A Methodology for Troubleshooting Interdomain IP Multicast

Bill Nickless & Caren Litvanyi

Math & Computer Science Division, Argonne Nat'l Laboratory
Chicago IL, USA

NANOG 27 **Phoenix AZ**

What this tutorial is :

- A systematic approach to debugging mcast
  - Simple
  - Practical
  - Consistent
- Can be taught
- It has worked for us
- A good fallback

What this tutorial is not :

- ... the ONLY way, or the guaranteed fastest way
- ... a protocol taxonomy
- ... a configuration tutorial
- ... a command reference
- ... a religious statement
- ... a 'futures' talk
- ... an inventory of all tools
- ... an application demo

Why the need for a “methodology”?

- **Most engineers don’t troubleshoot multicast problems as often as unicast.**
- **Receiver-driven (somewhat backwards)**
  trench analogy
- **The problem can be far from the symptom.**
- **The same symptom can have many different causes, at different places in the path.**

Overview

- Gather information
- Verify receiver interest
- Verify DR knowledge of active source
- Trace forwarding state back

STEP 1:
GATHER INFORMATION

What is the problem?

Nobody can see me!

Multicast is broken … again

Some sites can hear us, but others can't.

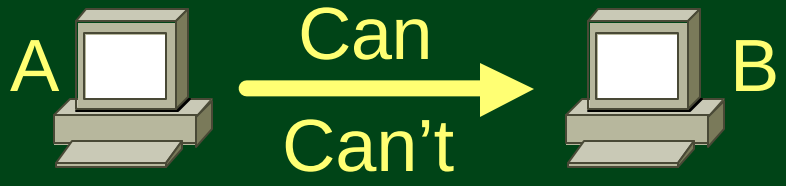
Site A can see B, but C can't hear D…

Site X called to say they can't see my presentation!

Multicast isn't working between here and there.

Gather Information

- **Pick ONE direction (that *is* the problem, or seems representative of the problem).**
- **Identify source end and receiving end.**
- **Remember, multicast is <u>unidirectional</u> in nature…**



Implies almost nothing about…

Gather Information

Now that you have a direction, you will need:

- **A constantly active source IP address**
- **A constantly active receiver IP address**
- **The group address**

It is impossible to debug a multicast problem without specifying all of these!!!

Gather Information

- **Is the beacon working?**

The beacon is an application to monitor multicast reachability and performance among beacon-group participants. Participants both send and receive on a known group, in this case, 233.2.171.1.

Packet Loss (%)		S0	S1	S2	S3	S4	S5	S6	S7	S8	S9
R0	beacon@ag-audio (206.75.91.25)	0	0	99	0	NA	2	2	NA	10	NA
R1	beacon@ag-video1 (156.56.104.3)	0	0	99	0	0	0	0	NA	0	NA
R2	beacon@audio (130.20.208.21)	2	0	0	0	19	23	0	NA	NA	NA
R3	beacon@backup2 (144.174.129.22)	NA	NA	NA	0	NA	NA	NA	NA	NA	NA
R4	beacon@dingdong (198.48.78.89)	0	0	99	0	0	0	0	NA	0	NA
R5	vu-amsterdam@display (130.37.42.36)	0	0	99	0	2	0	0	NA	0	NA
R6	beacon.noc.kreonet2.net@kreonet2 (134.75.20.90)	0	0	99	0	0	0	0	NA	0	NA
R7	beacon@mocha (128.208.20.215)	0	0	99	0	0	0	0	0	0	NA
R8	otter-ns3@ns3 (145.41.1.167)	0	0	0	0	0	0	0	NA	0	NA
R9	beacon@titania (128.111.55.97)	0	0	99	0	0	0	0	NA	2	0
Packet Loss (%)		**S0**	**S1**	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**	**S9**
R10	beacon@video (128.83.143.75)	2	0	0	0	2	0	0	NA	NA	NA
R11	beacon@beacon.sheridanc.on.ca (142.55.1.52)	0	7	99	0	0	12	7	NA	10	NA
R12	jnj@lab-disp.atr.jnj.com (192.168.3.96)	27	7	22	NA	7	20	22	NA	NA	NA
R13	beacon@hendrix.multicasttech.com (63.105.122.14)	0	0	99	0	0	0	0	NA	0	NA
R14	beacon@techie.multicasttech.com (216.177.62.40)	0	0	99	0	0	0	0	NA	0	NA
R15	beacon@nettest.arsc.edu (199.165.80.245)	0	0	99	0	NA	0	0	NA	0	NA
R16	beacon@agaudio.bu.edu (192.12.188.20)	0	0	0	0	0	0	0	NA	0	NA

Gather Information

http://dast.nlanr.net/Projects/Beacon/

Packet Loss (%)			S0	S1	S2	S3	S4	S5	S6	S7	S8	S9	
R80	[Click for FAQ(2)] agdisplay.chpc.**utah.edu**	155.101.28.13	NA	0	0	2	0	0	NA	NA	0	0	R80
R81	mcast1.gw.**utexas.edu**	128.83.6.240	0	0	0	0	0	0	NA	NA	0	0	R81
R82	tulip.as.**utk.edu**	160.36.8.67	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	R82
R83	[Click for FAQ(2)] ag02.cs.**utk.edu**	160.36.59.104	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	R83
R84	d-128-208-20-224.dhcp4.**washington.edu**	128.208.20.224	0	0	0	0	0	0	NA	NA	0	0	R84
R85	mbone-test.cs.**wisc.edu**	128.105.1.86	0	0	0	0	0	0	NA	NA	0	0	R85
R86	grid-op.trace.**wisc.edu**	128.104.192.212	0	0	0	2	0	0	NA	NA	0	0	R86
R87	[Click for FAQ(2)] ag-enc.**wpi.edu**	130.215.128.21	0	0	0	0	0	0	NA	NA	0	0	R87
R88	noc1.**wpi.edu**	130.215.201.81	7	0	0	10	10	0	NA	NA	0	7	R88
R89	[Click for FAQ(2)] ip-62-54.telcom.**wvu.edu**	157.182.62.54	NA	0	0	0	0	2	NA	NA	0	NA	R89
Packet Loss (%)			S0	S1	S2	S3	S4	S5	S6	S7	S8	S9	
R90	[Click for FAQ(2)] dsl-agvideo.mcs.**anl.gov**	140.221.8.157	0	0	0	0	0	0	NA	NA	0	0	R90
R91	[Click for FAQ(2)] lib-video.mcs.**anl.gov**	140.221.8.53	0	0	0	0	0	0	NA	NA	0	0	R91
R92	ws-video.mcs.**anl.gov**	140.221.34.1	0	0	0	0	0	0	NA	NA	0	0	R92
R93	[Click for FAQ(2)] micsagaudio.er.**doe.gov**	192.73.213.181	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	R93
R94	[Click for FAQ(2)] agaudio2.acl.**lanl.gov**	204.121.50.22	62	70	69	67	65	70	NA	NA	67	71	R94

Gather Information

- If the beacon is also broken between sites, it is sometimes possible to use it as the constantly active source and receiver.
- However, many times the beacon can be fine yet multicast is broken for a different group.
- It will not catch new/transient problems with source knowledge or state creation.

http://dast.nlanr.net/Projects/Beacon/

Gather Information

- **Example: GEANT** **http://beaconserver.geant.net:9999**

Time: **Sat Feb 08 23:24:51 GMT 2003**
Target: **233.81.229.1:56464**
Beacons: **12** details
Page: refresh in 60 seconds

Loss (%)	S0	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11
R0 beacon@62.40.99.107@ws2.lu	0	0	0	0	0	0	0	0	0	0	0	0
R1 beacon@62.40.100.11@ws2.si	0	0	0	0	0	0	0	0	0	0	0	0
R2 beacon@62.40.98.151@ws1.de	0	0	0	0	0	0	0	0	0	0	0	0
R3 beacon@62.40.98.180@ws1.es	0	0	0	0	0	0	0	0	0	0	0	0
R4 beacon@62.40.98.212@ws1.fr	0	0	0	0	0	2	0	0	0	0	0	0
R5 beacon@62.40.98.21@ws2.at	0	0	0	0	0	0	0	0	0	0	0	0
R6 beacon@62.40.99.245@ws2.se	0	0	0	0	0	0	0	0	0	0	0	0
R7 beacon@62.40.98.52@ws1.be	0	0	0	0	0	0	0	0	0	0	0	0
R8 beacon@62.40.100.52@ws1.sk	0	0	0	0	0	0	0	0	0	0	0	0
R9 beacon@62.40.98.85@ws2.ch	0	0	0	0	0	0	0	0	0	0	0	0
R10 beacon@62.40.99.85@ws2.it	0	0	0	0	0	0	0	0	0	0	0	0
R11 beacon@62.40.100.85@ws2.uk	0	0	0	0	0	0	0	0	0	0	0	0

Gather Information

- **OK – we know the IP addresses for the problem source, receiver, and group, and that the source and receiver are active.**

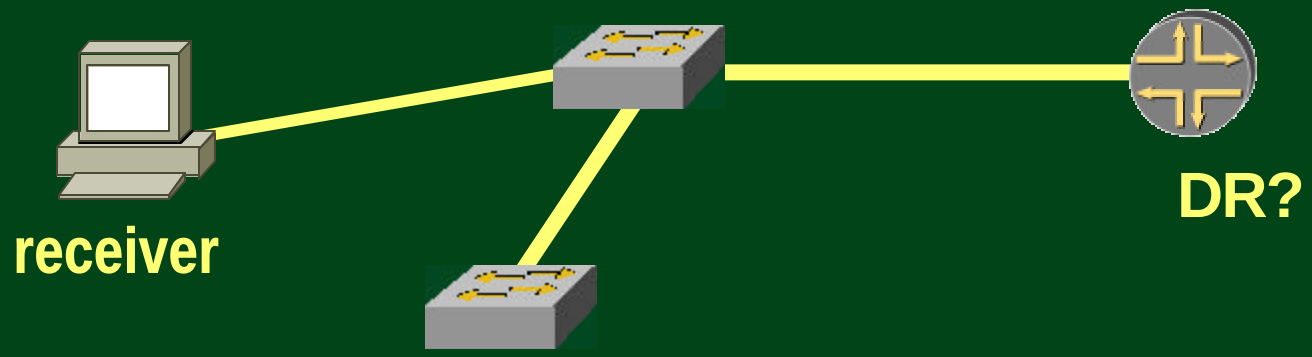
Move on to step 2…

STEP 2: VERIFY RECEIVER INTEREST

Verify Receiver Interest

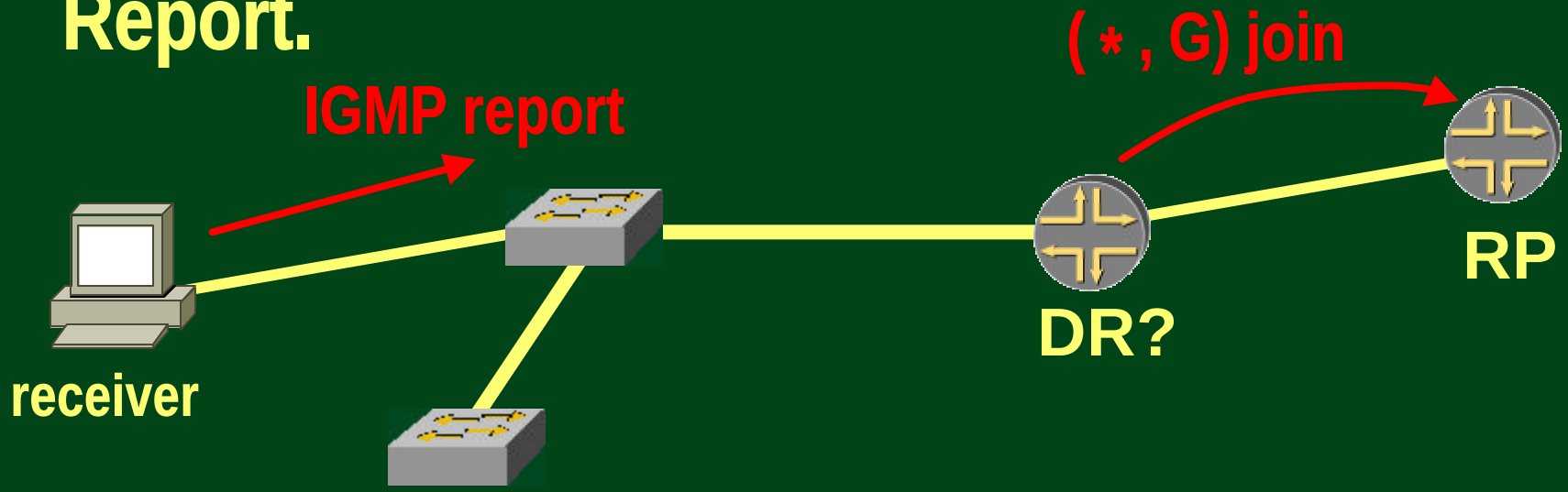
- Verify who is the PIM Designated Router (DR) on the receiving host's subnet.

You might think you know this, but you should not proceed until it has been verified.



Verify Receiver Interest

- Recall that the DR will need to send a (* , G) join towards the RP when it learns of a receiver's interest via an IGMP Membership Report.



Verify Receiver Interest

- To verify the DR, log into the router you think *should* be routing multicast for the receiver.
- 1) Find the interface that serves the receiver's subnet.
- 2) Check that there is no other PIM router that thinks *IT* is the DR for the subnet.

Verify Receiver Interest

Cisco: find the right interface

```
squash# show ip rpf 140.221.34.1
RPF information for ws-video.mcs.anl.gov
(140.221.34.1)
  RPF interface: GigabitEthernet5/7
  RPF neighbor: ? (0.0.0.0) - directly connected
  RPF route/mask: 140.221.34.0/28
  RPF type: unicast (connected)
  RPF recursion count: 0
  Doing distance-preferred lookups across tables
squash#
```

Verify Receiver Interest

Juniper: find the right interface

```
remote@MREN-M5> show multicast rpf 206.220.240.86
Multicast RPF table: inet.2, 5051 entries

206.220.240.64/27
    Protocol: Direct
    Interface: ge-0/0/0.108
```

Verify Receiver Interest

Cisco: verify DR for that interface

```
squash#sh ip igmp interface gig5/7
GigabitEthernet5/7 is up, line protocol is up
  Internet address is 140.221.34.13/28
  IGMP is enabled on interface
  Current IGMP host version is 2
  Current IGMP router version is 2
  IGMP query interval is 60 seconds
  IGMP querier timeout is 120 seconds
  IGMP max query response time is 10 seconds
  Last member query response interval is 1000 ms
  Inbound IGMP access group is not set
  IGMP activity: 867 joins, 866 leaves
  Multicast routing is enabled on interface
  Multicast TTL threshold is 0
  Multicast designated router (DR) is 140.221.34.13 (this system)
  IGMP querying router is 140.221.34.13 (this system)
  No multicast groups joined
squash#
```

Verify Receiver Interest

Juniper: verify DR for that interface

```
remote@MREN-M5> show pim interfaces

Instance: PIM.master
Name                Stat  Mode     IP V  State  Count   DR address
at-0/2/1.237        Up    Sparse    4 2  P2P        1
at-0/2/1.6325       Up    Sparse    4 2  P2P        1
at-0/2/1.9149       Up    Sparse    4 2  P2P        1
ge-0/0/0.108        Up    Sparse    4 2  DR         1   206.220.240.85
ge-0/0/0.109        Up    Sparse    4 2  NotDR      1   10.10.10.1

remote@MREN-M5>
```

Verify Receiver Interest

- **SO… now you are sure you are on your receiver's DR.**
- **Remember, multicast is receiver-driven**
- **QUESTION: *Does this DR know that there are interested receivers of your group on the receiving host's subnet ??***

Verify Receiver Interest

On the DR:

```
squash#sh ip igmp group 233.2.171.1
IGMP Connected Group Membership
Group Address  Interface         Uptime    Expires   Last Reporter
233.2.171.1    Vlan1              1d03h     00:02:16  140.221.10.87
233.2.171.1    GigabitEthernet5/7 7w0d      00:02:21  140.221.34.1
squash#
```

```
remote@MREN-M5> show igmp group 233.2.171.1
Interface: ge-0/0/0.108
    Group: 233.2.171.1
        Source: 0.0.0.0   Last Reported by: 206.220.240.86
        Timeout:     156           Type: Dynamic
remote@MREN-M5>
```

Receiver's interface should be in this list.
Might want to watch to ensure no timeouts.

Verify Receiver Interest

- What if your interface isn't listed with that group??

- You have a problem
  - Host OS / driver problem
  - Application problem
  - Broken IGMP snooping switches in the middle
  - Try tcpdump on the host

Verify Receiver Interest

- If your receiver's DR knows it has listeners of your group on that interface, you are done this step.

Move on to step 3…

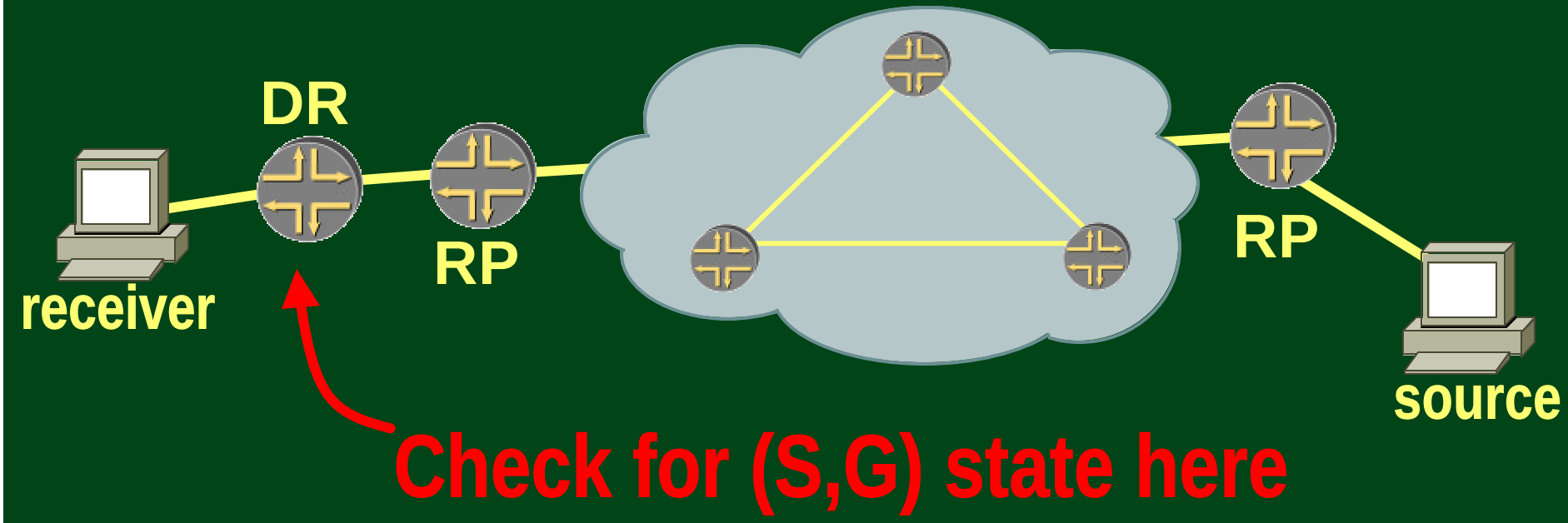
STEP 3: VERIFY DR KNOWLEDGE OF ACTIVE SOURCE

Verify DR knowledge of active source

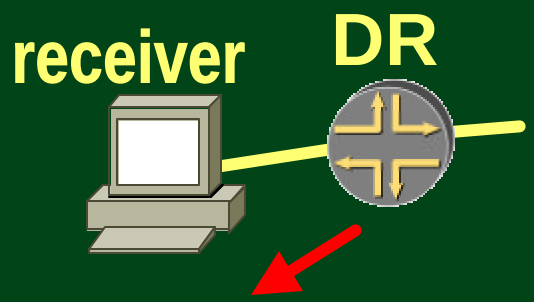
- This is the most complex part – the bulk of your work could be here.
- You MAY have view this from both ends
  - The receiver's RP
  - The source's RP
- For most interdomain cases, these RPs will not be the same, and MSDP will be involved.

Verify DR knowledge of active source

- First, let's check to see if this is a problem at all.
- If the receiver's DR has (S,G) state already, we know we are ok on knowledge of active source, and we can skip this whole step!



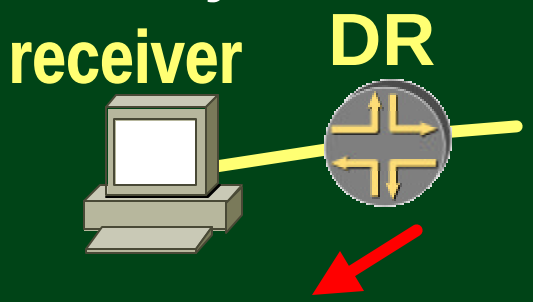
Verify DR knowledge of active source




```
squash# show ip mroute 233.2.171.1 141.142.64.104
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, 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 - Advertised via MSDP, 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

(141.142.64.104, 233.2.171.1), 1w0d/00:02:59, flags: CJT
  Incoming interface: Vlan669, RPF nbr 130.202.222.74
  Outgoing interface list:
    GigabitEthernet5/7, Forward/Sparse, 20:19:14/00:02:08
    Vlan1, Forward/Sparse, 1w0d/00:01:56
```

GOOD!

Verify DR knowledge of active source




```
remote@starlight-m10> show multicast route group 233.2.171.1
                                          source-prefix 140.221.34.1
Family: INET
Group           Source prefix      Act Pru InIf  NHid  Session Name
233.2.171.1     140.221.34.1    /32 A   F  6     246   Static Alloc
```

GOOD!

(...extensive)

```
Family: INET
Group           Source prefix     Act Pru NHid  Packets IfMi Timeout
233.2.171.1     140.221.34.1   /32 A   F  246   8702556  69    360
    Upstream interface: ge-0/0/0.0
    Session name: Static Allocations
    Forwarding rate: 1 kBps (9 pps)
```

Verify DR knowledge of active source

- **If the DR does NOT know about the source, we may only see a (* , G) entry on a Cisco DR, and we have some work to do.**

```
squash# show ip mroute 233.2.171.1 141.142.64.104
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, 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 - Advertised via MSDP, 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

(*, 233.2.171.1), 7w0d/00:02:59, RP 192.5.170.2, flags: SJCF
  Incoming interface: Vlan29, RPF nbr 140.221.20.97
  Outgoing interface list:
    GigabitEthernet5/7, Forward/Sparse, 20:22:27/00:02:52
    Vlan1, Forward/Sparse, 7w0d/00:02:45
```

BAD!

Verify DR knowledge of active source

- **If the DR does NOT know about the source, we may see nothing on a Juniper DR, and we have some work to do.**

```
remote@starlight-m10> show multicast route group 233.2.171.1
                                        source-prefix 141.142.64.104
Family: INET
Group           Source prefix     Act Pru InIf  NHid  Session Name


remote@starlight-m10>
```

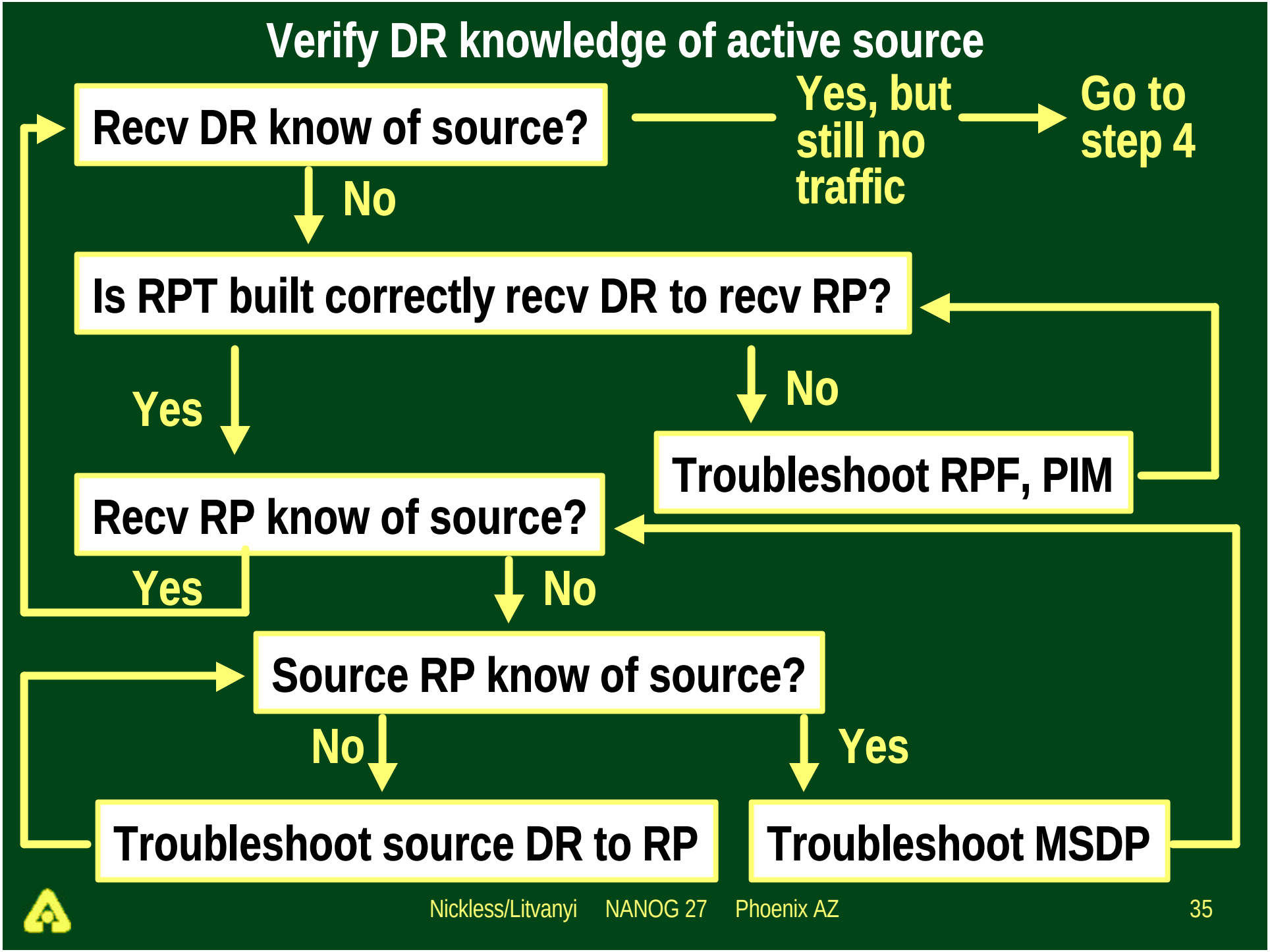
BAD!

Verify DR knowledge of active source

- Recall that knowledge of active sources is spread through a given PIM domain by per-group RP-rooted shared distribution trees.
- Current practice is to set the Source Path Tree (SPT) threshold to zero, so that (S,G) state is created by on the first packet sent through the RP.
- But if the shared tree doesn't get built properly, the SPT never will.

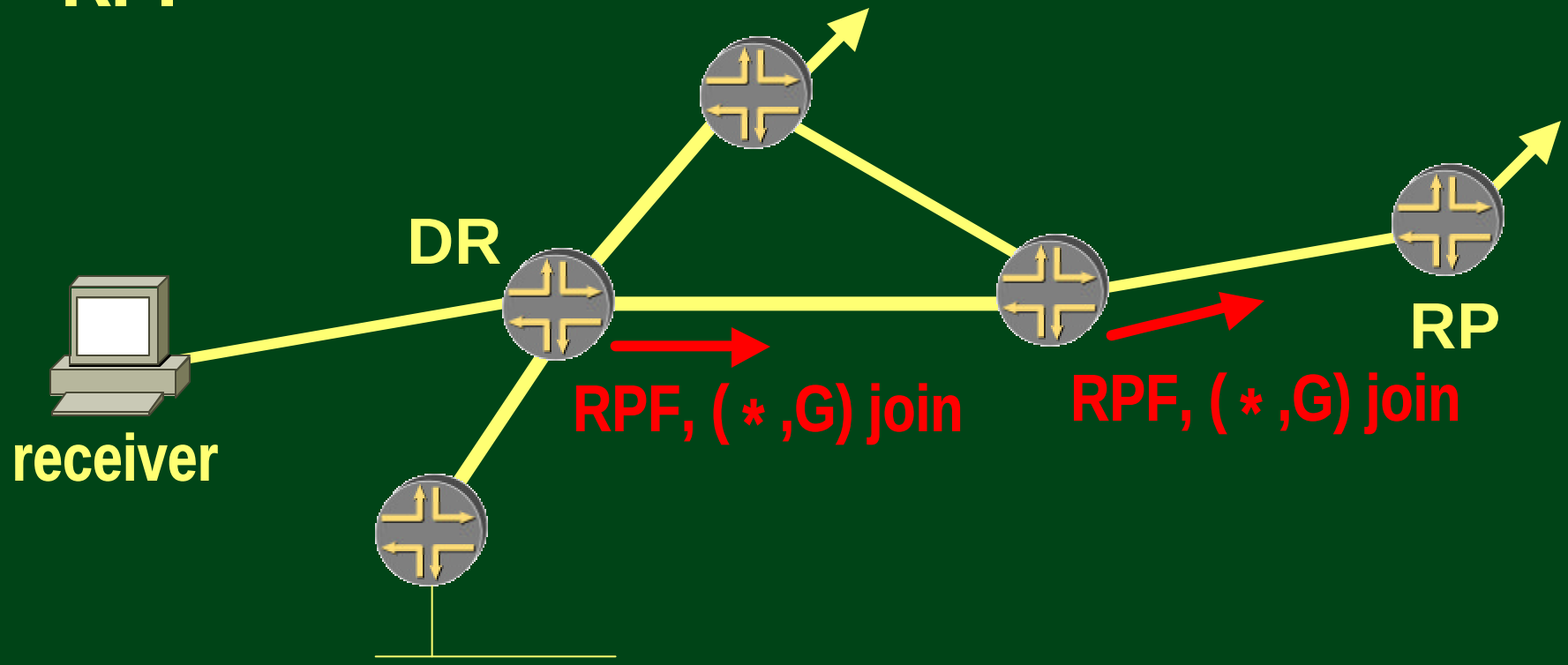
Verify DR knowledge of active source

- So, first, we will work back from the receiver's DR to it's RP, to be sure the RPT branch is built correctly.
- Second, we will check to see if the receiver's RP knows about the source.
- Third, we will check with the source end for their RP knowledge/advertisement of the source.
- Last, we will troubleshoot MSDP as needed.


Verify DR knowledge of active source
Recv DR know of source?
Yes, but still no traffic
Go to step 4
No
Is RPT built correctly recv DR to recv RP?
Yes
No
Troubleshoot RPF, PIM
Recv RP know of source?
Yes
No
Source RP know of source?
No
Yes
Troubleshoot source DR to RP
Troubleshoot MSDP

Verify DR knowledge of active source

- First, we check that the shared tree is built from the receiver's DR back to the receiver's RP.



Verify DR knowledge of active source

- **Does the DR have the right RP?**

```
squash# show ip pim rp mapping 233.2.171.1
PIM Group-to-RP Mappings
 Group(s) 224.0.0.0/4
  RP 192.5.170.2 (kiwi-loop.anchor.anl.gov), v2v1
    Info source: 140.221.20.97 (kiwi.anchor.anl.gov),
via Auto-RP, via bootstrap
         Uptime: 7w0d, expires: 00:02:47
Group(s): 224.0.0.0/4, Static
    RP: 192.5.170.2 (kiwi-loop.anchor.anl.gov)
```

Verify DR knowledge of active source

- **Does the DR have the right RP?**

```
remote@starlight-m10> show pim rps detail
Instance: PIM.master
Family: INET
RP: 206.220.240.220
Learned via: static configuration
Time Active: 13w2d 09:59:40
Holdtime: 0
Group Ranges:
        224.0.0.0/4
Active groups using RP:
        224.2.127.254
        233.2.171.1
        239.22.33.5
        total 3 groups active

remote@starlight-m10>
```

Verify DR knowledge of active source

- Now that you are sure of what the RP is, starting at the receiver's DR, work your way back to the receiver's RP.
- Check that the RPF is pointing the way you expect.
- Check that PIM is working properly on the interface.

Verify DR knowledge of active source

- show ip rpf <RP ip address>
- show ip pim neighbor <rpf interface>

```
squash# show ip rpf 192.5.170.2
RPF information for kiwi-loop.anchor.anl.gov
(192.5.170.2)
  RPF interface: Vlan29
  RPF neighbor: kiwi.anchor.anl.gov (140.221.20.97)
  RPF route/mask: 192.5.170.2/32
  RPF type: unicast (ospf 683)
  RPF recursion count: 0
  Doing distance-preferred lookups across tables


squash# show ip pim neighbor Vlan29
PIM Neighbor Table
Neighbor Address  Interface  Uptime Expires  Ver  Mode
140.221.20.97     Vlan29      7w0d  00:01:35  v2  (DR)
squash#
```

Verify DR knowledge of active source

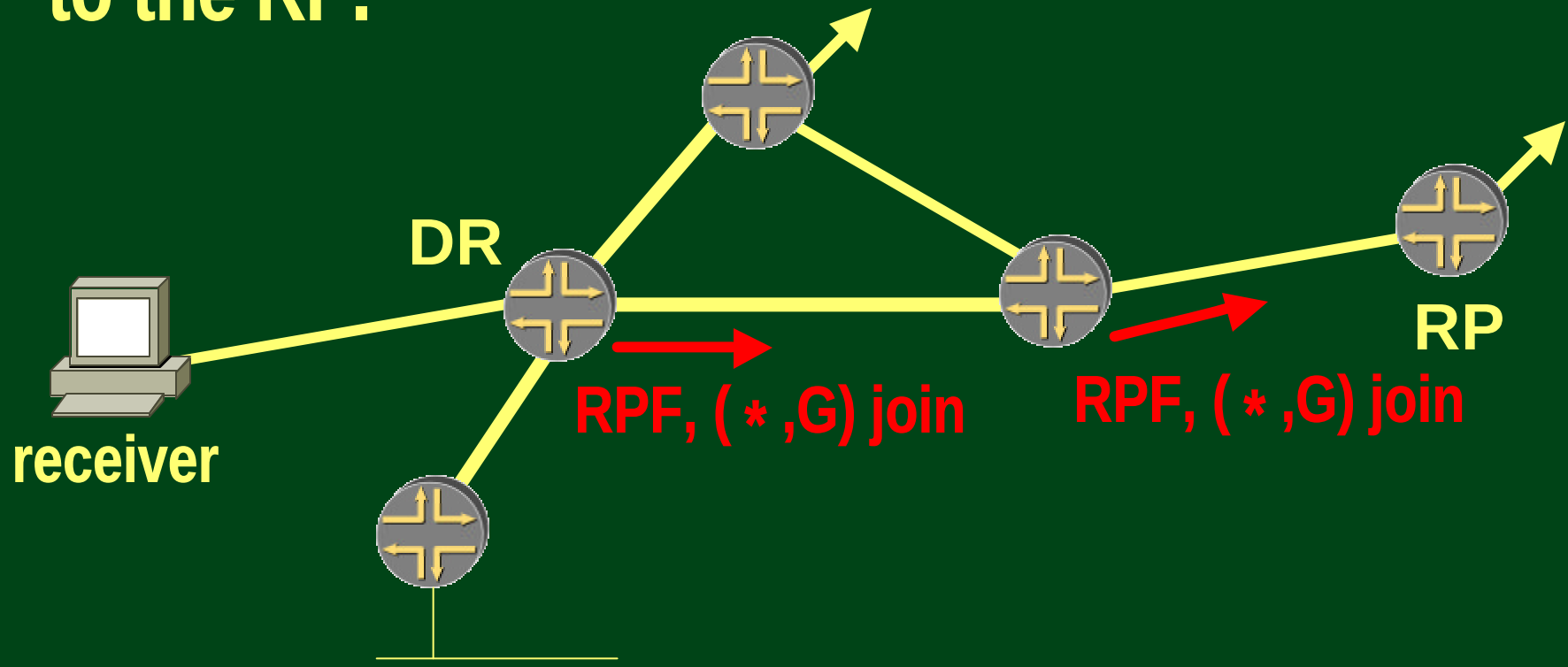
- show multicast rpf <RP ip address>
- show pim neighbors

```
remote@MREN-M5> show multicast rpf 206.220.241.254
Multicast RPF table: inet.2, 5061 entries

206.220.241.0/24
    Protocol: BGP
    Interface: ge-0/0/0.108
```

```
remote@MREN-M5> show pim neighbors
Instance: PIM.master

Interface           IP V Mode    Option   Uptime   Neighbor addr
at-0/2/1.237         4 2         H        4w6d11h  192.122.182.13
at-0/2/1.6325        4 2         H        4w6d11h  206.166.9.33
at-0/2/1.9149        4 2         HP  B    4w6d11h  199.104.137.245
ge-0/0/0.108         4 2         H   G    4w6d11h  206.220.240.86
```

Verify DR knowledge of active source

- **Repeat that process until you have verified the RPF paths and the PIM adjacencies back to the RP.**



Verify DR knowledge of active source

- Next Big Question: Does the RP have knowledge of the active source?
- If it doesn't, (*, G) only, and no MSDP SA cache entry for that source, we will have to find out some information about the source end of things.
- Objective here is to get MSDP SA to the receiver's RP from the ***source's RP***.

Verify DR knowledge of active source

On the receiver's RP:

```
Kiwi#sh ip mroute 233.2.171.1 141.142.64.102
IP Multicast Routing Table
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, Z - Mcast Tunnel
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 233.2.171.1), 6w6d/stopped, RP 192.5.170.2, flags: S
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    GigabitEthernet5/0, Forward/Sparse, 6w6d/00:03:01


  Kiwi#sh ip msdp sa-cache 233.2.171.1 141.142.64.102
  MSDP Source-Active Cache
  Entry not found
```

BAD!

Verify DR knowledge of active source

- But… how do we know the source's RP if we run only the receiver network?
  - May have to pick up phone and walk them through verifying the source's DR and finding the group RP mapping there.
  - Get them to tell you they have verified the source is sending, and the IP of their RP is ___.
  - You might want to have them look to see that they mark the mroute as a candidate for MSDP advertisement.

Verify DR knowledge of active source

On the source's RP:

Source IP

```
Kiwi#sh ip mroute 233.2.171.1 140.221.34.1
IP Multicast Routing Table
Flags: D-Dense, S-Sparse, B-BidirGroup, 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 - Recv Source Specific Host Report, Z - Multicast Tunnel,
       Y - Joined MDT-data group, y - Sending to MDT-data group
Outgoing interface flags: H - Hardware switched
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(140.221.34.1, 233.2.171.1), 6w6d/00:03:26, flags: TA
  Incoming interface: GigabitEthernet5/0, RPF nbr 140.221.20.124
  Outgoing interface list:
    ATM3/0.6200, Forward/Sparse, 2w0d/00:02:42 (ttl-threshold 32)
Kiwi#
```

Verify DR knowledge of active source

- So now we have the information to see how we are supposed to be learning about that source
  - The receiver's RP
  - The source's RP
  - The fact that the receiver's MSDP speaking RP doesn't know about this source
- Trace back reachability / reverse path from the receiver's RP towards the source's RP into the upstream network.
- MSDP uses "peer-RPF rules" to determine from where it will accept source-active notifications.

Verify DR knowledge of active source

- Peer-RPF rules are not all that straight-forward or well defined.
- An SA message is only accepted and forwarded to other peers if it came from the RPF peer.
- When using MSDP mesh groups, this becomes easier since the RPF rules are only applied to external peers.
  - If an SA is received from an external peer, it is flooded to all internal peers.
  - If an SA is received from an internal peer, it is sent only to external peers, and is always accepted.

Verify DR knowledge of active source

- The idea here is we are trying to figure out which of our MSDP peers we should expect to get knowledge of that source from.
  - If the source RP is an MSDP peer of our RP, the source RP is the RPF peer.
  - If we look at “show ip mbgp <source RP IP>”, the MSDP peer in the adjacent AS is the RPF peer.
  - In practice, “show ip rpf <source RP IP>” and “show ip mbgp <source RP IP>” will ususally get you going in the right direction.

Verify DR knowledge of active source

```
guava#sh ip rpf 206.220.241.254
RPF information for lsd6509.sl.startap.net (206.220.241.254)
  RPF interface: Vlan109
  RPF neighbor: mren-anl-gige.anchor.anl.gov (192.5.170.214)
  RPF route/mask: 206.220.241.0/24
  RPF type: mbgp
  RPF recursion count: 0
  Doing distance-preferred lookups across tables
guava#sh ip mbgp 206.220.241.254
BGP routing table entry for 206.220.241.0/24, version 734283
Paths: (2 available, best #1, table NULL)  Flag: 0x278
  Advertised to peer-groups: imbgp-mesh
  22335
    192.5.170.214 from 192.5.170.214 (206.220.241.254)
      Origin IGP, metric 0, localpref 40100, valid, external, best
      Community: 683:65001 22335:22335
  293 10764 22335
    192.5.170.78 from 192.5.170.78 (134.55.29.97)
      Origin IGP, metric 100, localpref 10000, valid, external
      Community: 293:52 683:293 no-export
guava#
```

Verify DR knowledge of active source

- At this point, you may need to open a ticket with your upstream provider or peer. You can give them the following:
  - Our RP which MSDP peers with you is <IP address>.
  - We are not getting an SA for <source IP address>
  - The source's RP is <source RP IP address>
  - We expected to get this from <MSDP peer's IP address>
- PIM will need to be checked along the way as well.
- You will know they have fixed it when you get knowledge of the source on your RP.

Verify DR knowledge of active source

- **Since you have already checked your path back from the receiver to the RP, you should then get (S,G) state on the receiver's DR when your upstream provider or peer works the ticket.**

Move on to step 4…

Overview Refresher!

Gather information

Verify receiver interest

Verify DR knowledge of active source

Trace forwarding state back

STEP 4:
TRACE FORWARDING STATE BACK

Trace forwarding state back

- We now have (S,G) state on the receiver's DR.
- Need to check to see if traffic is actually flowing now...

```
squash# show ip mroute 233.2.171.1 204.121.50.22 count
IP Multicast Statistics
226 routes using 103842 bytes of memory
42 groups, 4.38 average sources per group
Forwarding Counts: Pkt Count/Pkts per second/Avg PktSize/Kilobits per sec
Other counts: Total/RPF fail/Other drops(OIF-null,rate-limit,etc)

Group: 233.2.171.1, Source count: 100, Group pkt count: 987910557
  Source: 204.121.50.22/32, Forwarding: 0/0/0/0, Other: 6/0/6
squash#
```

- If this is zero, you still have a problem.

Trace forwarding state back

- Start on your receiver's DR.
- This time, rpf back towards the actual source IP address (as opposed to the source RP).

```
squash# show ip rpf 204.121.50.22
RPF information for agaudio2.acl.lanl.gov (204.121.50.22)
  RPF interface: Vlan669
  RPF neighbor: guava-stardust.anchor.anl.gov (130.202.222.74)
  RPF route/mask: 0.0.0.0/0
  RPF type: unicast (ospf 683)
  RPF recursion count: 0
  Doing distance-preferred lookups across tables
```

- You are looking to see how you are expecting the SPT tree to be built, where you actually expect the packet flow to come from.

Trace forwarding state back

- **Work your way back towards the source IP, looking for PIM problems along the way.**

```
squash# show ip pim neighbor Vlan669
PIM Neighbor Table
Neighbor Address  Interface                Uptime    Expires   Ver  Mode
130.202.222.74    Vlan669                  7w0d      00:01:35  v2    (DR)
squash#
```

Trace forwarding state back

- **Also double-check that the receiver DR has sent a PIM join towards the right upstream neighbor:**

```
squash# show ip mroute 233.2.171.1 204.121.50.22
IP Multicast Routing Table
Flags: D - Dense, S - Sparse, 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 - Advertised via MSDP, 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

(204.121.50.22, 233.2.171.1), 00:00:41/00:02:18, flags: CJ
  Incoming interface: Vlan669, RPF nbr 130.202.222.74
  Outgoing interface list:
    Vlan1, Forward/Sparse, 00:00:41/00:02:18
    GigabitEthernet5/7, Forward/Sparse, 00:00:41/00:02:20
```

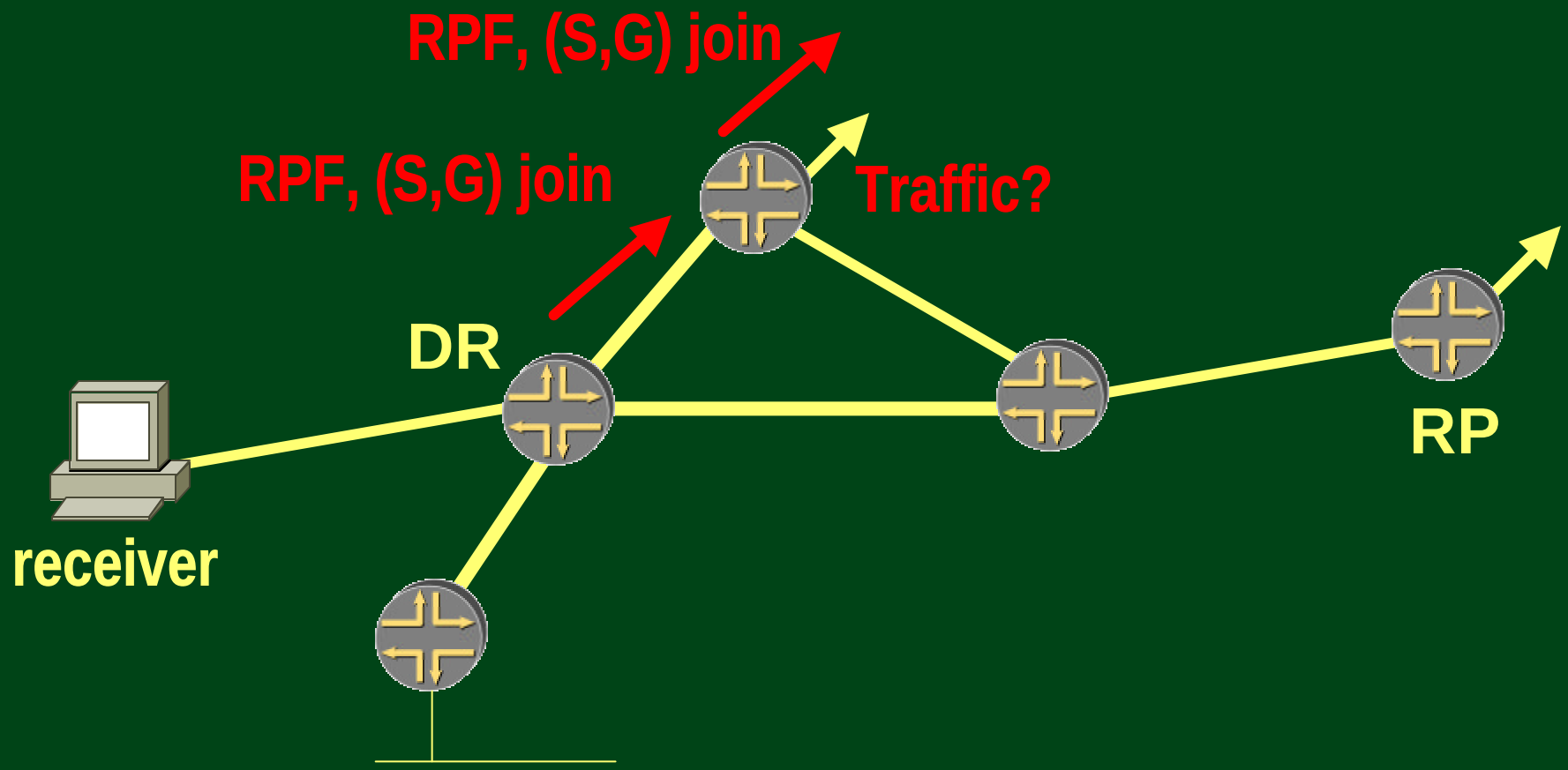
Trace forwarding state back

- Log into that upstream router and check state there with:
  - > show ip mroute <group> <source>
  - > show ip mroute <group> <source> count
  - Or (Juniper):

  sh multi route group <group> source <source> ext
- Look to see if the downstream router is in the outgoing interface list, and to see if you see a positive traffic rate.

Trace forwarding state back

We are tracing back the SPT….



Trace forwarding state back

```
Kiwi#sh ip mroute 233.2.171.1 140.221.34.1
IP Multicast Routing Table
Flags: <cut>
Outgoing interface flags: H - Hardware switched
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(140.221.34.1, 233.2.171.1), 6w6d/00:03:26, flags: TA
  Incoming interface: GigabitEthernet5/0, RPF nbr 140.221.20.124
  Outgoing interface list:
    ATM3/0.6200, Forward/Sparse, 2w0d/00:02:46 (ttl-threshold 32)
Kiwi#


Kiwi#sh ip mroute 233.2.171.1 140.221.34.1 count
IP Multicast Statistics
493 routes using 224398 bytes of memory
71 groups, 5.94 average sources per group
Forwarding Counts: Pkt Count/Pkts per second/Avg Pkt Size/Kbits per sec
Other counts: Total/RPF failed/Other drops(OIF-null, rate-limit etc)

Group: 233.2.171.1, Source count: 123, Group pkt count: 82381322
  Source: 140.221.34.1/32, Forwarding: 37847545/9/89/6, Other: 33/0/0
```

Trace forwarding state back

- If you get to a point where the upstream router IS showing it is receiving the packets, but your downstream is not, you need to figure out why those packets are getting lost.
  - ACLs?
  - Broken IGMP snooping switch in the middle?

Trace forwarding state back

- You may work this back to the edge of your area of responsibility, and may have to open a ticket with your upstream to continue the process towards the source. Give them:
  - The active source IP address
  - The group address
  - The circuit / link towards which your router has sent the (S,G) join
  - The fact that you are not receiving packets for that (S,G) on that shared link.

Summary

Gather information

Verify receiver interest

Verify DR knowledge of active source

Trace forwarding state back

Summary

Gather information

A direction
Active source and receiver IP addresses
Group address

Summary

Verify receiver interest

Identify the DR for the receiver
Verify the DR knows of interest in that group
Check that the DR is not receiving traffic

Summary

Get DR knowledge of active source

Might mean fixing multicast reachability topology or PIM state
Probably will involve MSDP SA debugging

Summary

Trace forwarding state back

Trace forwarding state from receiver's DR
Work towards the source
Verify reachability, PIM state, and whether traffic is flowing at each step

Thank you – comments welcome!

A Methodology for Troubleshooting Interdomain IP Multicast

Bill Nickless & Caren Litvanyi

Math & Computer Science Division, Argonne Nat'l Laboratory
Chicago IL, USA

NANOG 27 Phoenix AZ