S
  Incoming interface: Serial0, RPF nbr 10.1.5.1,
  Outgoing interface list:
    Serial1, Forward/Sparse-Dense, 00:01:43/00:02:11
    Serial2, Forward/Sparse-Dense, 00:00:32/00:02:28

(171.68.37.121, 224.1.1.1), 00:13:28/00:02:53, flags: R
  Incoming interface: Serial0, RPF nbr 10.1.5.1
  Outgoing interface list:
    Serial2, Forward/Sparse-Dense, 00:00:32/00:02:28
```

State in C After Receiving the (S_i, G) RP-bit Prune

PIM SM SPT-Switchover

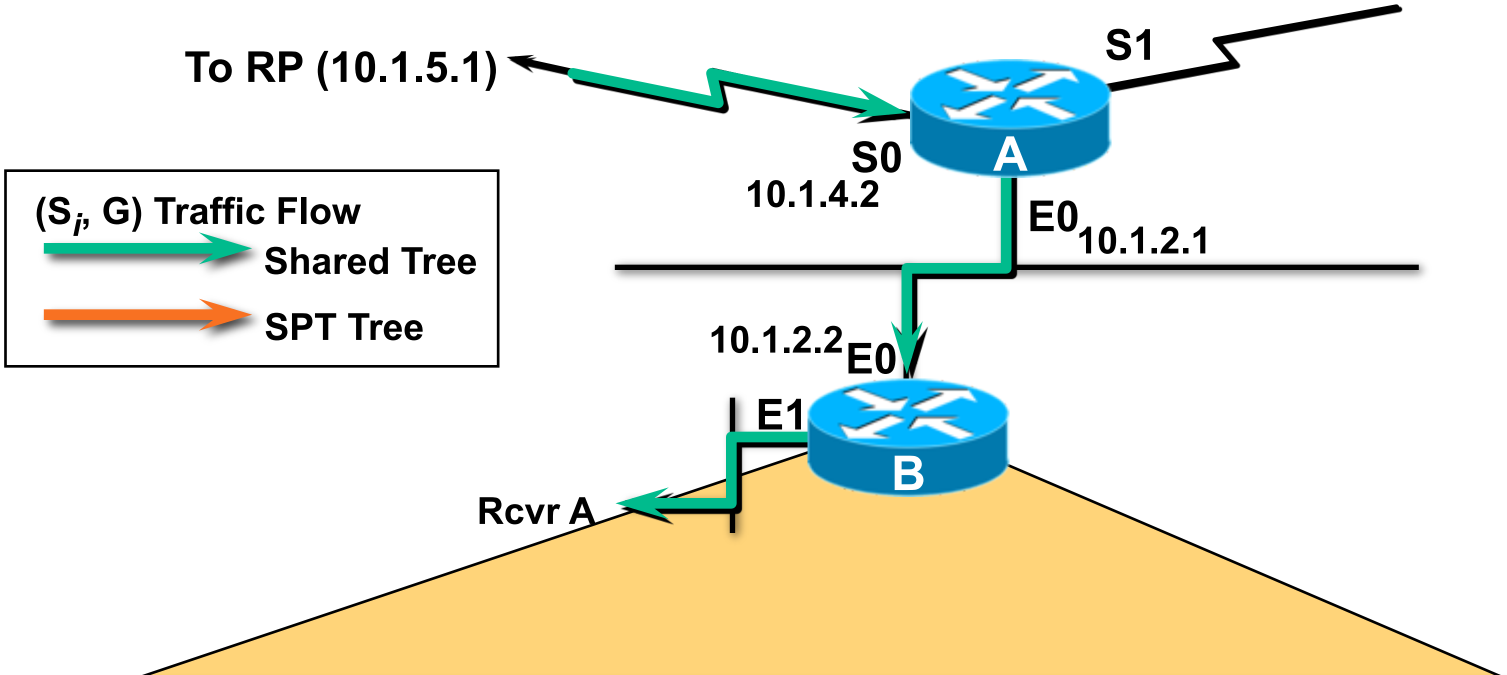


7 **Unnecessary (S_i, G) traffic is pruned from the Shared tree.**

PIM SM Pruning

PIM SM Pruning

Shared Tree Case

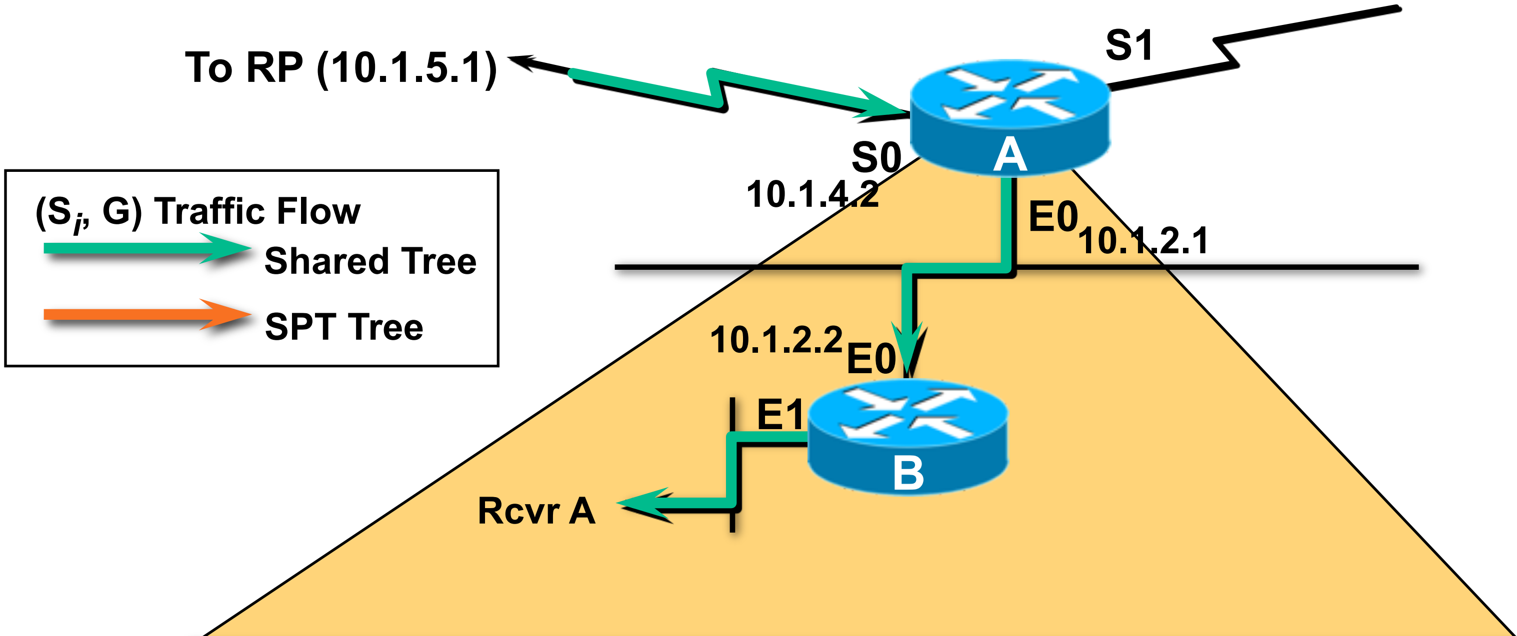



```
(*, 224.1.1.1), 00:01:43/00:02:13, RP 10.1.5.1, flags: SC
  Incoming interface: Ethernet0, RPF nbr 10.1.2.1,
  Outgoing interface list:
    Ethernet1, Forward/Sparse-Dense, 00:01:43/00:02:11
```

State in B Before Pruning

PIM SM Pruning

Shared Tree Case

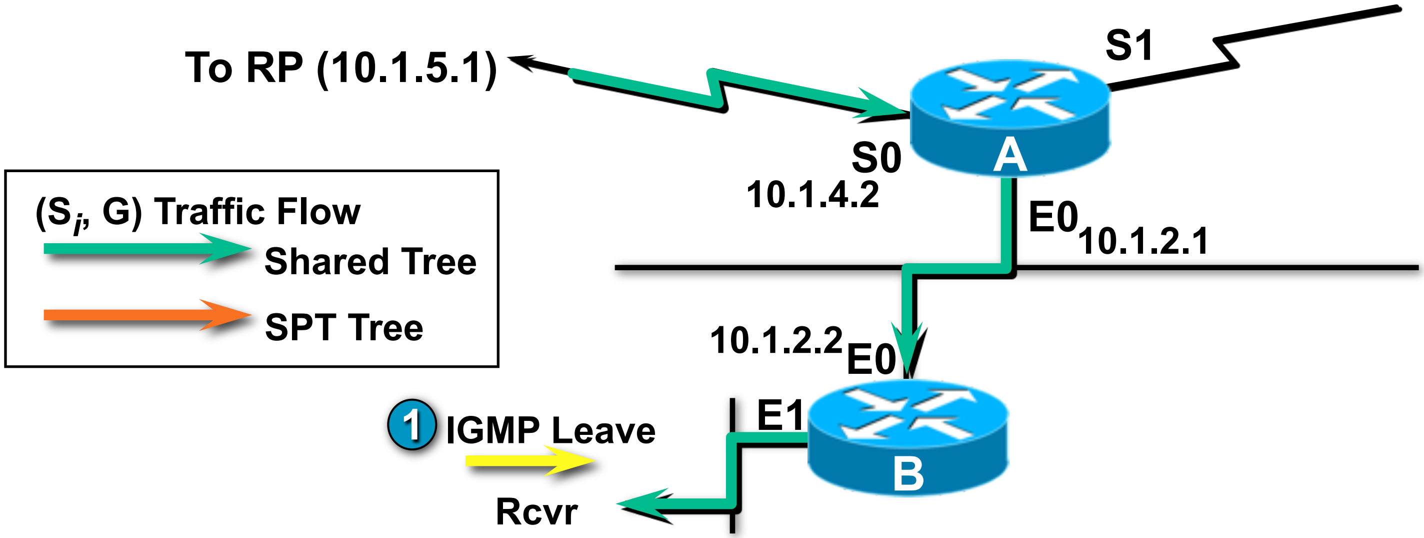



```
(*, 224.1.1.1), 00:01:43/00:02:13, RP 10.1.5.1, flags: S
  Incoming interface: Serial0, RPF nbr 10.1.4.1,
  Outgoing interface list:
    Ethernet0, Forward/Sparse-Dense, 00:01:43/00:02:11
```

State in A Before Pruning

PIM SM Pruning

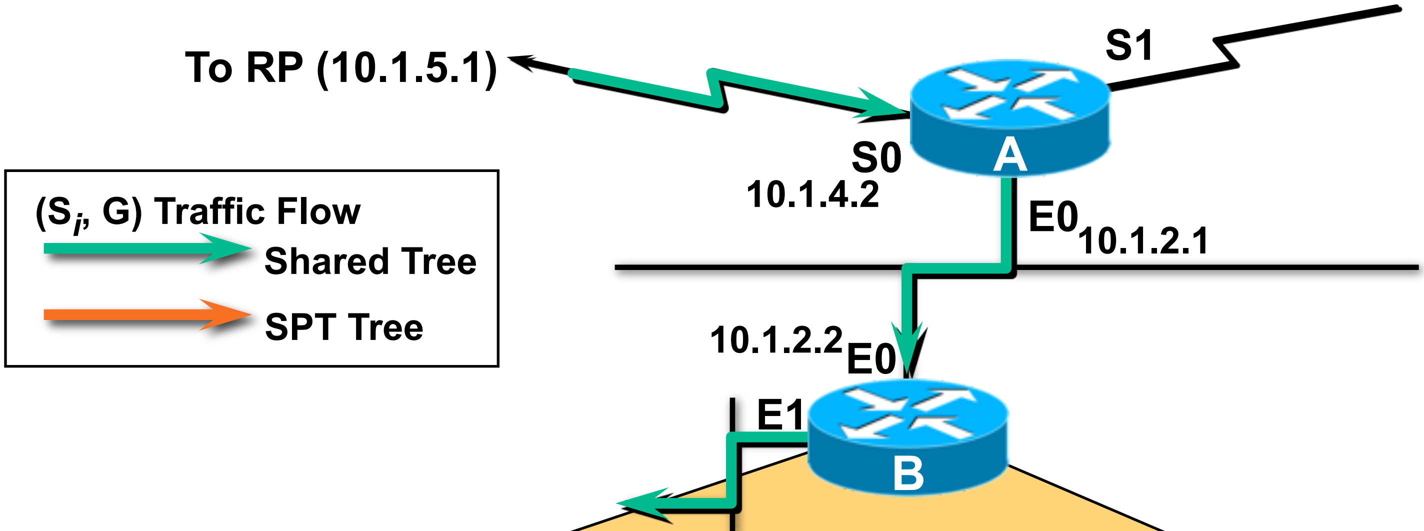
Shared Tree Case



1 B is a Leaf router. Last Rcvr, leaves group G.

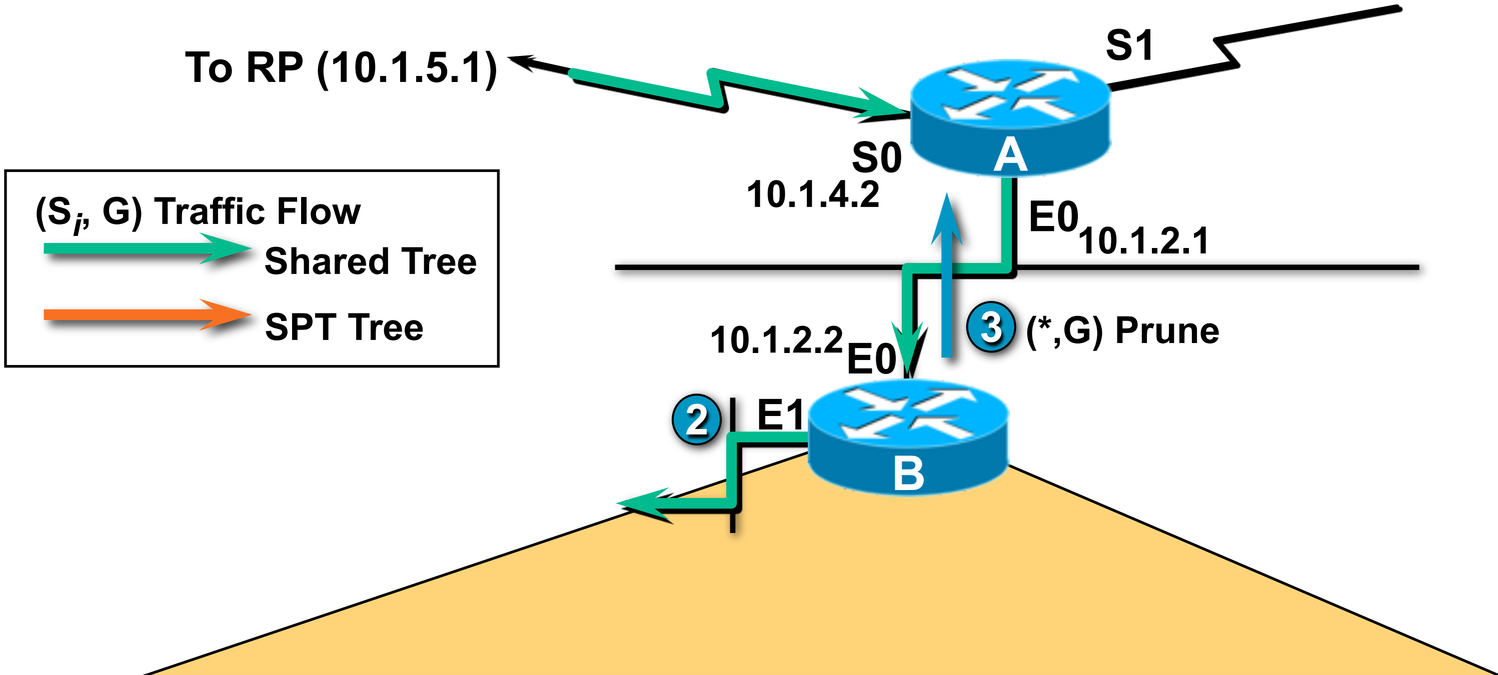
PIM SM Pruning

Shared Tree Case




```
(*, 224.1.1.1), 00:01:43/00:02:13, RP 10.1.5.1, flags: S
  Incoming interface: Ethernet0, RPF nbr 10.1.2.1,
  Outgoing interface list:
    Ethernet1, Forward/Sparse-Dense, 00:01:43/00:02:11
```

PIM SM Pruning
Shared Tree Case

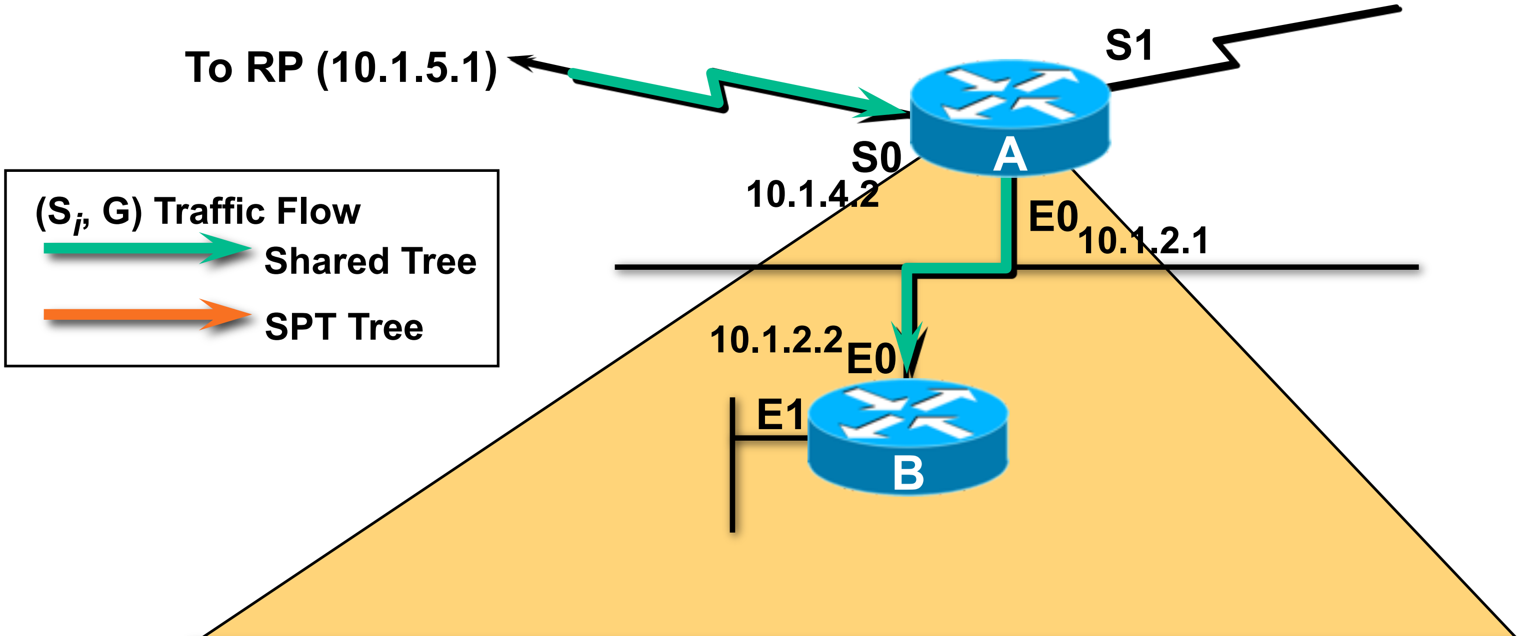



```
(*, 224.1.1.1), 00:01:43/00:02:13, RP 10.1.5.1, flags: SP
  Incoming interface: Ethernet0, RPF nbr 10.1.2.1,
  Outgoing interface list:
```

2. **B removes E1 from (*,G) and any (S_i,G) *"oilists"*.**
3. **B's (*,G) *"oilist"* now empty; triggers (*,G) Prune toward RP.**

PIM SM Pruning

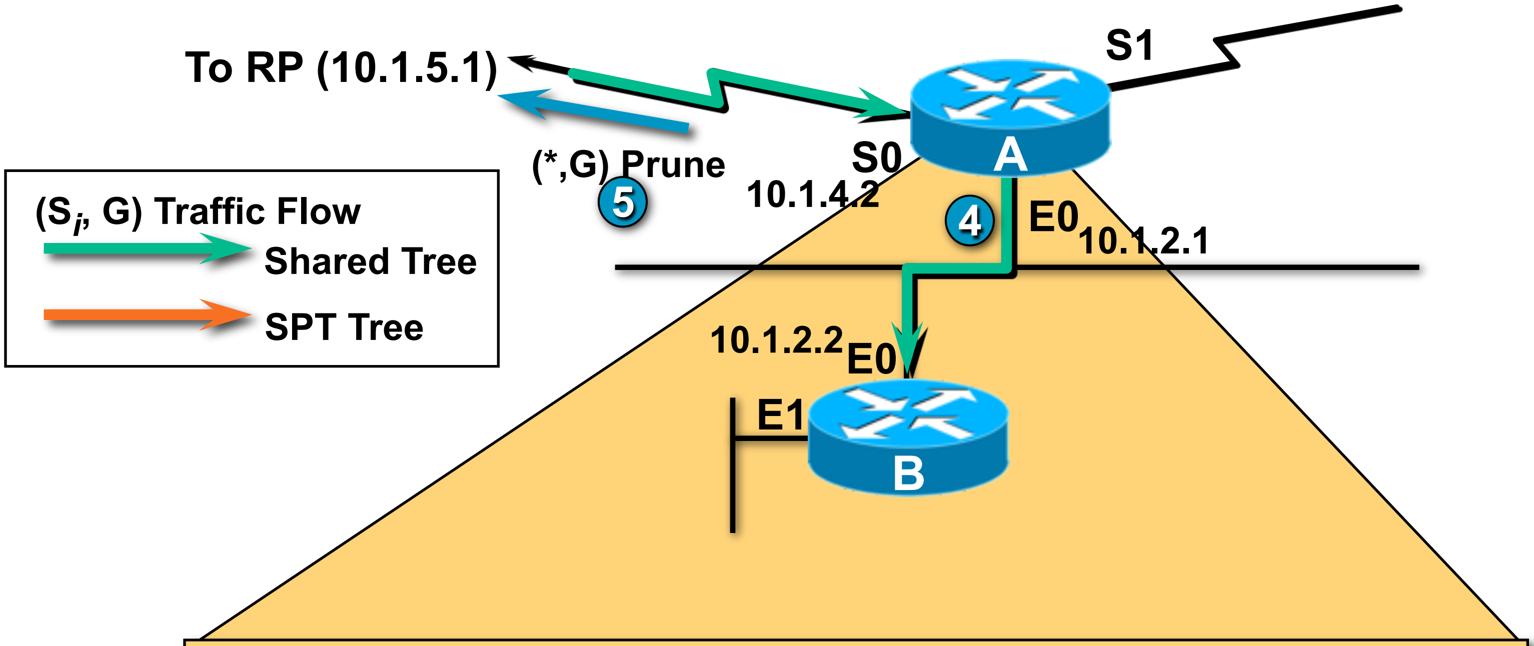
Shared Tree Case




```
(*, 224.1.1.1), 00:01:43/00:02:13, RP 10.1.5.1, flags: S
  Incoming interface: Serial0, RPF nbr 10.1.4.1,
  Outgoing interface list:
    Ethernet0, Forward/Sparse-Dense, 00:01:43/00:02:11
```

PIM SM Pruning

Shared Tree Case



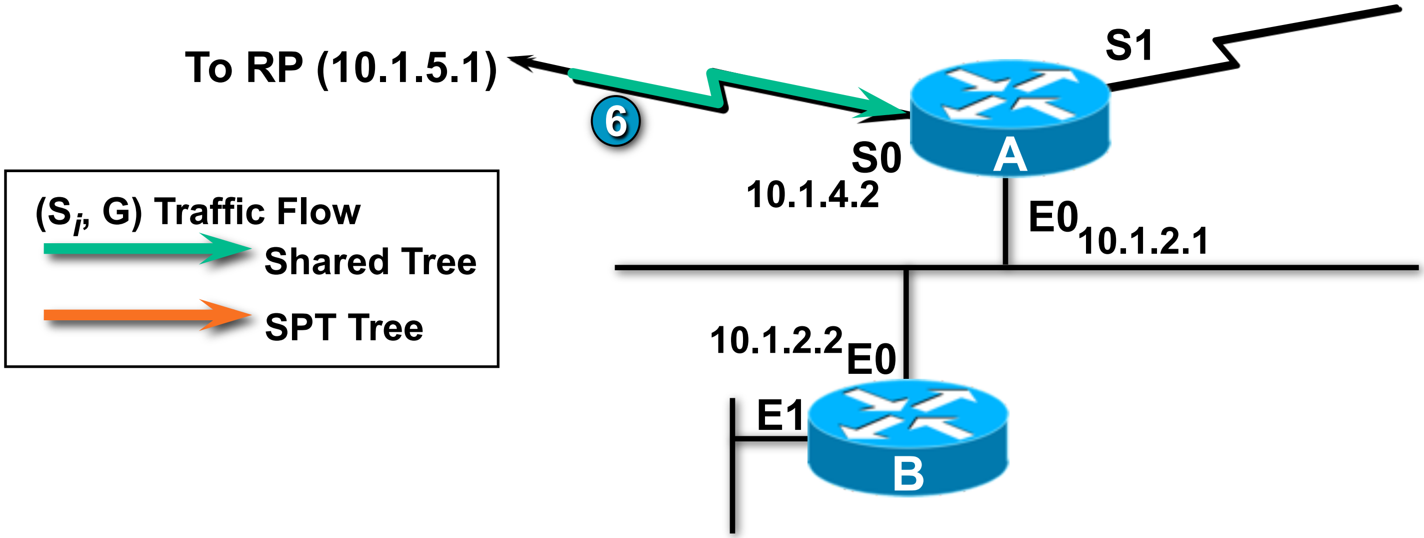

```
(*, 224.1.1.1), 00:01:43/00:02:13, RP 10.1.5.1, flags: SP
  Incoming interface: Serial0, RPF nbr 10.1.4.1,
  Outgoing interface list:
```

4 A receives Prune; removes E0 from (*,G) *"oilist"*.

(After the 3 second Multi-access Network Prune delay.)

5 A's (*,G) *"oilist"* now empty; triggers (*,G) Prune toward RP.

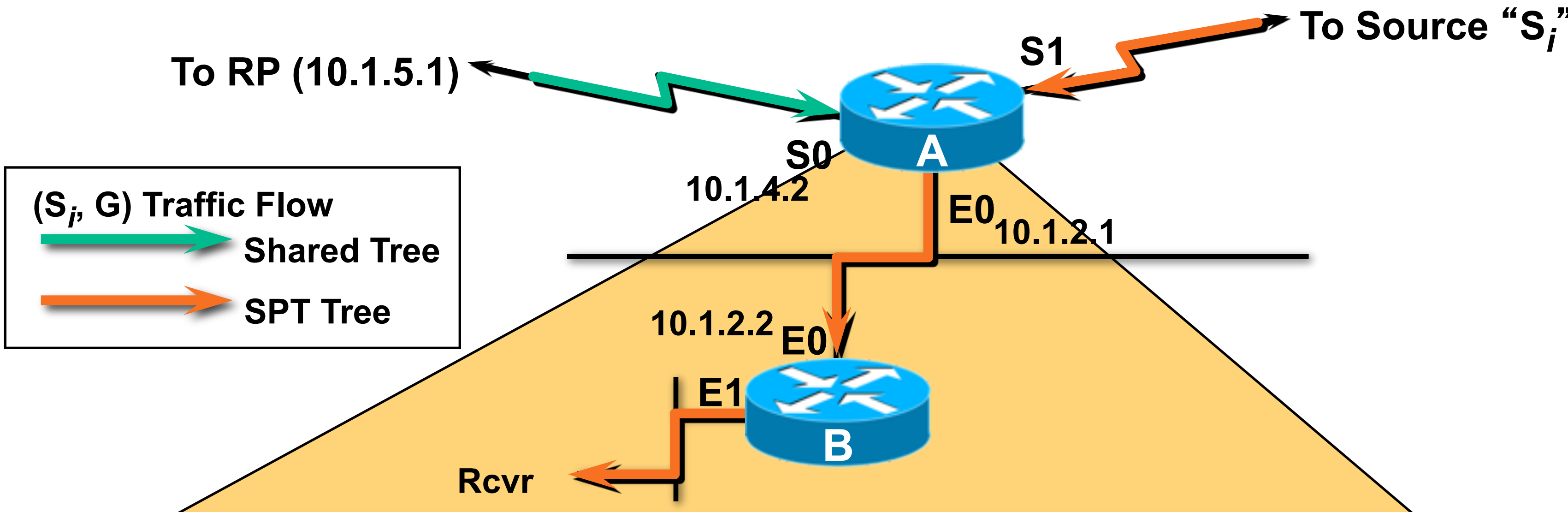
PIM SM Pruning
Shared Tree Case



6 **Pruning continues back toward RP.**

PIM SM Pruning

Source (SPT) Case

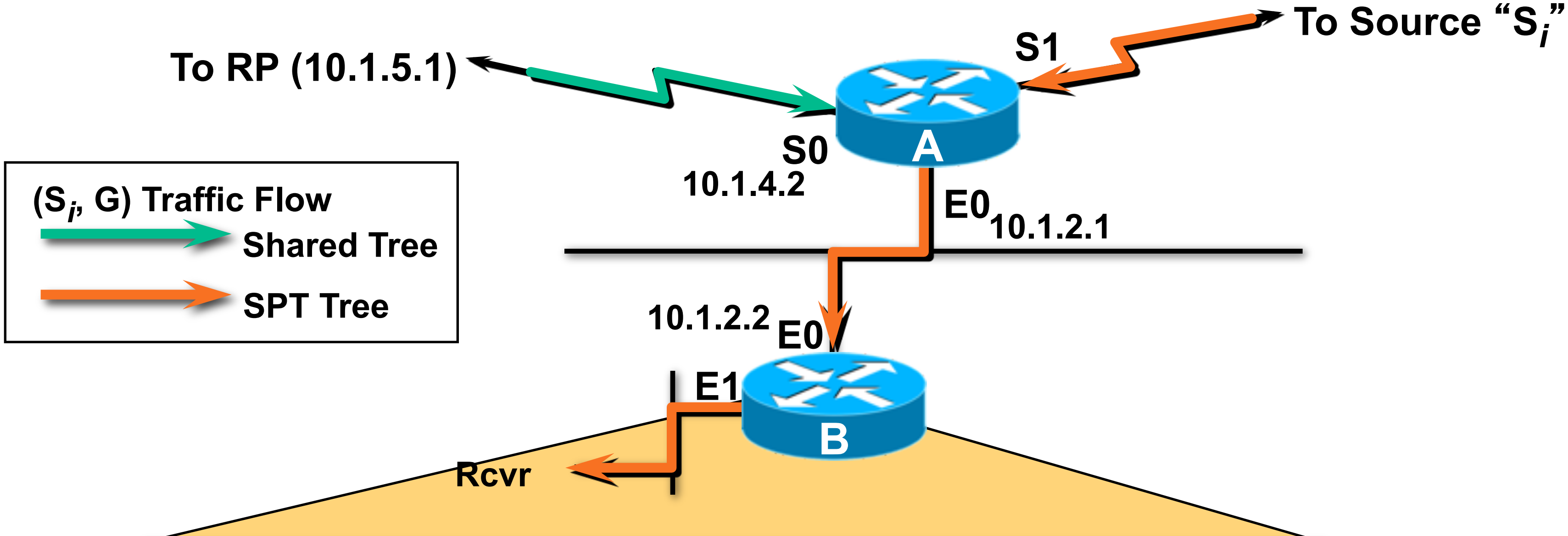



```
(*, 224.1.1.1), 00:01:43/00:00:00, RP 10.1.5.1, flags: S
  Incoming interface: Serial0, RPF nbr 10.1.4.1,
  Outgoing interface list:
    Ethernet0, Forward/Sparse-Dense, 00:01:43/00:02:11

(171.68.37.121, 224.1.1.1), 00:01:05/00:01:55, flags: T
  Incoming interface: Serial1, RPF nbr 10.1.9.2
  Outgoing interface list:
    Ethernet0, Forward/Sparse-Dense, 00:01:05/00:02:55
```

State in A Before Pruning

PIM SM Pruning

Source (SPT) Case

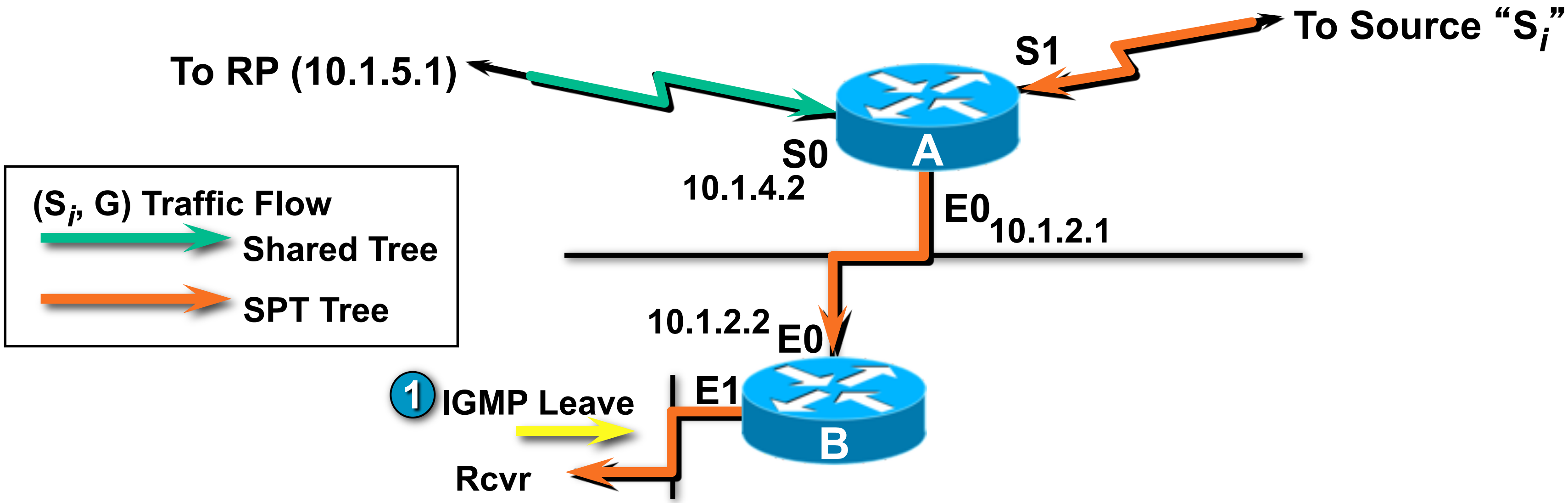



```
(*, 224.1.1.1), 00:01:43/00:00:00, RP 10.1.5.1, flags: SC
  Incoming interface: Ethernet0, RPF nbr 10.1.2.1,
  Outgoing interface list:
    Ethernet1, Forward/Sparse-Dense, 00:01:43/00:02:11

(171.68.37.121, 224.1.1.1), 00:01:05/00:01:55, flags: CJT
  Incoming interface: Ethernet0, RPF nbr 10.1.2.1
  Outgoing interface list:
    Ethernet1, Forward/Sparse-Dense, 00:01:05/00:02:55
```

State in B Before Pruning

PIM SM Pruning

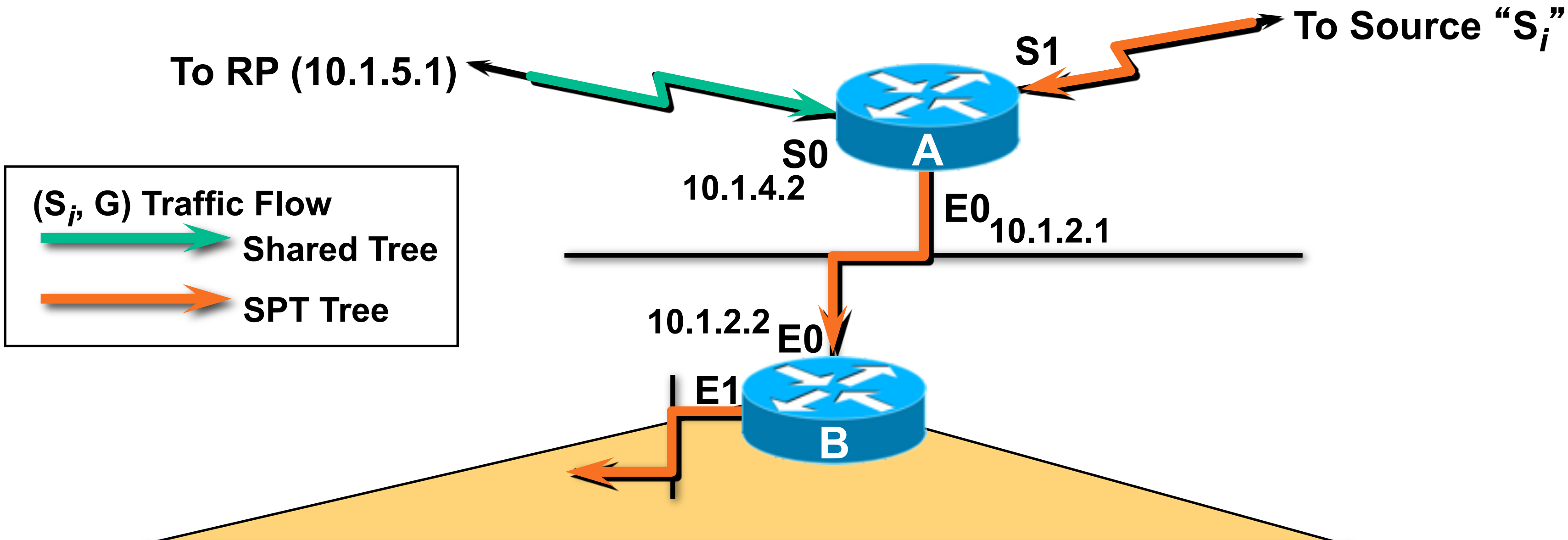
Source (SPT) Case



1 **B is a Leaf router. Last Rcvr leaves group G.**

PIM SM Pruning

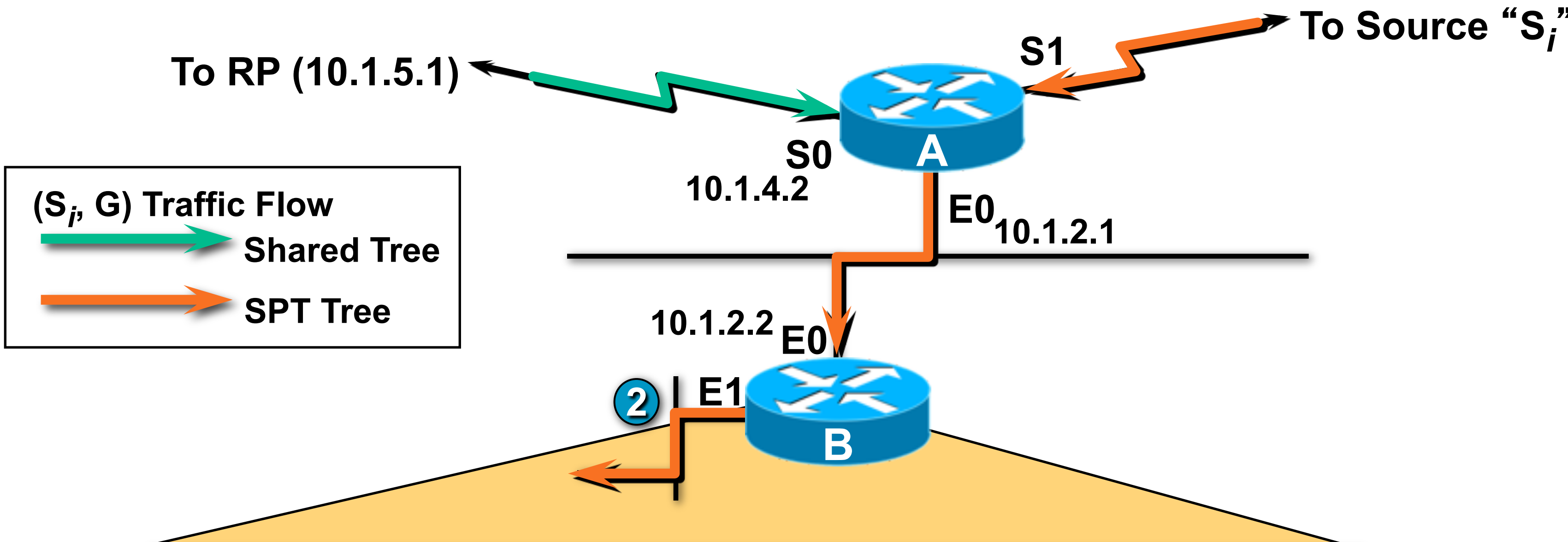
Source (SPT) Case




```
(*, 224.1.1.1), 00:01:43/00:02:59, RP 10.1.5.1, flags: SC
  Incoming interface: Ethernet0, RPF nbr 10.1.2.1,
  Outgoing interface list:
    Ethernet1, Forward/Sparse-Dense, 00:01:43/00:02:11
(171.68.37.121, 224.1.1.1), 00:01:05/00:01:55, flags: CJT
  Incoming interface: Ethernet0, RPF nbr 10.1.2.1
  Outgoing interface list:
    Ethernet1, Forward/Sparse-Dense, 00:01:05/00:02:55
```


PIM SM Pruning

Source (SPT) Case

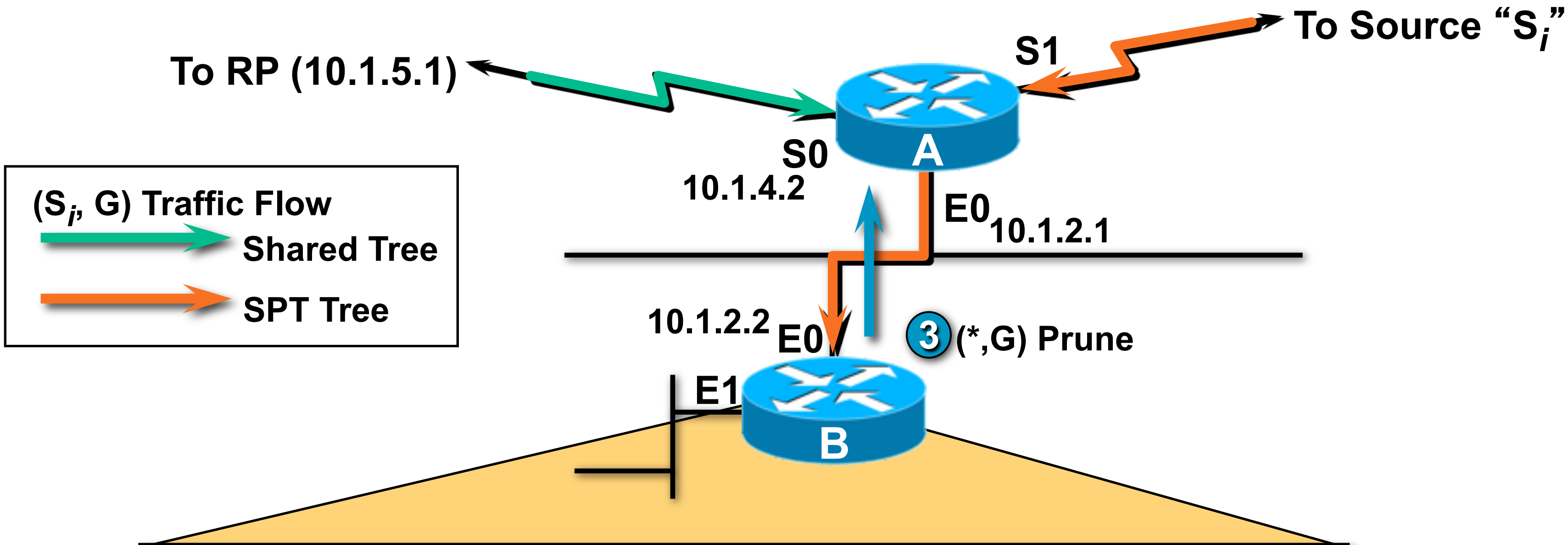



```
(*, 224.1.1.1), 00:01:43/00:00:00, RP 10.1.5.1, flags: SP
  Incoming interface: Ethernet0, RPF nbr 10.1.2.1,
  Outgoing interface list:

(171.68.37.121, 224.1.1.1), 00:01:05/00:01:55, flags: CJPT
  Incoming interface: Ethernet0, RPF nbr 10.1.2.1
  Outgoing interface list:
```

2 B removes E1 from (*,G) and all (S,G) *OILs*.

PIM SM Pruning

Source (SPT) Case

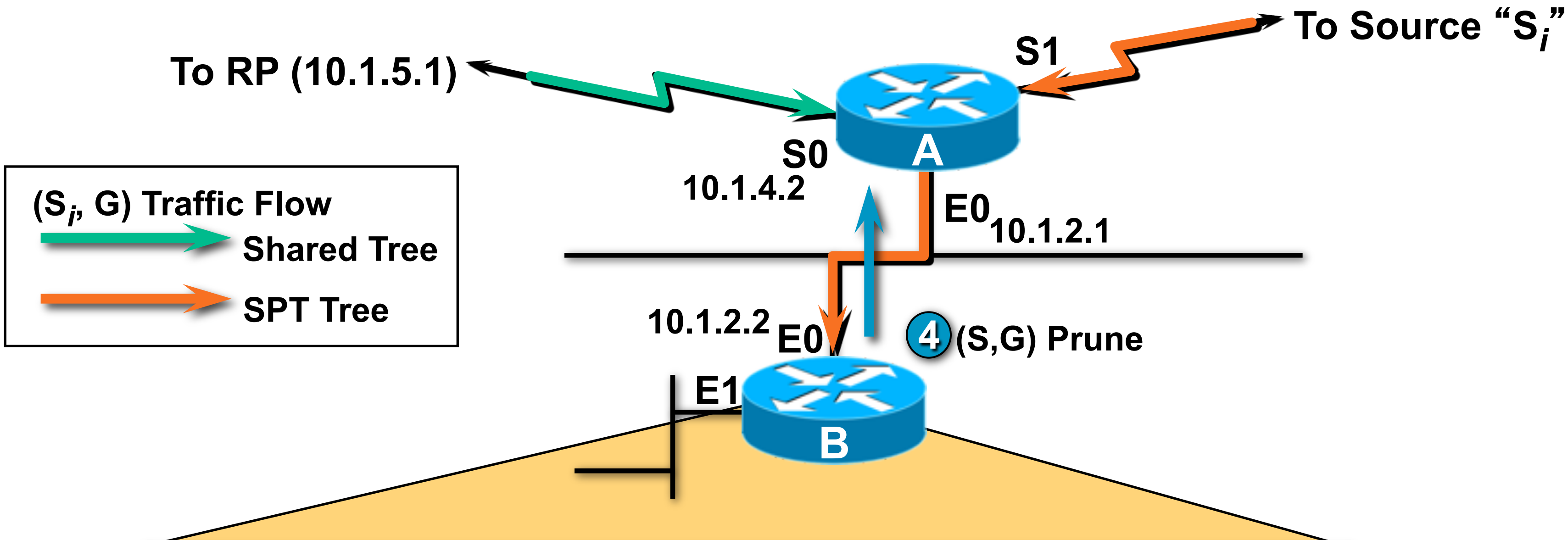



```
(*, 224.1.1.1), 00:01:43/00:00:00, RP 10.1.5.1, flags: SP
  Incoming interface: Ethernet0, RPF nbr 10.1.2.1,
  Outgoing interface list:

(171.68.37.121, 224.1.1.1), 00:01:05/00:01:55, flags: CJPT
  Incoming interface: Ethernet0, RPF nbr 10.1.2.1
  Outgoing interface list:
```

3 **B's (*,G) OIL now empty; triggers (*,G) Prune toward RP.**

PIM SM Pruning

Source (SPT) Case

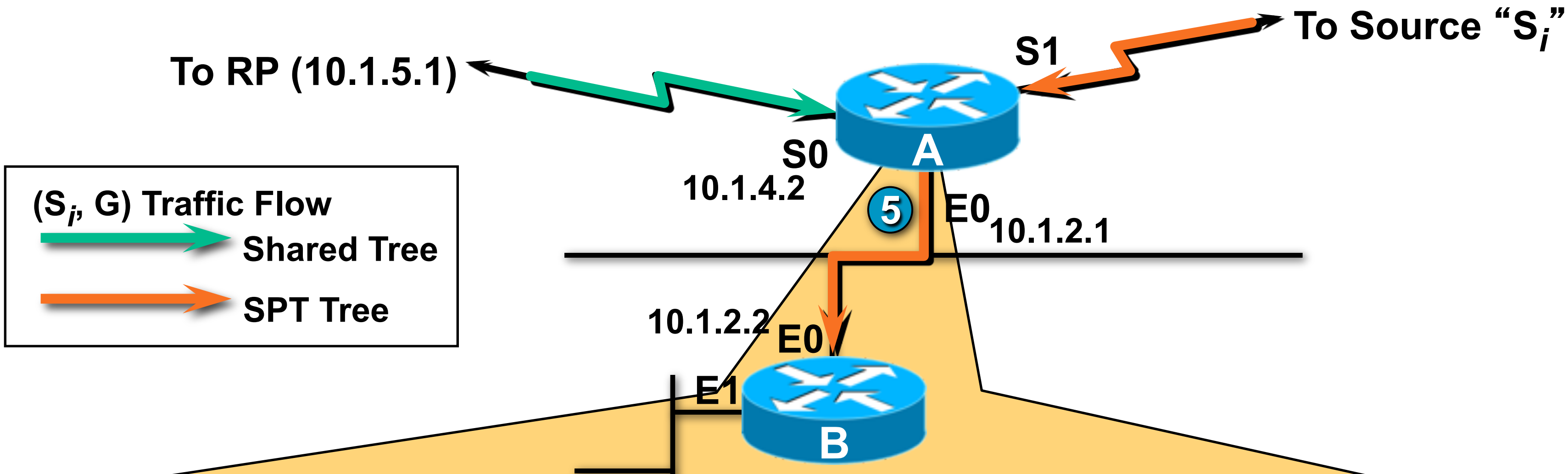



```
(*, 224.1.1.1), 00:01:43/00:00:00, RP 10.1.5.1, flags: SP
  Incoming interface: Ethernet0, RPF nbr 10.1.2.1,
  Outgoing interface list:

(171.68.37.121, 224.1.1.1), 00:01:05/00:01:55, flags: CJPT
  Incoming interface: Ethernet0, RPF nbr 10.1.2.1
  Outgoing interface list:
```

4 B’s (S,G) OIL also now empty; triggers (S, G) Prune towards Si .

PIM SM Pruning

Source (SPT) Case



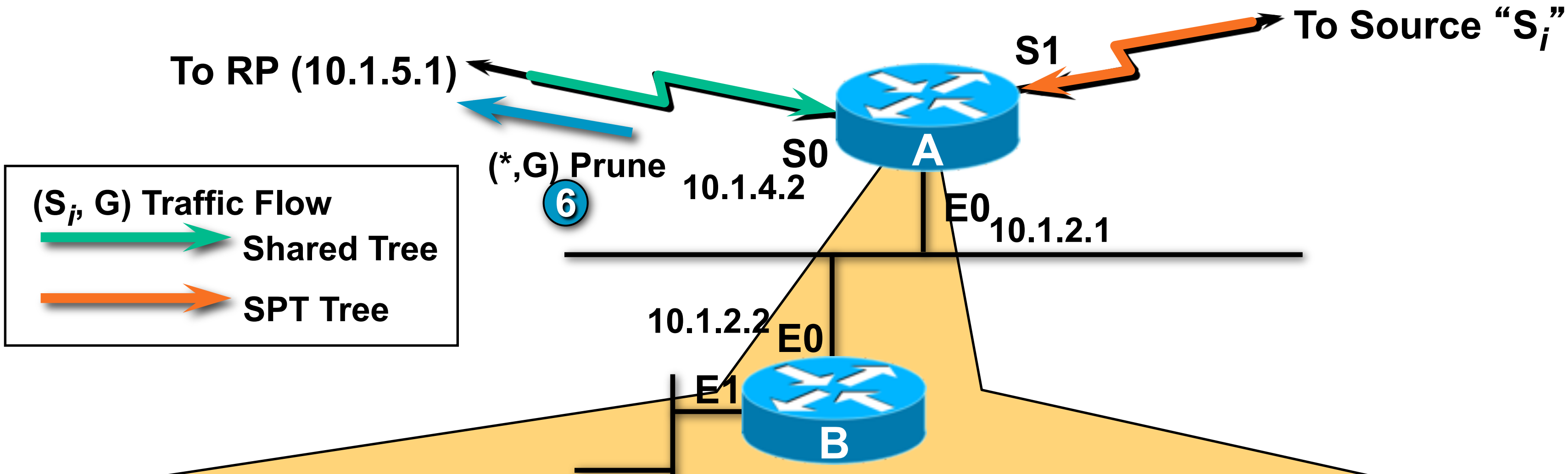

```
(*, 224.1.1.1), 00:02:32/00:00:00, RP 10.1.5.1, flags: SP
  Incoming interface: Serial0, RPF nbr 10.1.4.1,
  Outgoing interface list:

(171.68.37.121, 224.1.1.1), 00:01:56/00:00:53, flags: PT
  Incoming interface: Serial1, RPF nbr 10.1.9.2
  Outgoing interface list:
```

5 A receives (*, G) Prune; removes E0 from (*,G) & (S,G) OILs

(After the 3 second Multi-access Network Prune delay.)

PIM SM Pruning

Source (SPT) Case

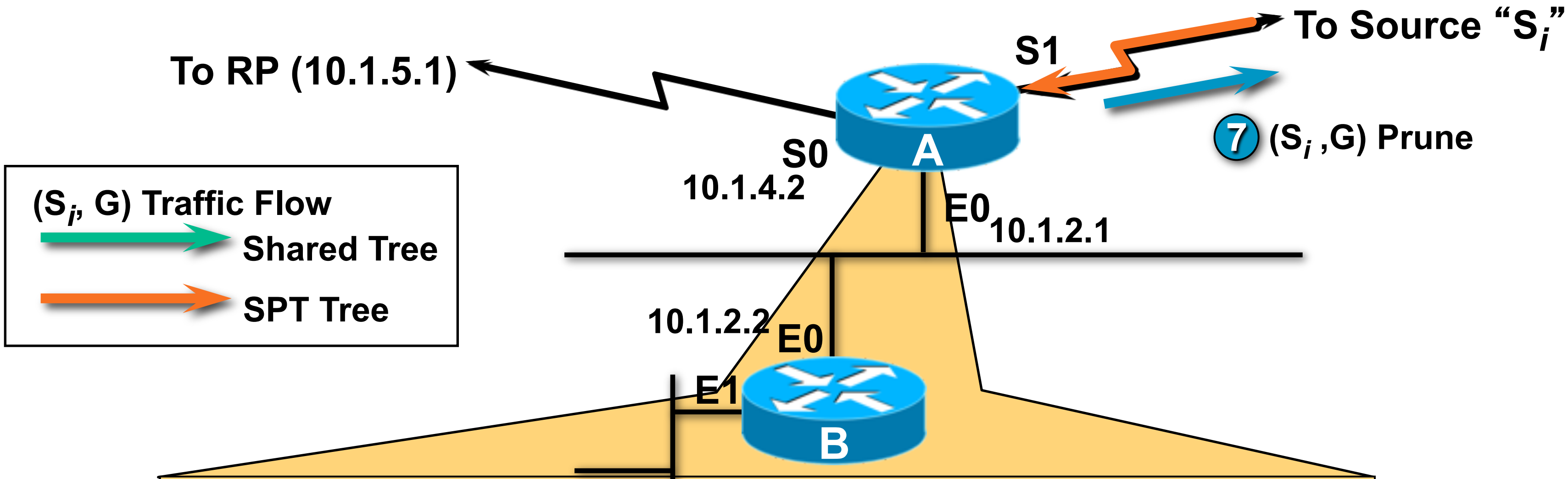



```
(*, 224.1.1.1), 00:02:32/00:00:00, RP 10.1.5.1, flags: SP
  Incoming interface: Serial0, RPF nbr 10.1.4.1,
  Outgoing interface list:

(171.68.37.121, 224.1.1.1), 00:01:56/00:00:53, flags: PT
  Incoming interface: Serial1, RPF nbr 10.1.9.2
  Outgoing interface list:
```

6 **A's (*,G) *OIL* now empty; triggers (*,G) Prune toward RP.**

PIM SM Pruning

Source (SPT) Case

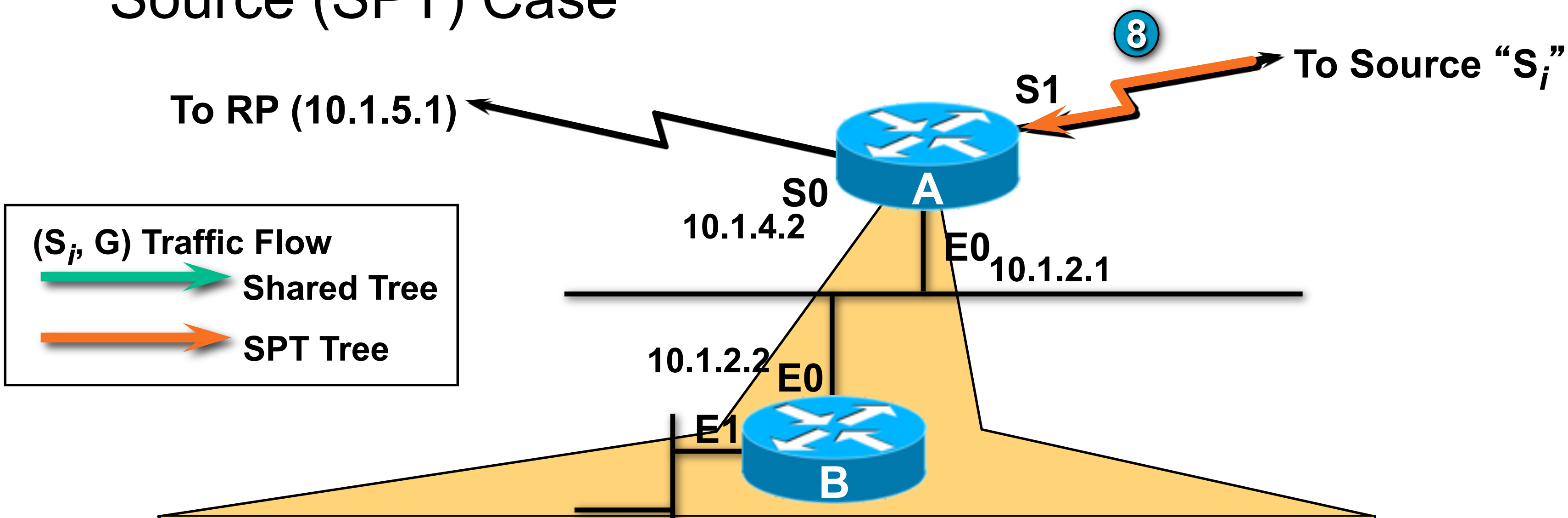



```
(*, 224.1.1.1), 00:02:32/00:00:00, RP 10.1.5.1, flags: SP
  Incoming interface: Serial0, RPF nbr 10.1.4.1,
  Outgoing interface list:

(171.68.37.121, 224.1.1.1), 00:01:56/00:00:53, flags: PT
  Incoming interface: Serial1, RPF nbr 10.1.9.2
  Outgoing interface list:
```

7 **A's (S,G) *OIL* also now empty; triggers (S,G) Prune towards S_i.**

PIM SM Pruning

Source (SPT) Case




```
(*, 224.1.1.1), 00:02:32/00:00:00, RP 10.1.5.1, flags: SP
  Incoming interface: Serial0, RPF nbr 10.1.4.1,
  Outgoing interface list:

(171.68.37.121, 224.1.1.1), 00:01:56/00:00:53, flags: PT
  Incoming interface: Serial1, RPF nbr 10.1.9.2
  Outgoing interface list:
```

8 **(S_i,G) traffic ceases flowing down SPT.**

Recap: Common Multicast Flags—IOS

- S: Sparse Mode (in contrast to D for Dense Mode)
- s: SSM; only seen on (S,G) entries
- B: Bidir
- F: Register; set on first-hop router
- P: Prune; entry has an empty OIL
- J: Join-SPT; (*,G) traffic exceeds SPT Threshold
- T: SPT; set on (S,G) entries after SPT join
- L: Local; router should receive and process this traffic
- C: Connected; seen primarily with IGMP

Source Specific Multicast

Barriers to Multicast Deployment

- Global Multicast Address Allocation
  - Dynamic Address Allocation
    - No adequate dynamic address allocation methods exist
    - SDR—Doesn't scale
    - MASC—Long ways off!
  - Static Address Allocation (GLOP)
    - Based on AS number
    - Insufficient address space for large Content Providers
- Multicast Content "Jammers"
  - Undesirable sources on a multicast group
    - "Capt. Midnight" sources bogus data/noise to group
    - Can cause DoS attack by congesting low speed links

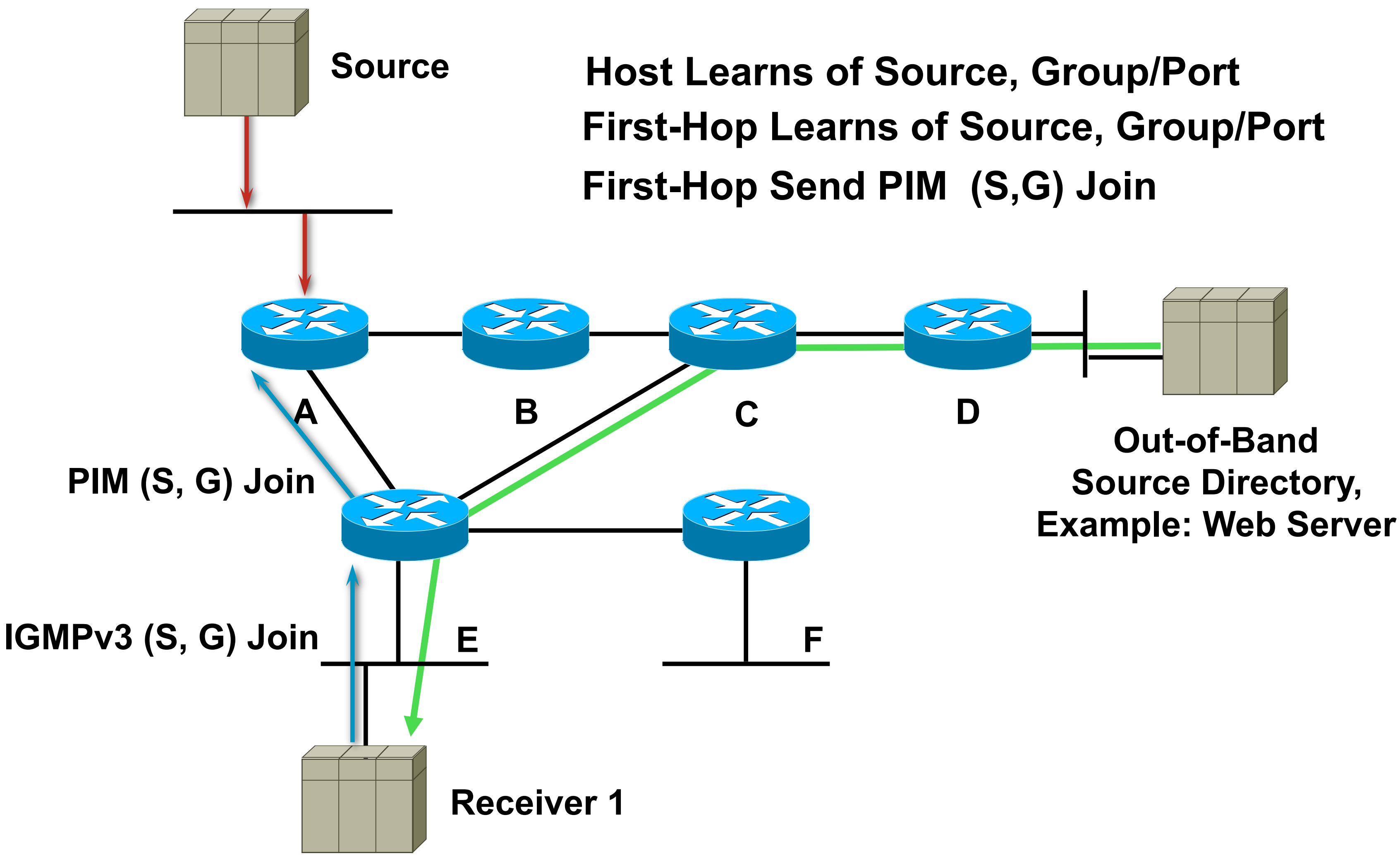
Source Specific Multicast (SSM)

- Uses Source Trees only
- Assumes one-to-many model
  - Most Internet multicast fits this model
  - IP/TV also fits this model
- Hosts responsible for source discovery
  - Typically via some out-of-band mechanism
    - Web page, Content Server, etc.
  - Eliminates need for RP and Shared Trees
  - Eliminates need for MSDP

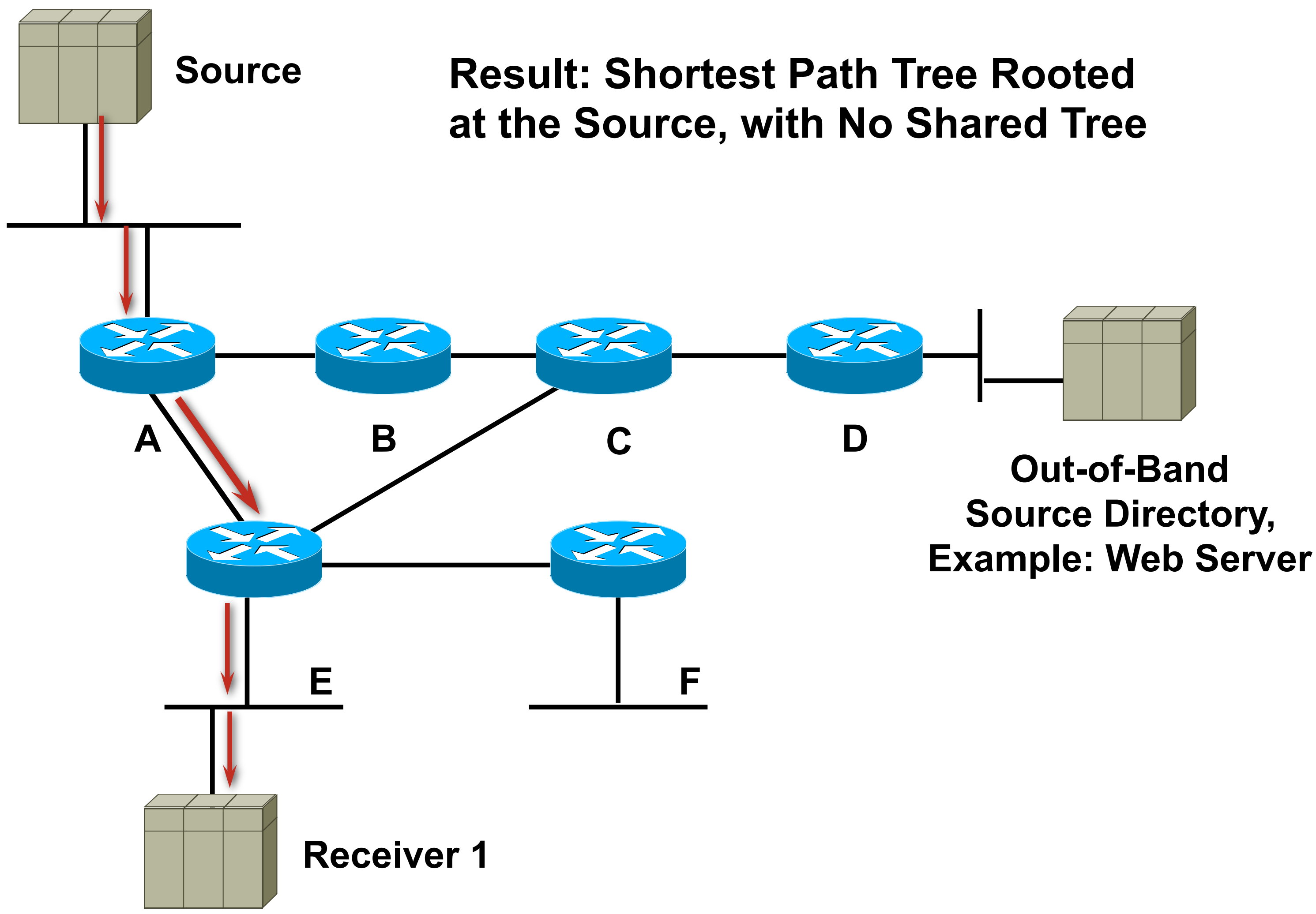
SSM Overview

- Hosts join a specific source within a group
  - Content identified by specific (S,G) instead of (*,G)
  - Hosts responsible for learning (S,G) information
- Last-hop router sends (S,G) join toward source
  - Shared Tree is never Joined or used
  - Eliminates possibility of content Jammers
  - Only specified (S,G) flow is delivered to host
- Eliminates Networked-Based Source Discovery
  - No RPs for SSM groups
- Simplifies address allocation
  - Dissimilar content sources can use same group without fear of interfering with each other

SSM Example


Source
Host Learns of Source, Group/Port
First-Hop Learns of Source, Group/Port
First-Hop Send PIM (S,G) Join
A
B
C
D
Out-of-Band
Source Directory,
Example: Web Server
PIM (S, G) Join
IGMPv3 (S, G) Join
E
F
Receiver 1

SSM Example



Source
Result: Shortest Path Tree Rooted at the Source, with No Shared Tree
A
B
C
D
Out-of-Band Source Directory, Example: Web Server
E
F
Receiver 1

SSM Configuration

- Global command

  `ip pim ssm {default | range <acl>}`

  Defines SSM address range

    Default range = 232.0.0.0/8

    Use ACL for other ranges

  Prevents Shared Tree Creation

    (*, G) Joins never sent or processed

    PIM Registers never sent or processed

  Available in Cisco IOS versions

    12.1(5)T, 12.2, 12.0(15)S, 12.1(8)E

SSM—Summary

- Uses Source Trees only
  - Hosts are responsible for source and group discovery
  - Hosts must signal router which (S,G) to join
- Solves multicast address allocation problems
  - Flows differentiated by **both** source and group
  - Content providers can use same group ranges
    - Since each (S,G) flow is unique
- Helps prevent certain DoS attacks
  - “Bogus” source traffic:
    - Can’t consume network bandwidth
    - Not received by host application

Bidirectional (BiDir) PIM

Multicast Application Categories

- One-to-many applications
  - Video, TV, radio, concerts, stock ticker, etc.
- Few-to-few applications
  - Small (<10 member) video/audio conferences
- Few-to-many applications
  - TIBCO RV servers (publishing)
- Many-to-many applications
  - Stock trading floors, gaming
- Many-to-few applications
  - TIBCO RV clients (subscriptions)

Multicast Application Categories

PIM-SM (S, G) State

- One-to-many applications
  - Single (S,G) entry
- Few-to-few applications
  - Few (<10 typical) (S,G) entries
- Few-to-many applications
  - Few (<10 typical) (S,G) entries
- Many-to-many applications
  - **Unlimited (S,G) entries**
- Many-to-few applications
  - **Unlimited (S,G) entries**

Many-to-Any State Problem

- Creates huge amounts of (S,G) state
  - State maintenance workloads skyrocket
    - High OIL fan-outs make the problem worse
  - Router performance begins to suffer
- Using Shared-Trees only
  - Provides some (S,G) state reduction
    - Results in (S,G) state only along SPT to RP
    - Frequently still too much (S,G) state
    - Need a solution that only uses (*,G) state

Bidirectional (BiDir) PIM

- Idea:

  Use the same tree for traffic from sources towards RP and from RP to receivers

- Benefits:

  Less state in routers

    Only (*, G) state is used

    Source traffic follows the Shared Tree

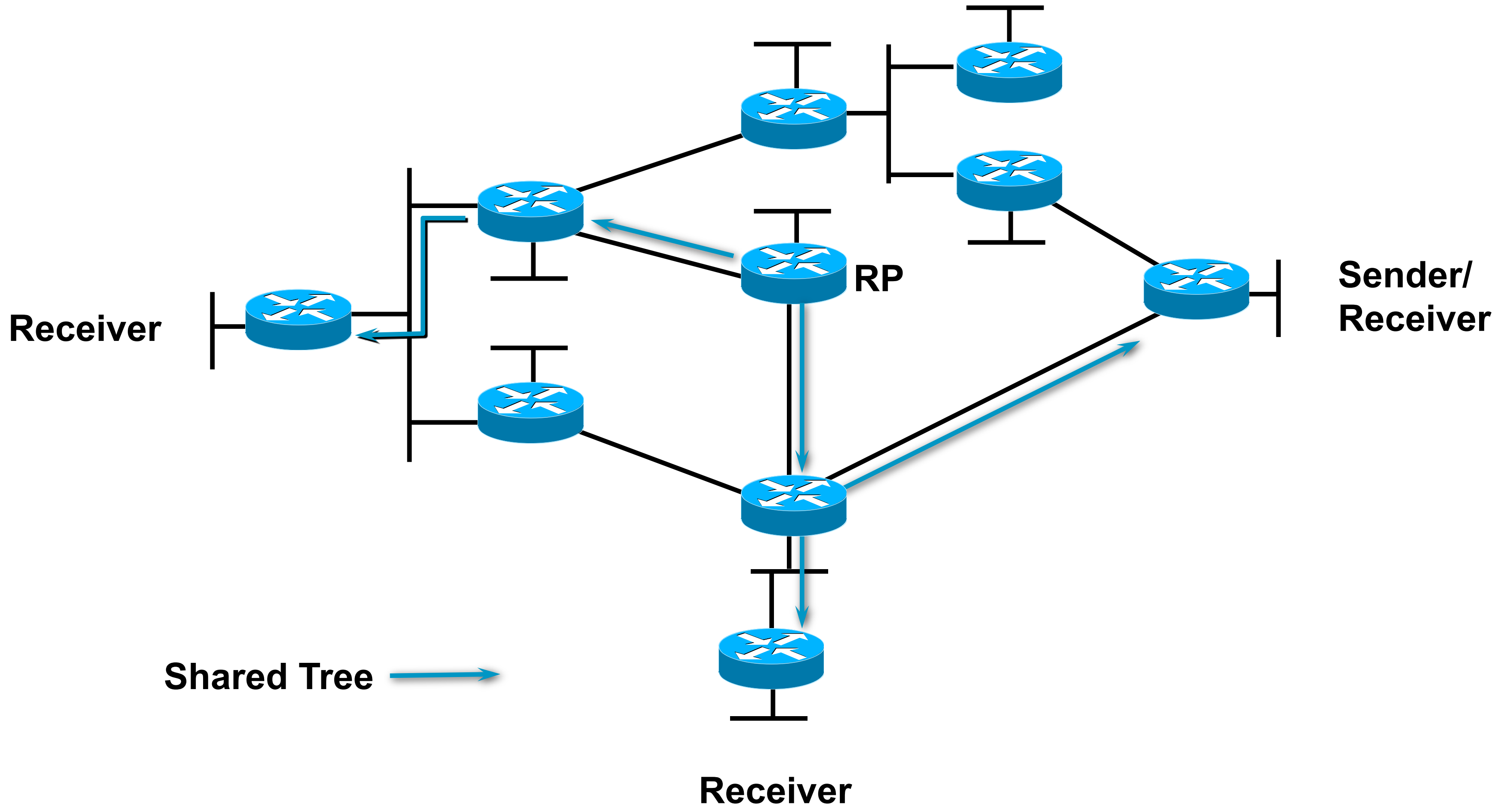
      Flows up the Shared Tree to reach the RP

      Flows down the Shared Tree to reach all other receivers

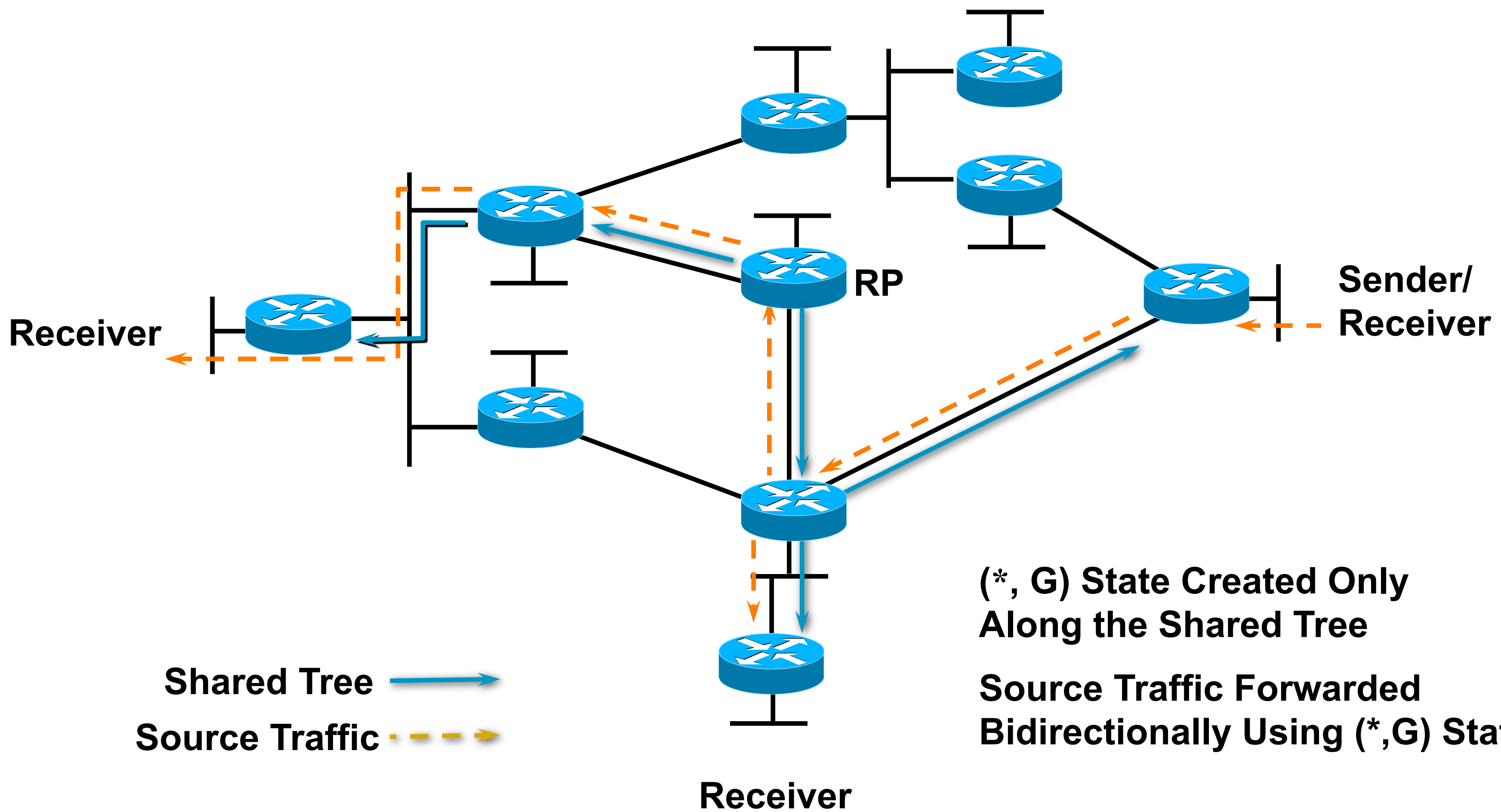
Bidirectional (BiDir) PIM

- Bidirectional Shared-Trees
    - Violates current (*,G) RPF rules
        - Traffic often accepted on **outgoing** interfaces
        - Care must be taken to avoid multicast loops
    - Requires a Designated Forwarder (DF)
        - Responsible for forwarding traffic up Shared Tree
            - DFs will accept data on the interfaces in their OIL
            - Then send it out all other interfaces (including the IIF)

Bidirectional PIM—Overview



Bidirectional PIM—Overview



(*, G) State Created Only Along the Shared Tree

Source Traffic Forwarded Bidirectionally Using (*,G) State

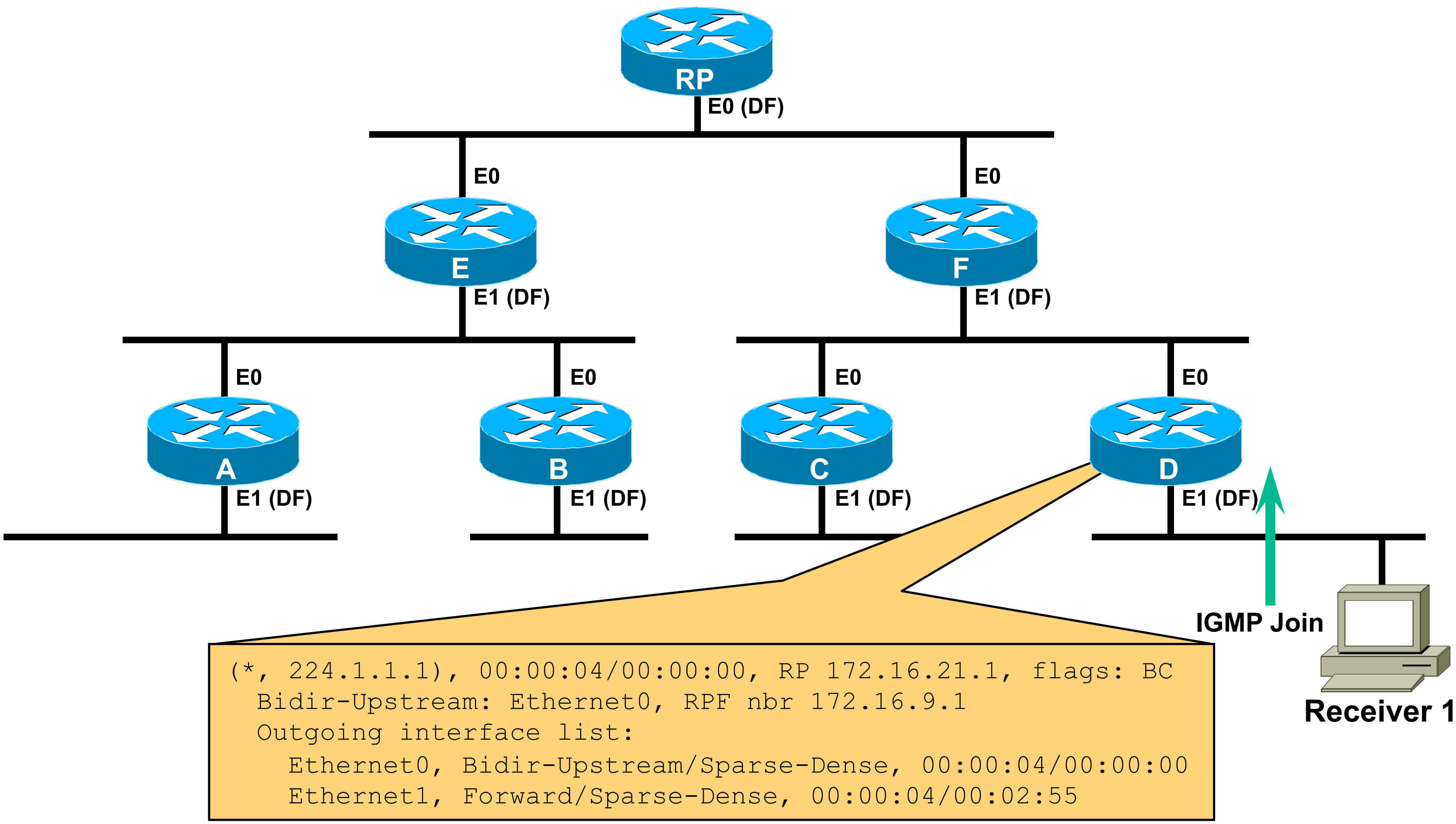
PIM Modifications for BiDir Operation

- Designated Forwarders (DF)
  - One DF per link
  - Router with best path to the RP is elected DF
    - Note: Designated Routers (DR) are not used for bidir groups
  - In addition to normal (*,G) forwarding rules:
    - Accepts traffic on outgoing interfaces
    - Forwards traffic out all other interfaces

Designated Forwarder Election

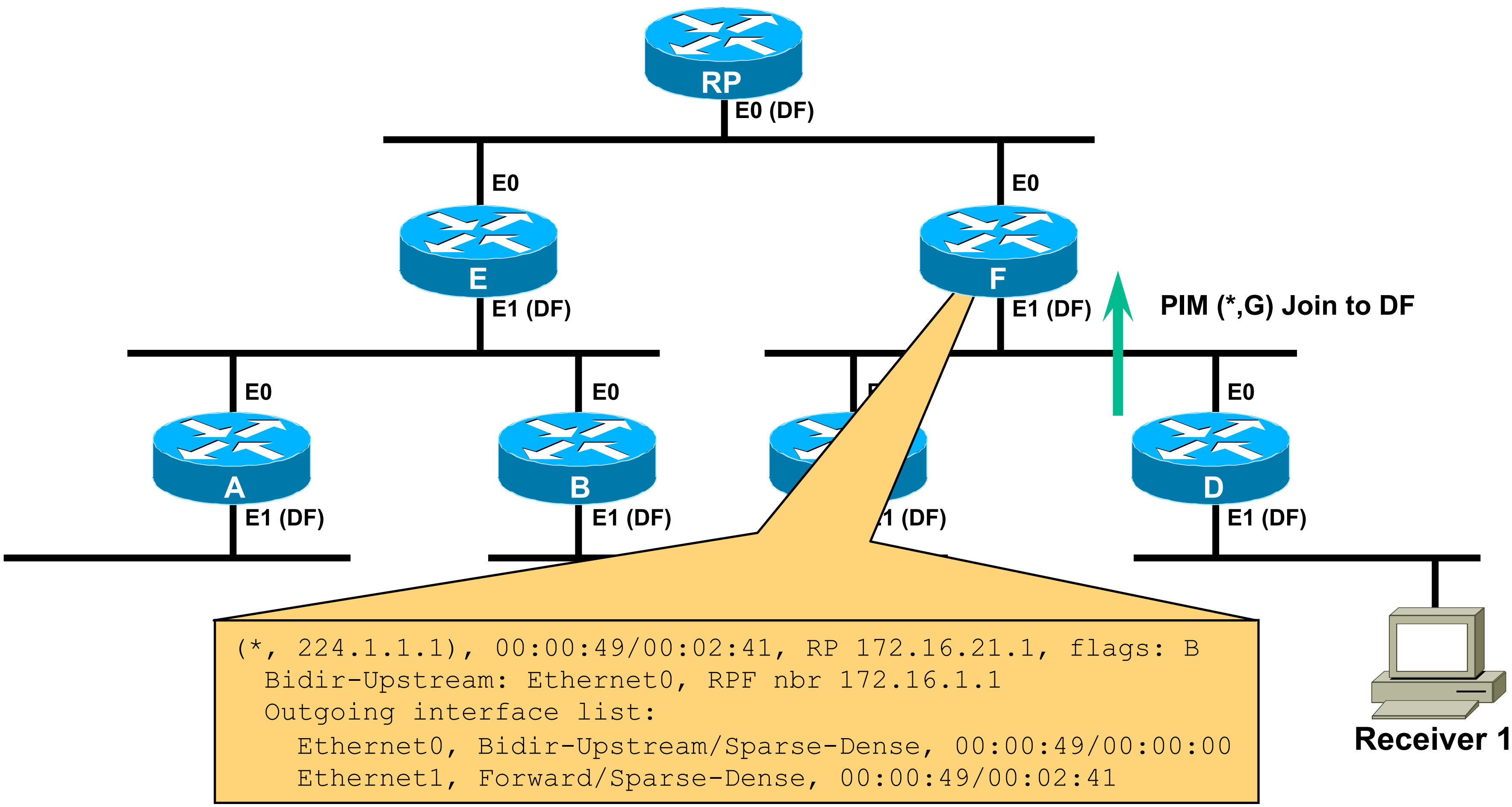
- Automatically performed on every link
  - When Bidir Group-range/RP is learned or configured
  - Router with the best path to the RP elected DF
  - Uses assert-like metric comparison to pick best path
- Purpose:
  - Ensures all routers on link agree on who is DF
    - Prevents route loops from forming

Forwarding/Tree Building



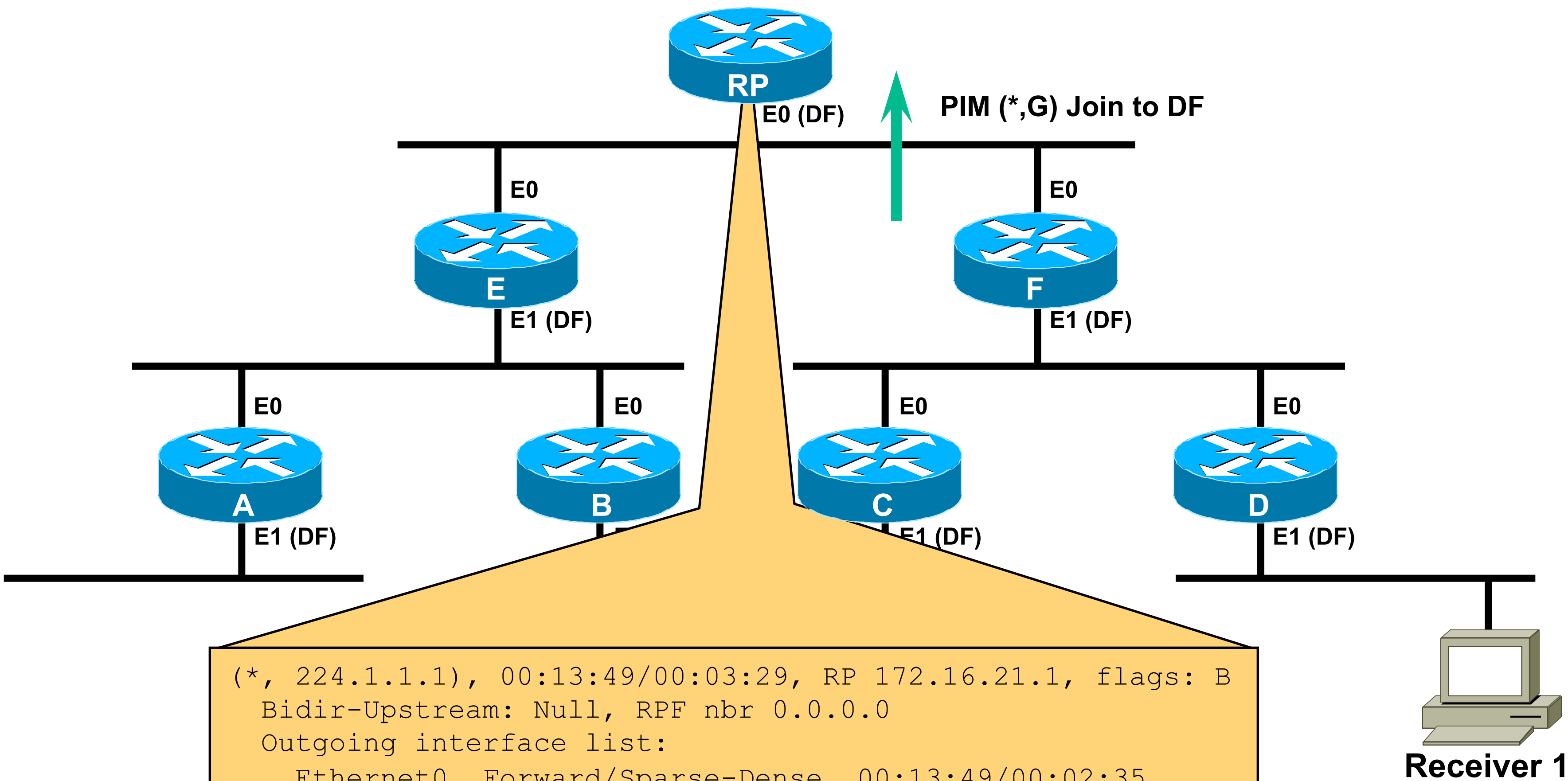
Receiver 1 Joins Group Causing Router “D” to Create (*, G) State

Forwarding/Tree Building



Router "D" Sends (*, G) Join to Router "F" (DF) Causing It to Create (*, G) State

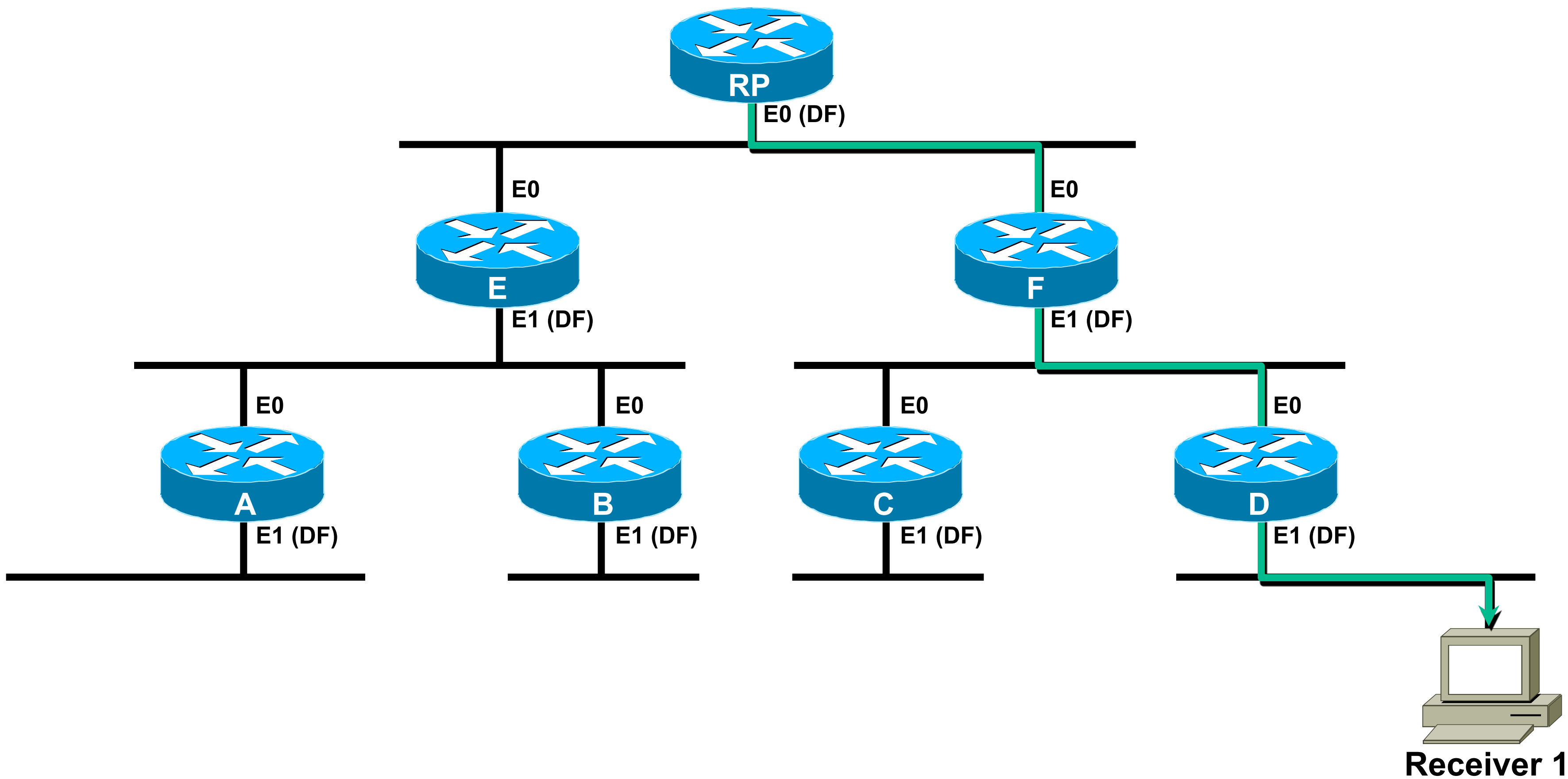
Forwarding/Tree Building




```
(*, 224.1.1.1), 00:13:49/00:03:29, RP 172.16.21.1, flags: B
  Bidir-Upstream: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Ethernet0, Forward/Sparse-Dense, 00:13:49/00:02:35
```

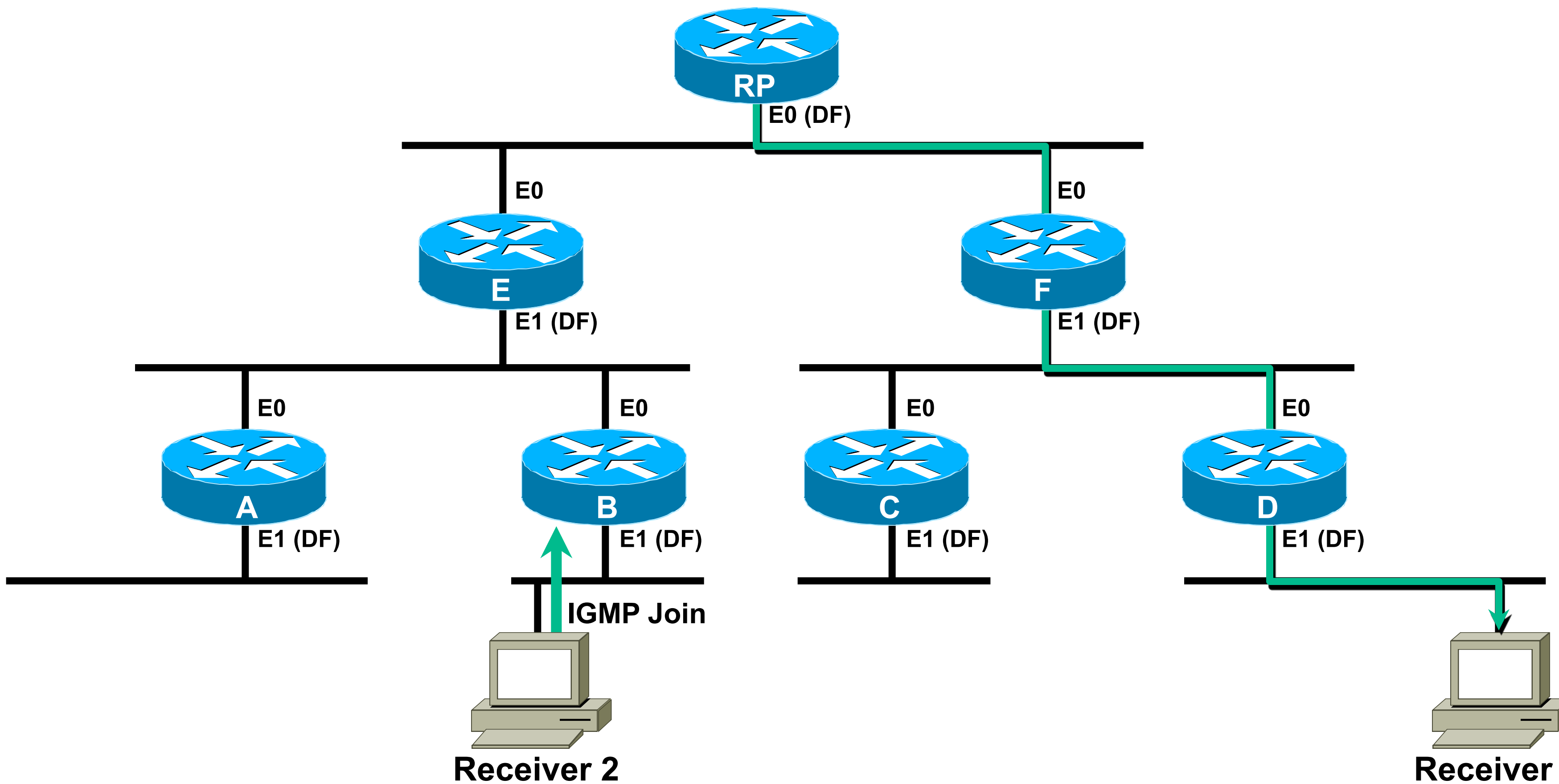
Router "F" Sends (*, G) Join to "RP" Causing It to Create (*, G) State

Forwarding/Tree Building



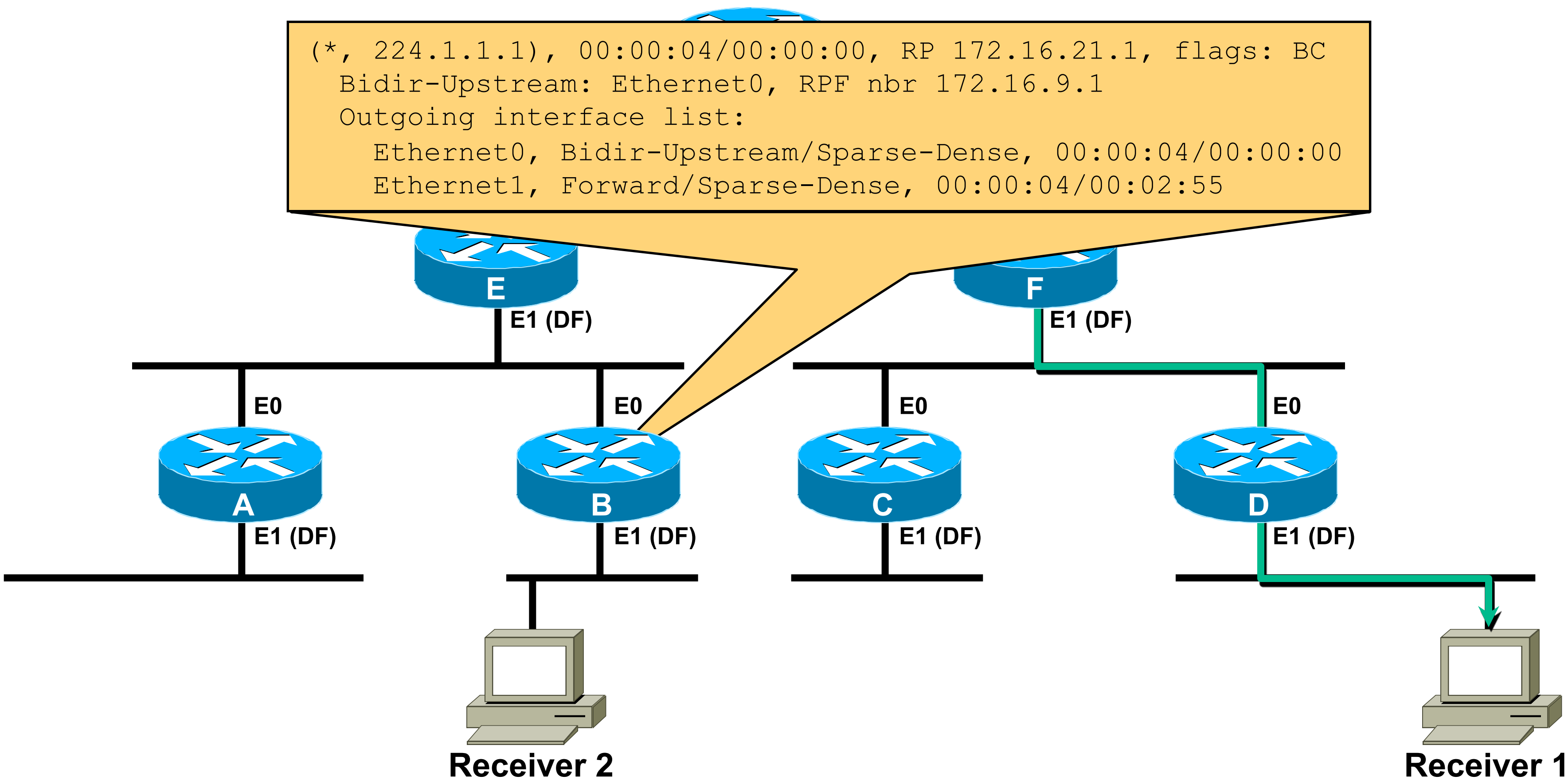
Branch of Shared Tree Is Now Built Down to Receiver 1

Forwarding/Tree Building



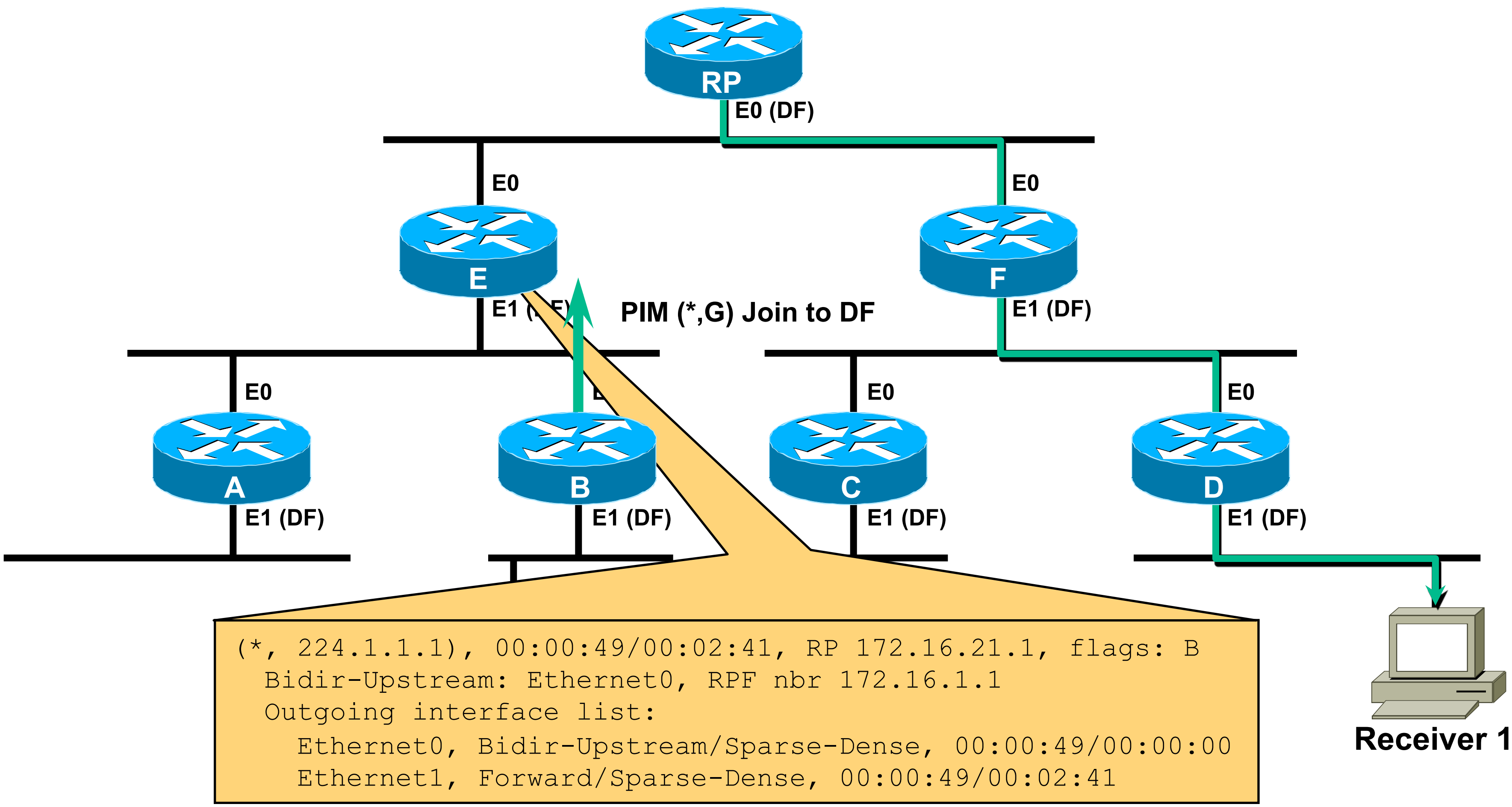
Receiver 2 Also Joins Group

Forwarding/Tree Building



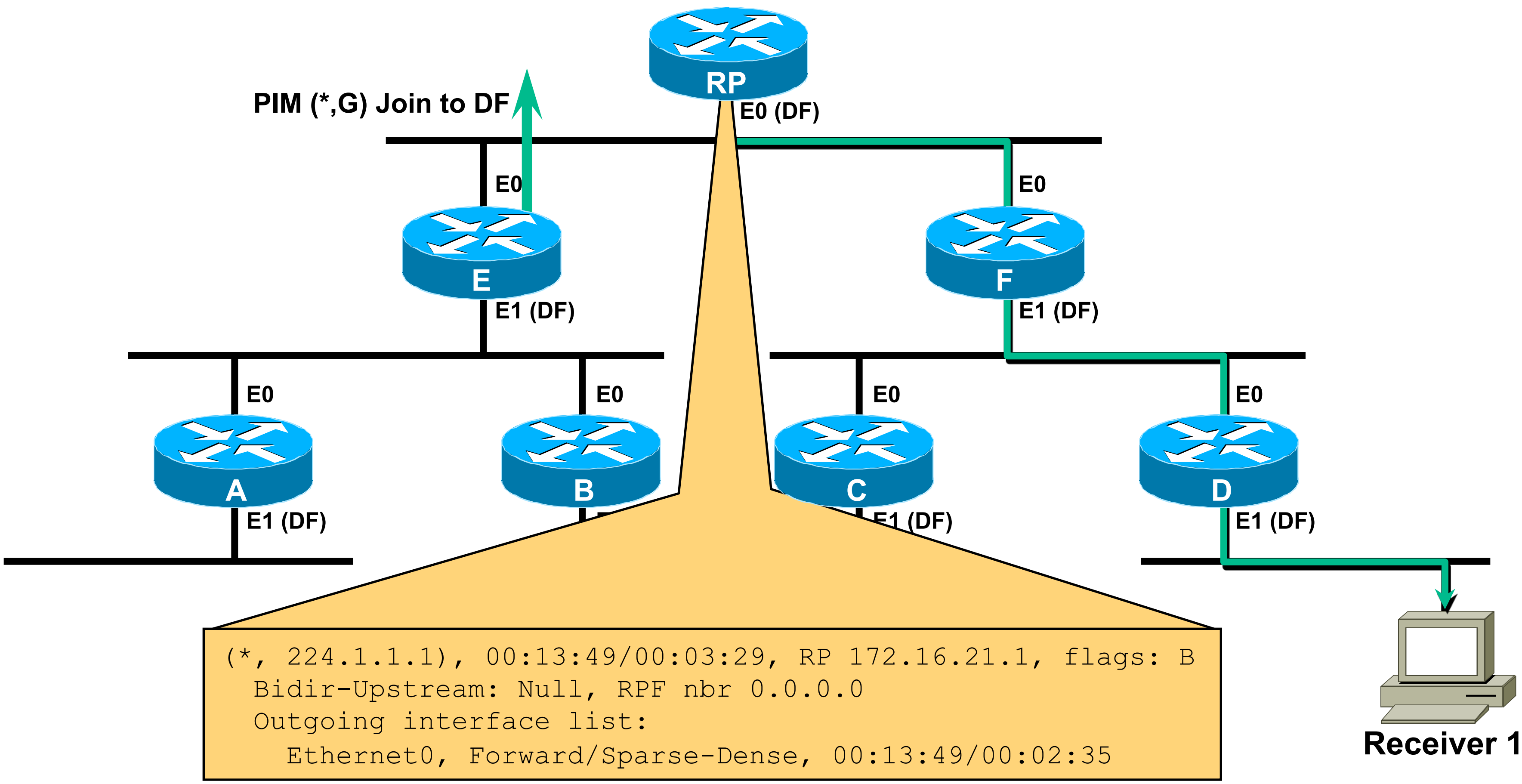
Router “B” Creates (*, G) State

Forwarding/Tree Building



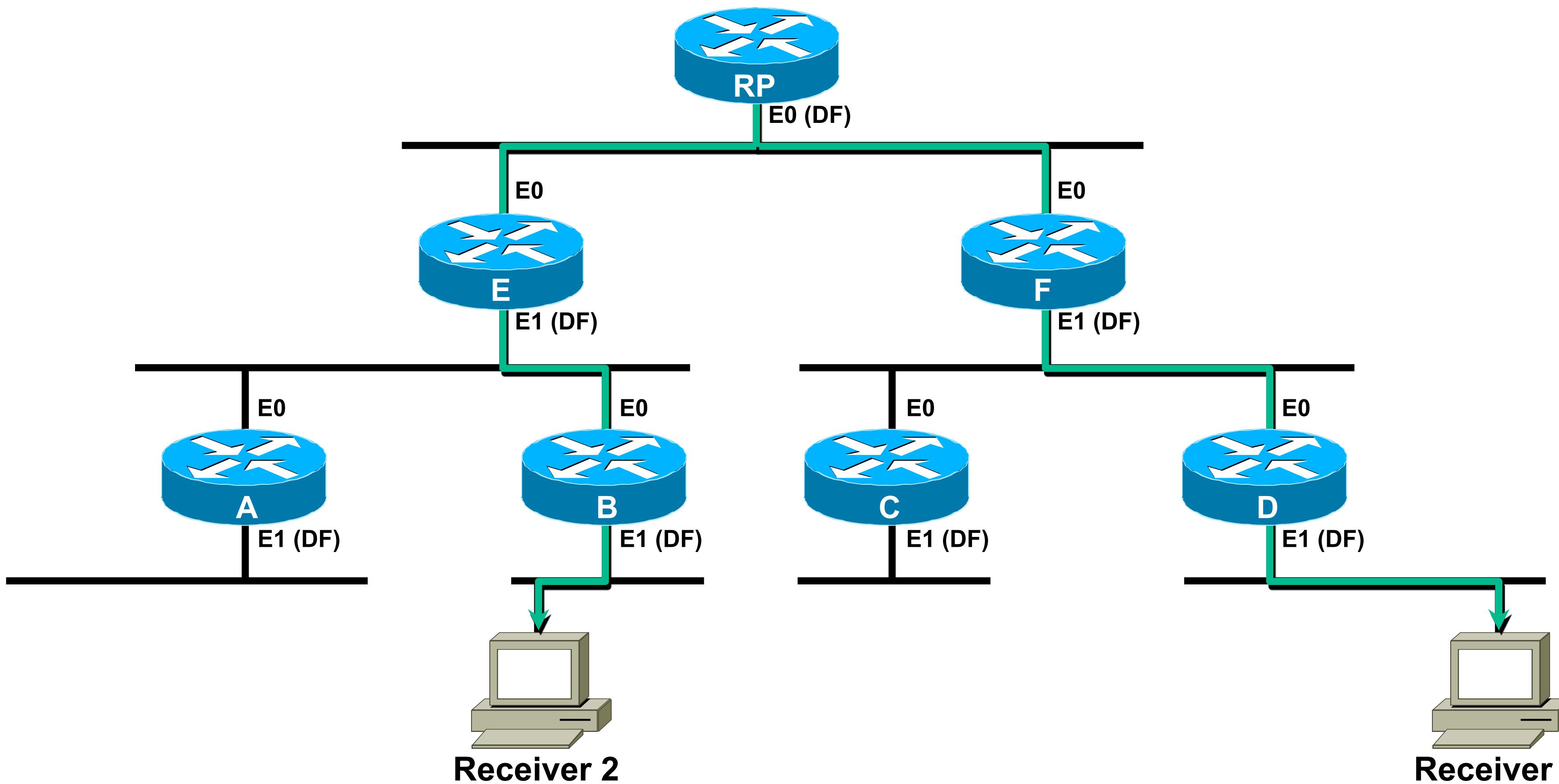
Router “B” Sends (*, G) Join to “E” (DF) Causing It to Create (*, G) State

Forwarding/Tree Building



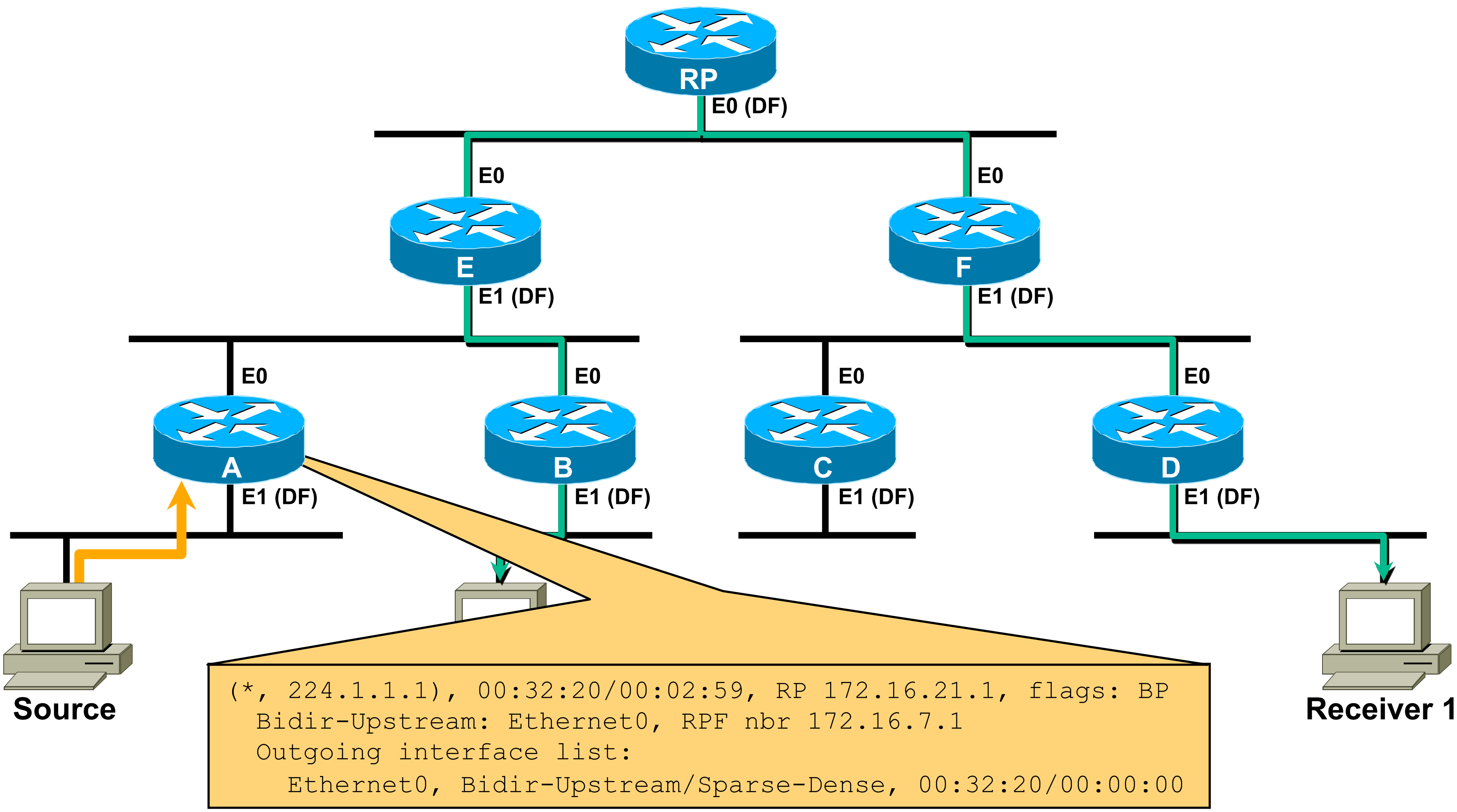
Router “E” Sends (*, G) Join to “RP” (State on RP Remains Unchanged)

Forwarding/Tree Building



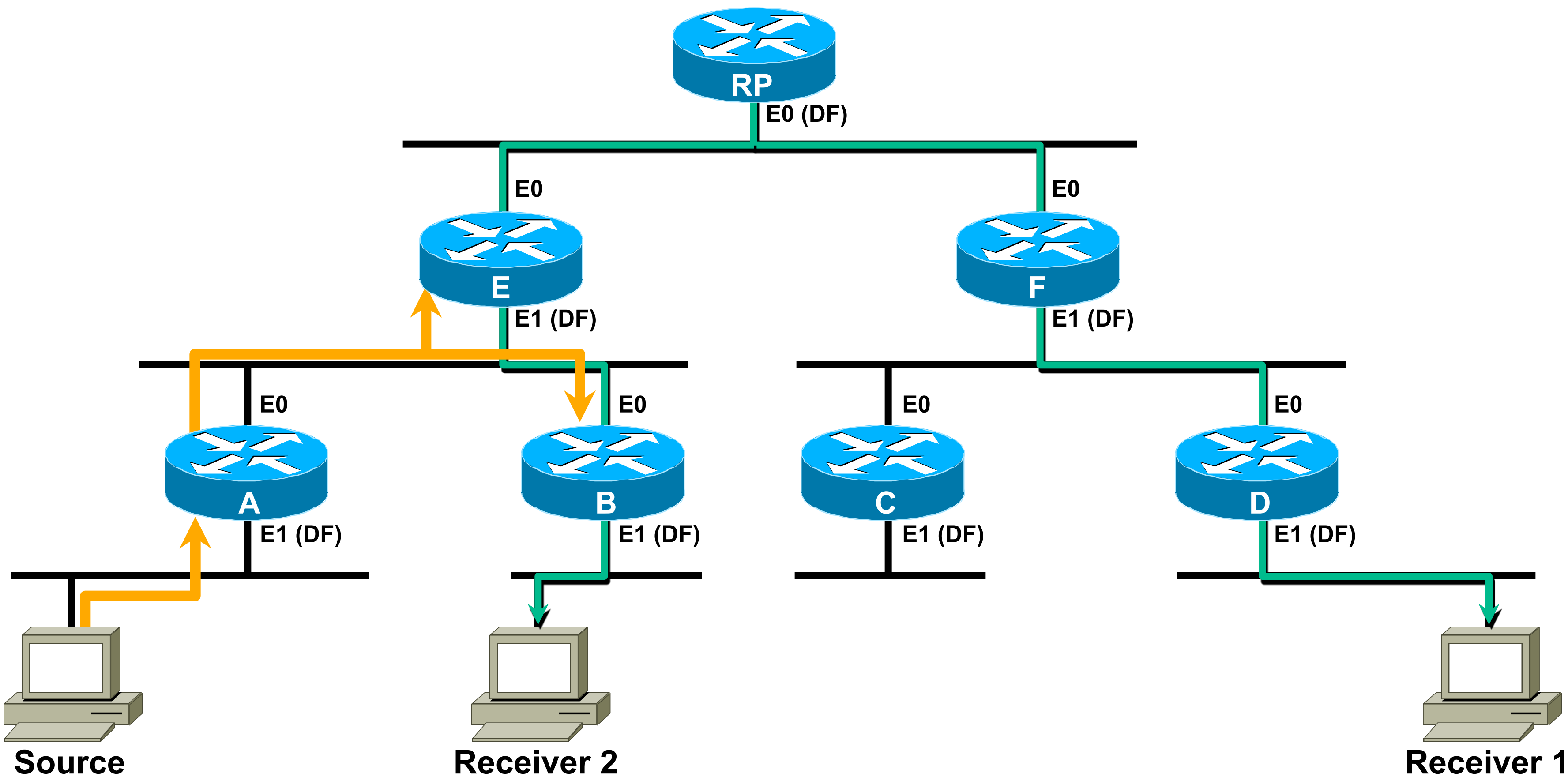
New Branch of Shared Tree Is Built to Receiver 2

Forwarding/Tree Building



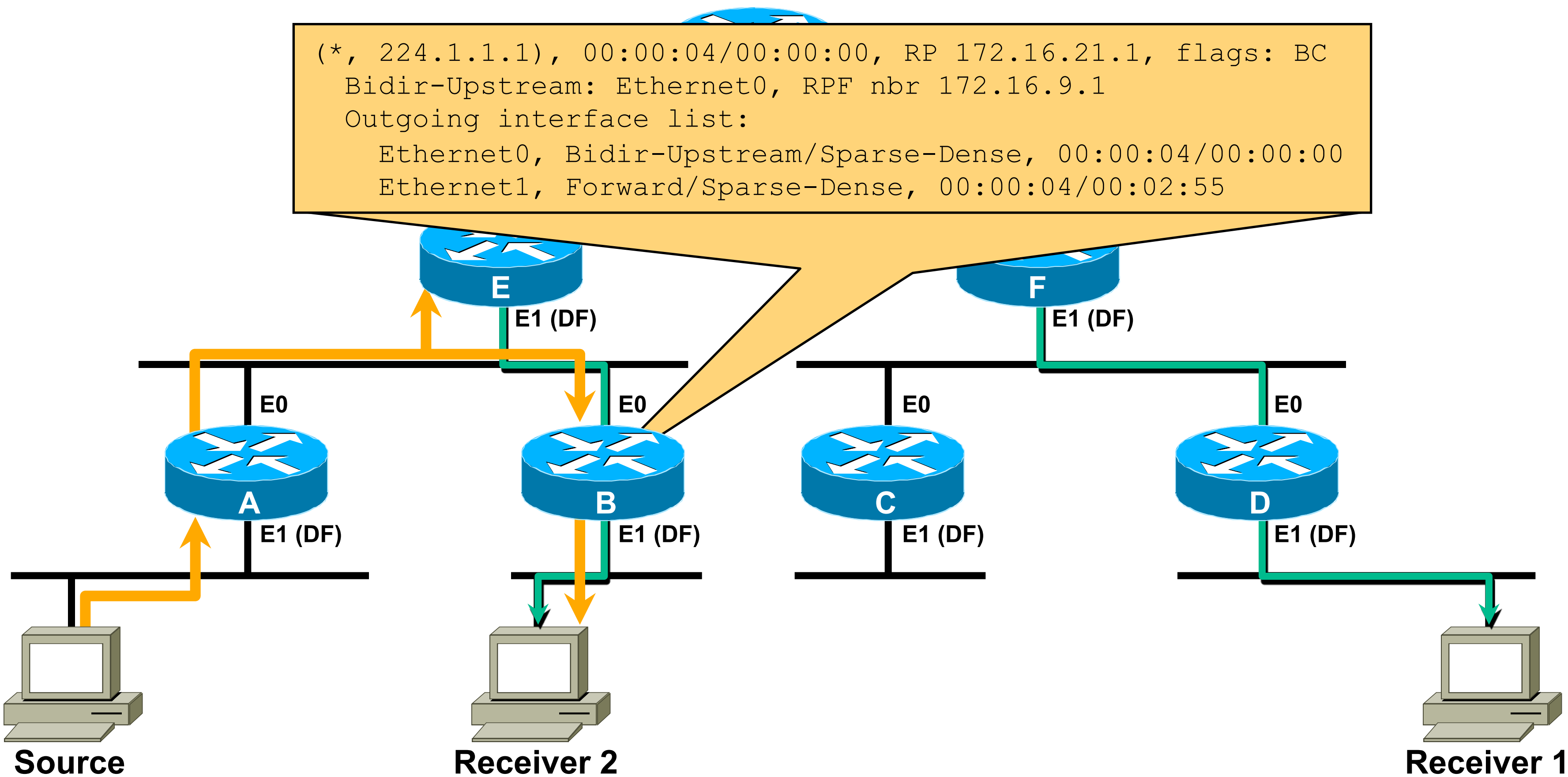
Arriving Traffic from Source Causes Router “A” to Create (*, G) State

Forwarding/Tree Building



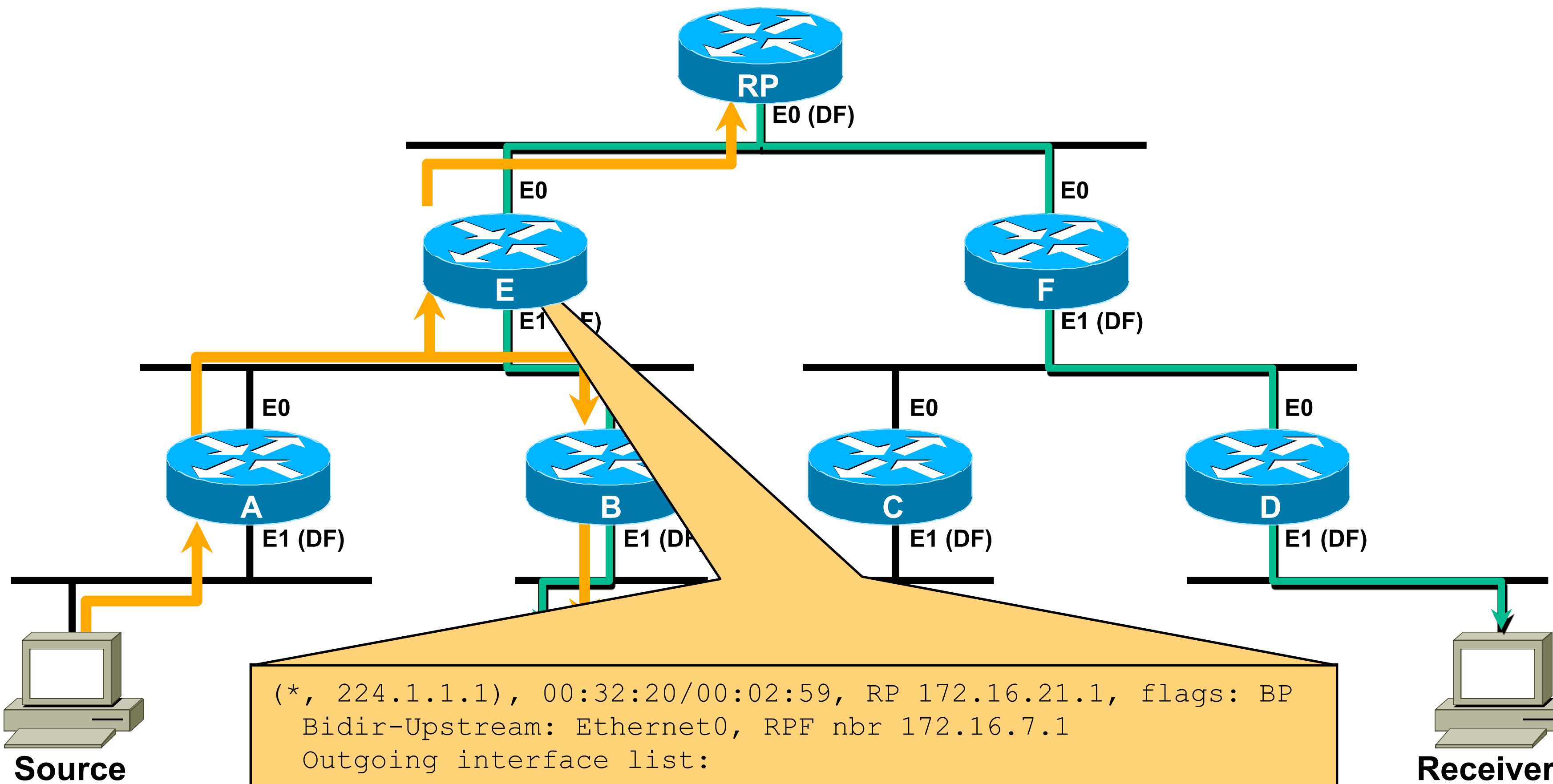
Traffic Is Forwarded Toward Router "E" and Also Arrives at IIF of Router "B"

Forwarding/Tree Building



Router "B" Forwards Traffic Back Down Shared Tree ala Normal PIM-SM

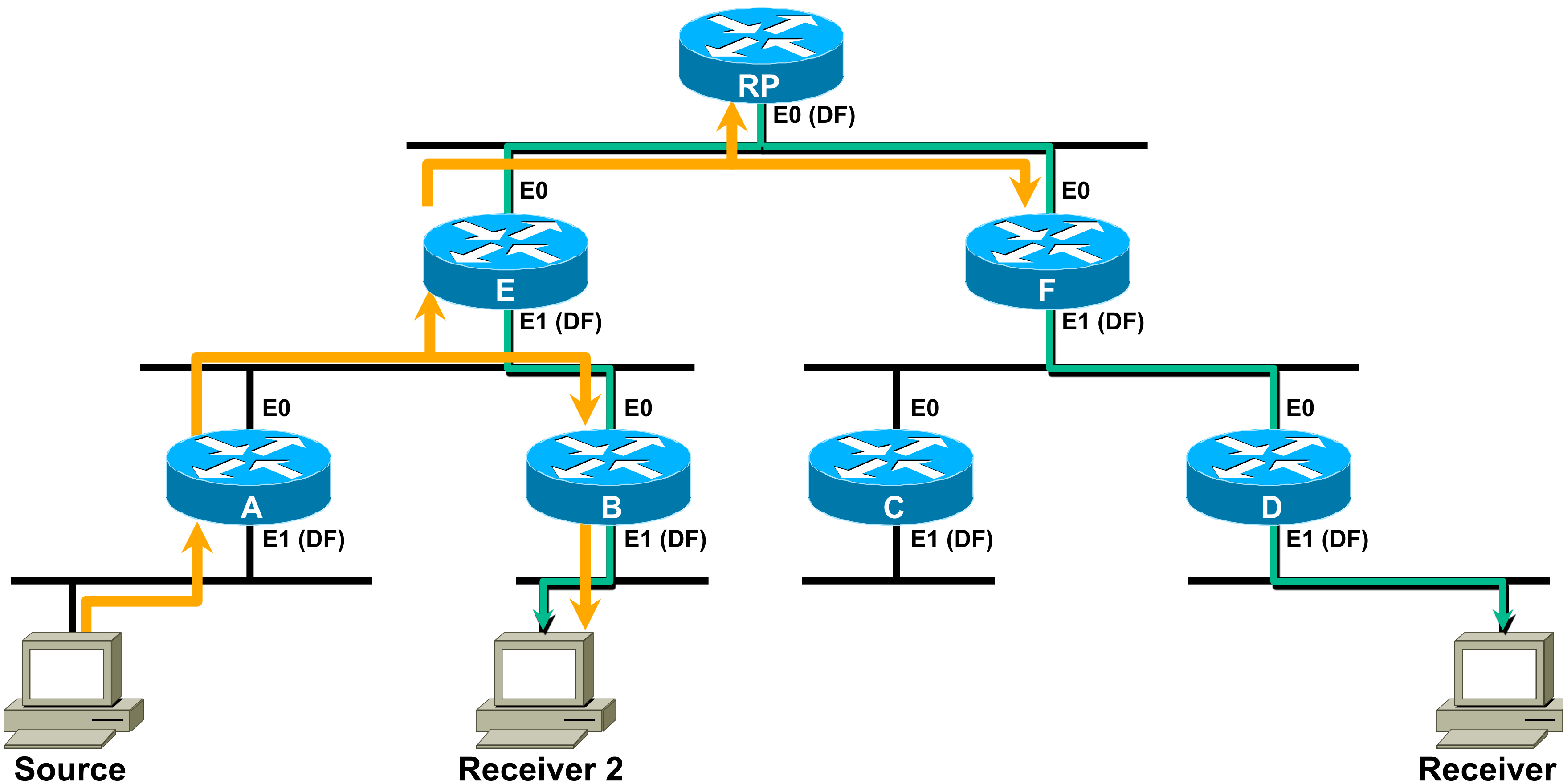
Forwarding/Tree Building




```
(*, 224.1.1.1), 00:32:20/00:02:59, RP 172.16.21.1, flags: BP
  Bidir-Upstream: Ethernet0, RPF nbr 172.16.7.1
  Outgoing interface list:
    Ethernet0, Bidir-Upstream/Sparse-Dense, 00:32:20/00:00:00
    Ethernet1, Forward/Sparse-Dense, 00:00:04/00:02:55
```

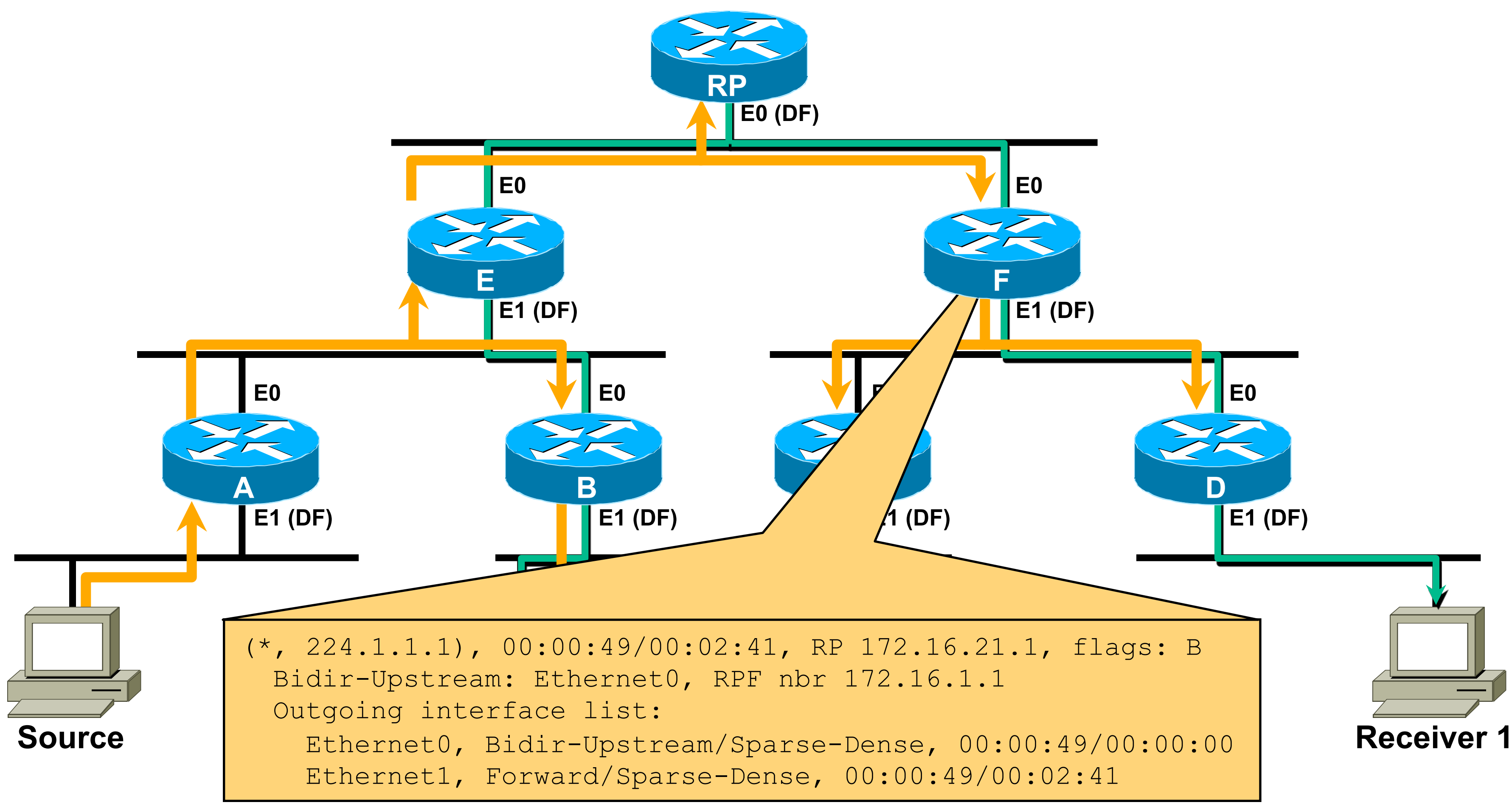
Router “E” Forwards Traffic on Toward RP

Forwarding/Tree Building



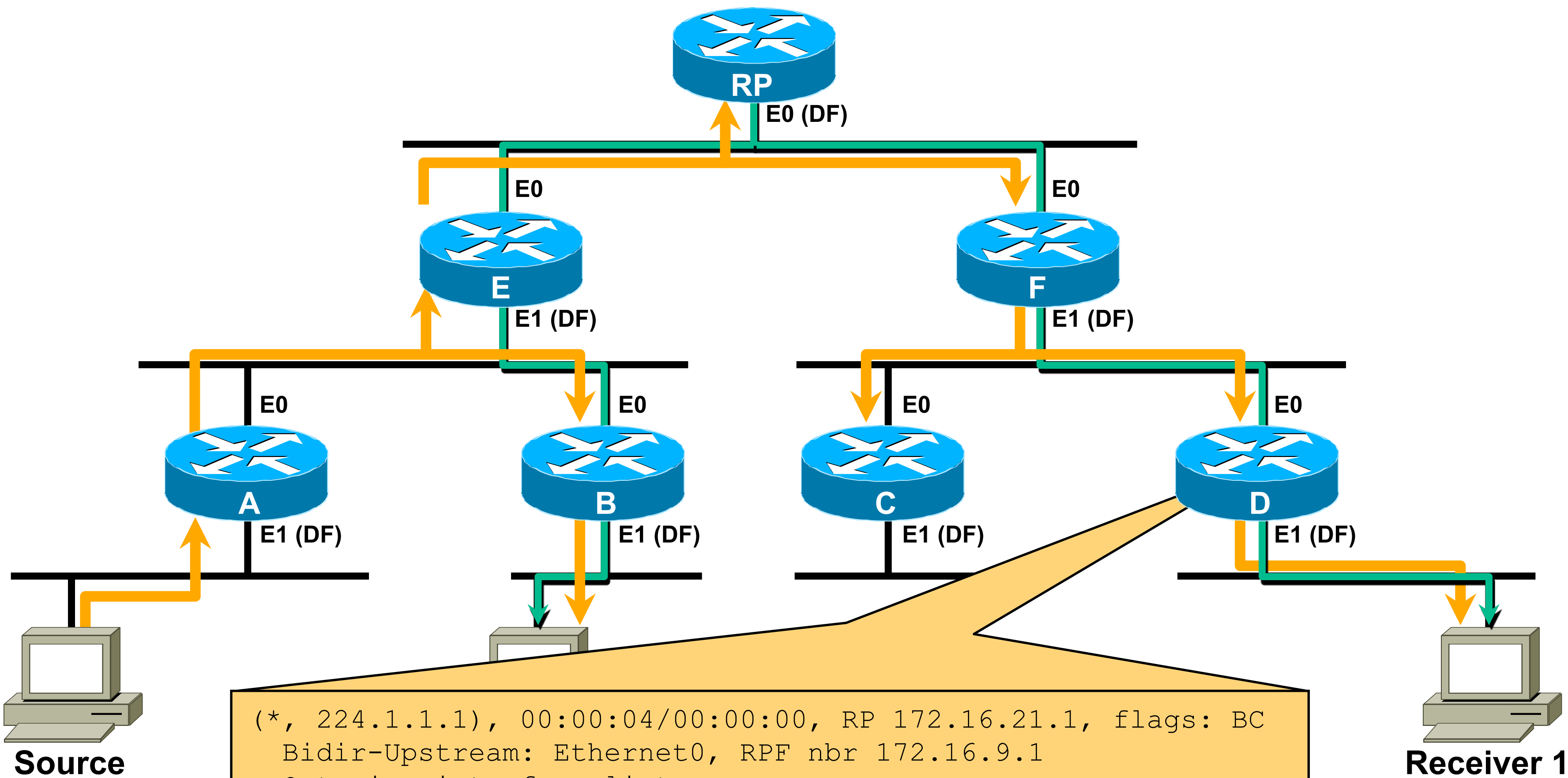
Traffic Forwarded Toward RP Also Arrives at the IIF of Router "F"

Forwarding/Tree Building



Router "F" Forwards Traffic on Down the Shared Tree ala Normal PIM-SM

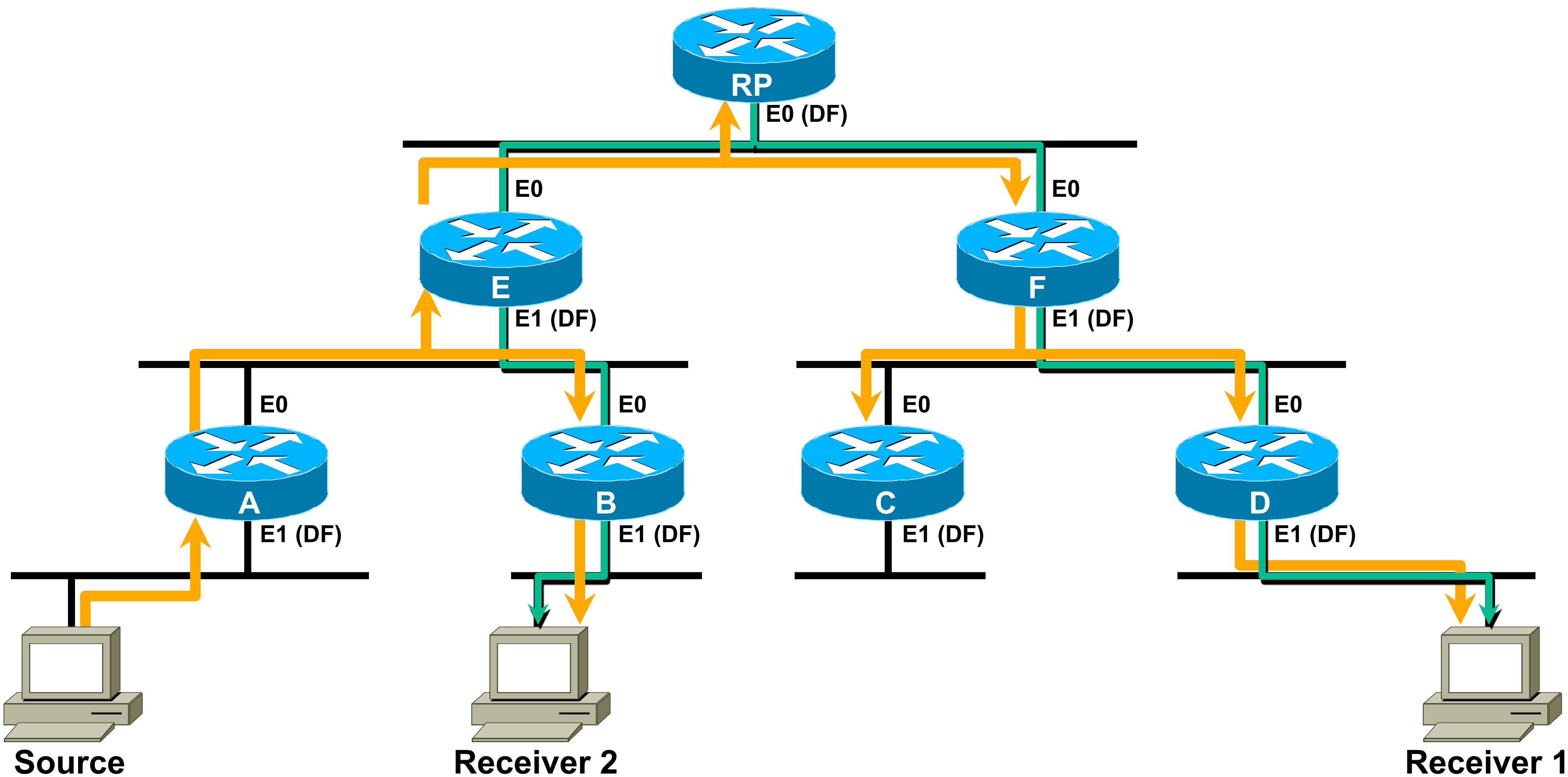
Forwarding/Tree Building




```
(*, 224.1.1.1), 00:00:04/00:00:00, RP 172.16.21.1, flags: BC
  Bidir-Upstream: Ethernet0, RPF nbr 172.16.9.1
  Outgoing interface list:
    Ethernet0, Bidir-Upstream/Sparse-Dense, 00:00:04/00:00:00
    Ethernet1, Forward/Sparse-Dense, 00:00:04/00:02:55
```

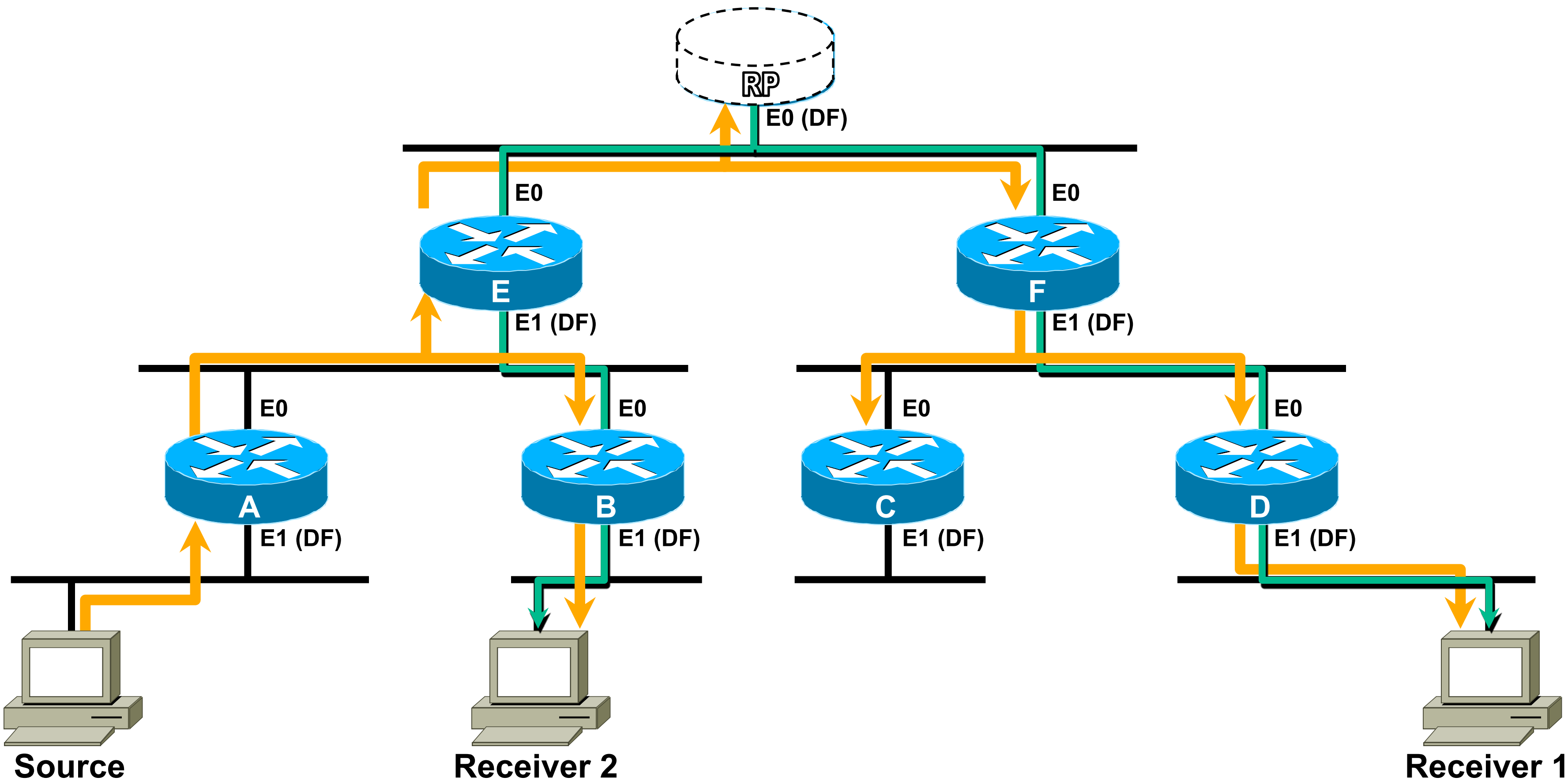
Router “D” Forwards Traffic to Receiver 1 via the Shared Tree

Forwarding/Tree Building



Question: Does the RP even have to physically exist?

Forwarding/Tree Building



Question: Does the RP even have to physically exist?

Answer: No. It can just be a phantom address.

Bidir PIM—Summary

- Uses Shared Trees only
  - Single (*, G) forwarding entry per group
  - Source traffic flows up and down Shared Tree
- Drastically reduces network mroute state
  - Eliminates ALL (S,G) state in the network
    - By eliminating SPT between source and RP
  - Allows many-to-any applications to scale
    - Permits virtually an unlimited number of sources

Lab

IPv4 PIM Configuration

Enable multicast routing on every router

```
ip multicast-routing
```

ALL Modes of PIM on ALL interfaces of every router

```
interface <interface>
  ip pim sparse-mode
```

Sparse-mode and BiDir require an RP mapping on every router

```
ip pim rp-address x.x.x.x [bidir]
```

SSM: 232/8 is the default range

```
ip pim ssm default
```

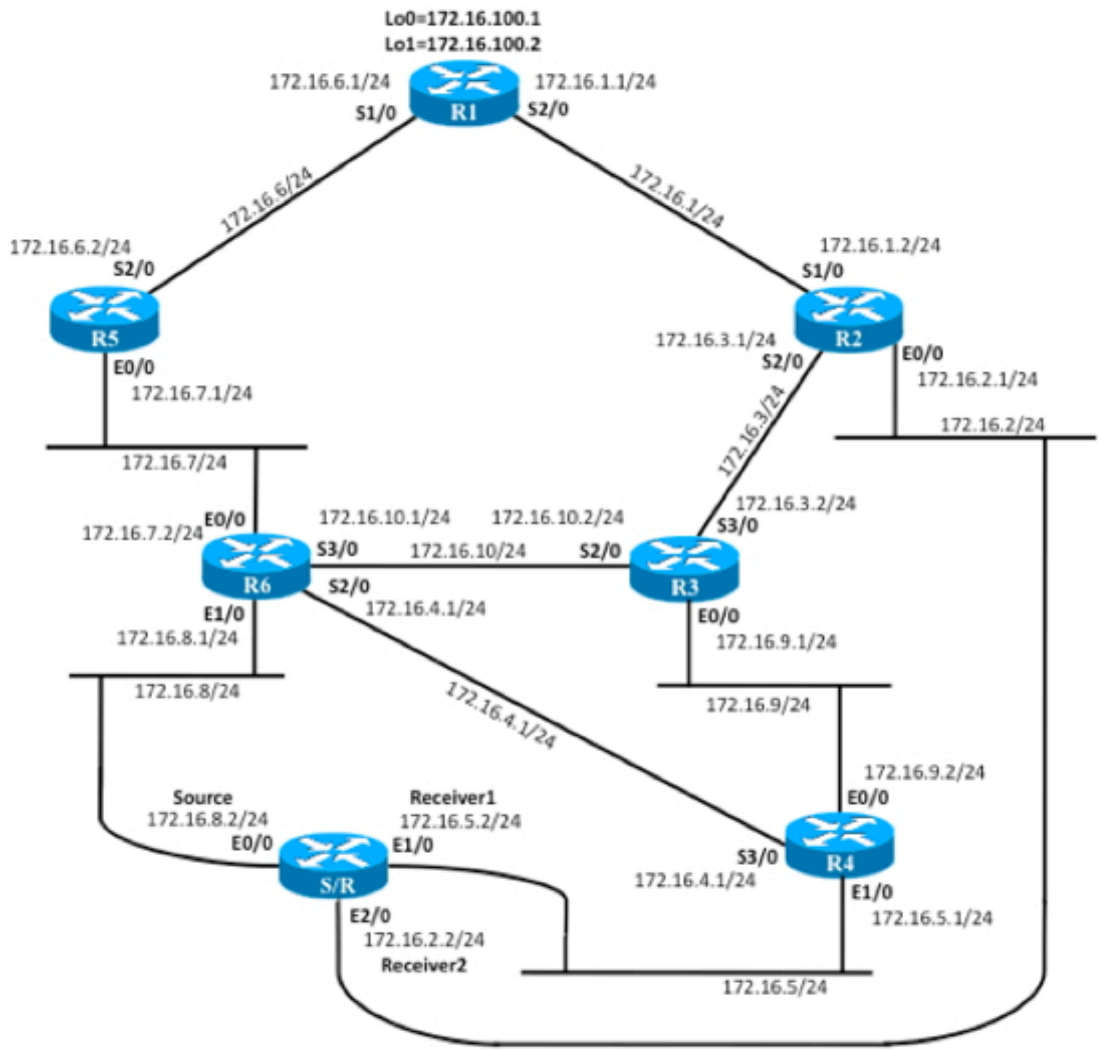
On the RP router ONLY

```
interface lo1
  ip address x.x.x.x 255.255.255.255
```

LAB #1
PIM-SM Mechanics - SSM / ASM / BiDir

- Get your username and password from the instructor
- Once your are logged in, DO NOT start the lab until instructed
- Lab templates PIM-Mechanics
- Refer to your lab handout

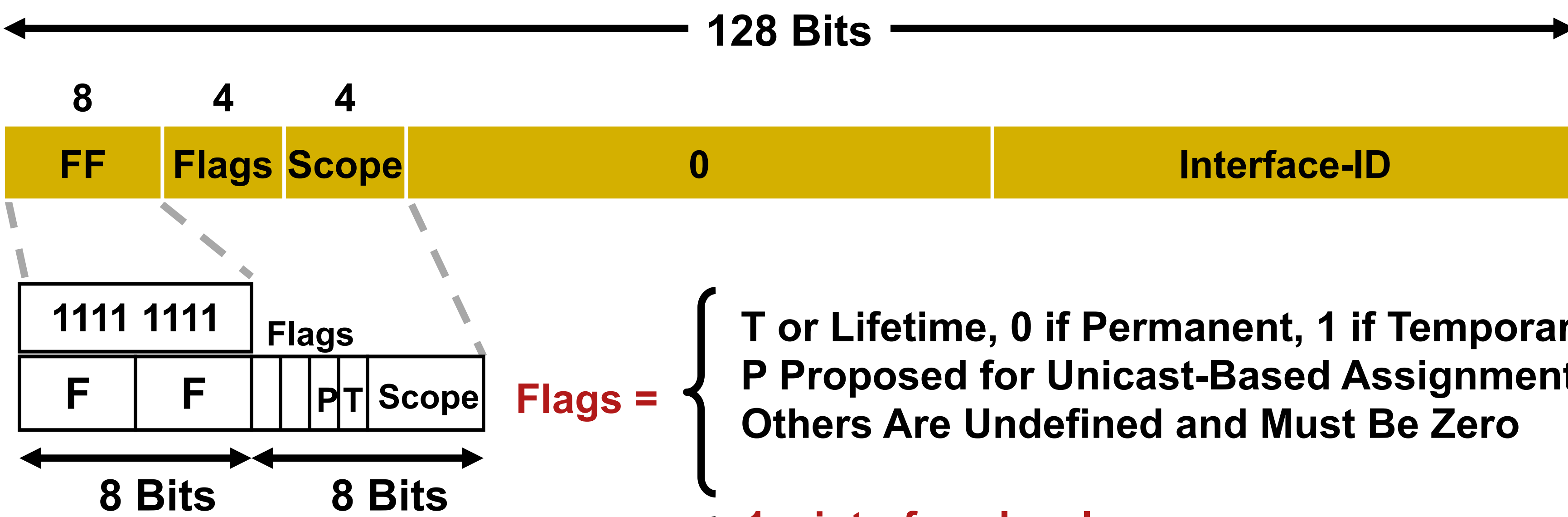
LAB #1 IPv4
PIM-SM Mechanics - SSM / ASM / BiDir



IPv4 vs. IPv6 Multicast

IP Service	IPv4 Solution	IPv6 Solution
Address Range	32-Bit, Class D	128-Bit (112-Bit Group)
Routing	Protocol-Independent All IGPs and GBP4+	Protocol-Independent All IGPs and BGP4+ with v6 Mcast SAFI
Forwarding	PIM-DM, PIM-SM: ASM, SSM, BiDir	PIM-SM: ASM, SSM, BiDir
Group Management	IBMPv1, v2, v3	MLDv1, v2
Domain Control	Boundary/Border	Scope Identifier
Interdomain Source Discovery	MSDP Across Independent PIM Domains	Single RP Within Globally Shared Domains

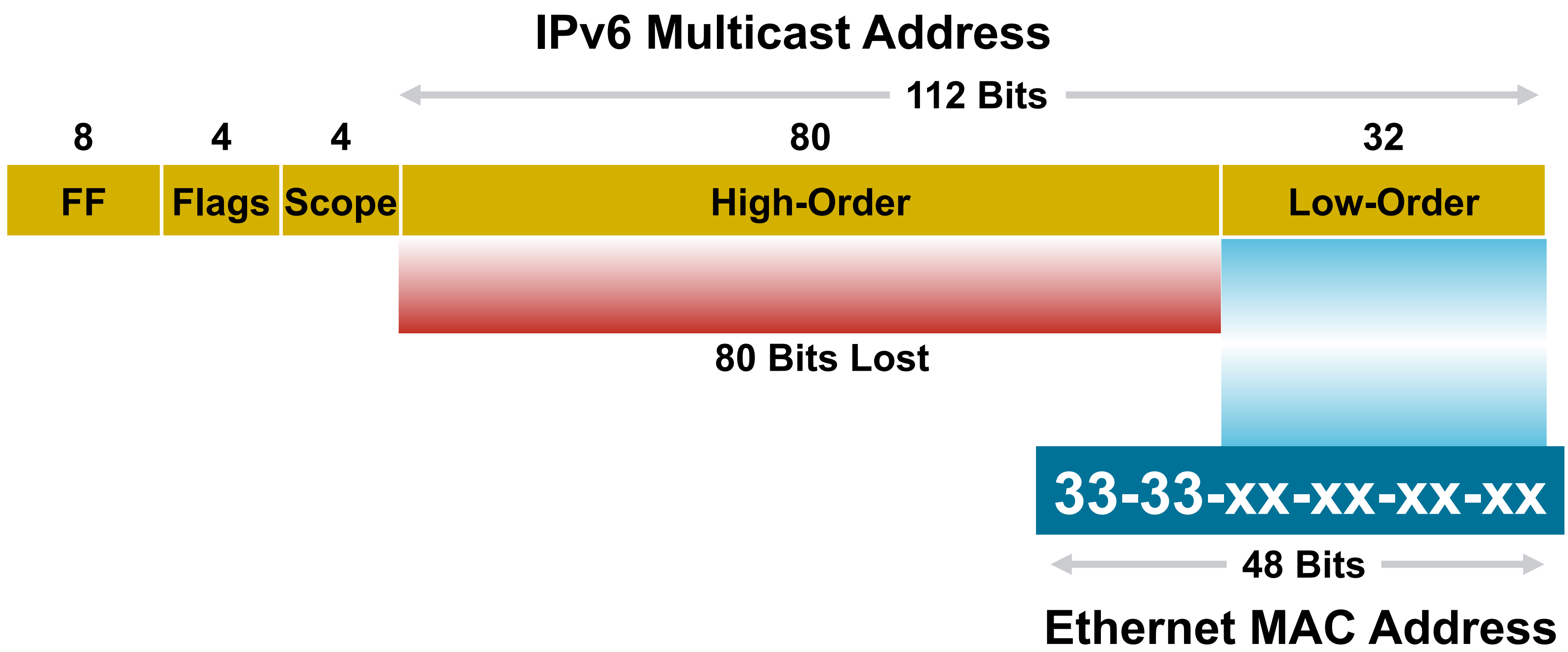
IPv6 Multicast Addresses (RFC 3513)



Flags =
- T or Lifetime, 0 if Permanent, 1 if Temporary
- P Proposed for Unicast-Based Assignments
- Others Are Undefined and Must Be Zero

Scope =
- 1 = interface-local
- 2 = link
- 4 = admin-local
- 5 = site
- 8 = organization
- E = global

IPv6 Layer 2 Multicast Addressing Mapping



Unicast-Based Multicast Addresses

FF	Flags	Scope	Rsvd	Plen	Network-Prefix	Group-ID
8	4	4	8	8	64	32

- RFC 3306—unicast-based multicast addresses
  - Similar to IPv4 GLOP addressing
  - Solves IPv6 global address allocation problem
  - Flags = 00PT
    - P = 1, T = 1 → Unicast-based multicast address
- Example
  - Content provider's unicast prefix
    - 1234:5678:9::/48
  - Multicast address
    - FF3x:0030:1234:5678:0009::0001

IP Routing for Multicast

- RPF-based on reachability to v6 source same as with v4 multicast
- RPF still protocol-independent
  - Static routes, mroutes
  - Unicast RIB: BGP, ISIS, OSPF, EIGRP, RIP, etc.
  - Multiprotocol BGP (mBGP)
    - Support for v6 mcast subaddress family
    - Provide translate function for nonsupporting peers

IPv6 Multicast Forwarding

- PIM-Sparse Mode (PIM-SM)

  RFC4601

- PIM Source Specific Mode (SSM)

  RFC3569 SSM overview (v6 SSM needs MLDv2)

  Unicast, prefix-based multicast addresses ff30::/12

  SSM range is ff3X::/96

- PIM Bi-Directional Mode (BiDir)

  draft-ietf-pim-bidir-09.txt

RP Mapping Mechanisms for IPv6

- Static RP assignment
- BSR
- Auto-RP—no current plans
- Embedded RP

Embedded RP Addressing—RFC3956

FF	Flags	Scope	Rsvd	RPadr	Plen	Network-Prefix	Group-ID
8	4	4	4	4	8	64	32

- Proposed new multicast address type

  Uses unicast-based multicast addresses (RFC 3306)

- RP address is embedded in multicast address
- Flag bits = 0RPT

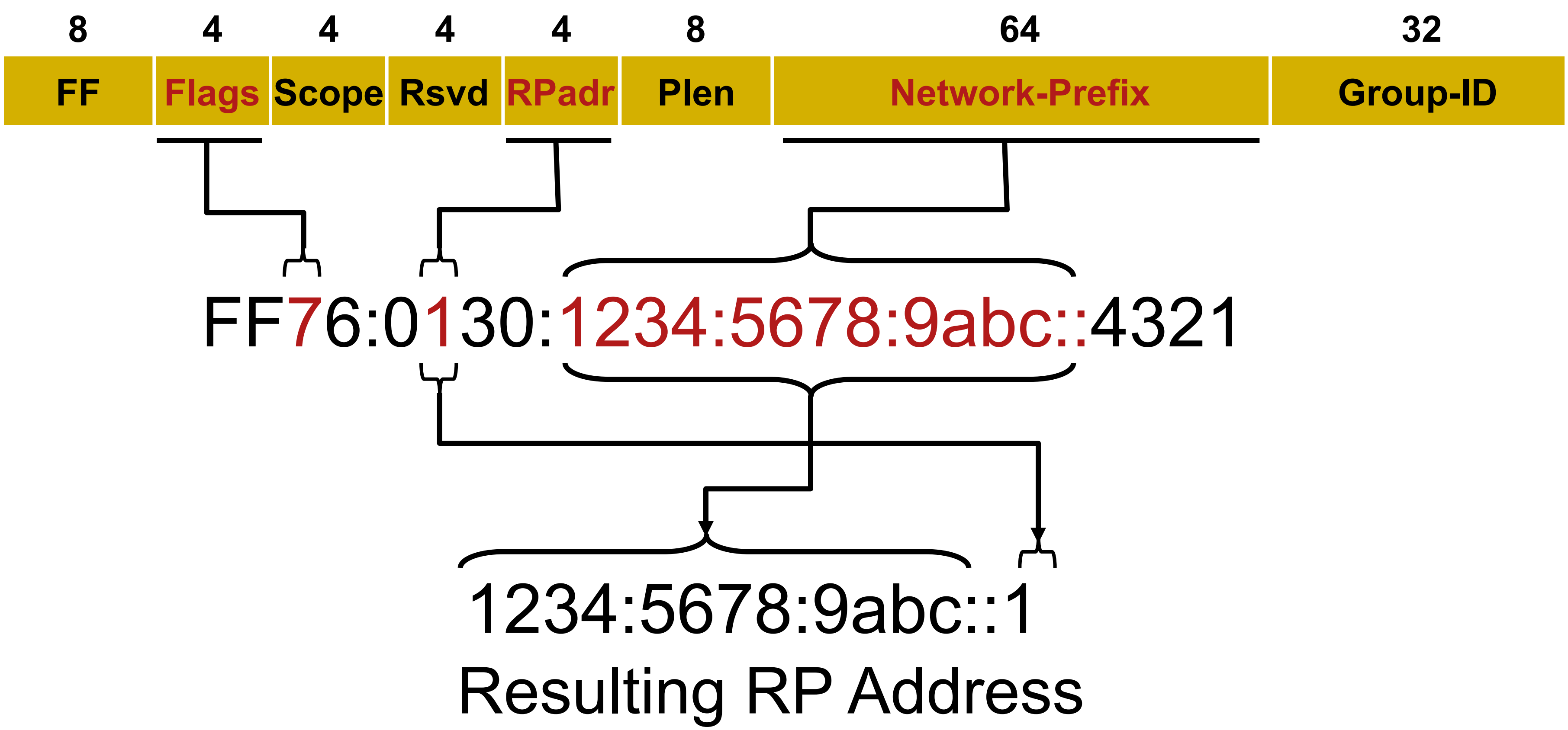
  R = 1, P = 1, T = 1 → Embedded RP address

- Network-Prefix::RPadr = RP address
- For each unicast prefix you own, you now also own:

  16 RPs for each of the 16 multicast scopes (256 total) with 2^32 multicast groups assigned to each RP (2^40 total)

Embedded RP Addressing—Example

Multicast Address with Embedded RP Address



Multicast Listener Discover—MLD

- MLD is equivalent to IGMP in IPv4
- MLD messages are transported over ICMPv6
- Version number confusion
  - MLDv1 corresponds to IGMPv2
    - RFC 2710
  - MLDv2 corresponds to IGMPv3, needed for SSM
    - RFC 3810
- MLD snooping
  - draft-ietf-magma-snoop-12.txt

IPv6 PIM Configuration

Enable multicast routing on every router

```
ipv6 multicast-routing
```

ALL Modes of PIM on ALL interfaces of every router

```
interface <interface>
  ip pim sparse-mode
```

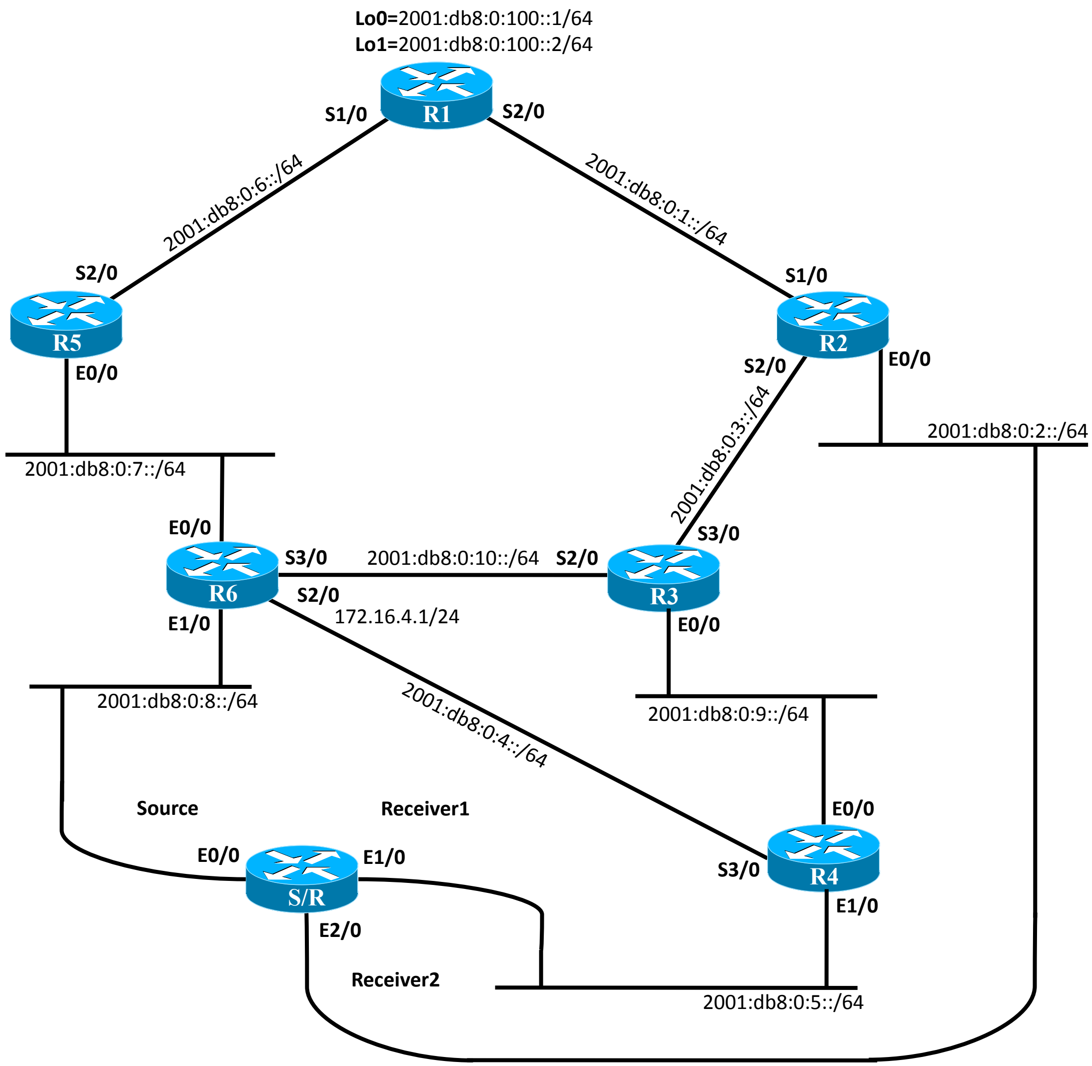
Sparse-mode and BiDir require an RP mapping on every router

```
ipv6 pim rp-address x.x.x.x [bidir]
```

On the RP router ONLY

```
interface lo1
  ipv6 address XXX::XXX/128
```

LAB #1 IPv6
PIM-SM Mechanics - SSM / ASM / BiDir



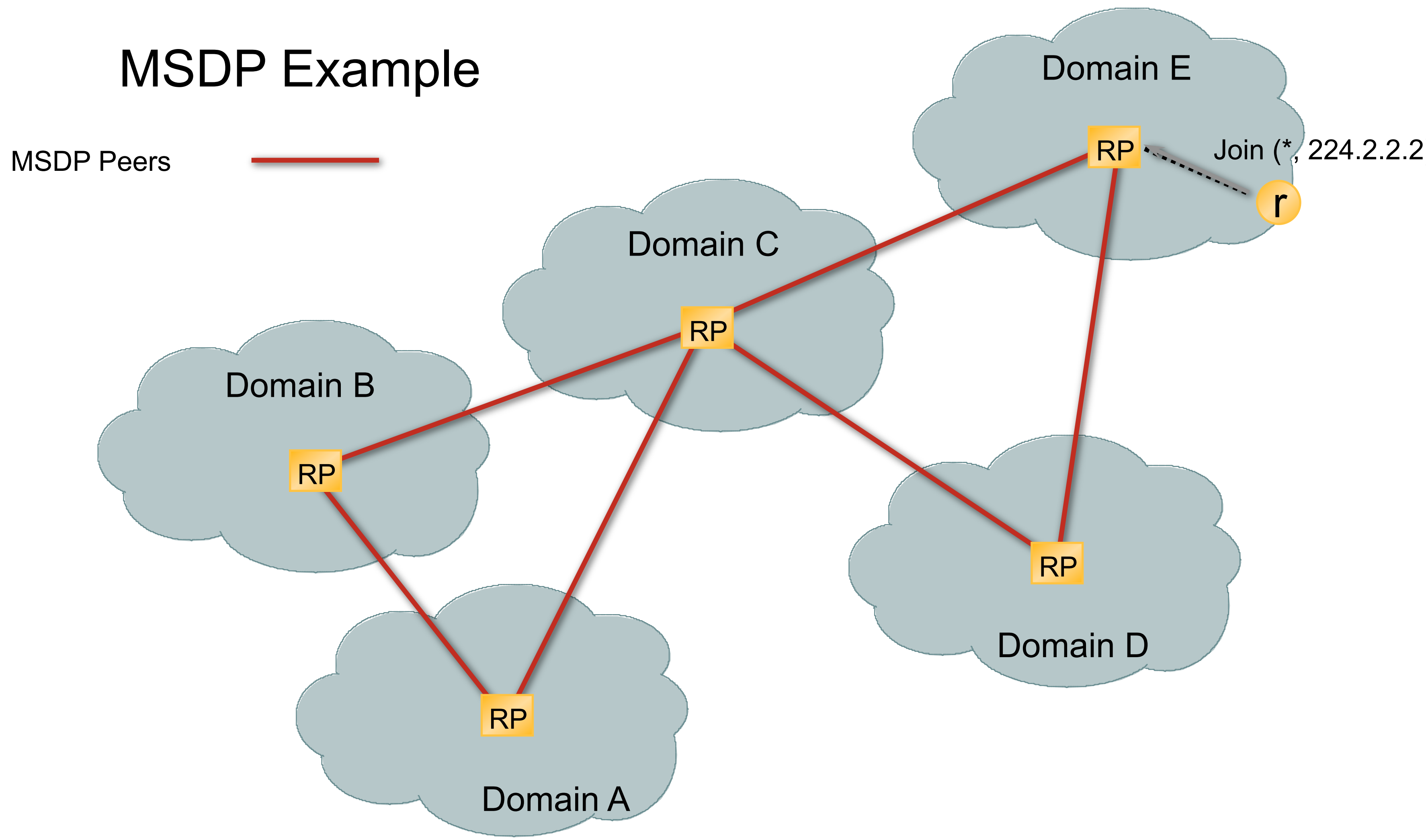
Agenda

- Introduction
- Multicast addressing
- Group Membership Protocol
- PIM-SM / SSM
- **MSDP**
- MBGP
- Summary

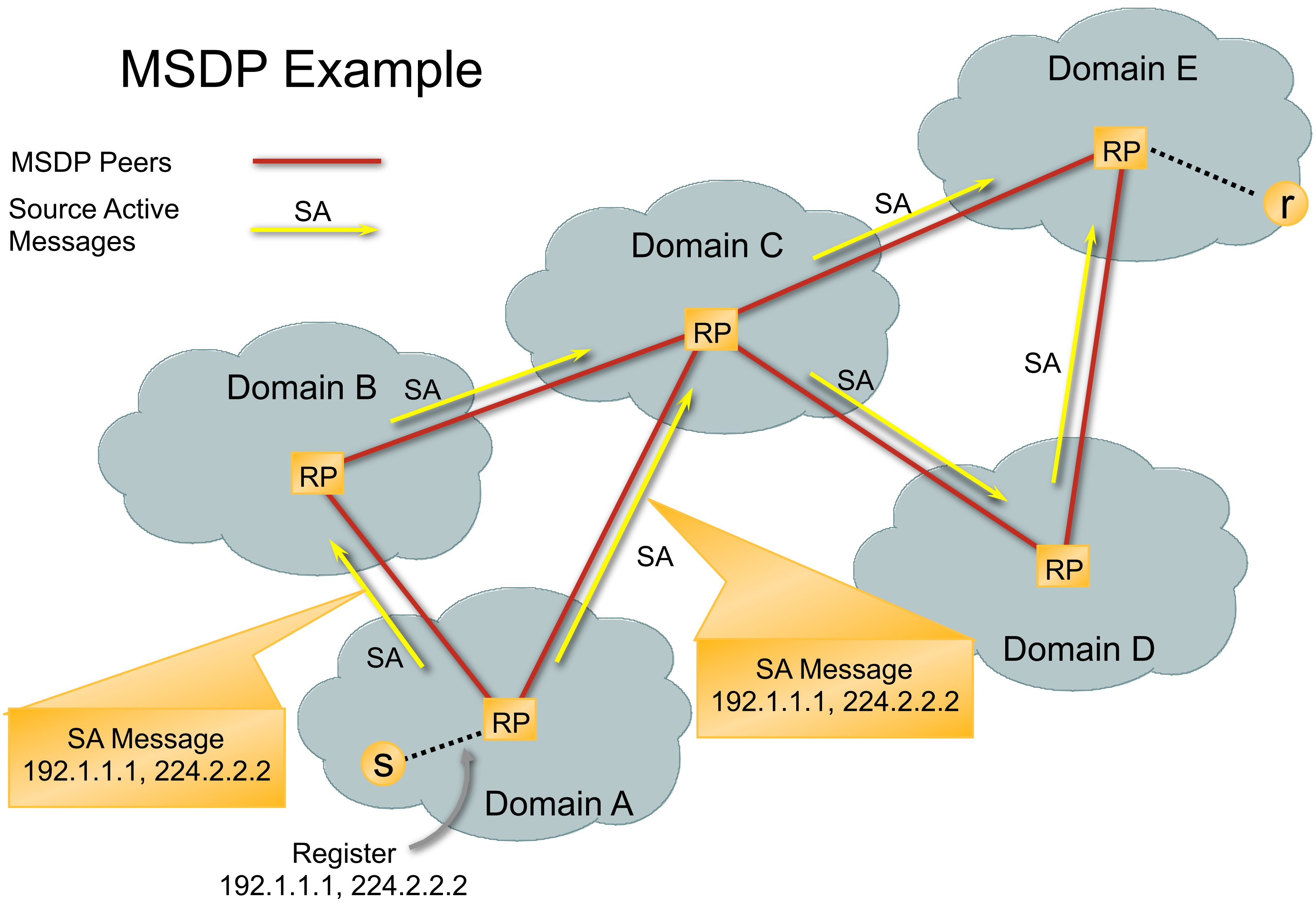
MSDP Overview

- **Uses inter-domain source trees only.**
  - RP's know about all sources in their domain
    - Sources cause a "PIM Register" to the RP
    - Can tell RP's in other domains of its sources
      - Via MSDP SA (Source Active) messages
  - RP's know about receivers in their domain
    - Receivers cause a "(*, G) Join" to the RP
    - RP can join the source tree in the peer domain
      - Via normal PIM (S, G) joins
      - Only necessary if there are receivers for the group
    - Last-hop routers then join source tree directly.

MSDP Overview

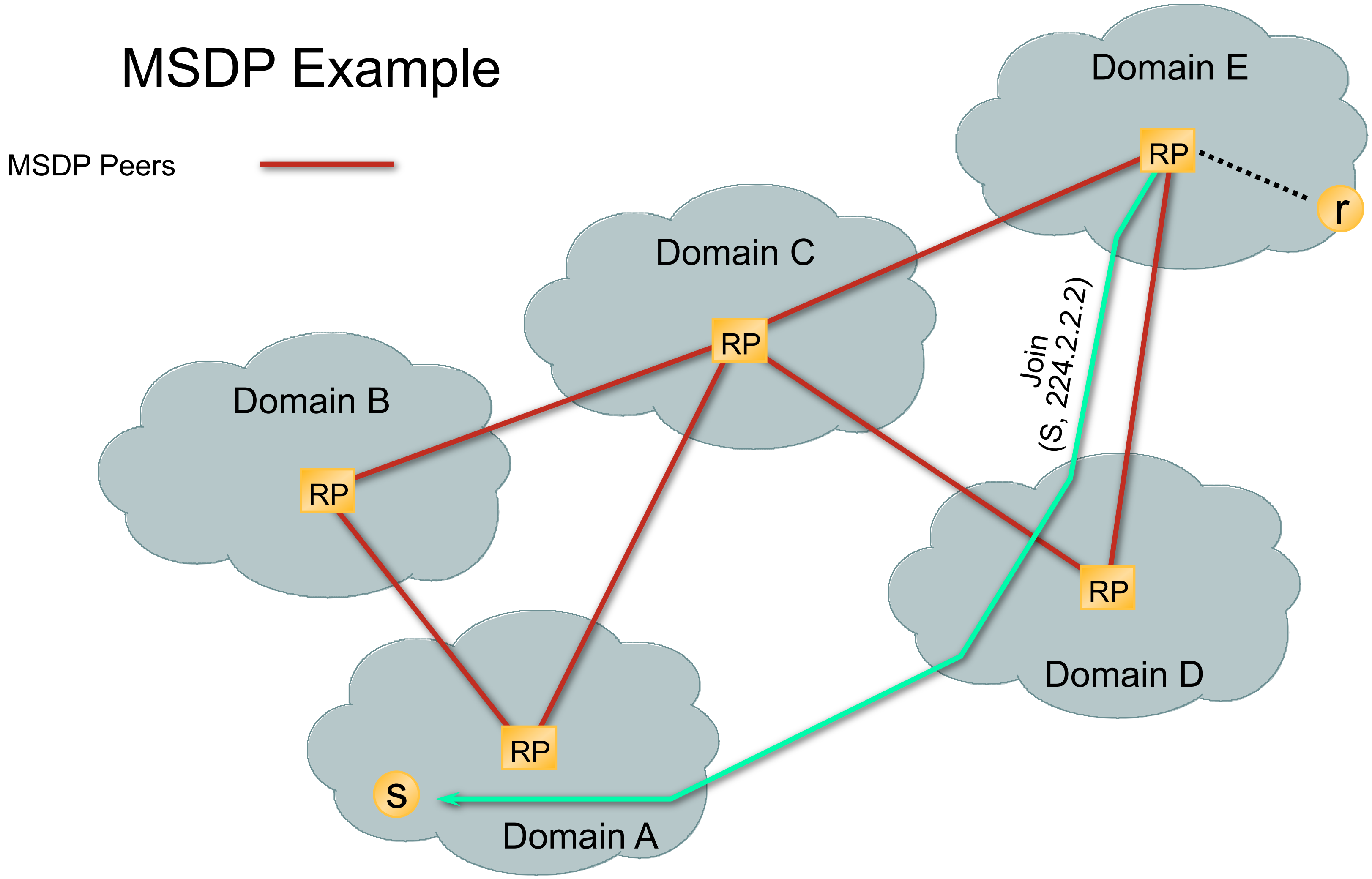


MSDP Overview

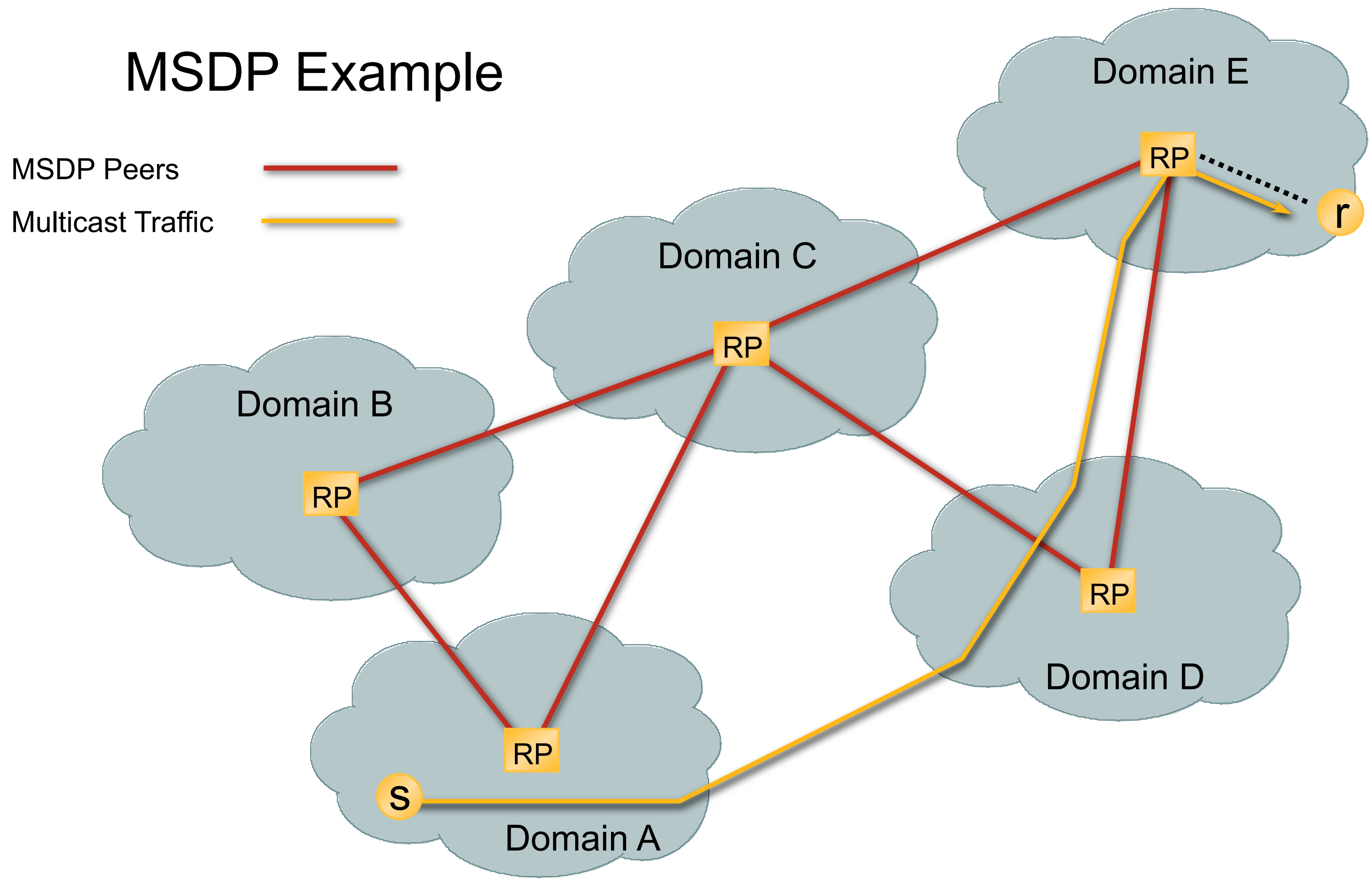


MSDP Overview

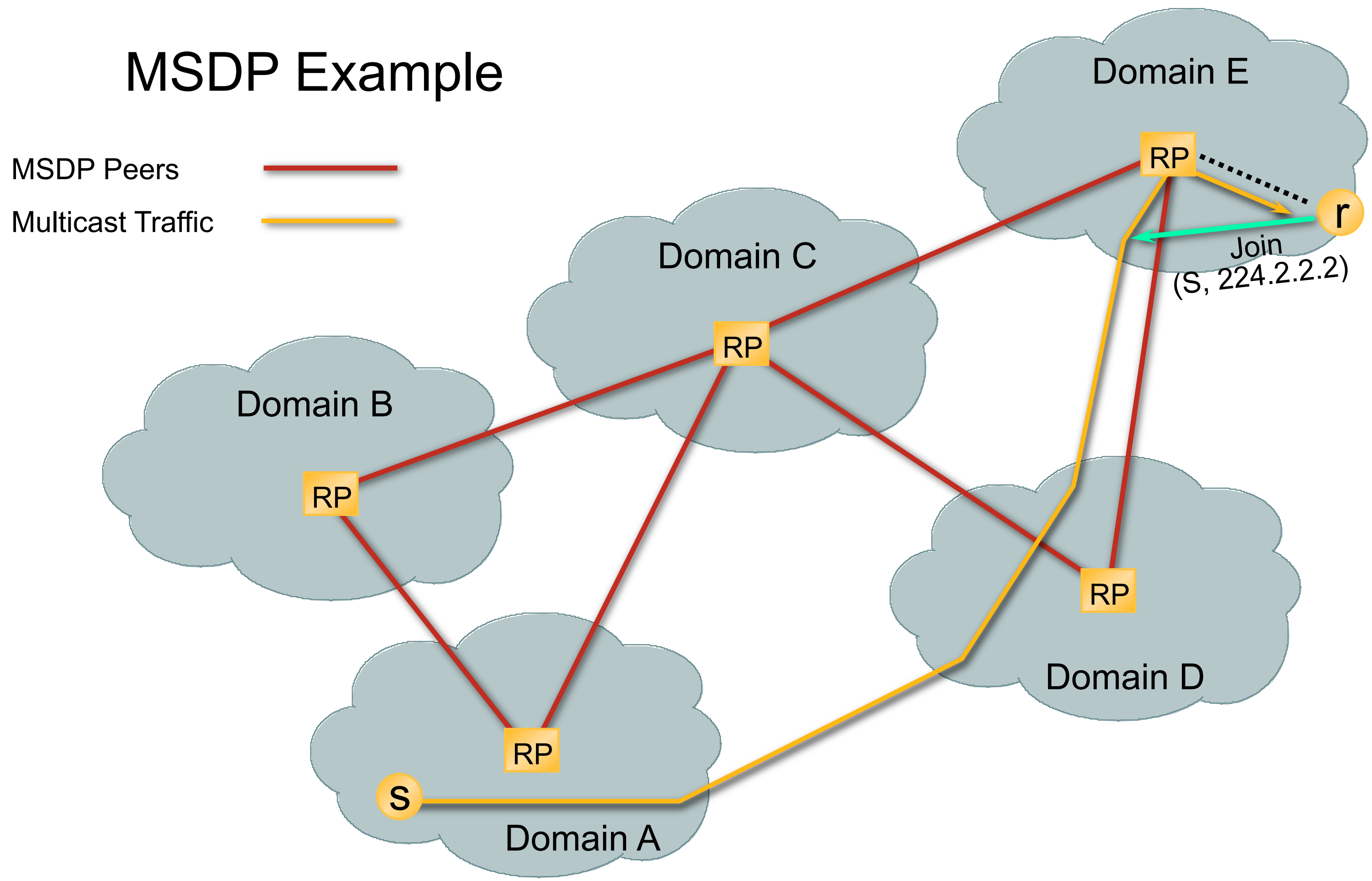
MSDP Example



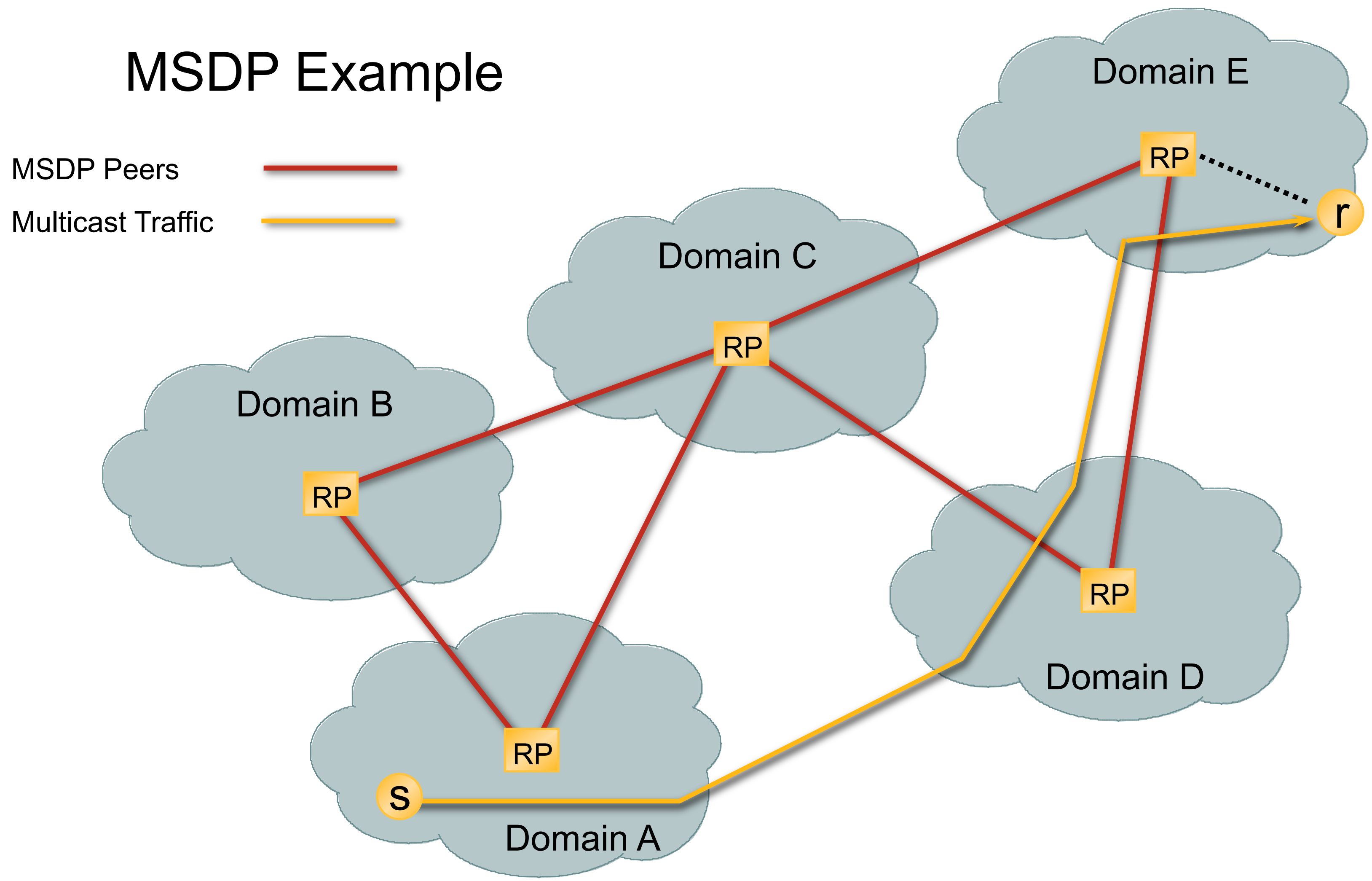
MSDP Overview



MSDP Overview



MSDP Overview



MSDP Peers

- **MSDP Peers configured similar to BGP**
- **Peers connect using TCP port 639**

  Lower address peer initiates connection

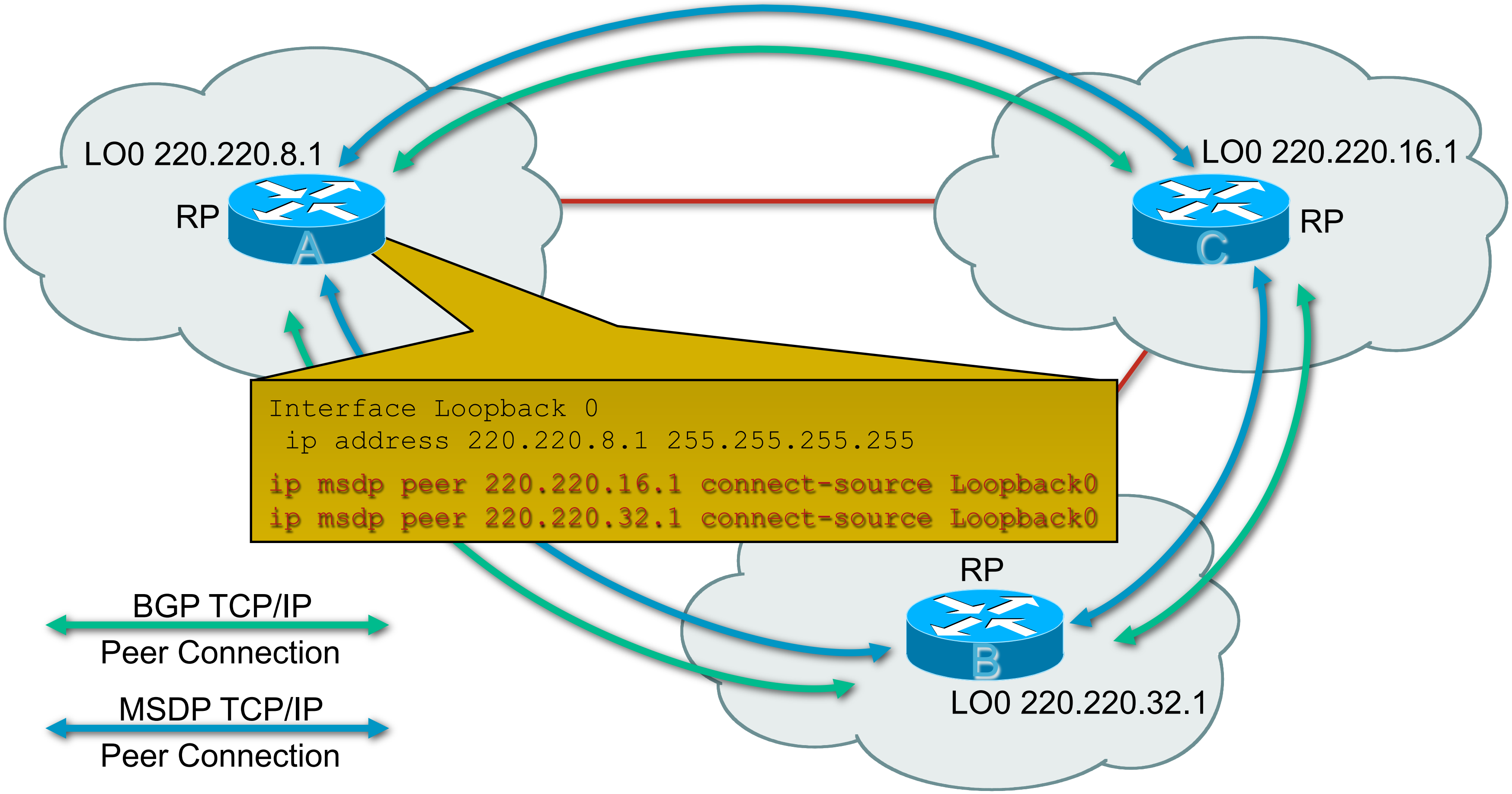
  Higher address peer waits in LISTEN state
- **Peers send keepalives every 60 secs.**
- **Connection reset after 75 seconds**

  If no MSDP packets or keepalives are received

MSDP Peers

- MSDP peers normally *must* run BGP!
  - BGP NLRI is used to RPF check SA messages.
    - May use NLRI from M-Table, U-Table or both.
  - RPF check prevents SA's from looping.
    - (More on that later.)
- Exceptions:
  - When peering with only a single MSDP peer.
  - When using an MSDP Mesh-Group.

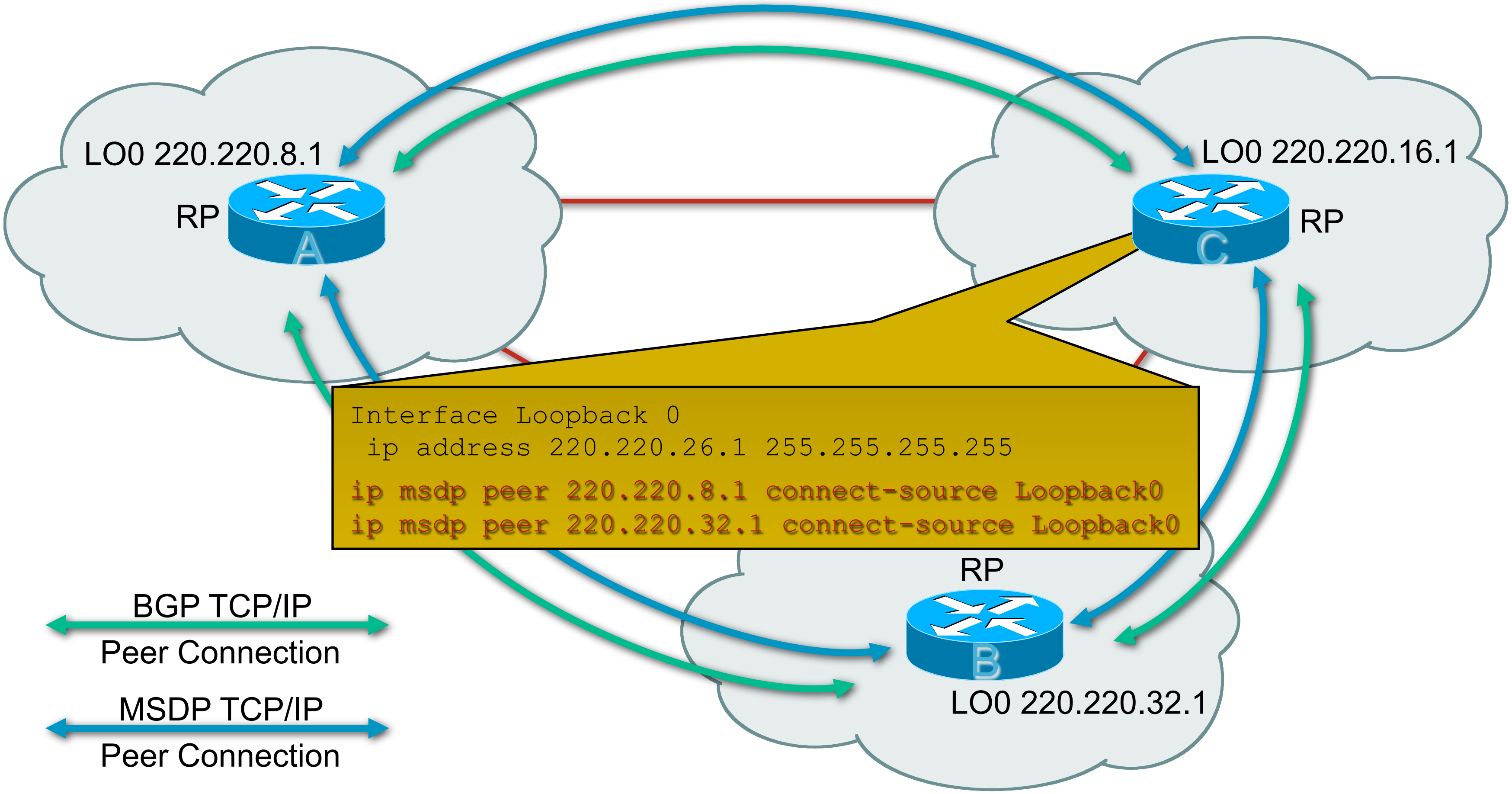
MSDP Peers



- MSDP peer connections are established using the MSDP “peer” configuration command

```
ip msdp peer <ip-address> [connect-source <intfc>]
```

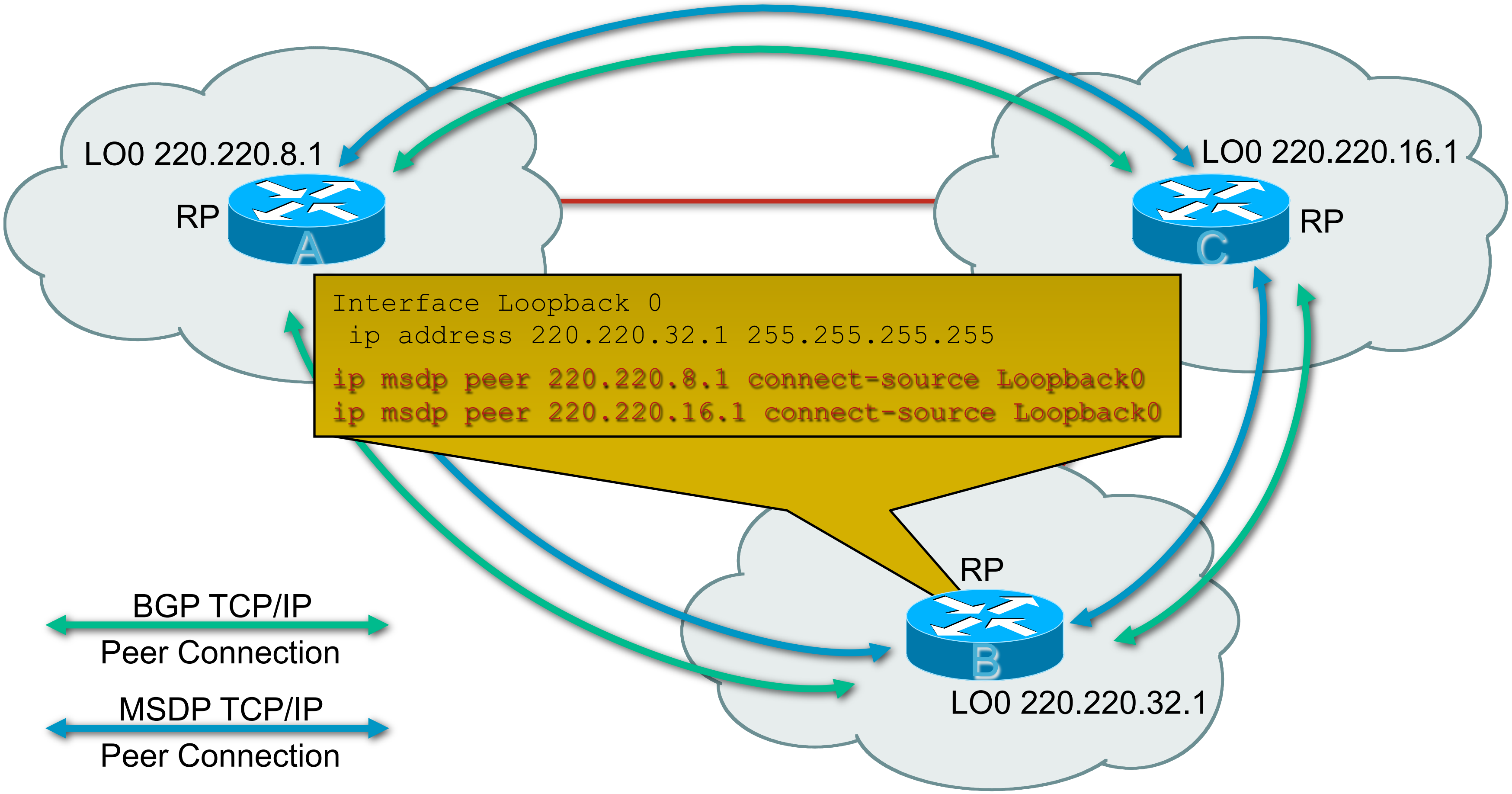
MSDP Peers



- MSDP peer connections are established using the MSDP “peer” configuration command

```
ip msdp peer <ip-address> [connect-source <intfc>]
```

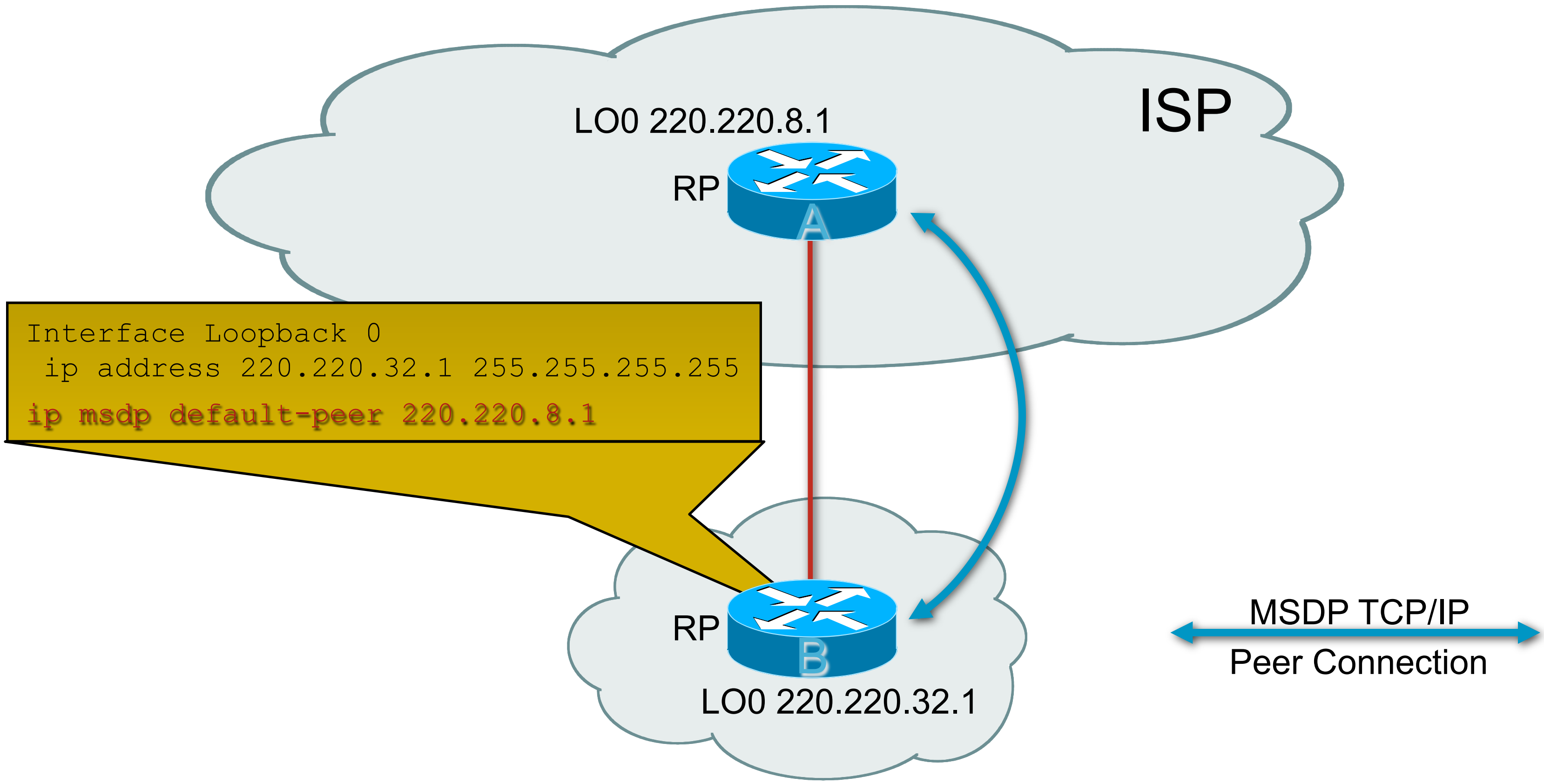
MSDP Peers



- MSDP peer connections are established using the MSDP “peer” configuration command

```
ip msdp peer <ip-address> [connect-source <intfc>]
```

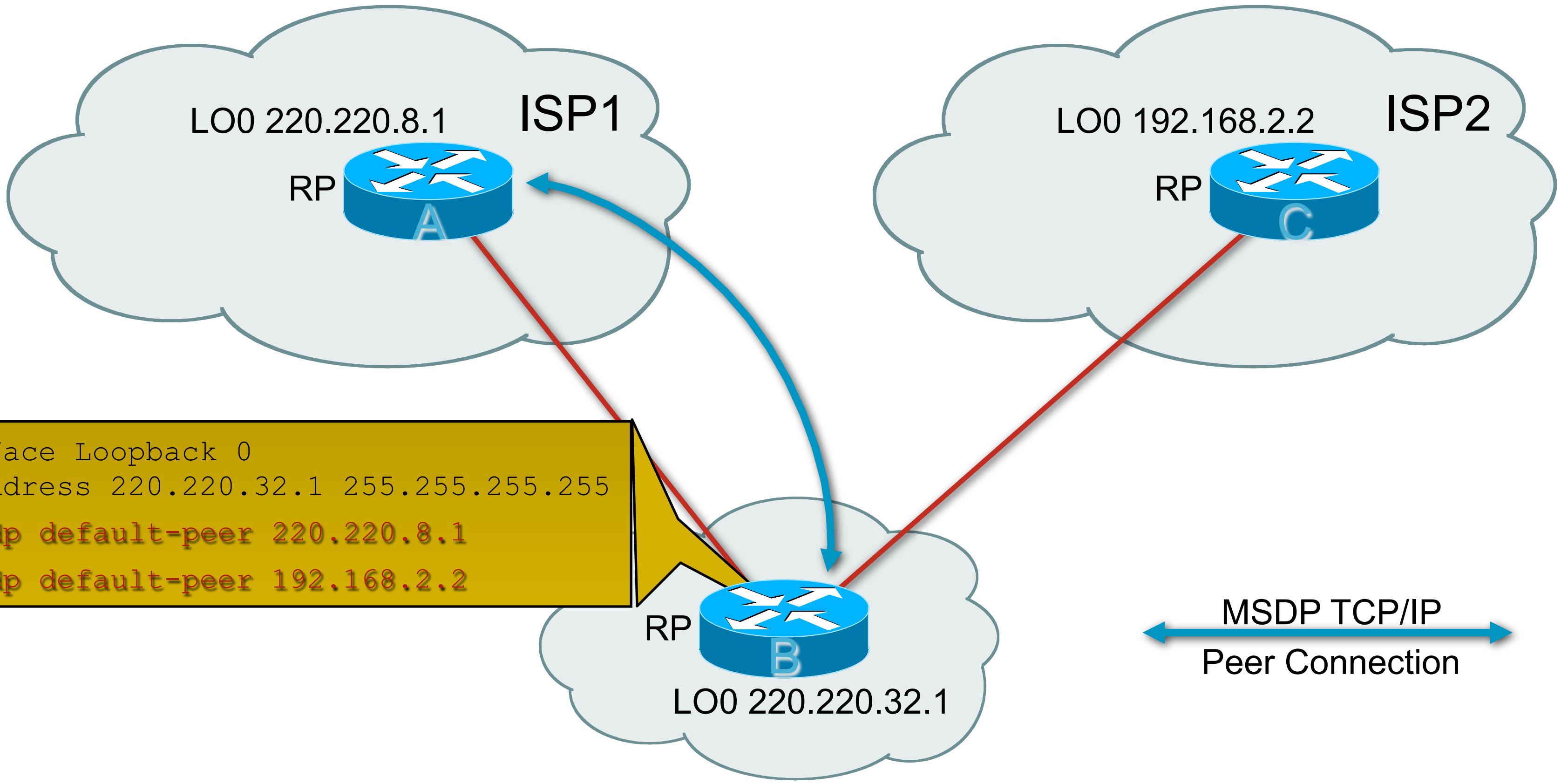
MSDP Peers



- Stub-networks may use "default" peering without being a BGP peer by using the MSDP "default-peer" configuration command.

```
ip msdp default-peer <ip-address>
```

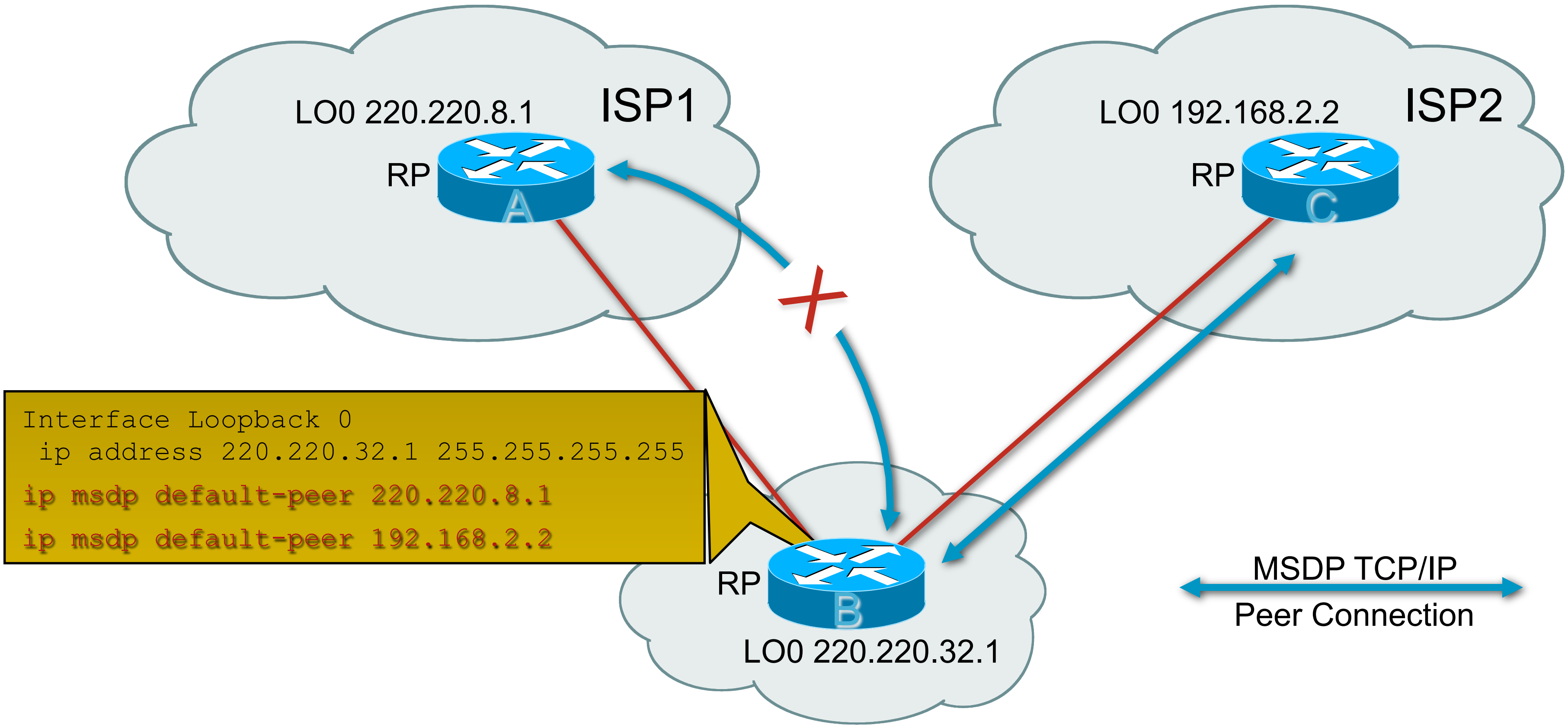

MSDP Peers




```
Interface Loopback 0
 ip address 220.220.32.1 255.255.255.255
ip msdp default-peer 220.220.8.1
ip msdp default-peer 192.168.2.2
```

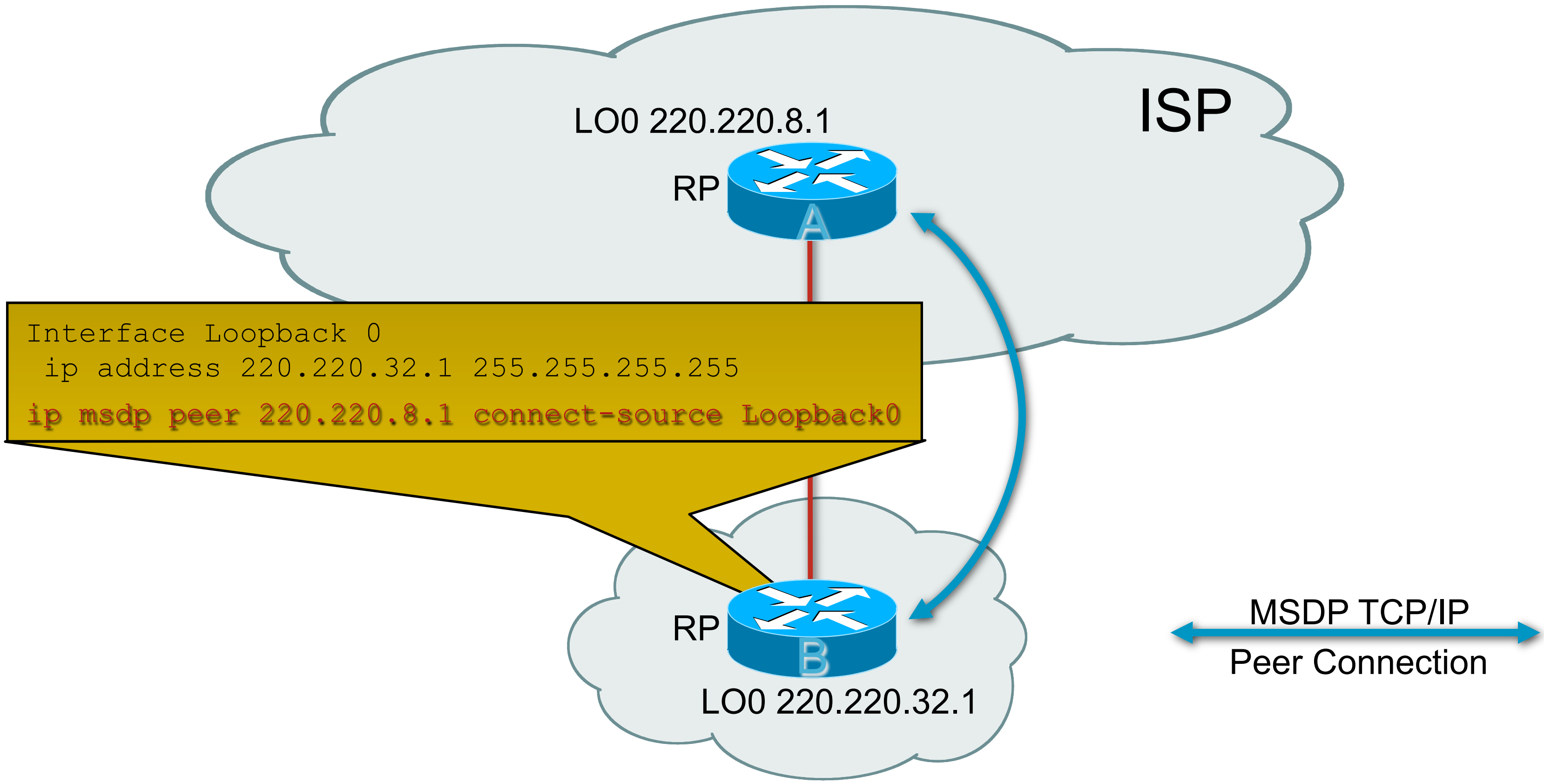
- Multiple “default-peers” may be configured in case connection to first default-peer goes down.

MSDP Peers



- When connection to first 'default-peer' is lost, the next one in the list is tried.

MSDP Peers



- Stub-networks configured with only a single MSDP peer are treated in the same manner as when a single "default-peer" is configured. (i.e. BGP is not required.)

SA Message Contents

- **MSDP Source Active (SA) Messages**
  - Used to advertise active Sources in a domain
  - Can also carry 1st multicast packet from source
    - Hack for Bursty Sources (a' la SDR)
  - SA Message Contents:
    - IP Address of Originating RP
    - Number of (S, G)'s pairs being advertised
    - List of active (S, G)'s in the domain
    - Encapsulated Multicast packet [optional]

Originating SA Messages

- **Local Sources**
  - RP's only originate SA's for local sources
    - Denoted by the "A" flag on an (S,G) entry on RP
  - A source is local if:
    - The RP received a "Register" for (S, G), or
    - The source is directly connected to RP

Originating SA Messages

- **Use ‘msdp redistribute’ to control what SA’s are originated.**
  - **Think of this as ‘msdp sa-originate-filter’ function**

```
ip msdp redistribute [list <acl>]
                     [asn <aspath-acl>]
                     [route-map <map>]
```

    - **Filter by (S,G) pair using ‘list <acl>’**
    - **Filter by AS-PATH using ‘asn <aspath-acl>’**
    - **Filter based on route-map ‘<map>’**
  - **Omitting all acl’s stops all SA origination**

```
Example: ip msdp redistribute
```

  - **Default: Originate SA’s for all local sources**
    - **If ‘msdp redistribute’ command is not configured**

Originating SA Messages

- **SA messages are triggered when any new source in the local domain goes active.**
  - **Initial multicast packet is encapsulated in an SA message.**
    - **This is an attempt at solving the bursty-source problem**

Originating SA Messages

- **Encapsulating Initial Multicast Packets**

  Can bypass TTL-Thresholds

    Original TTL is inside of data portion of SA message

    SA messages sent via Unicast with TTL = 255

- **Requires special command to control**

```
ip msdp ttl-threshold <peer-address> <ttl>
```

  Encapsulated multicast packets with a TTL lower than <ttl> for the specific MSDP peer are not forwarded or originated.

Originating SA Messages

- **Once a minute**
  - **Router scans mroute table**
  - **If group = sparse AND router = RP for group**
    - **For each (S,G) entry for the group:**
      - **If the ‘`msdp redistribute`’ filters permits**
      - **AND if the source is a local source**
      - **Then originate an SA message for (S,G)**

Receiving SA Messages

If SA message RPF checks OK
- Store in SA Cache
- If new SA cache entry
  - Immediately flood SA downstream
  - Set entry’s SA-expire-timer to 6 minutes.
  - If RP for group and receivers exist
    - Create (S,G) entry and trigger (S,G) Join
- If existing entry
  - Reset entry’s SA-expire-timer to 6 minutes.
    - When timer = zero, entry has expired and is deleted.

Else
- Discard SA

SA Message Cache

- **Enabling SA Caching**

```
ip msdp cache-sa-state [list <acl>]
```

Caching is now on by default.

- **Beginning with IOS versions 12.1(7), 12.0(14)S1.**
  - **Cannot be turned off.**

Router caches all SA messages.

- **Cached (S, G) entries timeout after 6 minutes.**
  - **If not refreshed by another (S,G) SA message.**

Once per minute, router scans SA cache.

- **Sends SA downstream for each entry in cache.**

SA Message Caching

- Listing the contents of the SA Cache

    `show ip msdp sa-cache [<group-or-source>] [<asn>]`

```
sj-mbone# show ip msdp sa-cache
MSDP Source-Active Cache - 1997 entries
(193.92.8.77, 224.2.232.0), RP 194.177.210.41, MBGP/AS 5408, 00:01:51/00:04:09
(128.119.167.221, 224.77.0.0), RP 128.119.3.241, MBGP/AS 1249, 06:40:59/00:05:12
(147.228.44.30, 233.0.0.1), RP 195.178.64.113, MBGP/AS 2852, 00:04:48/00:01:11
(128.117.16.142, 233.0.0.1), RP 204.147.128.141, MBGP/AS 145, 00:00:41/00:05:18
(132.250.95.60, 224.253.0.1), RP 138.18.100.1, MBGP/AS 668, 01:15:07/00:05:55
(128.119.40.229, 224.2.0.1), RP 128.119.3.241, MBGP/AS 1249, 06:40:59/00:05:12
(130.225.245.71, 227.37.32.1), RP 130.225.245.71, MBGP/AS 1835, 1d00h/00:05:29
(194.177.210.41, 227.37.32.1), RP 194.177.210.41, MBGP/AS 5408, 00:02:53/00:03:07
(206.190.42.106, 236.195.60.2), RP 206.190.40.61, MBGP/AS 5779, 00:07:27/00:04:04
                              .
                              .
                              .
```

- Clearing the contents of the SA Cache

    `clear ip msdp sa-cache [<group-address> | group-name]`

Filtering Incoming/Outgoing SA Messages

- **SA Filter Command:**

```
ip msdp sa-filter {in|out} <peer-address> [list <acl>]
                                           [route-map <map>]
```

Filters (S,G) pairs to / from peer based on specified ACL.

Can filter based on AS-Path by using optional route-map clause with a path-list acl.

You can filter flooded and originated SA's based on a specific peer, incoming and outgoing.

- **Caution: Filtering SA messages can break the Flood and Join mechanism!**

Recommended MSDP SA Filter

```
! domain-local applications
access-list 111 deny   ip any host 224.0.2.2    !
access-list 111 deny   ip any host 224.0.1.3    ! Rwhod
access-list 111 deny   ip any host 224.0.1.24   ! Microsoft-ds
access-list 111 deny   ip any host 224.0.1.22   ! SVRLOC
access-list 111 deny   ip any host 224.0.1.2    ! SGI-Dogfight
access-list 111 deny   ip any host 224.0.1.35   ! SVRLOC-DA
access-list 111 deny   ip any host 224.0.1.60   ! hp-device-disc
!-- auto-rp groups
access-list 111 deny   ip any host 224.0.1.39
access-list 111 deny   ip any host 224.0.1.40
!-- scoped groups
access-list 111 deny   ip any 239.0.0.0 0.255.255.255
!-- loopback, private addresses (RFC 1918)
access-list 111 deny   ip 10.0.0.0 0.255.255.255 any
access-list 111 deny   ip 127.0.0.0 0.255.255.255 any
access-list 111 deny   ip 172.16.0.0 0.15.255.255 any
access-list 111 deny   ip 192.168.0.0 0.0.255.255 any
access-list 111 permit ip any any
!-- Default SSM-range. Do not do MSDP in this range
access-list 111 deny   ip any 232.0.0.0 0.255.255.255
access-list 111 permit ip any any
```

See “ftp://ftp-eng.cisco.com/ipmulticast/msdp-sa-filter.txt” for the latest updates to this list.

SA Message RPF Checking

- **Purpose**

  Accept SA's via a single deterministic path

  Ignore all other arriving SA's

  Necessary to prevent SA's from looping endlessly

- **Problem**

  Need to know MSDP topology of Internet

  But, MSDP does not distribute topology data!

- **Solution**

  Use BGP data to *infer* MSDP topology.

  Impact:

  The MSDP topology must follow BGP topology.

  An MSDP peer must *generally* also be an BGP peer

SA Message RPF Checking

- **RPF Check Rules depend on peering**

  Rule 1: Sending MSDP peer = iBGP peer

  Rule 2: Sending MSDP peer = eBGP peer

  Rule 3: Sending MSDP peer != BGP peer

- **Exceptions:**

  RPF check is skipped when:

  Sending MSDP peer = Originating RP

  Sending MSDP peer = Mesh-Group peer

  Sending MSDP peer = only MSDP peer

  (i.e. the 'default-peer' or the only 'msdp-peer' configured.)

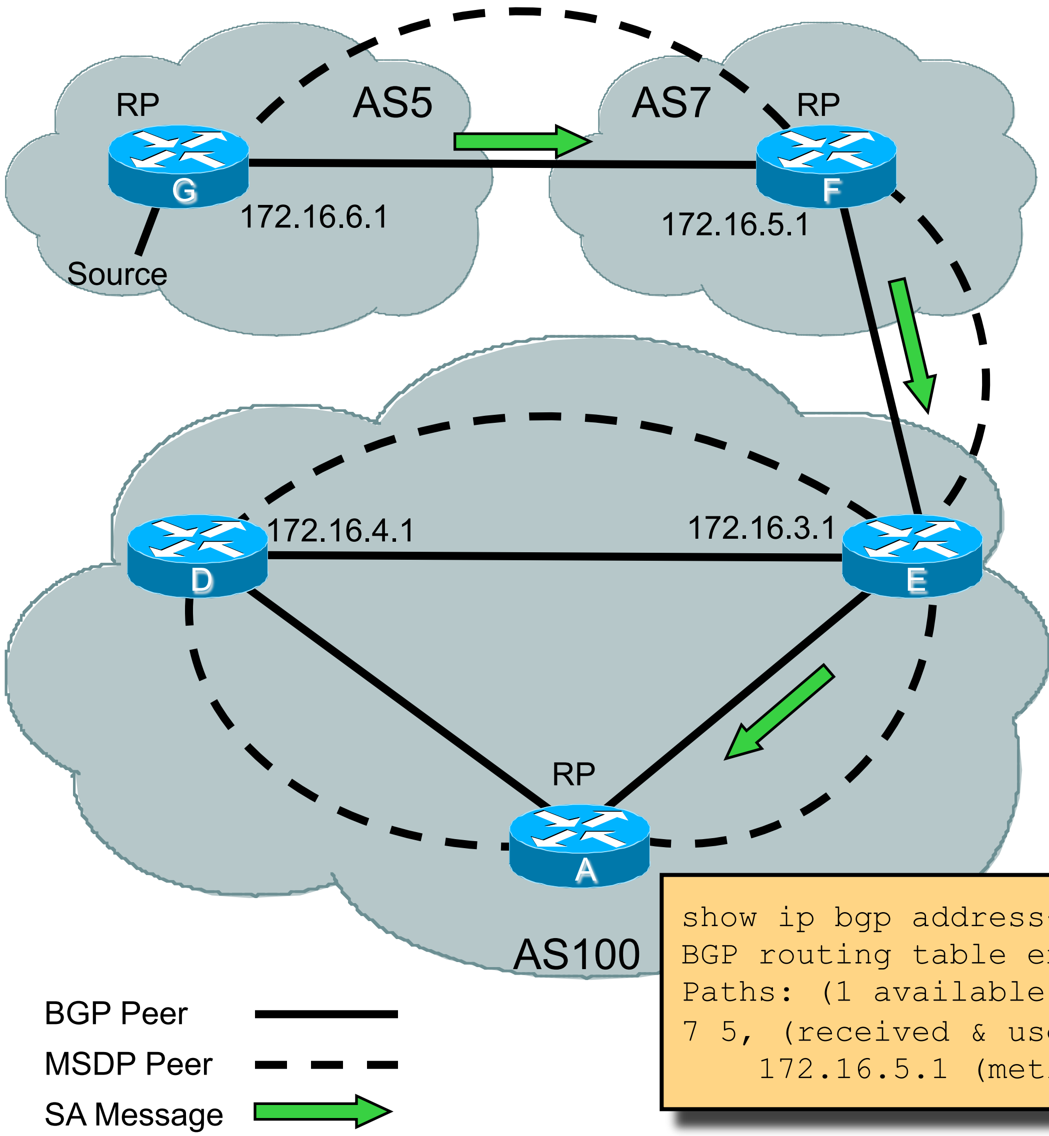
SA Message RPF Checking

- **Determining Applicable RPF Rule**
  - **Use IP address of sending MSDP peer**
    - **Find BGP neighbor w/matching IP address**
    - **IF (no match found)**
      - **Apply Rule 3**
    - **IF (matching neighbor = iBGP peer)**
      - **Apply Rule 1**
    - **ELSE {matching neighbor = eBGP peer}**
      - **Apply Rule 2**
- ***Implication***
  - ***The MSDP peer address must be configured using the same IP address as the BGP peer!***

RPF Check Rule 1

- **When MSDP peer = iBGP peer**
  - **Find "Best Path" to RP in BGP Tables**
    - **Search M-Table first then U-Table.**
    - **If no path to Originating RP found, RPF Fails**
  - **Note "BGP Neighbor" that advertised path**
    - **(i.e IP Address of BGP peer that sent us this path)**
    - ***Warning:***
      - ***This is not the same as the Next-hop of the path!!!***
      - ***iBGP peers normally do not set Next-hop = Self.***
      - ***This is also not necessarily the same as the Router-ID!***
  - **Rule 1 Test Condition:**
    - **MSDP Peer address = BGP Neighbor address?**
      - **If Yes, RPF Succeeds**

Rule1: MSDP peer = iBGP peer



iBGP peer address = 172.16.3.1
(advertising best-path to RP)

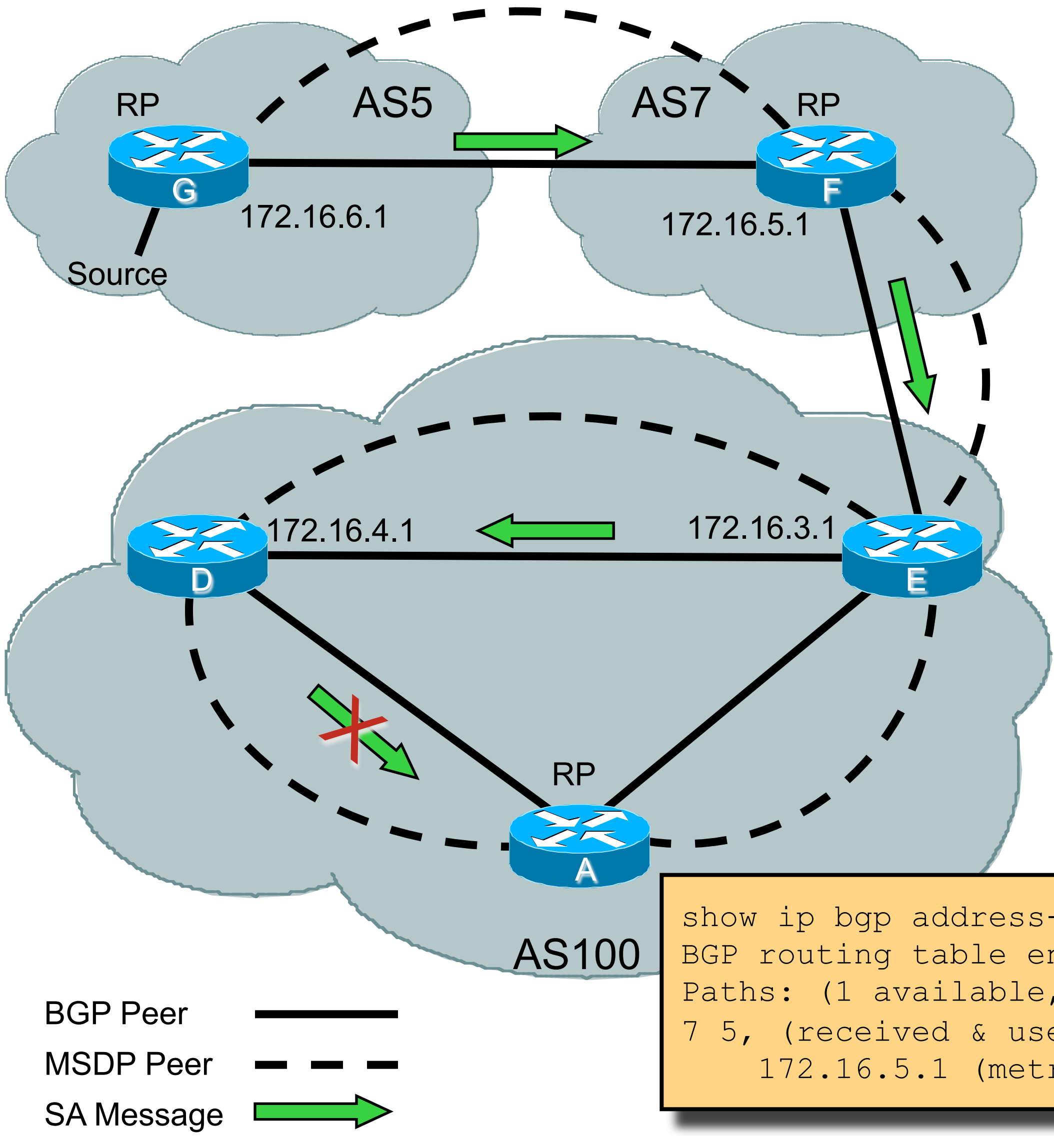
MSDP Peer address = 172.16.3.1

MSDP Peer address = iBGP Peer address

SA RPF Check Succeeds

```
show ip bgp address-family ipv4 multicast 172.16.6.1
BGP routing table entry for 172.16.6.0/24, version 8745118
Paths: (1 available, best #1)
7 5, (received & used)
    172.16.5.1 (metric 68096) from 172.16.3.1 (172.16.3.1)
```

Rule1: MSDP peer = iBGP peer



iBGP Peer address = 172.16.3.1
(advertising best-path to RP)

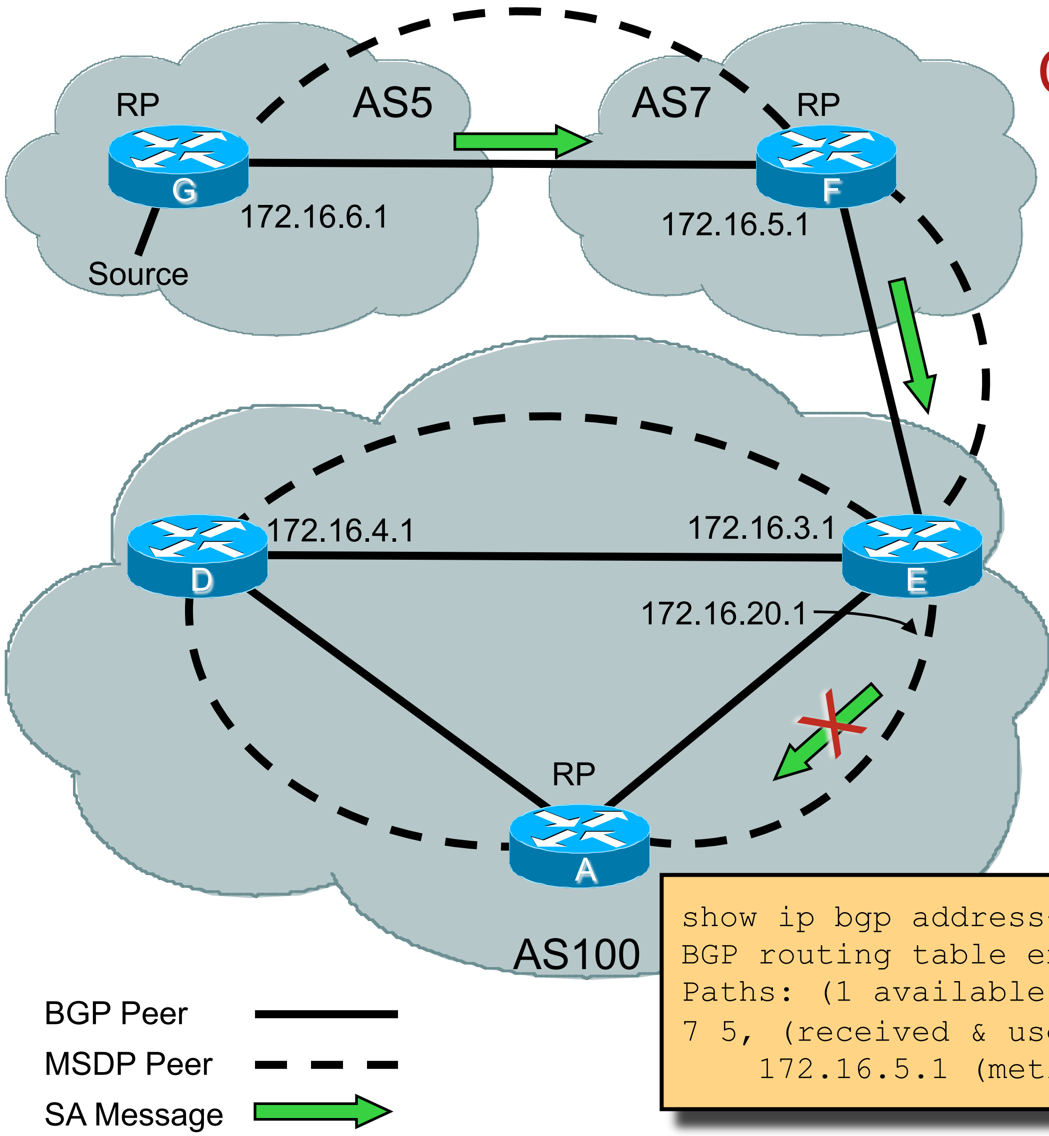
MSDP Peer address = 172.16.4.1

MSDP Peer address != iBGP Peer address

SA RPF Check Fails

```
show ip bgp address-family ipv4 multicast 172.16.6.1
BGP routing table entry for 172.16.6.0/24, version 8745118
Paths: (1 available, best #1)
7 5, (received & used)
   172.16.5.1 (metric 68096) from 172.16.3.1 (172.16.3.1)
```


Rule1: MSDP peer = iBGP peer



Common Mistake #1:

Failure to use same addresses for MSDP peers as iBGP peers!

iBGP Peer address = 172.16.3.1
(advertising best-path to RP)

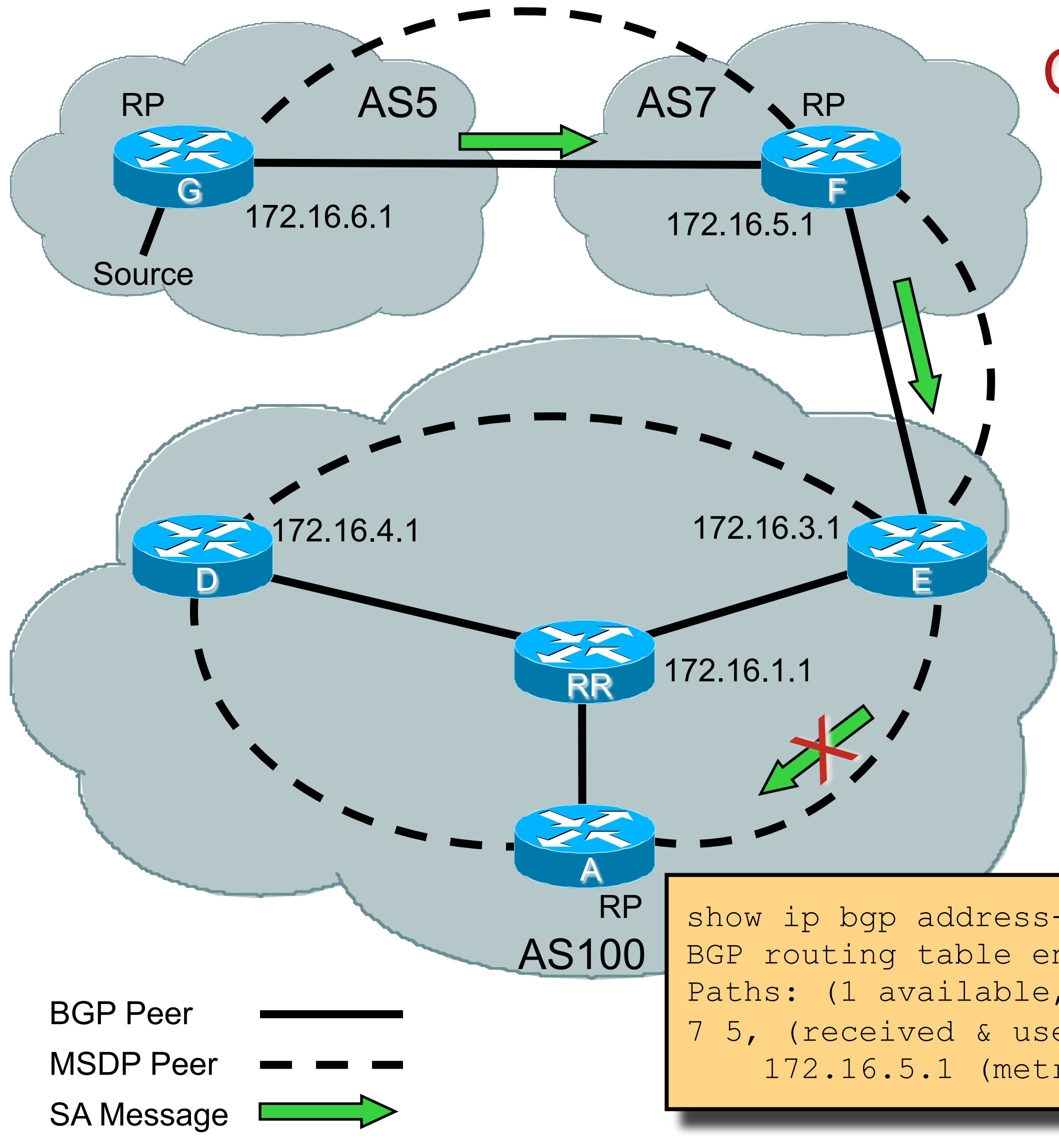
MSDP Peer address = 172.16.20.1

MSDP Peer address != iBGP Peer address

SA RPF Check Fails

```
show ip bgp address-family ipv4 multicast 172.16.6.1
BGP routing table entry for 172.16.6.0/24, version 8745118
Paths: (1 available, best #1)
7 5, (received & used)
   172.16.5.1 (metric 68096) from 172.16.3.1 (172.16.3.1)
```


Rule1: MSDP peer = iBGP peer



Common Mistake #2:

Failure to follow iBGP topology!
Can happen when RR's are used.

iBGP Peer address = 172.16.1.1
(advertising best-path to RP)

MSDP Peer address = 172.16.3.1

MSDP Peer address != iBGP Peer address

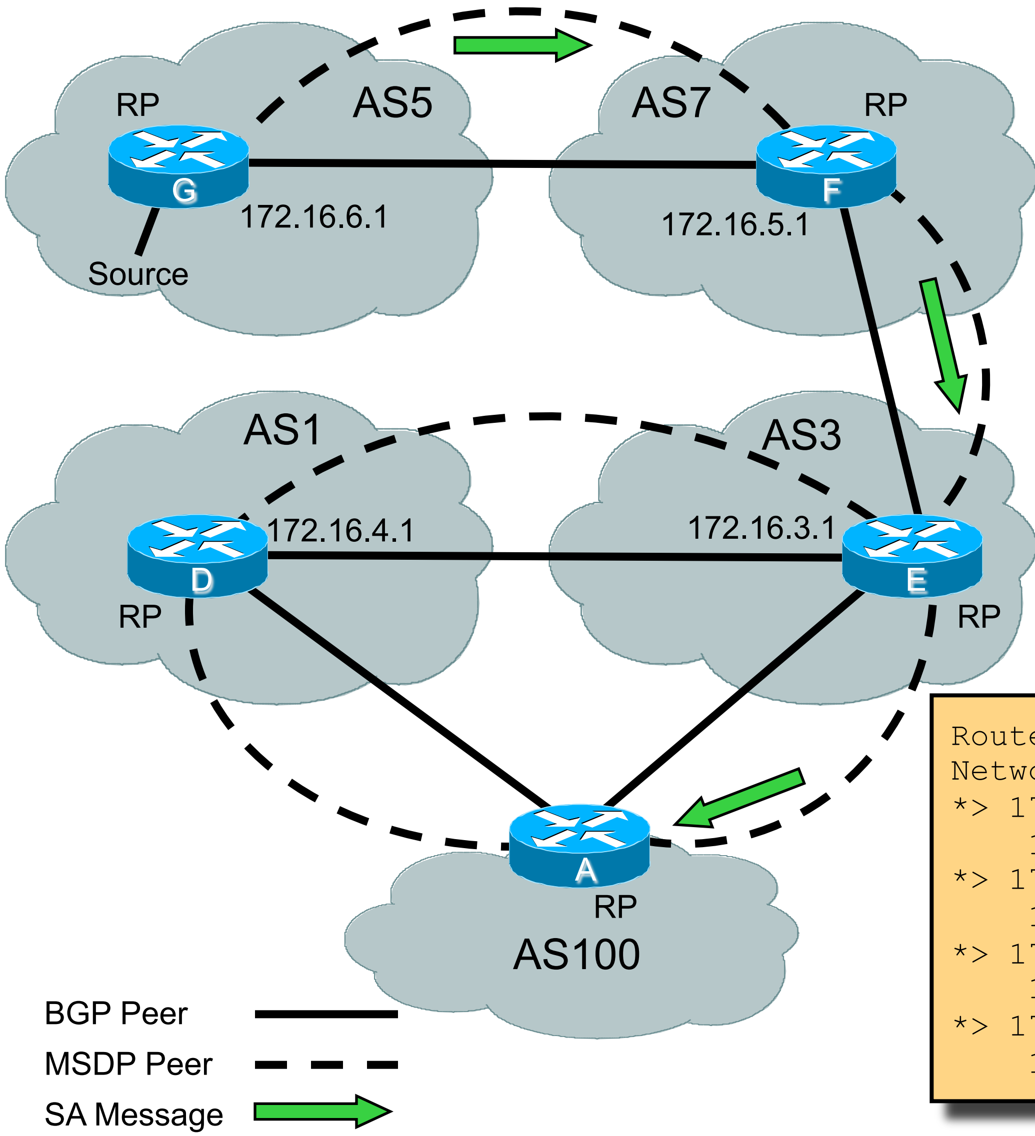
SA RPF Check Fails

```
show ip bgp address-family ipv4 multicast 172.16.6.1
BGP routing table entry for 172.16.6.0/24, version 8745118
Paths: (1 available, best #1)
7 5, (received & used)
   172.16.5.1 (metric 68096) from 172.16.1.1 (172.16.1.1)
```

RPF Check Rule 2

- **When MSDP peer = eBGP peer**
  - **Find BGP "Best Path" to RP**
    - **Search M-Table first then U-Table.**
      - **If no path to Originating RP found, RPF Fails**
  - **Rule 2 Test Condition:**
    - **First AS in path to the RP = AS of eBGP peer?**
      - **If Yes, RPF Succeeds**

Rule2: MSDP peer = eBGP peer



First-AS in best-path to RP = 3
AS of MSDP Peer = 3

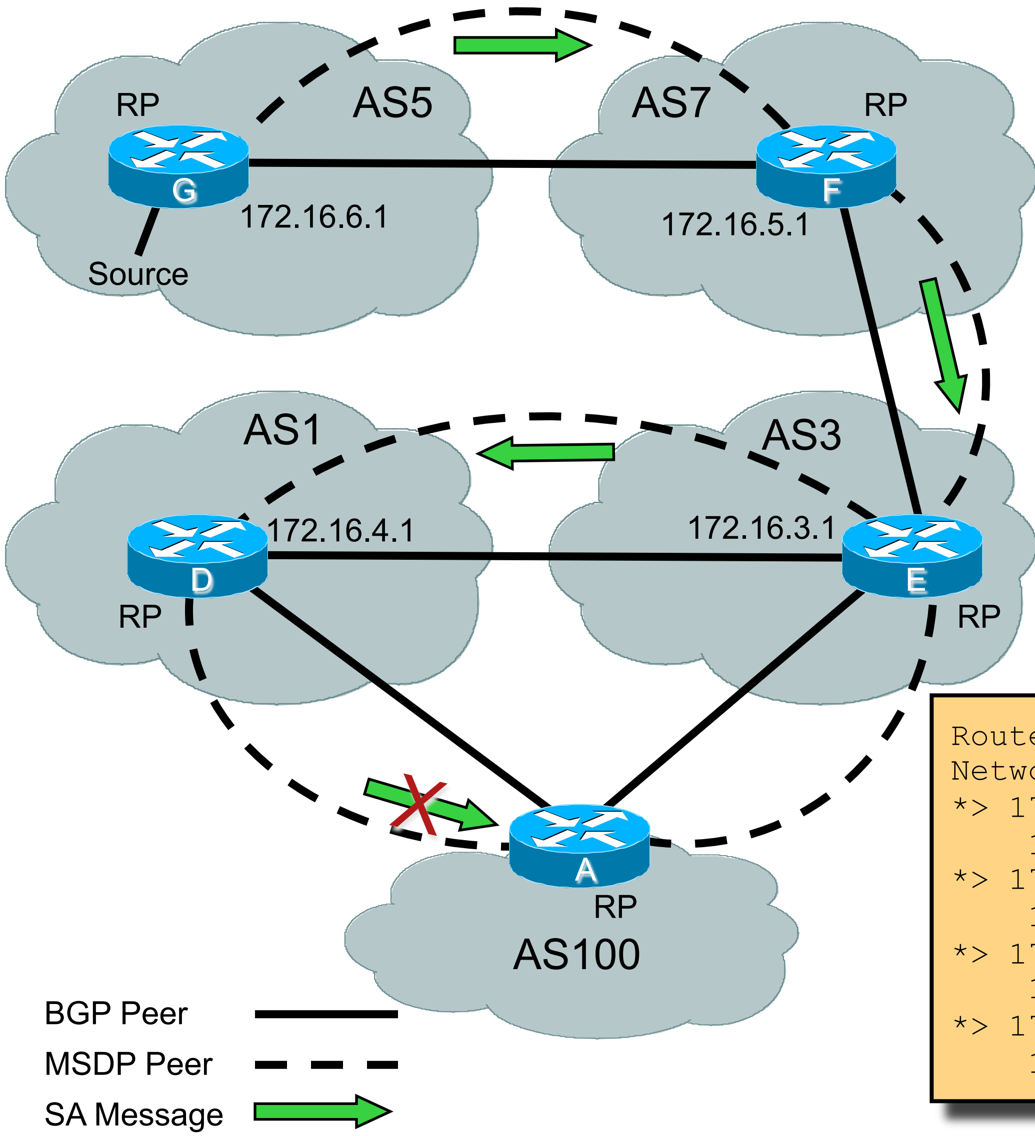
First-AS in best-path to RP = AS of eBGP Peer

SA RPF Check Succeeds

Router A's ipv4 multicast BGP Table

	Network	Next Hop	Path
*>	172.16.3.0/24	172.16.3.1	3 i
	172.16.3.0/24	172.16.4.1	1 3 i
*>	172.16.4.0/24	172.16.4.1	1 i
	172.16.4.0/24	172.16.3.1	3 1 i
*>	172.16.5.0/24	172.16.3.1	3 7 i
	172.16.5.0/24	172.16.4.1	1 3 7 i
*>	172.16.6.0/24	172.16.3.1	3 7 5 i
	172.16.6.0/24	172.16.4.1	1 3 7 5 i

Rule2: MSDP peer = eBGP peer



First-AS in best-path to RP = 3
AS of eBGP Peer = 1

First-AS in best-path to RP != AS of eBGP Peer

SA RPF Check Fails!

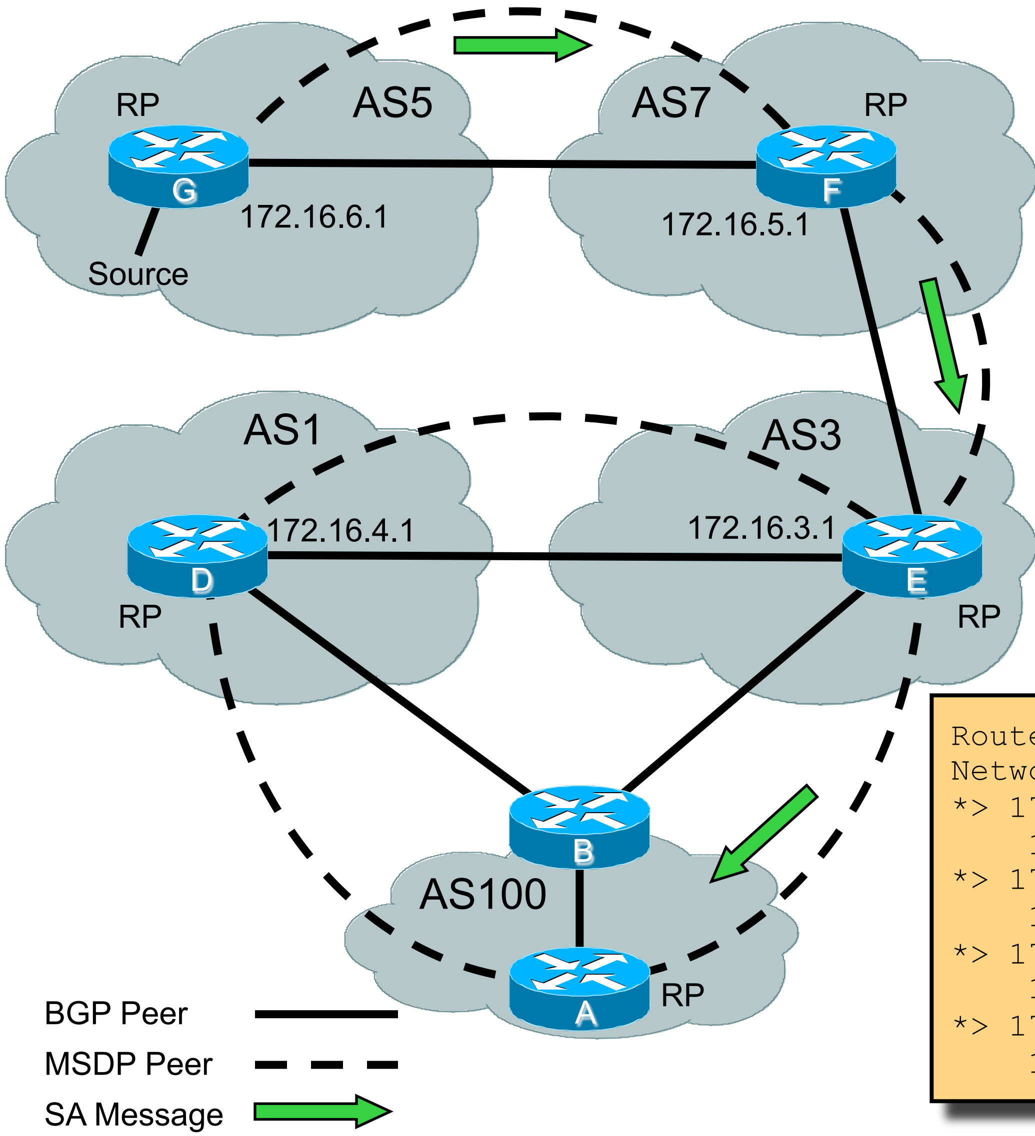
Router A's ipv4 multicast BGP Table

Network	Next Hop	Path
*> 172.16.3.0/24	172.16.3.1	3 i
172.16.3.0/24	172.16.4.1	1 3 i
*> 172.16.4.0/24	172.16.4.1	1 i
172.16.4.0/24	172.16.3.1	3 1 i
*> 172.16.5.0/24	172.16.3.1	3 7 i
172.16.5.0/24	172.16.4.1	1 3 7 i
*> 172.16.6.0/24	172.16.3.1	3 7 5 i
172.16.6.0/24	172.16.4.1	1 3 7 5 i

RPF Check Rule 3

- **When MSDP peer != BGP peer**
  - Find BGP "Best Path" to RP
    - Search M-Table first then U-Table.
      - If no path to Originating RP found, RPF Fails
  - Find BGP "Best Path" to MSDP peer
    - Search M-Table first then U-Table.
      - If no path to sending MSDP Peer found, RPF Fails
  - Note AS of sending MSDP Peer
    - Origin AS (last AS) in AS-PATH to MSDP Peer
  - Rule 3 Test Condition:
    - First AS in path to RP = Sending MSDP Peer AS ?
      - If Yes, RPF Succeeds

Rule3: MSDP peer != BGP peer



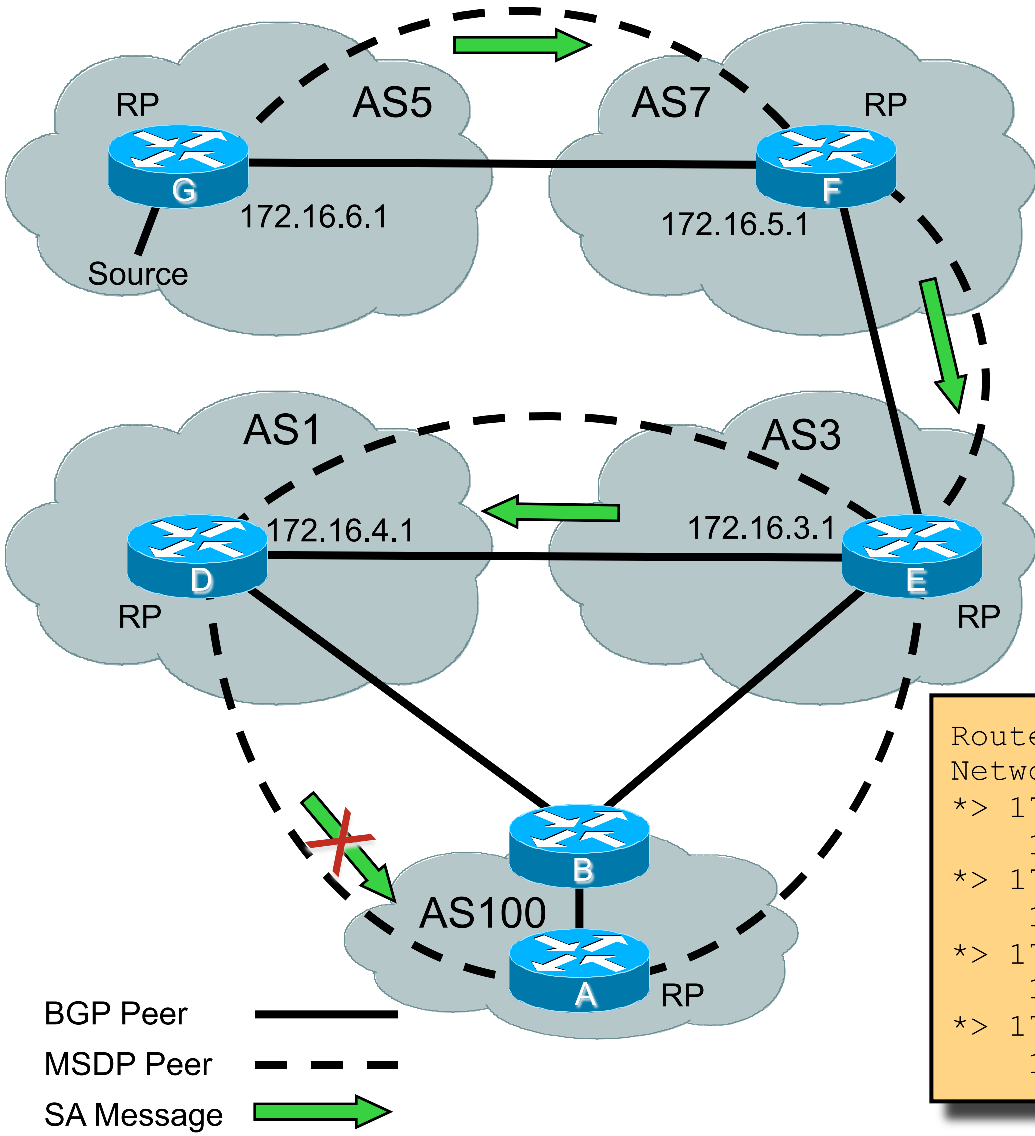
First-AS in best-path to RP = 3
AS of MSDP Peer = 3

First-AS in best-path to RP = AS of MSDP Peer

SA RPF Check Succeeds

```
Router A's ipv4 multicast BGP Table
Network                Next Hop       Path
*> 172.16.3.0/24       172.16.3.1     3 i
     172.16.3.0/24      172.16.4.1    1 3 i
*> 172.16.4.0/24       172.16.4.1     1 i
     172.16.4.0/24      172.16.3.1    3 1 i
*> 172.16.5.0/24       172.16.3.1     3 7 i
     172.16.5.0/24      172.16.4.1    1 3 7 i
*> 172.16.6.0/24       172.16.3.1     3 7 5 i
     172.16.6.0/24      172.16.4.1    1 3 7 5 i
```

Rule3: MSDP peer != BGP peer



First-AS in best-path to RP = 3
AS of MSDP Peer = 1

First-AS in best-path to RP != AS of MSDP Peer

SA RPF Check Fails

```
Router A's ipv4 multicast BGP Table
Network              Next Hop      Path
*> 172.16.3.0/24     172.16.3.1    3 i
   172.16.3.0/24      172.16.4.1   1 3 i
*> 172.16.4.0/24     172.16.4.1    1 i
   172.16.4.0/24      172.16.3.1   3 1 i
*> 172.16.5.0/24     172.16.3.1    3 7 i
   172.16.5.0/24      172.16.4.1   1 3 7 i
*> 172.16.6.0/24     172.16.3.1    3 7 5 i
   172.16.6.0/24      172.16.4.1   1 3 7 5 i
```


MSDP Mesh-Groups

- **Optimises SA flooding.**
  - Useful when 2 or more peers are in a group.
  - Requires full mesh of mesh group peers.
- **Reduces amount of SA traffic in the net.**
  - SA’s not flooded to other mesh-group peers.
- **Suspends RPF check of SA messages.**
  - When received from a mesh-group peer.
  - SA’s always accepted from mesh-group peers.
  - Eliminates need for BGP.

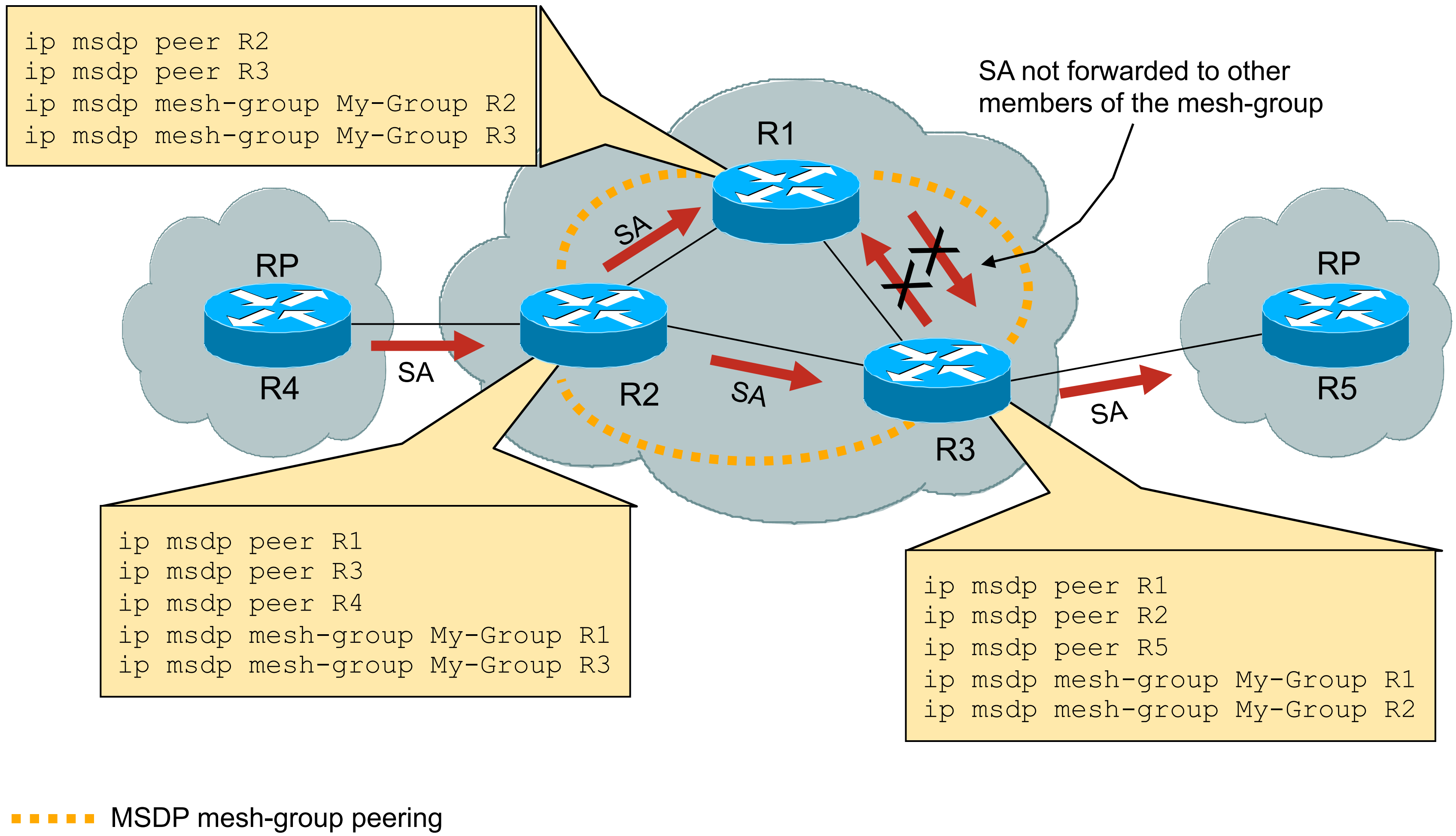
MSDP Mesh-Groups

- Configured with:

```
ip msdp mesh-group <name> <peer-address>
```

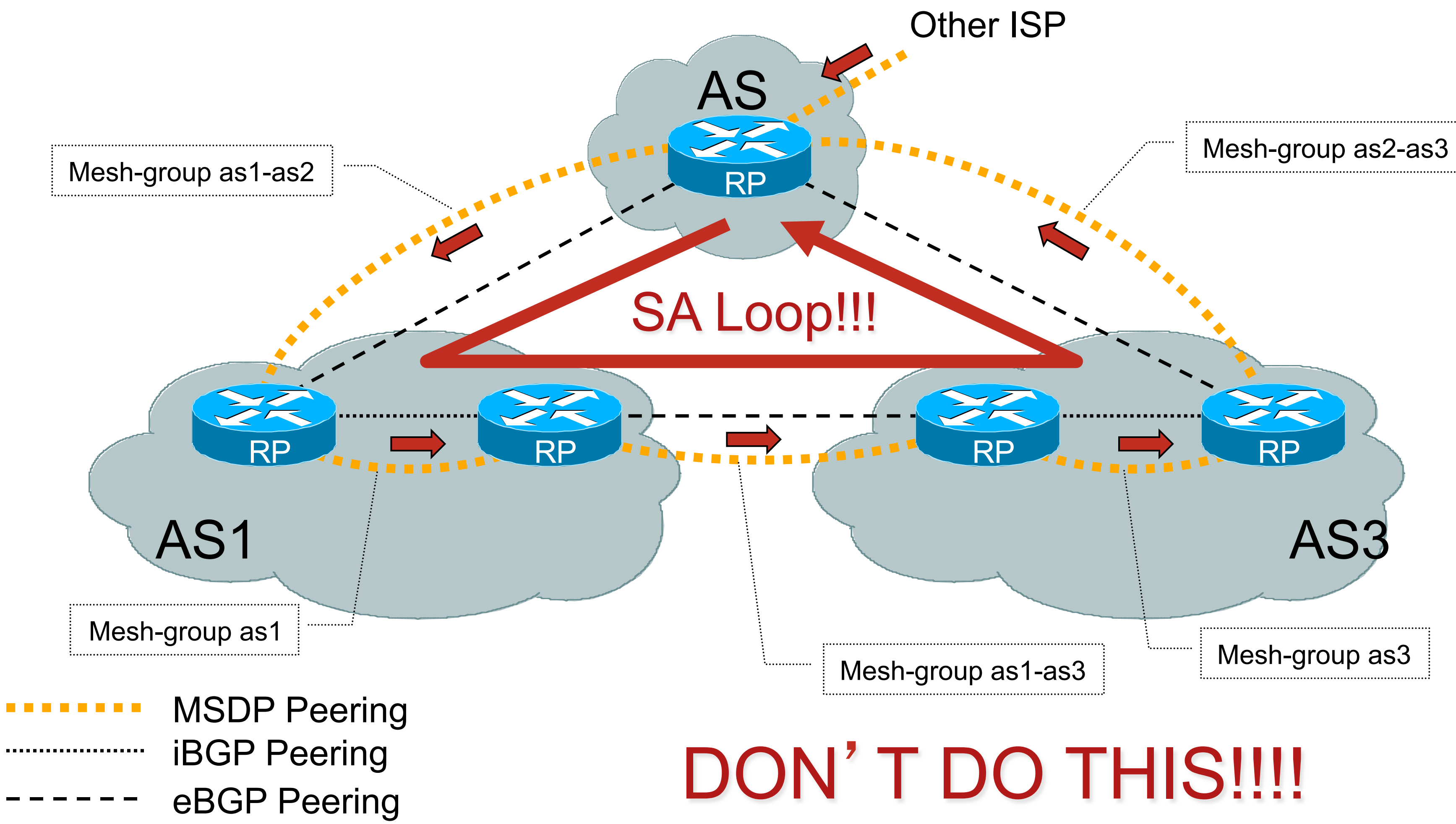
- Peers in the mesh-group must be fully meshed.
- Multiple mesh-groups per router are supported.

MSDP Mesh-Group Example



Avoid Mesh-Group Loops!!!

WARNING: There is no RPF check between Mesh-groups!!!



DON' T DO THIS!!!!

MSDP Mroute Flags

New 'mroute' Flags for MSDP

```
sj-mbone#show ip mroute summary
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, C - Connected, L - Local, P - Pruned
       R - RP-bit set, F - Register flag, T - SPT-bit set, J - Join SPT
       M - MSDP created entry, X - Proxy Join Timer Running
       A - Advertised via MSDP
Timers: Uptime/Expires
Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 224.2.246.13), 5d17h/00:02:59, RP 171.69.10.13, flags: S
  (171.69.185.51, 224.2.246.13), 3d17h/00:03:29,   flags: TA
  (128.63.58.45, 224.2.246.13), 00:02:16/00:00:43, flags: M
  (128.63.58.54, 224.2.246.13), 00:01:16/00:01:43, flags: M
```

"M" flag indicates source was learned via MSDP

"A" flag indicates source is a *candidate* for advertisement by MSDP

MSDP Enhancements

- **New IOS command**

```
ip msdp new-rpf-rules
```

MSDP SA RPF check using IGP

Accept SA's from BGP NEXT HOP

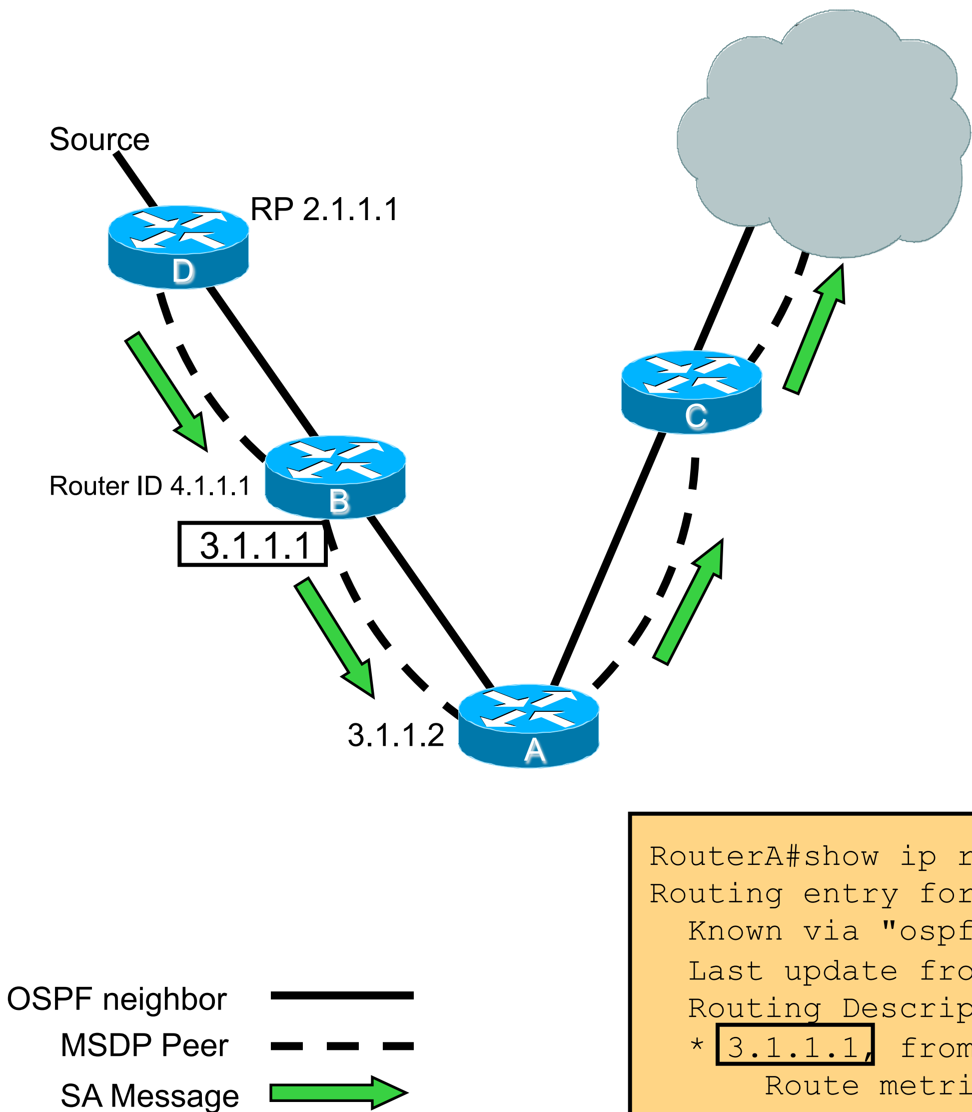
Accept SA's from closest peer along the best path to the originating RP

"show ip msdp rpf"

MSDP RPF check using IGP

- **When MSDP peer = IGP peer (No BGP)**
  - **Find best IGP route to RP**
    - **Search URIB**
      - **If route to Originating RP found and:**
        - **If IGP next hop (or advertiser) address for RP is the MSDP peer and in UP state, then that is the RPF peer.**
        - **If route not found: Fall through to the next rule.**

IGP Rule: MSDP peer = IGP peer (Next hop)



MSDP Peer = 3.1.1.1

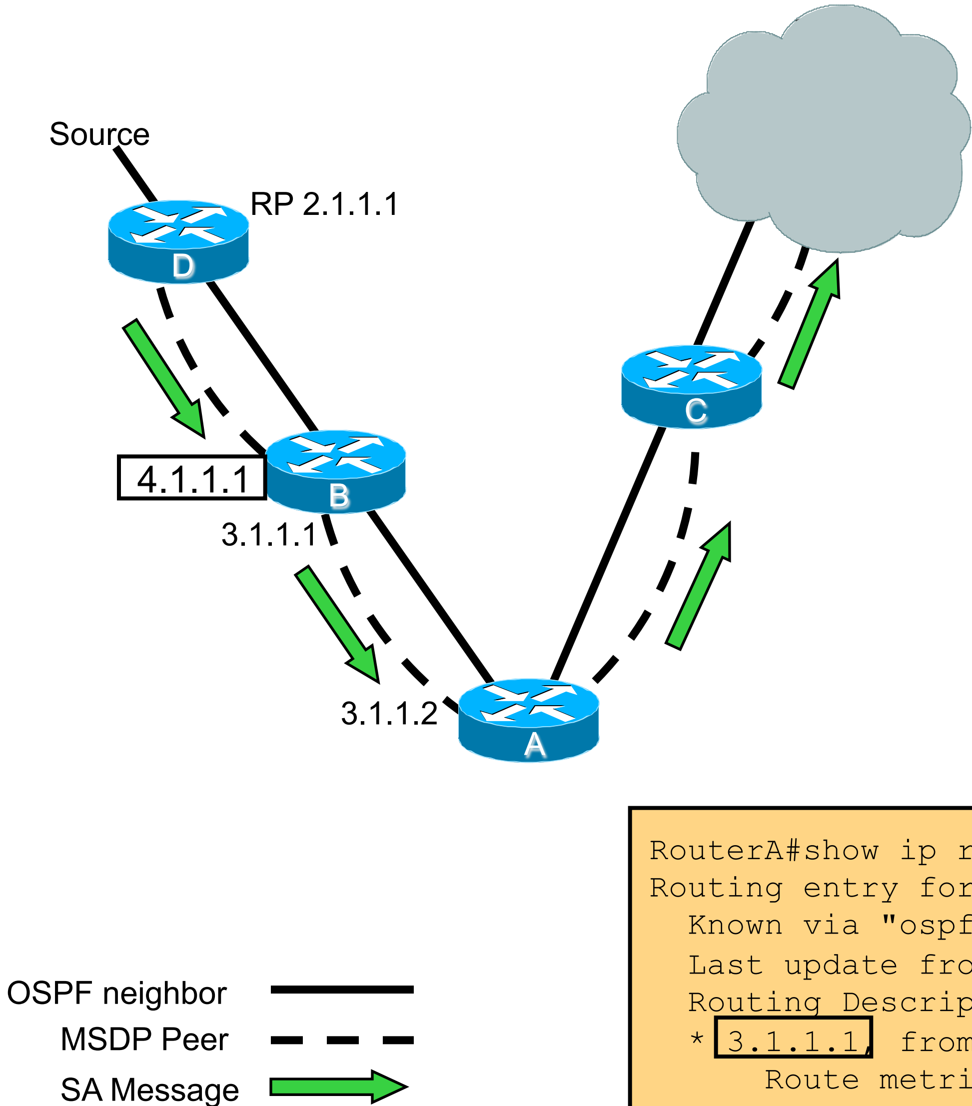
IGP next hop to originating RP = 3.1.1.1

IGP next hop to originating RP = MSDP peer

SA RPF Check Succeeds

```
RouterA#show ip route 2.1.1.1
Routing entry for 2.1.1.0/24
  Known via "ospf 1", distance 110, metric 20, type intra area
  Last update from 3.1.1.1 on Ethernet2, 00:35:10 ago
  Routing Descriptor Blocks:
  * 3.1.1.1, from 4.1.1.1, 00:35:10 ago, via Ethernet2
      Route metric is 20, traffic share count is 1
```

IGP Rule: MSDP peer = IGP peer (Advertiser)



MSDP Peer = 4.1.1.1

IGP next hop to originating RP = ~~3.1.1.1~~

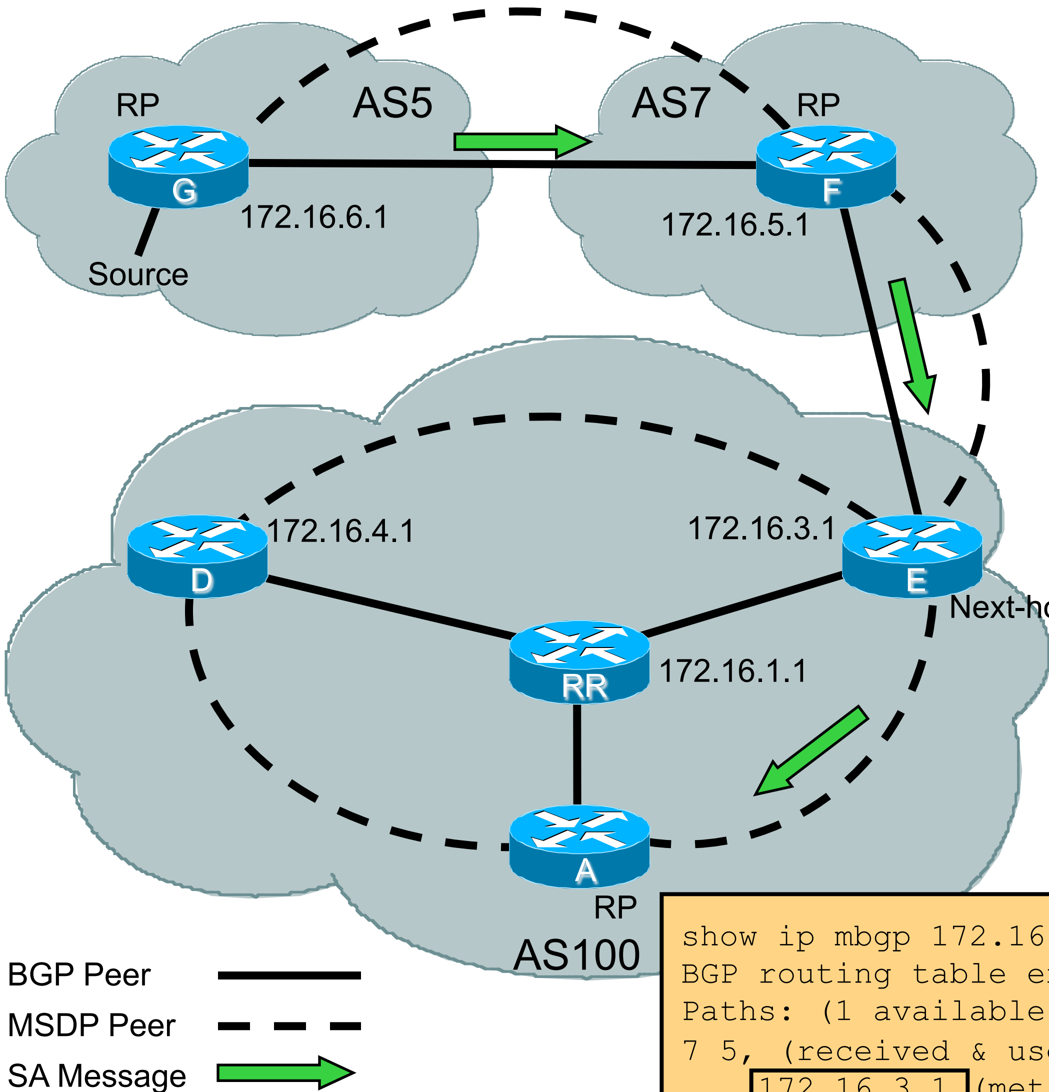
IGP advertiser to originating RP = 4.1.1.1

IGP advertiser to originating RP = MSDP peer

SA RPF Check Succeeds

```
RouterA#show ip route 2.1.1.1
Routing entry for 2.1.1.0/24
  Known via "ospf 1", distance 110, metric 20, type intra area
  Last update from 3.1.1.1 on Ethernet2, 00:35:10 ago
  Routing Descriptor Blocks:
  * 3.1.1.1, from 4.1.1.1, 00:35:10 ago, via Ethernet2
      Route metric is 20, traffic share count is 1
```


SA's accepted from Next Hop



i(m)BGP Peer address = 172.16.1.1
(Advertiser of next hop)

MSDP Peer address = 172.16.3.1

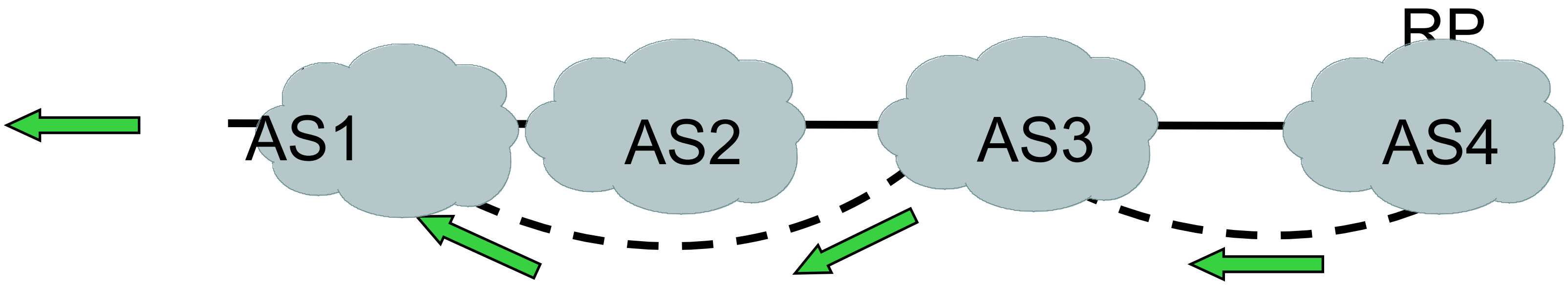
But, BGP next hop = 172.16.3.1

MSDP Peer address = BGP next hop address

SA RPF Check Succeeds

```
show ip mbgp 172.16.6.1
BGP routing table entry for 172.16.6.0/24, version 8745118
Paths: (1 available, best #1)
7 5, (received & used)
    172.16.3.1 (metric 68096) from 172.16.1.1 (172.16.1.1)
```

Accept SA along RPF path



Existing Rule: If first AS in best path to the RP != MSDP peer
RPF Fails

New code: Choose peer in CLOSEST AS along best AS path to th
Loosens rule a bit.

RPF Succeeds.

New MSDP RPF command

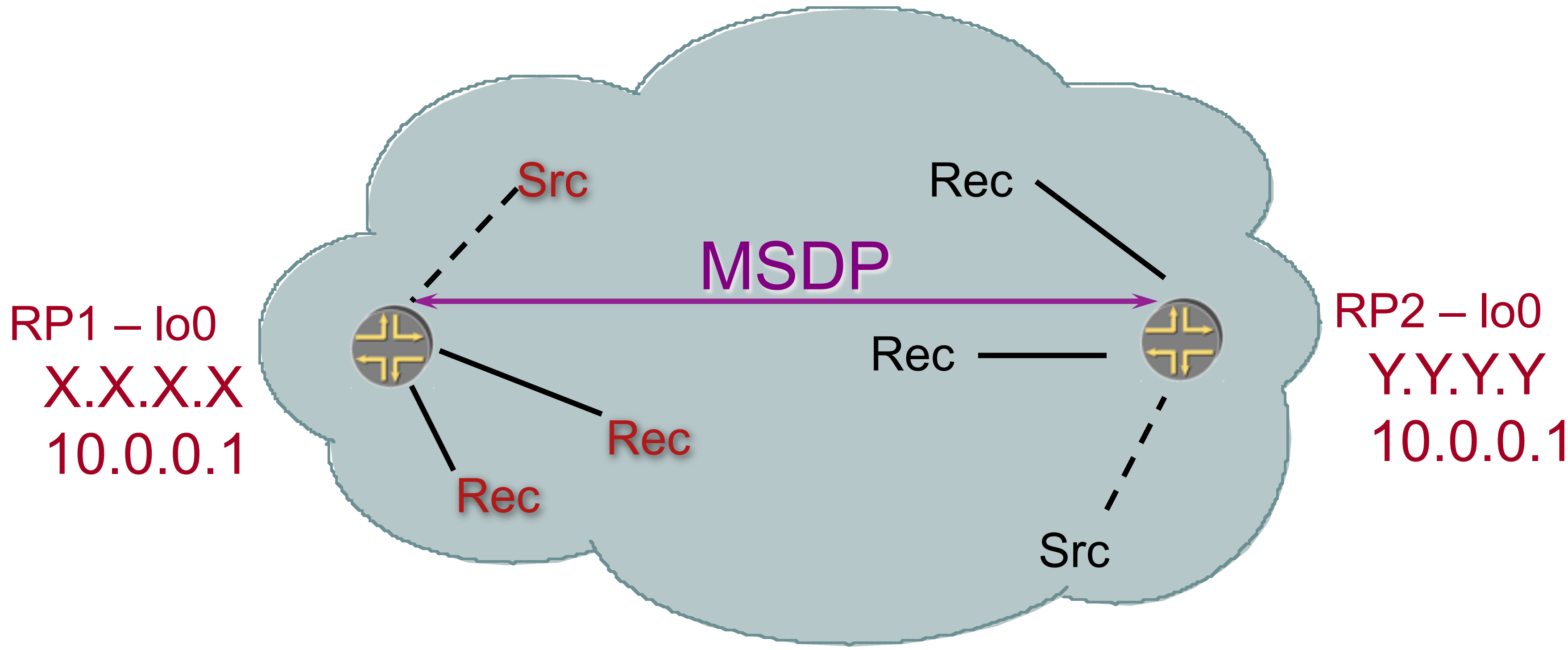
```
Router-A# show ip msdp rpf 2.1.1.1
RPF peer information for Router-B (2.1.1.1)
  RPF peer: Router-C (3.1.1.1)
  RPF route/mask: 2.1.1.0/24
  RPF rule: Peer is IGP next hop of best route
  RPF type: unicast (ospf 1)
```

Anycast-RP

- draft-ietf-mboned-anycast-rp-08.txt
- Within a domain, deploy more than one RP for the same group range
- Sources from one RP are known to other RPs using MSDP
- Give each RP the same /32 IP address
- Sources and receivers use closest RP, as determined by the IGP
- Used intra-domain to provide redundancy and RP load sharing, when an RP goes down, sources and receivers are taken to new RP via unicast routing

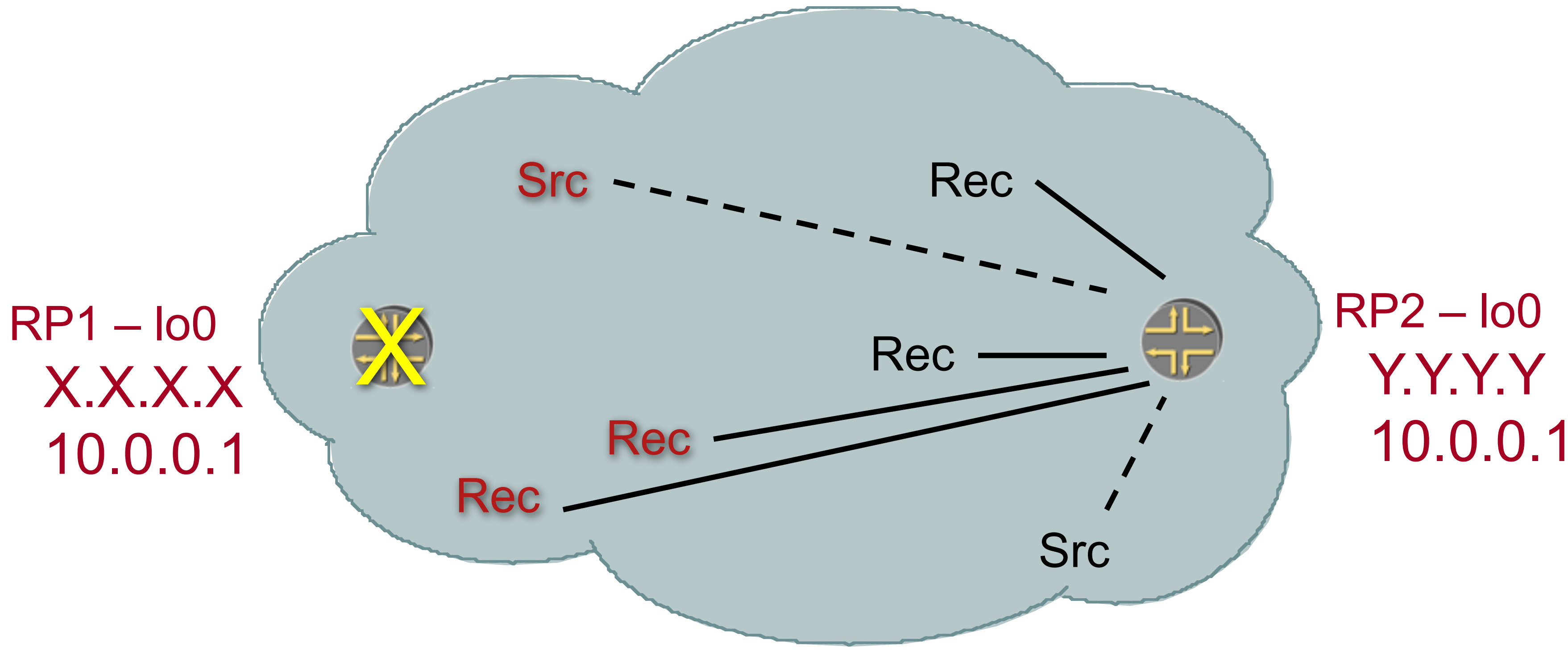
Fast convergence!

Anycast-RP



Src
Rec
MSDP
RP1 – lo0
X.X.X.X
10.0.0.1
Rec
RP2 – lo0
Y.Y.Y.Y
10.0.0.1
Rec
Rec
Src

Anycast-RP



MSDP Configuration

Your peer's IP address

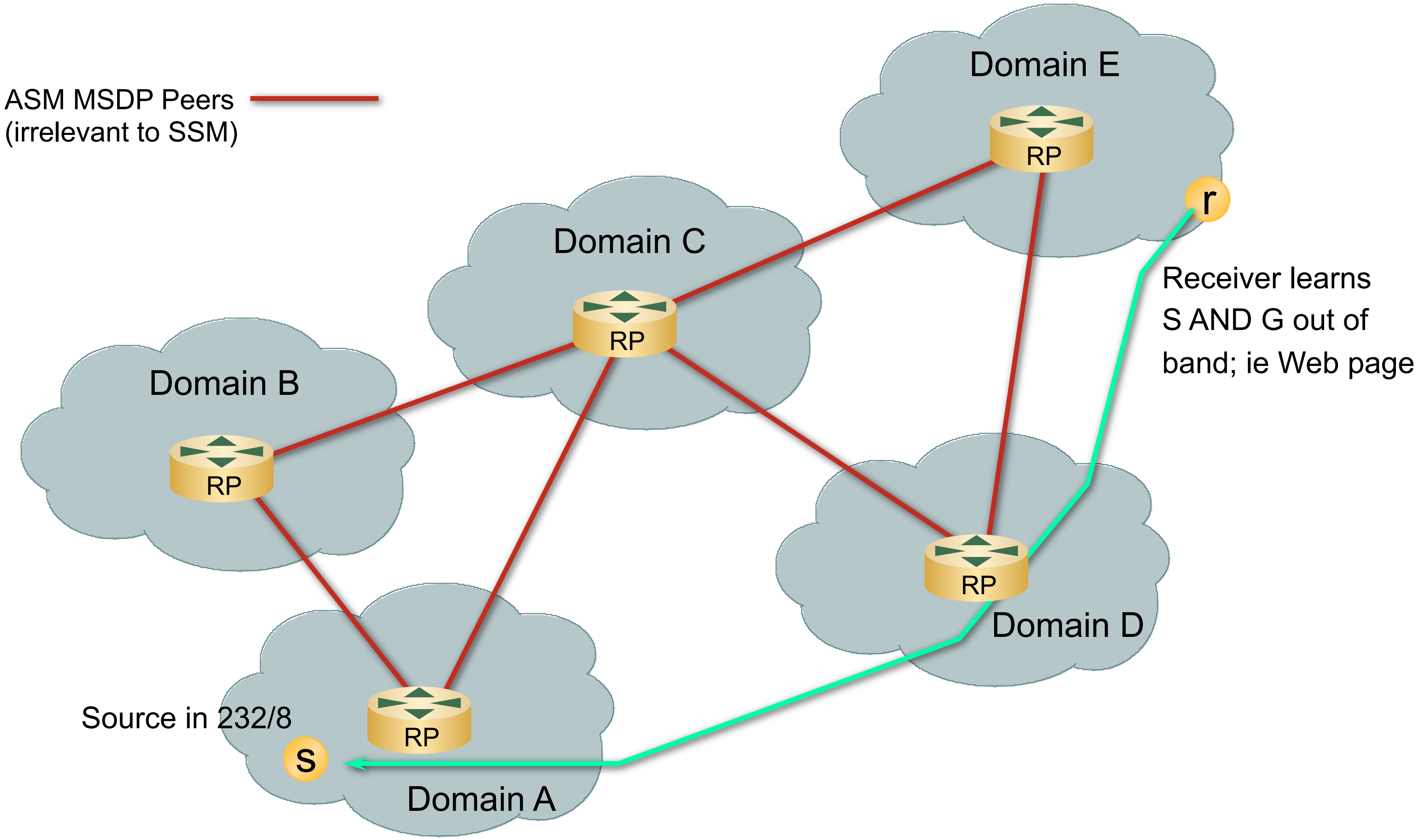
Your local connection interface.

Your peer's IP ASN

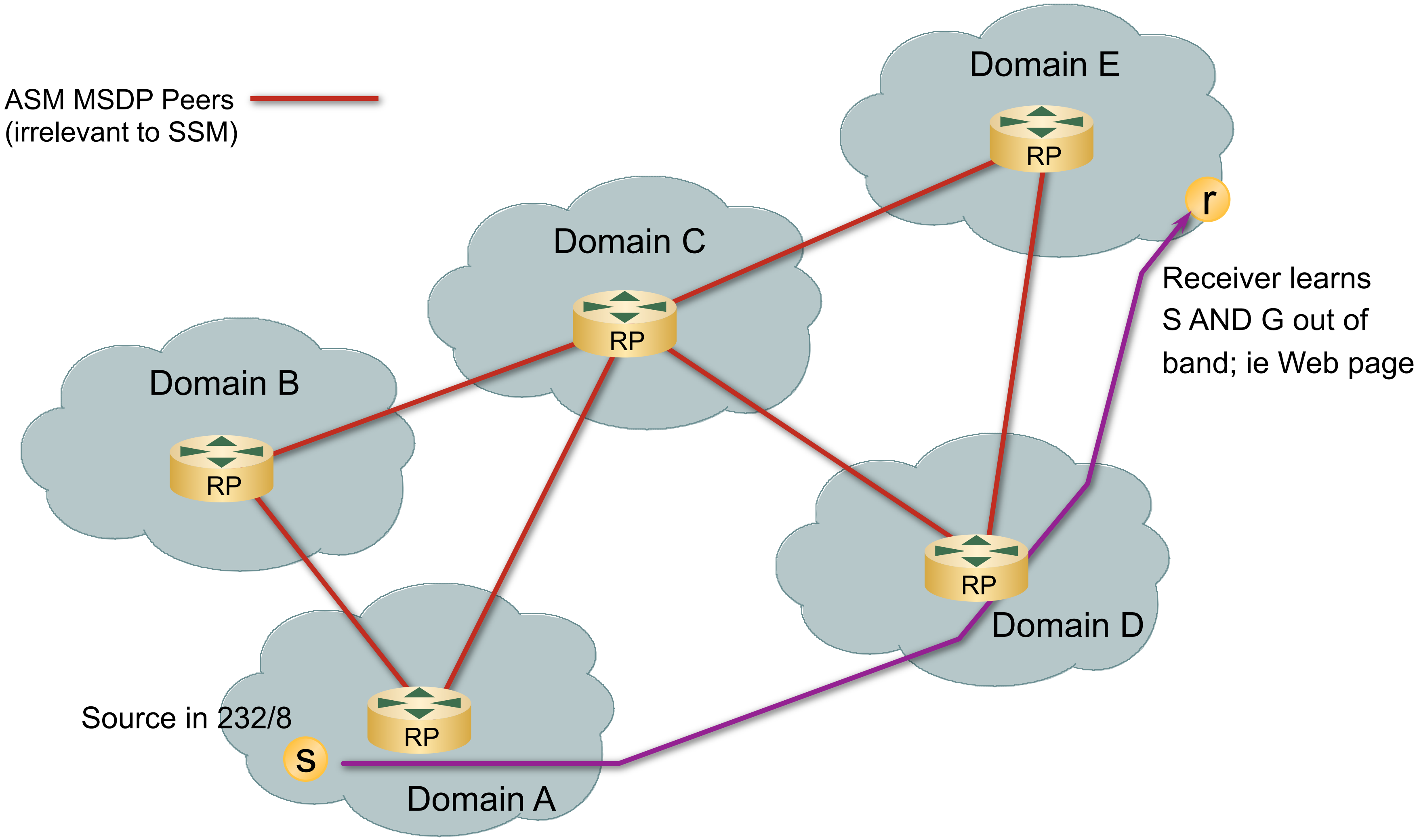
```
ip msdp peer 198.58.3.252 connect-source Ethernet0/0/2 remote-as 2
ip msdp originator-id Loopback1
```

Your local address which will appear as the RP in the MSDP SA TLV – used for MSDP peer-RPF checks

MSDP wrt SSM – Unnecessary!



MSDP wrt SSM – Unnecessary!



Agenda

- Introduction
- Multicast addressing
- Group Membership Protocol
- PIM-SM / SSM
- MSDP
- **MBGP**
- Summary

MBGP—Multiprotocol BGP

- MBGP overview
- MBGP capability negotiation
- MBGP NLRI exchange
- Configuration guidelines

MBGP

- Multiprotocol Extensions to BGP (RFC 2283).
- Tag unicast prefixes as multicast source prefixes for intra-domain mcast routing protocols to do RPF checks.
- WHY? Allows for interdomain RPF checking where unicast and multicast paths are non-congruent.
- DO I REALLY NEED IT?

  YES, if:

  - ISP to ISP peering
  - Multiple-homed networks

  NO, if:

  - You are single-homed

MBGP Overview

- **MBGP: Multiprotocol BGP**
  (aka multicast BGP in multicast networks)
  - Defined in RFC 2283 (extensions to BGP)
  - Can carry different route types for different purposes
    - Unicast
    - Multicast
  - Both route types carried in same BGP session
  - Does not propagate multicast state information
  - Same path selection and validation rules
    - AS-Path, LocalPref, MED, …

MBGP Overview

- New multiprotocol attributes

  MP_REACH_NLRI

  MP_UNREACH_NLRI

- MP_REACH_NLRI and MP_UNREACH_NLRI

  Address Family Information (AFI) = 1 (IPv4)

    Sub-AFI = 1 (NLRI is used for unicast)

    Sub-AFI = 2 (NLRI is used for multicast RPF check)

    Sub-AFI = 3 (NLRI is used for both unicast and multicast RPF check)

- Allows for different policies between multicast and unicast

MBGP—Capability Negotiation

- **BGP routers establish BGP sessions through the OPEN message**
- **OPEN message contains optional parameters**
- **BGP session is terminated if OPEN parameters are not recognised**
- **New parameter: CAPABILITIES**
  - **Multiprotocol extension**
  - **Multiple routes for same destination**
- **Configures router to negotiate either or both NLRI**
  - **If neighbor configures both or subset, common NRLI is used in both directions**
  - **If there is no match, notification is sent and peering doesn't come up**
  - **If neighbor doesn't include the capability parameters in open, session backs off and reopens with no capability parameters**

MBGP—Summary

- **Solves part of inter-domain problem**
  - Can exchange unicast prefixes for multicast RPF checks
  - Uses standard BGP configuration knobs
  - Permits separate unicast and multicast topologies if desired
- **Still must use PIM to:**
  - Build distribution trees
  - Actually forward multicast traffic
  - PIM-SM recommended

MBGP configuration (new)

Your ASN

Configure prefixes to advertise in both SAFI-1 and SAFI-2

```
router bgp 1
 address-family ipv4 unicast
    network 198.58.3.0/24
 address-family ipv4 multicast
    network 198.58.3.0/24
 neighbor 198.32.165.2 remote-as 2
    description LabPeer1
    update-source Ethernet0/0/1
    address-family ipv4 unicast
    address-family ipv4 multicast
```

Your peer's ASN

Local address for the BGP peering session

Configure to exchange both SAFI-1 and SAFI-2 prefixes

MBGP configuration (original)

Your ASN

Configure prefixes to advertise in both SAFI-1 and SAFI-2

```
router bgp 301 no synchronization
  network 172.16.2.0 mask 255.255.255.0 nlri unicast multicast
  neighbor 172.16.23.2 remote-as 201 nlri unicast multicast
  neighbor 172.16.23.2 update-source Loopback1
  next-hop-self
```

Your peer's ASN

Local address for the BGP peering session

Configure to exchange both SAFI-1 and SAFI-2 prefixes

LAB #2
Interdomain Multicast

- Do not launch lab until instructred to do so.
- Lab templates or cfgs: Interdomain-Multicast
- Refer to your lab handout

LAB #2
Interdomain Multicast