Understand the PIM Assert Mechanism

Contents

Introduction

This document describes Protocol Independent Multicast (PIM) assert mechanism, concentrates on PIM assert winner criteria, and dives deeper into certain corner cases.

Prerequisites

Requirements

Cisco recommends that you have knowledge of PIM assert mechanism.

Components Used

The information in this document is based on Cisco CSR1000V Version 16.4.1

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, ensure that you understand the potential impact of any command.

What is the PIM Assert Mechanism?

When there are multiple PIM enabled routers on a shared segment, it is possible that these routers encounter duplicate multicast traffic. This might be the case because two or more routers on the same shared segment might have a valid (S,G) or (*,G) entry which populates the outgoing interface towards the shared segment for the same source IP/destination group.

PIM assert mechanism is used to detect and eliminate duplication of multicast traffic on a shared segment. It is important to note that this mechanism does not prevent duplication of occurring, instead it uses duplication of multicast traffic as a trigger in order to activate this mechanism which elects a single forwarder for this stream.

When you have duplication of multicast traffic on a shared segment, you can assume that there are multiple routers that send the same (S,G) or (*,G) on a shared segment. If you elect one router to forward this stream effectively, it eliminates duplication.

PIM leverages on PIM assert messages which are triggered when you receive a multicast packet on the Outgoing Interface List (OIL). These assert messages contain metrics which are then used to calculate who will become assert winner. Downstream routers on the LAN also receive PIM assert messages. These messages are then used by downstream devices to send appropriate Join/Prune messages to the upstream router that won the assert ellection.

Scenario 1. LHR Rationale

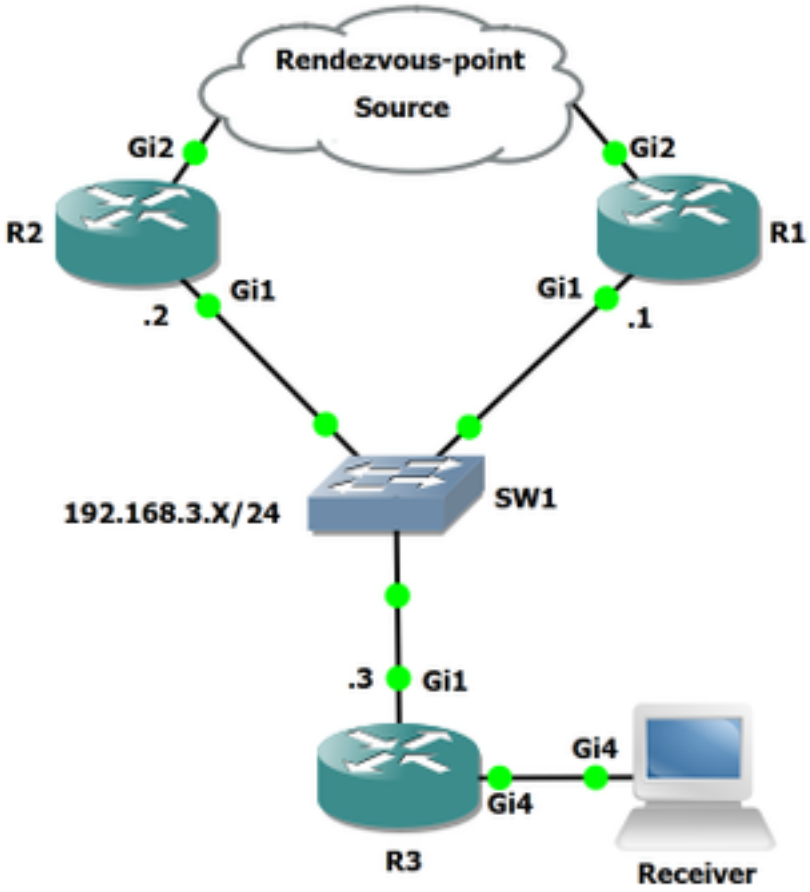


Figure 1.

In the network diagram, R3 is the Last Hop Router (LHR), R3 connects to both R2 and R1 via a shared segment.

When you receive an Internet Group Management Protocol (IGMP) report from the receiver, R3 checks who the RPF neighbour towards the RP is. In the topology, R1 is the RPF neighbour towards the RP, hence R3 sends a (*,G) join towards R1. Once R1 pulls down the stream (assume the group is active) R3 sends an (S,G) join towards the source and pulls the source tree down. R2 is the RPF neighbour towards the source tree which means R3 will send the (S,G) join towards R2. R3 has the same RPF interface towards both RP and the source. Here you can see the R3 mroute table for group 239.1.1.1.

```
R3#show ip mroute
IP Multicast Routing Table
```

```
Outgoing interface flags: H - Hardware switched, A - Assert winner, p - PIM Join
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode
(*, 239.1.1.1), 00:00:55/stopped, RP 192.168.0.100, flags: SJC
  Incoming interface: GigabitEthernet1, RPF nbr 192.168.3.1
  Outgoing interface list:
    GigabitEthernet4, Forward/Sparse, 00:00:55/00:02:04

(10.0.0.2, 239.1.1.1), 00:00:52/00:02:07, flags: JT
  Incoming interface: GigabitEthernet1, RPF nbr 192.168.3.2, Mroute
  Outgoing interface list:
    GigabitEthernet4, Forward/Sparse, 00:00:52/00:02:07

(*, 224.0.1.40), 00:01:22/00:02:09, RP 192.168.0.100, flags: SJPCL
  Incoming interface: GigabitEthernet1, RPF nbr 192.168.3.1
```

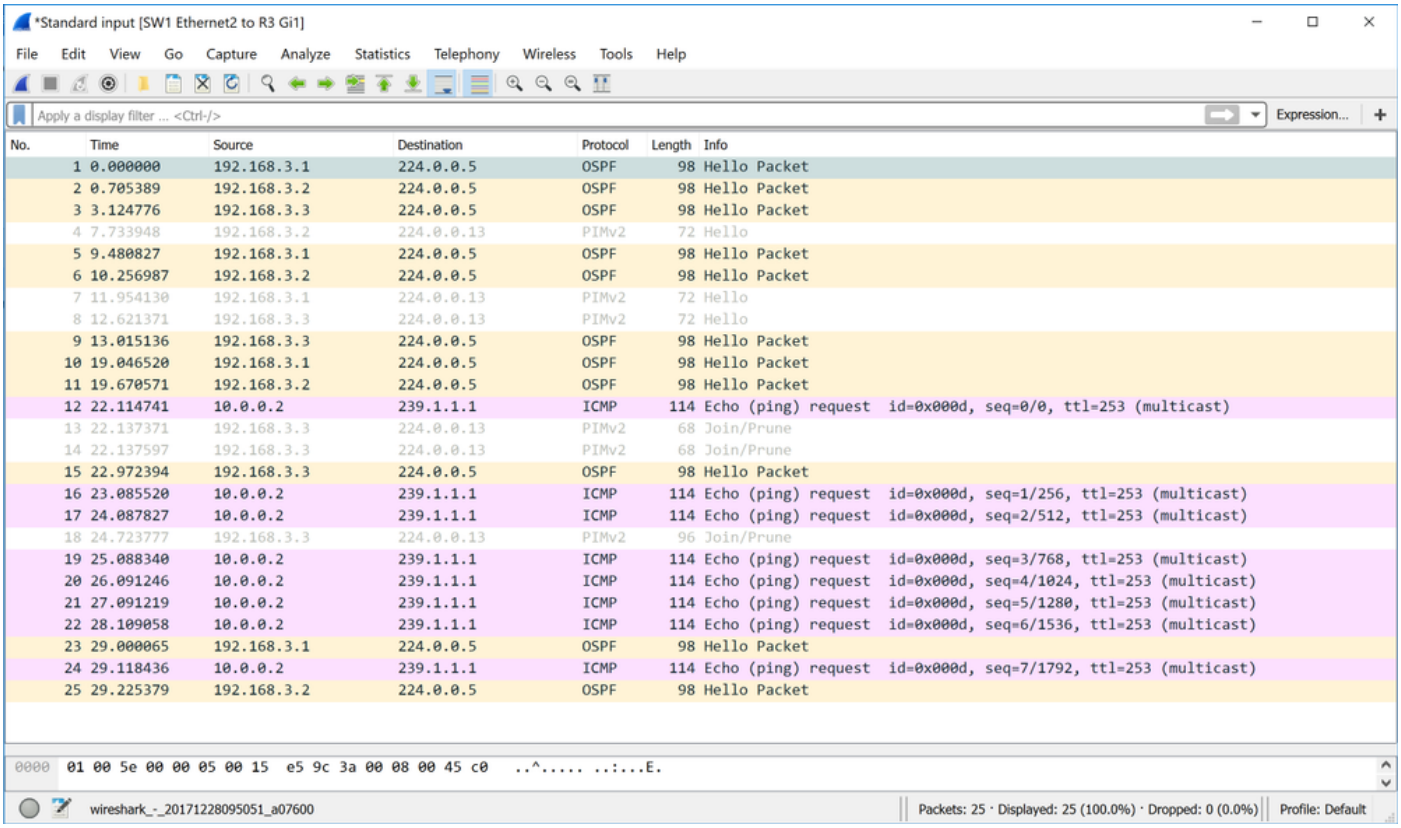
As you can see on R3 the (*,G) RPF neighbour is 192.168.3.1 and the RPF neighbour towards the (S,G) is 192.168.3.2. Now, **this should result in both R1 and R2 to have a valid OIL towards R3**. Let's have a look at these entries:

```
R1#show ip mroute
Outgoing interface flags: H - Hardware switched, A - Assert winner, p - PIM Join
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 239.1.1.1), 00:15:02/00:02:33, RP 192.168.0.100, flags: S
  Incoming interface: GigabitEthernet2, RPF nbr 192.168.5.2
  Outgoing interface list:
    GigabitEthernet1, Forward/Sparse, 00:15:02/00:02:33

(10.0.0.2, 239.1.1.1), 00:13:24/00:02:33, flags: PR
  Incoming interface: GigabitEthernet2, RPF nbr 192.168.5.2
  Outgoing interface list: Null

(*, 224.0.1.40), 00:29:17/00:02:51, RP 192.168.0.100, flags: SJCL
  Incoming interface: GigabitEthernet2, RPF nbr 192.168.5.2
  Outgoing interface list:
    GigabitEthernet1, Forward/Sparse, 00:16:06/00:02:51
  Outgoing interface list: Null
```

```
R2#show ip mroute
IP Multicast Routing Table
Outgoing interface flags: H - Hardware switched, A - Assert winner, p - PIM Join
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 239.1.1.1), 00:08:00/stopped, RP 192.168.0.100, flags: SP
  Incoming interface: GigabitEthernet2, RPF nbr 192.168.4.1
  Outgoing interface list: Null

(10.0.0.2, 239.1.1.1), 00:00:03/00:02:56, flags: T
  Incoming interface: GigabitEthernet2, RPF nbr 192.168.4.1
  Outgoing interface list:
    GigabitEthernet1, Forward/Sparse, 00:00:03/00:03:26

(*, 224.0.1.40), 01:37:30/00:02:22, RP 192.168.0.100, flags: SJPL
  Incoming interface: GigabitEthernet2, RPF nbr 192.168.4.1
```

As mentioned, assert can be triggered when there are two upstream routers which have a valid OIL populated on a shared segment. Since both R1 and R2 have a valid OIL, check if there is an assert mechanism in packet capture.

This packet capture was captured on R3 interface Gi1 towards SW1.



In this packet capture, you do not see any assert packets even though there are all the prerequisites to create duplication on the shared segment between R1, R2, and R3. Why do you not see any PIM assert packets when the (S,G) stream was activated?

It seems that RFC 7761 might hold the answer to these questions.

Abstract from RFC 7761 Section 4.2.2.

```
4.2.2.  Setting and Clearing the (S,G) SPTbit

   Basically, Update_SPTbit(S,G,iif) will set the SPTbit if we have the
   appropriate (S,G) join state, and if the packet arrived on the
   correct upstream interface for S, and if one or more of the following
   conditions apply:

   1.  The source is directly connected, in which case the switch to the
       SPT is a no-op.

   2.  The RPF interface to S is different from the RPF interface to the
       RP.  The packet arrived on RPF_interface(S), and so the SPT must
       have been completed.

   3.  No one wants the packet on the RP tree.

   4.  RPF'(S,G) == RPF'(*,G).  In this case, the router will never be
       able to tell if the SPT has been completed, so it should just
```

```
switch immediately.  The RPF'(S,G) != NULL check ensures that the
SPTbit is set only if the RPF neighbor towards S is valid.
```

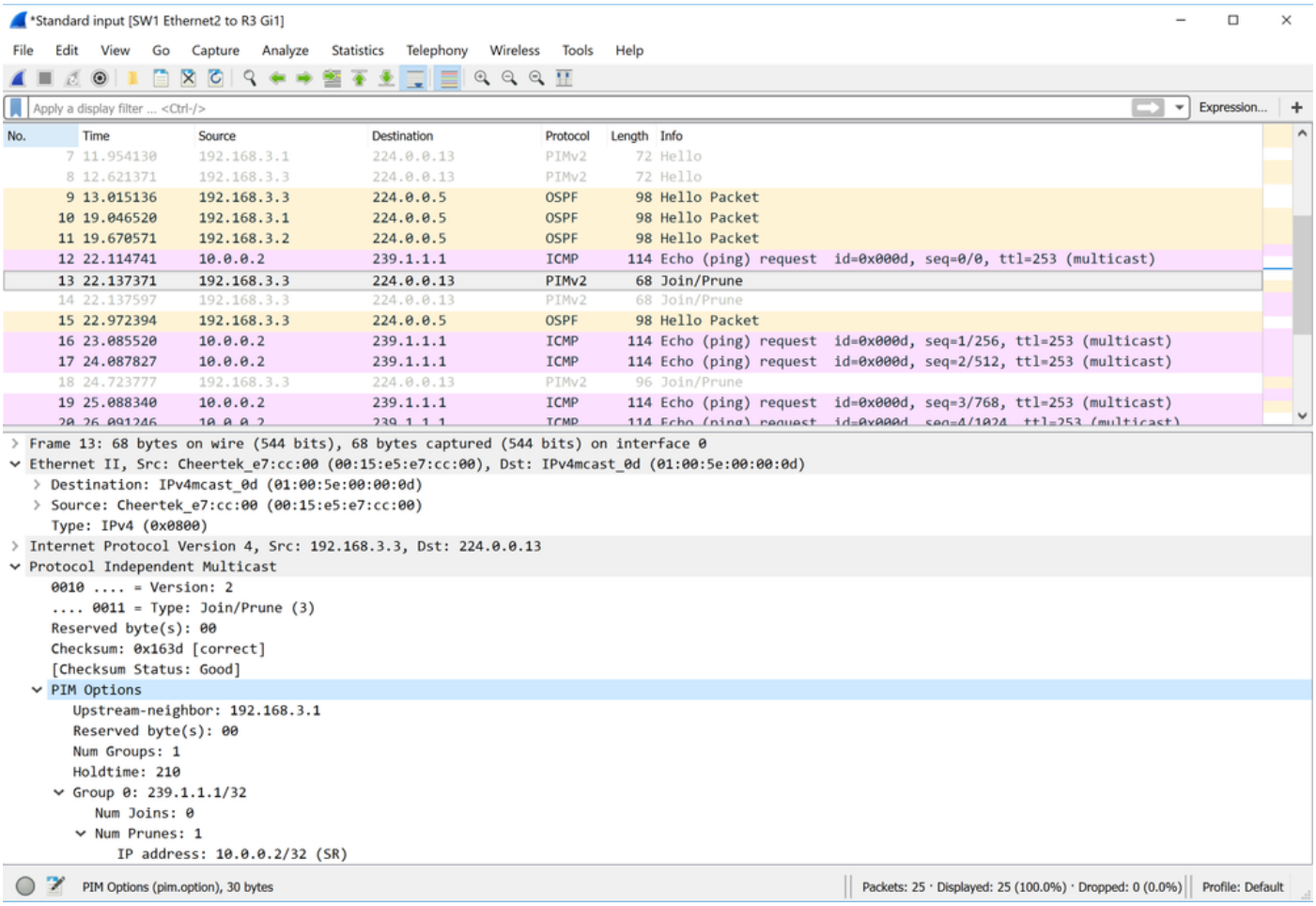
In the case where the RPF interface is the same for the RP and for S, but RPF'(S,G) and RPF'(*,G) differ, we wait for an Assert(S,G), which indicates that the upstream router with (S,G) state believes the SPT has been completed.

The (S,G) SPTbit is used to distinguish whether to forward on (*,G) or on (S,G) state. When you switch from the RP tree to the source tree, there is a transition period when data arrives due to upstream (*,G) state while upstream (S,G) state is established, at that time, the router should continue to forward only on (*,G) state. This prevents temporary black holes that would be caused by sending a Prune(S,G,rpt) before the upstream (S,G) state has finished being established.

Although it seems that the scenario can correlate to the last point mentioned above. In the case where the RPF interface is the same for the RP and for S,
but RPF'(S,G) and RPF'(*,G) differ, we wait for an Assert(S,G), which indicates that the upstream router with (S,G) state believes the SPT has been completed.

For assert to be triggered, the router must receive a duplicate packet on its already populated OIL for the same source IP/destination group on the segment. R3 is also a LHR, which means it is designated to switch from (*,G) towards SPT (S,G) when a packet is received from (*,G).

In the packet capture we observe that no asserts are triggered. Although we do see a prune sent immediately after the first ICMP echo is received.



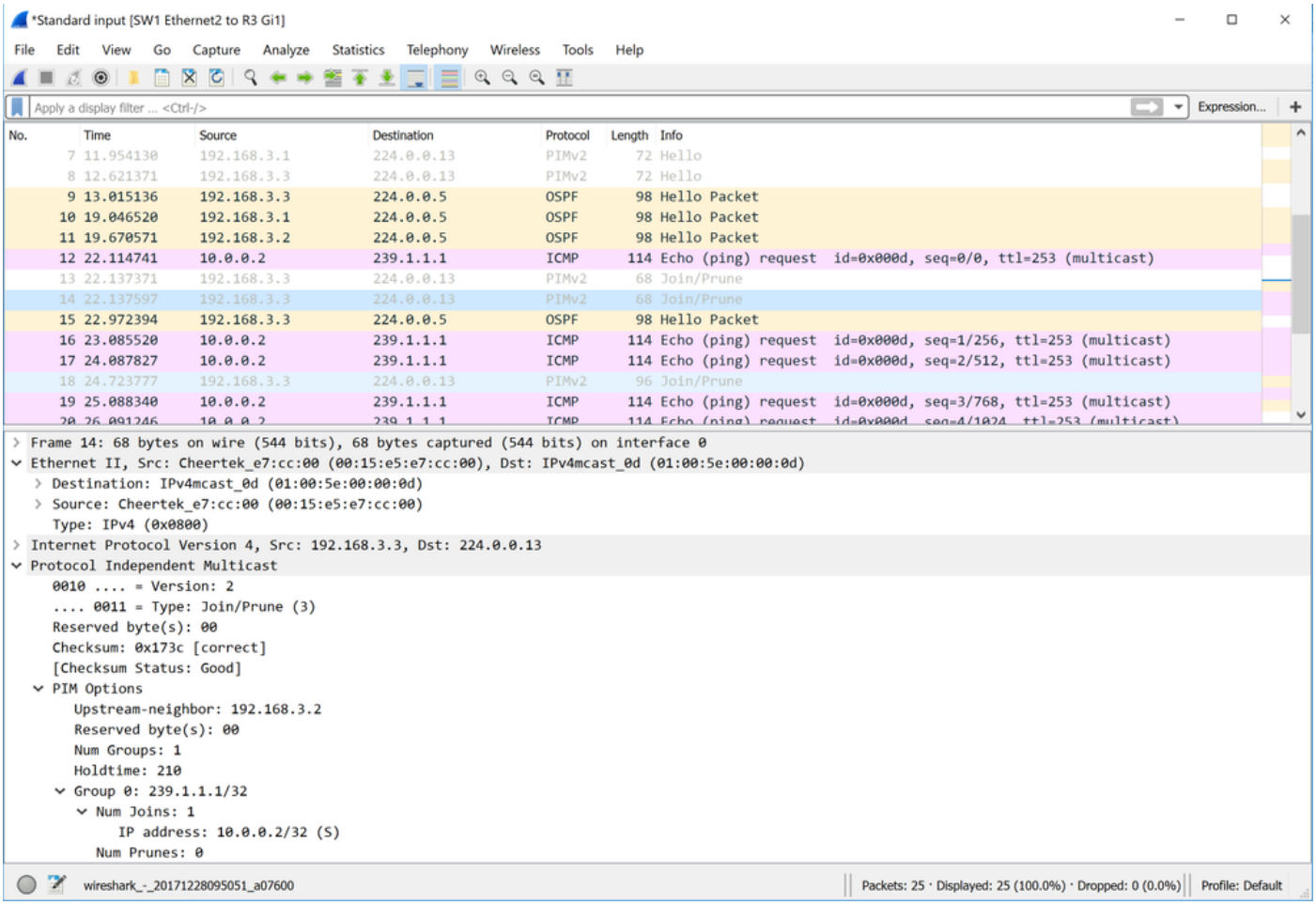
As you can see, once the first Internet Control Message Protocol (ICMP) request packet is received on R3 interface G1, a (*,G) SR-bit prune is sent towards upstream neighbour

192.168.3.1. This prunes (*,G) for the specific source defined.

You can see these flags also set: (SR):

```
The S flag: indicates that the multicast group is a sparse mode group.
The R flag: The R flag is the RP-bit flag and indicates that the information in the (S, G) entry
is applicable to the shared tree.
```

In the second PIM packet No. 14, you can see that R3 tries to join the (S,G) tree.



It is observed that once the first data plane is received, packet R3 prunes the (*,G) and builds the (S,G). This is the reason why you do not see PIM assert packets. This certain scenario is in effect when you have a LHR who has same RPF interface for (S,G) and (*,G). Although this behaviour may slightly differ from RFC 7761, it should not cause any issues.

Now let’s continue with Scenario 2., the diagram of this scenario can be seen here:

Scenario 2. Assert Path Selection

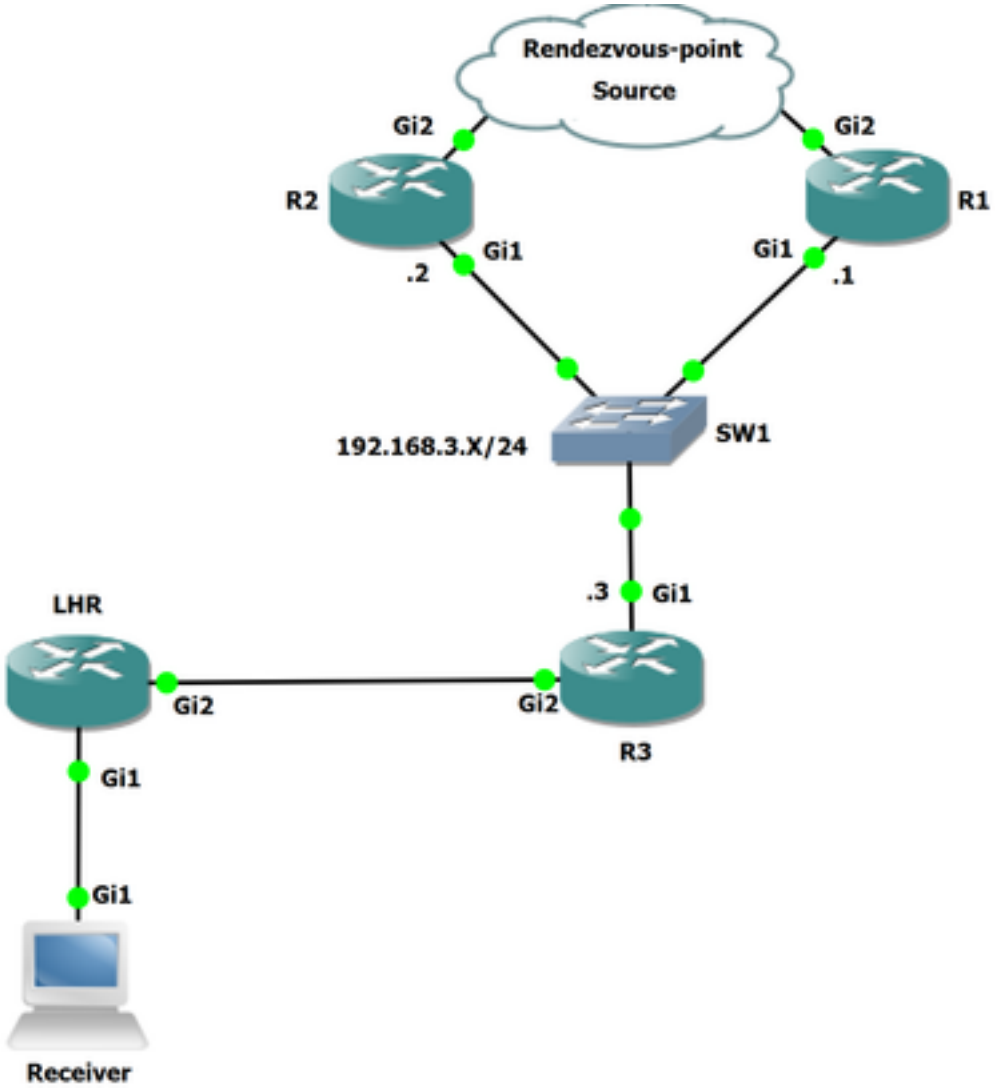


Figure 2.

In this topology, there is another router connected on R3 which is the LHR. The LHR connects directly to the receiver. The source and RP are both above R2 and R1. The RPF neighbour on R3 towards the RP is R1 and the RPF neighbour towards the source is R2.

Let's check the RPF neighbour for both the source and RP.

Here you see the RPF neighbour towards the RP: 192.168.0.100 is 192.168.3.1.

```
R3#show ip rpf 192.168.0.100
RPF information for ? (192.168.0.100)
  RPF interface: GigabitEthernet1
  RPF neighbor: ? (192.168.3.1)
  RPF route/mask: 192.168.0.100/32
  RPF type: unicast (ospf 1)
  Doing distance-preferred lookups across tables
  RPF topology: ipv4 multicast base, originated from ipv4 unicast base
```

Here you see the RPF neighbour towards the Source: 10.0.0.2 is 192.168.3.2.

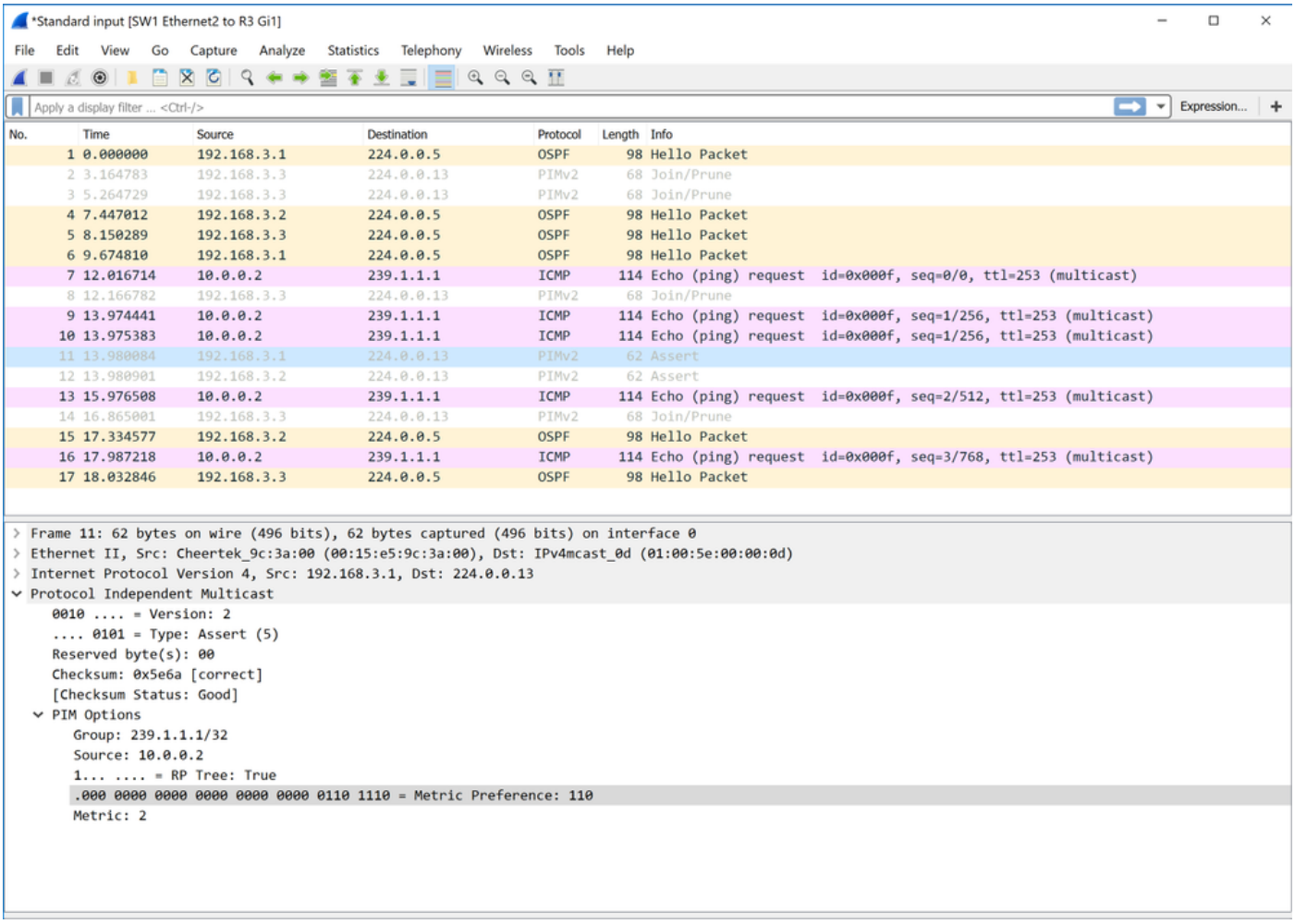
```
R3#show ip rpf 10.0.0.2
RPF information for ? (10.0.0.2)
  RPF interface: GigabitEthernet1
```

```
  RPF neighbor: ? (192.168.3.2)
  RPF route/mask: 10.0.0.0/24
  RPF type: unicast (ospf 1)
  Doing distance-preferred lookups across tables
  RPF topology: ipv4 multicast base, originated from ipv4 unicast base
```

Before we activate the source, lets take a look at the mroute table on R3, as you can see there is already (*,G) for the group 239.1.1.1. This is because the receiver connected to LHR has already requested for the specified group.

```
R3#show ip mroute
IP Multicast Routing Table
Outgoing interface flags: H - Hardware switched, A - Assert winner, p - PIM Join
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 239.1.1.1), 00:00:57/00:02:32, RP 192.168.0.100, flags: S
  Incoming interface: GigabitEthernet1, RPF nbr 192.168.3.1
  Outgoing interface list:
    GigabitEthernet2, Forward/Sparse, 00:00:57/00:02:32

(*, 224.0.1.40), 00:11:24/00:02:41, RP 192.168.0.100, flags: SJCL
  Incoming interface: GigabitEthernet1, RPF nbr 192.168.3.1
  Outgoing interface list:
    GigabitEthernet2, Forward/Sparse, 00:02:02/00:02:41
```

Now, activate the source and capture packets on R3 interface Gi1.



As you can see in this packet capture, PIM asserts packets are already present.

Frame 11:

```
> Frame 11: 62 bytes on wire (496 bits), 62 bytes captured (496 bits) on interface 0
> Ethernet II, Src: Cheertek_9c:3a:00 (00:15:e5:9c:3a:00), Dst: IPv4mcast_0d (01:00:5e:00:00:0d)
> Internet Protocol Version 4, Src: 192.168.3.1, Dst: 224.0.0.13
v Protocol Independent Multicast
    0010 .... = Version: 2
    .... 0101 = Type: Assert (5)
    Reserved byte(s): 00
    Checksum: 0x5e6a [correct]
    [Checksum Status: Good]
  v PIM Options
      Group: 239.1.1.1/32
      Source: 10.0.0.2
      1... .... = RP Tree: True
      .000 0000 0000 0000 0000 0000 0110 1110 = Metric Preference: 110
      Metric: 2
```

Frame 12:

```
> Frame 12: 62 bytes on wire (496 bits), 62 bytes captured (496 bits) on interface 0
> Ethernet II, Src: Cheertek_8b:3e:00 (00:15:e5:8b:3e:00), Dst: IPv4mcast_0d (01:00:5e:00:00:0d)
> Internet Protocol Version 4, Src: 192.168.3.2, Dst: 224.0.0.13
v Protocol Independent Multicast
    0010 .... = Version: 2
    .... 0101 = Type: Assert (5)
    Reserved byte(s): 00
    Checksum: 0xde6a [correct]
    [Checksum Status: Good]
  v PIM Options
      Group: 239.1.1.1/32
      Source: 10.0.0.2
      0... .... = RP Tree: False
      .000 0000 0000 0000 0000 0000 0110 1110 = Metric Preference: 110
      Metric: 2
```

When you look at these packets, you should be able to determine who is the assert winner. Now let's take a look at the PIM assert forwarder selection.

The metric preference is the Administrative Distance (AD). This refers to the administrative distance of the routing protocol installing the route in the routing table, which is used to look up the source IP address and the Metric is the cost of the route.

There are also other attributes which are used to determine who is the assert winner. You can see these details in RFC 7761.

Abstract from RFC 7761 Section 4.6.3.

```
4.6.3.  Assert Metrics

   Assert metrics are defined as:

     struct assert_metric {
       rpt_bit_flag;
       metric_preference;
```

```
      route_metric;
      ip_address;
    };
  When comparing assert_metrics, the rpt_bit_flag, metric_preference,
  and route_metric fields are compared in order, where the first lower
  value wins.  If all fields are equal, the primary IP address of the
  router that sourced the Assert message is used as a tie-breaker, with
  the highest IP address winning.
```

With the use of these fields defined and path selection, you can determine who the assert winner will be in this scenario. If you take a look at the assert packets again, you can see that metric preference is not compared since the decision is made on the very first selection criteria which is rpt_bit_flag.

In this scenario, comparing R1 and R2 is compared. Both routers send assert messages which were seen previously and once both devices see each other's assert messages they can compare metrics between each other in order to determine who the winner is.

Since R2 sends an assert message with the RP tree: False which has a value of 0, it is indeed lower than what R1 sent with a RP tree: True which has a value of 1. RP tree bit is set to either 0 or 1.

RP tree bit when set to 1 means that you are currently on the shared tree; the RPT bit cleared indicates that the sender of the assert had (S,G) forwarding state on an interface.

As (S,G) asserts have priority over (*,G) asserts, R2 should be the assert winner. Transition to the "I am Assert Winner" state. As mentioned in the earlier statement in RFC 7761, the lower value is more preferred.

Let's have a look at both R1 and R2 in order to see who the assert winner is.

```
R2#show ip mroute
IP Multicast Routing Table
Outgoing interface flags: H - Hardware switched, A - Assert winner, p - PIM Join
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 239.1.1.1), 00:42:52/stopped, RP 192.168.0.100, flags: SP
  Incoming interface: GigabitEthernet2, RPF nbr 192.168.4.1
  Outgoing interface list: Null

(10.0.0.2, 239.1.1.1), 00:42:52/00:01:40, flags: T
  Incoming interface: GigabitEthernet2, RPF nbr 192.168.4.1
  Outgoing interface list:
    GigabitEthernet1, Forward/Sparse, 00:42:52/00:03:07, A

(*, 224.0.1.40), 00:43:23/00:02:25, RP 192.168.0.100, flags: SJPL
  Incoming interface: GigabitEthernet2, RPF nbr 192.168.4.1
  Outgoing interface list: Null
```

In this output, you can see that the (S,G) on R2 has the A flag set on the OIL which indicates that it is the assert winner. Here on R1, you do not have an OIL on the (S,G) and the P flag is set which means the particular (S,G) has been pruned off in this case: it is not the assert winner.

Note: When assert is present on a shared segment, downstream neighbours send Join(*,G)

and Join(S,G) periodic messages to the appropriate RPF neighbour, i.e., the RPF neighbour as modified by the assert process. They are not always sent to the RPF neighbour as indicated by the MRIB.

```
R1#show ip mroute
IP Multicast Routing Table
Outgoing interface flags: H - Hardware switched, A - Assert winner, p - PIM Join
 Timers: Uptime/Expires
 Interface state: Interface, Next-Hop or VCD, State/Mode

(*, 239.1.1.1), 00:44:32/00:03:09, RP 192.168.0.100, flags: S
  Incoming interface: GigabitEthernet2, RPF nbr 192.168.5.2
  Outgoing interface list:
    GigabitEthernet1, Forward/Sparse, 00:44:32/00:03:09, A

(10.0.0.2, 239.1.1.1), 00:44:19/00:03:09, flags: PR
  Incoming interface: GigabitEthernet2, RPF nbr 192.168.5.2
  Outgoing interface list: Null

(*, 224.0.1.40), 00:44:50/00:02:53, RP 192.168.0.100, flags: SJCL
  Incoming interface: GigabitEthernet2, RPF nbr 192.168.5.2
  Outgoing interface list:
    GigabitEthernet1, Forward/Sparse, 00:43:56/00:02:53
```

If it is the case that both R1 and R2 have RP tree bit set to 1. you can then consider the router with the lowest AD; if equal, then take a look at the metric. If RP tree bit is true on both routers, the metric is compared toward RP IP address. If RP tree bit is 0, the metric is compared towards the source of the multicast stream.

If all these values are the same, the highest IP address sourcing assert message is the winner.

Summary

In scenario one, you did not observe assert packets, however, per RFC they should have been triggered. As mentioned, this was because R3 was pruning (*,G) before control plane for (S,G) was built.

While in scenario two, you see assert packets. When the first packet was received on LHR, it would send a (S,G) join/prune towards R3 to pull the source/group. R3 will then send a join/prune packet towards R2 for the same source/group. This would then cause both R1 and R2 to have valid OILs populated. Now R3 only prunes (S,G) with RP-bit set when the T flag is populated on R3s (S,G) state. For this to happen, you need to receive another data plane packet from the shared segment. Because control plane has already been built for (S,G) this then leads to duplication on the shared segment triggering assert messages.