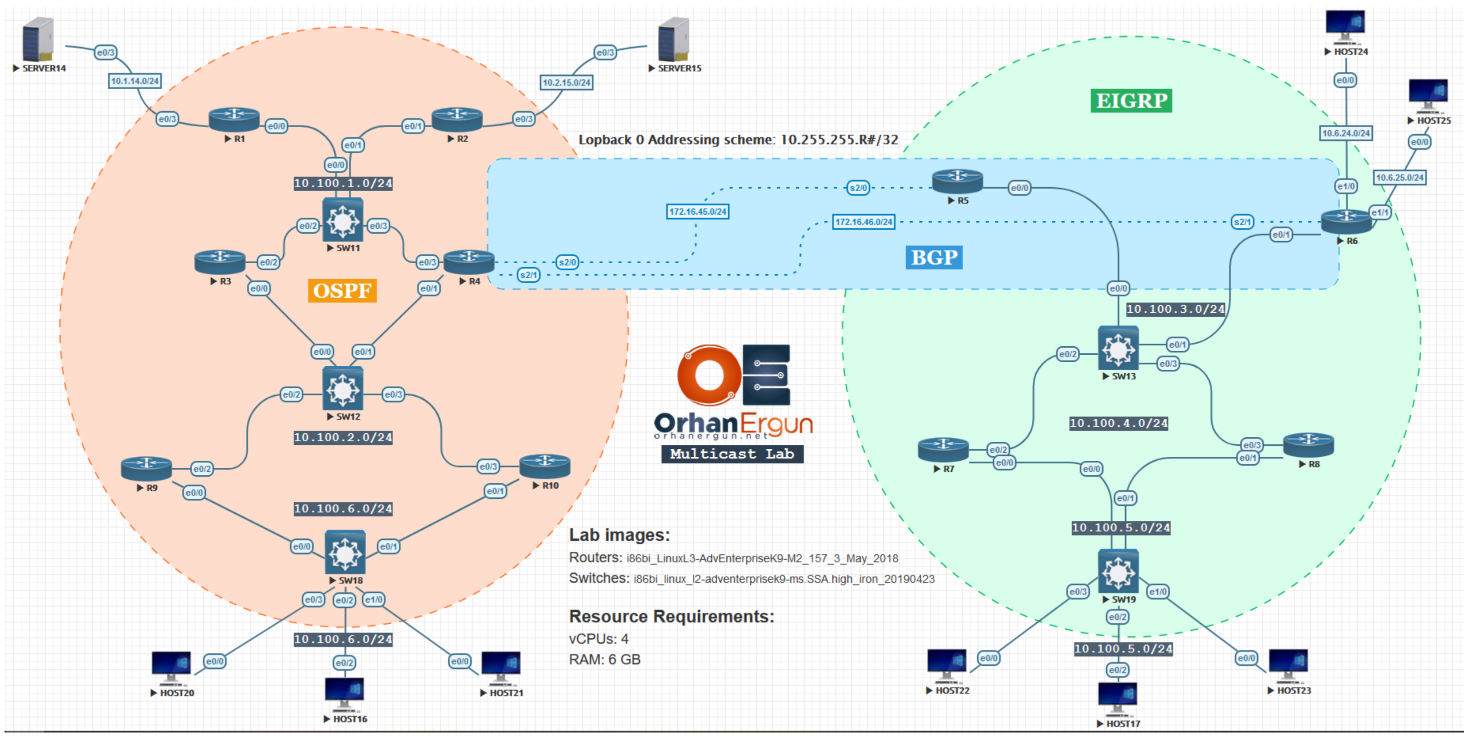
IP Multicast Lab

Internet Protocol Multicast Lab

Topology:



Lab images:
Routers: i86bi_LinuxL3-AdvEnterpriseK9-M2_157_3_May_2018
Switches: i86bi_linux_l2-adventerprisek9-ms.SSA.high_iron_20190423

Resource Requirements:
vCPUs: 4
RAM: 6 GB

Note: All Loopback 0 interface IPv4 addressing scheme: 10.255.255.R#/32

IGPs: OSPF, EIGRP, BGP

PIM:

- Dense Mode
- Sparse Mode
- Sparse Dense Mode
- SSM
- Bidirectional-PIM

RP:

- Static RP
- Auto-RP
- BSR
- Phantom RP
- Anycast RP (MSDP usage)

L2 Multicast:

- IGMP
- IGMP-Snooping
- MLD

Task 01:

- Enable IPv4 Multicast-Routing on all routers in the topology
- Enable PIM Dense-Mode on all routers in the topology
- All clients must receive pings destined to 227.27.27.27 group sent from both servers

Solution:

In this task we imagine that there are a lot of receivers, and our two servers are supposed to stream some video for them. First of all IPv4 multicast-routing must be enable on all of the routers, then we need to simply enable PIM Dense-Mode for all the interfaces that are going to forward multicast data:

```
On all routers (and also SW13):
ip multicast-routing
!

R1:
interface Ethernet0/0
 ip pim dense-mode
!
interface Ethernet0/3
 ip pim dense-mode
!

R2:
interface Ethernet0/1
 ip pim dense-mode
!
interface Ethernet0/3
```

```
 ip pim dense-mode
!

R3:
interface Ethernet0/0
 ip pim dense-mode
!
interface Ethernet0/2
 ip pim dense-mode
!

R4:
interface Ethernet0/1
 ip pim dense-mode
!
interface Ethernet0/3
 ip pim dense-mode
!
interface Serial2/0
 ip pim dense-mode
!
interface Serial2/1
 ip pim dense-mode
!
!

R9:
interface Ethernet0/0
 ip pim dense-mode
!
interface Ethernet0/2
 ip pim dense-mode
!
```

R10:

```
interface Ethernet0/1

 ip pim dense-mode

!

interface Ethernet0/3

 ip pim dense-mode

!
```

R5:

```
interface Ethernet0/0

 ip pim dense-mode

!

interface Serial2/0

 ip pim dense-mode

!
```

R6:

```
interface Serial2/1

 ip pim dense-mode

!

interface Ethernet0/1

 ip pim dense-mode

!

interface Ethernet1/0

 ip pim dense-mode

!

interface Ethernet1/1

 ip pim dense-mode

!
```

SW13:

```
interface Vlan3

 ip pim dense-mode

!

interface Vlan4
```

```
 ip pim dense-mode
!

R7:
interface Ethernet0/0
 ip pim dense-mode
!
interface Ethernet0/2
 ip pim dense-mode
!

R8:
interface Ethernet0/1
 ip pim dense-mode
!
interface Ethernet0/3
 ip pim dense-mode
!
```

Now clients must join the group (227.27.27.27):

```
Host20-Host25:
interface Ethernet0/0
 ip igmp join-group 227.27.27.27
!

Host16-Host17:
interface Ethernet0/2
 ip igmp join-group 227.27.27.27
!
```

Verification:

In this Lab for any verification purpose we will use simple group pings, I know it is not much interesting! ☺ but in Multicast VPN Lab (MVPN Lab), VLC has been used to stream a real video and do the verifications. After doing this lab, please visit the MVPN Lab!

```
SERVER14#ping 227.27.27.27

Type escape sequence to abort.

Sending 1, 100-byte ICMP Echos to 227.27.27.27, timeout is 2 seconds:


Reply to request 0 from 10.100.6.21, 6 ms

Reply to request 0 from 10.100.5.17, 13 ms

Reply to request 0 from 10.100.5.22, 13 ms

Reply to request 0 from 10.100.5.23, 9 ms

Reply to request 0 from 10.6.24.24, 9 ms

Reply to request 0 from 10.6.25.25, 9 ms

Reply to request 0 from 10.100.6.16, 6 ms

Reply to request 0 from 10.100.6.20, 6 ms




R1(config)#do sh ip mroute | begin \(

(*, 227.27.27.27), 00:00:38/stopped, RP 0.0.0.0, flags: D

  Incoming interface: Null, RPF nbr 0.0.0.0

  Outgoing interface list:

    Ethernet0/0, Forward/Dense, 00:00:38/stopped


(10.1.14.14, 227.27.27.27), 00:00:38/00:02:21, flags: T

  Incoming interface: Ethernet0/3, RPF nbr 0.0.0.0

  Outgoing interface list:

    Ethernet0/0, Forward/Dense, 00:00:38/stopped


(*, 224.0.1.40), 02:48:28/00:02:33, RP 0.0.0.0, flags: DCL

  Incoming interface: Null, RPF nbr 0.0.0.0

  Outgoing interface list:

    Ethernet0/0, Forward/Dense, 00:33:26/stopped



R4(config)#do sh ip mroute | begin \(

(*, 227.27.27.27), 00:01:06/stopped, RP 0.0.0.0, flags: D

  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial2/1, Forward/Dense, 00:01:06/stopped
    Serial2/0, Forward/Dense, 00:01:06/stopped
    Ethernet0/3, Forward/Dense, 00:01:06/stopped
    Ethernet0/1, Forward/Dense, 00:01:06/stopped

(10.1.14.14, 227.27.27.27), 00:01:06/00:01:53, flags: T
  Incoming interface: Ethernet0/3, RPF nbr 10.100.1.1
  Outgoing interface list:
    Ethernet0/1, Forward/Dense, 00:01:06/stopped, A
    Serial2/0, Prune/Dense, 00:01:06/00:01:53
    Serial2/1, Forward/Dense, 00:01:06/stopped

(*, 224.0.1.40), 02:48:58/00:02:08, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Ethernet0/1, Forward/Dense, 00:32:55/stopped
    Ethernet0/3, Forward/Dense, 02:47:56/stopped
```

As you realized, there is a wildcard entry (*, 227.27.27.27) to forward traffic everywhere. There is no RP in our topology (RP 0.0.0.0).

For each of the senders there will be a source Tree with RPF (Reverse Path Forwarding) neighbor check. For example for SERVER14 as the sender of the stream (in this scenarion stream = pings!) and for the group of receivers (227.27.27.27), an specific entry has been created:

```
R4(config)#do sh ip mroute | begin \(
(*, 227.27.27.27), 00:01:06/stopped, RP 0.0.0.0, flags: D
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Serial2/1, Forward/Dense, 00:01:06/stopped
    Serial2/0, Forward/Dense, 00:01:06/stopped
    Ethernet0/3, Forward/Dense, 00:01:06/stopped
    Ethernet0/1, Forward/Dense, 00:01:06/stopped

(10.1.14.14, 227.27.27.27), 00:01:06/00:01:53, flags: T
```

```
  Incoming interface: Ethernet0/3, RPF nbr 10.100.1.1
  Outgoing interface list:
    Ethernet0/1, Forward/Dense, 00:01:06/stopped, A
    Serial2/0, Prune/Dense, 00:01:06/00:01:53
    Serial2/1, Forward/Dense, 00:01:06/stopped

(*, 224.0.1.40), 02:48:58/00:02:08, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Ethernet0/1, Forward/Dense, 00:32:55/stopped
    Ethernet0/3, Forward/Dense, 02:47:56/stopped
```

This RPF check must be walid towards that incoming interface, in other words:

When R4 receives ping from R1 on interface e0/3, the source address of the packet must be reachable (sender IP address) via that interface. PIM is going to check the routing table information to perform that RPF check. It does not matter which routing protocol has been used to collect the information, that is why the protocol is called PIM (Protocol Independent Multicast). We have used OSPF, EIGRP, BGP in some parts of topology but PIM still works thanks to protocol independency!

Let's check the routing table of R4:

```
R4(config)#do sh ip route | include 10.1.14.0
O        10.1.14.0/24 [110/20] via 10.100.1.1, 03:08:30, Ethernet0/3
```

The sender address is reachable through Ethernet 0/3 so RPF check should be OK.

Let's send the traffic using both servers at the same time:

```
SERVER14#ping 227.27.27.27
Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 227.27.27.27, timeout is 2 seconds:

Reply to request 0 from 10.100.6.16, 5 ms
Reply to request 0 from 10.100.5.17, 13 ms
Reply to request 0 from 10.100.5.22, 13 ms
Reply to request 0 from 10.100.5.23, 9 ms
Reply to request 0 from 10.6.24.24, 9 ms
Reply to request 0 from 10.6.25.25, 9 ms
Reply to request 0 from 10.100.6.21, 6 ms

Reply to request 0 from 10.100.6.20, 6 ms


SERVER15#ping 227.27.27.27

Type escape sequence to abort.

Sending 1, 100-byte ICMP Echos to 227.27.27.27, timeout is 2 seconds:


Reply to request 0 from 10.100.6.20, 4 ms

Reply to request 0 from 10.100.5.17, 11 ms

Reply to request 0 from 10.100.5.23, 11 ms

Reply to request 0 from 10.100.5.22, 7 ms

Reply to request 0 from 10.6.24.24, 7 ms

Reply to request 0 from 10.6.25.25, 7 ms

Reply to request 0 from 10.100.6.16, 7 ms

Reply to request 0 from 10.100.6.21, 4 ms


R4(config)#do sh ip mroute | begin \(

(*, 227.27.27.27), 00:01:04/stopped, RP 0.0.0.0, flags: D

  Incoming interface: Null, RPF nbr 0.0.0.0

  Outgoing interface list:

    Serial2/1, Forward/Dense, 00:01:04/stopped

    Serial2/0, Forward/Dense, 00:01:04/stopped

    Ethernet0/3, Forward/Dense, 00:01:04/stopped

    Ethernet0/1, Forward/Dense, 00:01:04/stopped


(10.2.15.15, 227.27.27.27), 00:00:42/00:02:17, flags: T

  Incoming interface: Ethernet0/3, RPF nbr 10.100.1.2

  Outgoing interface list:

    Ethernet0/1, Forward/Dense, 00:00:42/stopped, A

    Serial2/0, Prune/Dense, 00:00:38/00:02:21

    Serial2/1, Forward/Dense, 00:00:42/stopped


(10.1.14.14, 227.27.27.27), 00:01:04/00:01:55, flags: T

  Incoming interface: Ethernet0/3, RPF nbr 10.100.1.1
  Outgoing interface list:
    Ethernet0/1, Forward/Dense, 00:01:04/stopped, A
    Serial2/0, Prune/Dense, 00:01:04/00:01:55
    Serial2/1, Forward/Dense, 00:01:04/stopped

(*, 224.0.1.40), 03:13:04/00:02:58, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Ethernet0/1, Forward/Dense, 00:57:01/stopped
    Ethernet0/3, Forward/Dense, 03:12:03/stopped
```

For each of the senders (Server14, Server15) an entry has been created, these are the source trees.

Take a look at the RPF nbr fields, those are different neighbors because R1 forwarded some of the traffic (from Server14) and R2 forwarded some of the traffic also.

Both of the RPF checks are successful because R4 knows how to reach to both senders thanks to OSPF!

Let's check LHRs (Last Hop Routers) IGMP and multicast route information:

```
R8(config)#do sh ip igmp groups
IGMP Connected Group Membership
Group Address    Interface                Uptime    Expires   Last Reporter   Group Accounted
227.27.27.27     Ethernet0/1              03:25:41  00:02:16  10.100.5.22
224.0.1.40       Ethernet0/1              03:26:44  00:02:20  10.100.5.7

R7(config)#do sh ip igmp groups
IGMP Connected Group Membership
Group Address    Interface                Uptime    Expires   Last Reporter   Group Accounted
227.27.27.27     Ethernet0/0              03:25:55  00:02:02  10.100.5.22
224.0.1.40       Ethernet0/0              03:26:58  00:02:06  10.100.5.7



R7(config)#do sh ip mroute verbose | begin \(
(*, 227.27.27.27), 01:14:22/stopped, RP 0.0.0.0, flags: DC
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Ethernet0/2, Forward/Dense, 01:14:22/stopped
    Ethernet0/0, Forward/Dense, 01:14:22/stopped

(10.2.15.15, 227.27.27.27), 00:00:25/00:02:34, flags: PT
  Incoming interface: Ethernet0/2, RPF nbr 10.100.4.13
  Outgoing interface list:
    Ethernet0/0, Prune/Dense, 00:00:25/00:02:34

(10.1.14.14, 227.27.27.27), 00:04:09/00:02:35, flags: T
  Incoming interface: Ethernet0/2, RPF nbr 10.100.4.13
  Outgoing interface list:
    Ethernet0/0, Forward/Dense, 00:04:09/stopped, A



(*, 224.0.1.40), 03:28:23/00:02:36, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Ethernet0/0, Forward/Dense, 01:15:02/stopped


R8(config)#do sh ip mroute verbose | begin \(
(*, 227.27.27.27), 01:14:29/stopped, RP 0.0.0.0, flags: DC
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Ethernet0/3, Forward/Dense, 01:14:29/stopped
    Ethernet0/1, Forward/Dense, 01:14:29/stopped

(10.2.15.15, 227.27.27.27), 00:00:32/00:02:27, flags: T
  Incoming interface: Ethernet0/3, RPF nbr 10.100.4.13
  Outgoing interface list:
    Ethernet0/1, Forward/Dense, 00:00:32/stopped, A
```

```
(10.1.14.14, 227.27.27.27), 00:04:16/00:02:29, flags: PT
  Incoming interface: Ethernet0/3, RPF nbr 10.100.4.13
  Outgoing interface list:
    Ethernet0/1, Prune/Dense, 00:04:16/00:01:40

(*, 224.0.1.40), 03:28:30/00:02:29, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Ethernet0/1, Forward/Dense, 03:28:29/stopped
```

Only one of two Last Hop Routers (LHRs) is in the forwarding state (R8).

This forwarder was elected using PIM Assert mechanism. In this mechanism both of the LHRs send assert messages on the shared segment, that message includes some fields such as assert_metrics (route metrics), metric_preference (which is Administrative Distance of the route), then comparison happens, the router that has the lowest value wins, if they stuck (for example AD values are the same on both LHRs) the router with the highest IP address (Assert message source IP address) wins.

You have also noticed that Hosts do not care about source (sender) of the group, thy just accept all traffic destined to the group 227.27.27.27, this is called Any Source Multicast (ASM).

Task 02:

- Configure PIM Sparse-Mode on all routers
- Configure R4 as the static Rendezvous-point
- All clients must receive pings destined to 227.27.27.27 group sent from both servers

Solution:

In this task we imagine that we don't have a lot of receivers, instead we have some multicast applications (groups) and some receivers for each of them.

```
R1:
interface Ethernet0/0
 ip pim sparse-mode
!
interface Ethernet0/3
 ip pim sparse-mode
!

R2:
interface Ethernet0/1
 ip pim sparse-mode
!
interface Ethernet0/3
 ip pim sparse-mode
!

R3:
interface Ethernet0/0
 ip pim sparse-mode
!
interface Ethernet0/2
 ip pim sparse-mode
!
```

```
R4:
interface Ethernet0/1
 ip pim sparse-mode
!
interface Ethernet0/3
 ip pim sparse-mode
!
interface Serial2/0
 ip pim sparse-mode
!
interface Serial2/1
 ip pim sparse-mode
!

R9:
interface Ethernet0/0
 ip pim sparse-mode
!
interface Ethernet0/2
 ip pim sparse-mode
!

R10:
interface Ethernet0/1
 ip pim sparse-mode
!
interface Ethernet0/3
 ip pim sparse-mode
!

R5:
interface Ethernet0/0
 ip pim sparse-mode
!
interface Serial2/0
```

```
 ip pim sparse-mode
!

R6:
interface Serial2/1
 ip pim sparse-mode
!
interface Ethernet0/1
 ip pim sparse-mode
!
interface Ethernet1/0
 ip pim sparse-mode
!
interface Ethernet1/1
 ip pim sparse-mode
!

SW13:
interface Vlan3
 ip pim sparse-mode
!
interface Vlan4
 ip pim sparse-mode
!

R7:
interface Ethernet0/0
 ip pim sparse-mode
!
interface Ethernet0/2
 ip pim sparse-mode
!

R8:
```

```
interface Ethernet0/1
 ip pim sparse-mode
!
interface Ethernet0/3
 ip pim sparse-mode
!
```

Before any Rendezvous-Point configuration let's ping 227.27.27.27 to see if it works or not:

```
SERVER14(config)#do ping 227.27.27.27
Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 227.27.27.27, timeout is 2 seconds:
.
SERVER14(config)#do ping 227.27.27.27
Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 227.27.27.27, timeout is 2 seconds:
.



SERVER15(config)#do ping 227.27.27.27
Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 227.27.27.27, timeout is 2 seconds:
.
SERVER15(config)#do ping 227.27.27.27
Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 227.27.27.27, timeout is 2 seconds:
.
```

It is not working without RP configuration, The FHR needs to do registeration on RP and LHR needs to join the group, they cannot find any RP to do so… .

Let's configure an static RP (R4 Loopback 0 address as the RP):

```
On all routers and also SW13 MLS:
ip pim rp-address 10.255.255.4
R4:
interface Loopback 0
 ip pim sparse-mode
```

Verification:

```
R1(config)#ip pim rp-address 10.255.255.4

R1(config)#

*Oct 19 22:04:27.691: %LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel0, changed state to up

R1(config)#do sh ip int br | include Tunnel0

Tunnel0                    10.100.1.1      YES unset  up                    up


R1(config)#do sh int tunnel 0

Tunnel0 is up, line protocol is up

  Hardware is Tunnel

  Description: Pim Register Tunnel (Encap) for RP 10.255.255.4

  Interface is unnumbered. Using address of Ethernet0/0 (10.100.1.1)

  MTU 17912 bytes, BW 100 Kbit/sec, DLY 50000 usec,

     reliability 255/255, txload 1/255, rxload 1/255

  Encapsulation TUNNEL, loopback not set

  Keepalive not set

  Tunnel linestate evaluation up

  Tunnel source 10.100.1.1 (Ethernet0/0), destination 10.255.255.4

   Tunnel Subblocks:

      src-track:

         Tunnel0 source tracking subblock associated with Ethernet0/0

          Set of tunnels with source Ethernet0/0, 1 member (includes iterators), on interface <OK>

  Tunnel protocol/transport PIM/IPv4

--More--
```

A tunnel interface for PIM registration has been created destined to the RP address, the important thing here is: RP address must be reachable otherwise this tunnel cannot be established.

```
R1(config)#do sh ip pim rp

Group: 224.0.1.40, RP: 10.255.255.4, uptime 00:05:57, expires never

R1(config)#do sh ip pim rp mapping

PIM Group-to-RP Mappings


Group(s): 224.0.0.0/4, Static

    RP: 10.255.255.4 (?)
```

It is possible to configure and RP for some specific groups. It is possible to define multiple RPs for different groups:

```
R1:

no ip pim rp-address 10.255.255.4

!

ip access-list standard GROUP227

 permit 227.0.0.0 0.255.255.255

!

ip pim rp-address 10.255.255.4 GROUP227
```

This time we have specified R4 as the RP for GROUP227 (227.0.0.0/8):

```
R1(config)#do sh ip pim rp mapp

PIM Group-to-RP Mappings


Acl: GROUP227, Static

    RP: 10.255.255.4 (?)
```

Let's test the configuration:

```
SERVER14#ping 227.27.27.27 re 1

Type escape sequence to abort.

Sending 1, 100-byte ICMP Echos to 227.27.27.27, timeout is 2 seconds:


Reply to request 0 from 10.6.25.25, 38 ms

Reply to request 0 from 10.6.24.24, 38 ms

SERVER14#ping 227.27.27.27 re 1

Type escape sequence to abort.

Sending 1, 100-byte ICMP Echos to 227.27.27.27, timeout is 2 seconds:


Reply to request 0 from 10.100.6.20, 10 ms

Reply to request 0 from 10.100.5.22, 22 ms

Reply to request 0 from 10.100.5.23, 22 ms

Reply to request 0 from 10.100.5.17, 18 ms

Reply to request 0 from 10.6.24.24, 18 ms

Reply to request 0 from 10.6.25.25, 18 ms

Reply to request 0 from 10.100.5.22, 17 ms

Reply to request 0 from 10.100.5.23, 15 ms
Reply to request 0 from 10.100.5.17, 15 ms

Reply to request 0 from 10.6.24.24, 15 ms

Reply to request 0 from 10.6.25.25, 15 ms

Reply to request 0 from 10.100.6.21, 14 ms

Reply to request 0 from 10.100.6.20, 14 ms

Reply to request 0 from 10.100.6.16, 14 ms

Reply to request 0 from 10.100.6.21, 10 ms

Reply to request 0 from 10.100.6.16, 10 ms

SERVER14#ping 227.27.27.27 re 1

Type escape sequence to abort.

Sending 1, 100-byte ICMP Echos to 227.27.27.27, timeout is 2 seconds:


Reply to request 0 from 10.100.6.20, 6 ms

Reply to request 0 from 10.100.5.23, 14 ms

Reply to request 0 from 10.100.5.22, 14 ms

Reply to request 0 from 10.100.5.17, 9 ms

Reply to request 0 from 10.6.24.24, 9 ms

Reply to request 0 from 10.6.25.25, 9 ms

Reply to request 0 from 10.100.6.16, 6 ms

Reply to request 0 from 10.100.6.21, 6 ms




R1(config)#do sh ip mroute | begin \(

(*, 227.27.27.27), 00:00:50/stopped, RP 10.255.255.4, flags: SPF

  Incoming interface: Ethernet0/0, RPF nbr 10.100.1.4

  Outgoing interface list: Null


(10.1.14.14, 227.27.27.27), 00:00:50/00:02:48, flags: FT

  Incoming interface: Ethernet0/3, RPF nbr 0.0.0.0

  Outgoing interface list:

    Ethernet0/0, Forward/Sparse, 00:00:50/00:02:44


(*, 224.0.1.40), 00:06:30/00:02:34, RP 0.0.0.0, flags: DPL

  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list: Null
```

There are some duplicate packets at first tries, that is because of RP Shared Tree and Source Tree switchover at the left side of the topology (OSPF domain) and also before Assert procedure between R7 and R8 at the right side of the topology.

Task 03:

- Remove Static RP configuration
- Enable Auto-RP (Configure R3 and R4 as the Rendezvous-Points)

Solution:

Auto-RP is a Cisco proprietary automatic RP discovery and announcement mechanism. Today in most production networks BSR (Bootstrap) which is a standard protocol and included in PIMv2 is being used widely.

We can use both protocols to automatically find the RP in our multicast network.

```
R1:
no ip pim rp-address 10.255.255.4 GROUP227


On all other routers and also SW13 MLS:
no ip pim rp-address 10.255.255.4


R3 and R4:
interface loopback 0
 ip pim sparse-mode
!
ip pim send-rp-announce loopback 0 scope 16 interval 10
ip pim send-rp-discovery loopback 0 scope 16
!
```

Verification:

```
R6(config)#do sh ip pim rp
Group: 227.27.27.27, RP: 10.255.255.4, uptime 00:02:08, expires 00:02:46
R6(config)#do sh ip pim rp mapping
PIM Group-to-RP Mappings


Group(s) 224.0.0.0/4
  RP 10.255.255.4 (?), v2v1
    Info source: 10.255.255.4 (?), elected via Auto-RP
         Uptime: 00:02:10, expires: 00:02:44
```

```
R6(config)#do sh ip mroute | begin \(
(*, 227.27.27.27), 00:47:40/stopped, RP 10.255.255.4, flags: SJC
  Incoming interface: Serial2/1, RPF nbr 172.16.46.4
  Outgoing interface list:
    Ethernet1/0, Forward/Sparse, 00:47:40/00:02:26
    Ethernet1/1, Forward/Sparse, 00:47:40/00:02:24

(10.1.14.14, 227.27.27.27), 00:19:13/00:01:37, flags: T
  Incoming interface: Serial2/1, RPF nbr 172.16.46.4
  Outgoing interface list:
    Ethernet1/1, Forward/Sparse, 00:19:13/00:02:24
    Ethernet1/0, Forward/Sparse, 00:19:13/00:02:26

(*, 224.0.1.40), 00:05:47/stopped, RP 0.0.0.0, flags: DPL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list: Null

(10.255.255.4, 224.0.1.40), 00:01:43/00:02:15, flags: PLT
  Incoming interface: Serial2/1, RPF nbr 172.16.46.4
  Outgoing interface list: Null
```

Take a look at (*, 224.0.1.40), the RP has been found using this Shared tree entry. All cisco routers understands this group number, this multicast group address (224.0.1.40) is a reserved multicast group for Cisco Auto-RP discovery and all cisco routers are members of this group by default so they listen to this group messages.

```
R3(config)#do sh ip pim rp mapping
PIM Group-to-RP Mappings
This system is an RP (Auto-RP)
This system is an RP-mapping agent (Loopback0)

Group(s) 224.0.0.0/4
  RP 10.255.255.4 (?), v2v1
    Info source: 10.255.255.4 (?), elected via Auto-RP
         Uptime: 00:00:56, expires: 00:00:24
  RP 10.255.255.3 (?), v2v1
    Info source: 10.255.255.3 (?), via Auto-RP
         Uptime: 00:08:17, expires: 00:00:23



R4(config)#do sh ip pim rp mapping
PIM Group-to-RP Mappings
This system is an RP (Auto-RP)
This system is an RP-mapping agent (Loopback0)

Group(s) 224.0.0.0/4
  RP 10.255.255.4 (?), v2v1
    Info source: 10.255.255.4 (?), elected via Auto-RP
         Uptime: 00:08:08, expires: 00:00:22
  RP 10.255.255.3 (?), v2v1
    Info source: 10.255.255.3 (?), via Auto-RP
         Uptime: 00:02:29, expires: 00:00:21
```

Both R3 and R4 routers are RP and also RP-mapping agents. They are redundant, but one of them has been elected as RP and RP-Mapping agent (R4: 10.255.255.4).

Task 04:

- Remove Auto-RP configuration
- Configure R3 and R4 as being the RPs and also Bootstrap candidates
- R3 must be the preferred RP

Solution:

BSR (Bootstrap protocol) is a standard based protocol that is available in PIMv2 natively.

All of the industry routers that are supporting PIMv2 can work with BSR to find RPs automatically.

```
R3 and R4:

no ip pim send-rp-announce loopback 0 scope 16 interval 10

no ip pim send-rp-discovery loopback 0 scope 16

!


R3:

ip pim bsr-candidate loopback 0

ip pim rp-candidate loopback 0 priority 50


R4:

ip pim bsr-candidate loopback 0

ip pim rp-candidate loopback 0 priority 60
```

Verification:

The router with the lowest priority becomes the preferred RP:

```
R6(config)#do sh ip pim rp

Group: 227.27.27.27, RP: 10.255.255.3, uptime 00:09:21, expires 00:01:52

R6(config)#do sh ip pim rp mapping

PIM Group-to-RP Mappings


Group(s) 224.0.0.0/4

  RP 10.255.255.3 (?), v2

    Info source: 10.255.255.4 (?), via bootstrap, priority 50, holdtime 150

         Uptime: 00:09:24, expires: 00:01:49
  RP 10.255.255.4 (?), v2

    Info source: 10.255.255.4 (?), via bootstrap, priority 60, holdtime 150

         Uptime: 00:07:12, expires: 00:01:48


R6(config)#do sh ip mroute | begin \(

(*, 227.27.27.27), 01:19:00/stopped, RP 10.255.255.3, flags: SJC

  Incoming interface: Serial2/1, RPF nbr 172.16.46.4

  Outgoing interface list:

    Ethernet0/1, Forward/Sparse, 00:08:43/00:02:40

    Ethernet1/0, Forward/Sparse, 01:19:00/00:02:59

    Ethernet1/1, Forward/Sparse, 01:19:00/00:01:59


(10.1.14.14, 227.27.27.27), 00:00:15/00:02:44, flags: T

  Incoming interface: Serial2/1, RPF nbr 172.16.46.4

  Outgoing interface list:

    Ethernet1/1, Forward/Sparse, 00:00:15/00:02:44

    Ethernet1/0, Forward/Sparse, 00:00:15/00:02:59

    Ethernet0/1, Forward/Sparse, 00:00:15/00:03:14


(*, 224.0.1.40), 00:37:07/00:02:00, RP 0.0.0.0, flags: DPL

  Incoming interface: Null, RPF nbr 0.0.0.0

  Outgoing interface list: Null
```

Task 05:

- R4 must be the RP for group 228.28.28.28

Solution:

RPs can be configured for specific groups too:

```
R3:
ip access-list standard GROUP-228
 permit 228.28.28.28 0.0.0.0
!
ip pim rp-candidate loopback 0 group-list GROUP-228 priority 30
!
```

Verification:

```
HOST22(config)#int e0/0
HOST22(config-if)#ip igmp join-group 228.28.28.28



SERVER14#ping 228.28.28.28
Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 228.28.28.28, timeout is 2 seconds:

Reply to request 0 from 10.100.5.22, 15 ms




R6(config)#do sh ip pim rp
Group: 228.28.28.28, RP: 10.255.255.3, uptime 00:03:05, expires 00:01:29
Group: 227.27.27.27, RP: 10.255.255.4, uptime 00:20:40, expires 00:01:29
```

Task 06:

- Enable PIM SSM (Source Specific Multicast)
- HOST22, HOST23 and HOST17 should only accept 227/8 group traffic sourced from SERVER14
- HOST20, HOST21 and HOST16 should only accept 227/8 group traffic sourced from SERVER15
- HOST16 and HOST17 should accept 227/8 group traffic sourced from both SERVER14 and SERVER15

Solution:

SSM can be enabled (By default, in IOS, IOS-XE it is not enabled to consider 232/8 range as the default SSM range) for default range (232/8) or any selective range using an ACL:

```
On all routers and also SW13:
ip access-list standard GROUP-227-8
 permit 227.0.0.0 0.255.255.255
!
ip pim ssm range GROUP-227-8
!


R9:
interface e0/0
 ip igmp version 3
!
R10:
interface e0/1
 ip igmp version 3
!
R7:
interface e0/0
 ip igmp version 3
!
R8:
interface e0/1
```

```
 ip igmp version 3
!

HOST23:
interface e0/0
 no ip igmp join-group 227.27.27.27
 ip igmp join-group 227.27.27.27 source 10.1.14.14
 ip igmp version 3
!
HOST22:
interface e0/0
 no ip igmp join-group 227.27.27.27
 ip igmp join-group 227.27.27.27 source 10.1.14.14
 ip igmp version 3
!
HOST20:
interface e0/0
 no ip igmp join-group 227.27.27.27
 ip igmp join-group 227.27.27.27 source 10.2.15.15
 ip igmp version 3
!
HOST21:
interface e0/0
 no ip igmp join-group 227.27.27.27
 ip igmp join-group 227.27.27.27 source 10.2.15.15
 ip igmp version 3
!
HOST17:
interface e0/2
 no ip igmp join-group 227.27.27.27
 ip igmp join-group 227.27.27.27 source 10.1.14.14
 ip igmp version 3
!
HOST16:
```

```
interface e0/2
 no ip igmp join-group 227.27.27.27
 ip igmp join-group 227.27.27.27 source 10.2.15.15
 ip igmp version 3
!
```

Note: PIM SSM is only works with IGMP v3 (in Layer 2).

Verification:

```
SERVER14#ping 227.27.27.27 re 1
Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 227.27.27.27, timeout is 2 seconds:

Reply to request 0 from 10.100.5.17, 14 ms
Reply to request 0 from 10.100.5.22, 23 ms
Reply to request 0 from 10.100.5.23, 14 ms




SERVER15#ping 227.27.27.27 re 1
Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 227.27.27.27, timeout is 2 seconds:

Reply to request 0 from 10.100.6.16, 4 ms
Reply to request 0 from 10.100.6.20, 4 ms
Reply to request 0 from 10.100.6.21, 4 ms
```

Why HOST24 and HOST25 do not answer to ping to the 227.27.27.27?

Because of IGMP v3 again! Let's enable IGMP version 3:

```
HOST24:
interface e0/0
 no ip igmp join-group 227.27.27.27
 ip igmp join-group 227.27.27.27 source 10.1.14.14
 ip igmp join-group 227.27.27.27 source 10.2.15.15
 ip igmp version 3
!
HOST25:
interface e0/0
 no ip igmp join-group 227.27.27.27
 ip igmp join-group 227.27.27.27 source 10.1.14.14
 ip igmp join-group 227.27.27.27 source 10.2.15.15
 ip igmp version 3
!
R6:
interface range e1/0-1
 ip igmp version 3
!
```

This time both HOST24 and HOST25 are requesting group traffic from both sources:

```
SERVER14#ping 227.27.27.27 re 1
Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 227.27.27.27, timeout is 2 seconds:

Reply to request 0 from 10.6.25.25, 11 ms
Reply to request 0 from 10.100.5.22, 15 ms
Reply to request 0 from 10.100.5.23, 15 ms
Reply to request 0 from 10.100.5.17, 11 ms
Reply to request 0 from 10.6.24.24, 11 ms




SERVER15#ping 227.27.27.27 re 1
Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 227.27.27.27, timeout is 2 seconds:

Reply to request 0 from 10.100.6.20, 5 ms
Reply to request 0 from 10.6.24.24, 11 ms
Reply to request 0 from 10.6.25.25, 11 ms
Reply to request 0 from 10.100.6.21, 5 ms
Reply to request 0 from 10.100.6.16, 5 ms
```

Task 07:

- Configure Bidirectional PIM for 229/8 group with static RP of 10.255.255.5 (it should override dynamically learned RPs)

Solution:

Bidirectional PIM (PIM-Bidir) is a solution for Many-to-Many multicast usages such as financial applications that rely on a many-to-many applications model. Bidir PIM enables these applications by allowing them to easily scale to a very large number of groups and sources by eliminating the maintenance of source states.

```
R5:
interface loopback 0
 ip pim sparse-mode
!

On all routers and also SW13 MLS:
no ip pim ssm range GROUP-227-8
!
ip access-list standard GROUP-229
 permit 229.0.0.0 0.255.255.255
!
ip pim bidir-enable
!
ip pim rp-address 10.255.255.5 GROUP-229 override bidir
```

Verification:

```
SW13(config)#do sh ip mroute 229.0.0.0/8
(*,229.0.0.0/8), 00:00:44/-, RP 10.255.255.5, flags: B
  Bidir-Upstream: Vlan3, RPF nbr: 10.100.3.5
  Incoming interface list:
    Vlan4, Accepting/Sparse
    Loopback0, Accepting/Dense
    Vlan3, Accepting/Sparse

R5(config)#do sh ip mroute 229.0.0.0/8
```

```
(*,229.0.0.0/8), 00:02:32/-, RP 10.255.255.5, flags: B
  Bidir-Upstream: Loopback0, RPF nbr: 10.255.255.5
  Incoming interface list:
    Serial2/0, Accepting/Sparse
    Ethernet0/0, Accepting/Sparse
    Loopback0, Accepting/Sparse



R6(config)#do sh ip pim rp mapp
PIM Group-to-RP Mappings

Group(s) 224.0.0.0/4
  RP 10.255.255.4 (?), v2
    Info source: 10.255.255.4 (?), via bootstrap, priority 60, holdtime 150
         Uptime: 00:37:59, expires: 00:01:49
Group(s) 228.28.28.28/32
  RP 10.255.255.3 (?), v2
    Info source: 10.255.255.4 (?), via bootstrap, priority 30, holdtime 150
         Uptime: 00:37:59, expires: 00:01:48
Acl: GROUP-229, Static, Bidir Mode
    RP: 10.255.255.5 (?)

HOST20(config)#int e0/0
HOST20(config-if)#ip igmp join-group 229.29.29.29

HOST21(config)#int e0/0
HOST21(config-if)#ip igmp join-group 229.29.29.29

HOST22(config)#int e0/0
HOST22(config-if)#ip igmp join-group 229.29.29.29

HOST23(config)#int e0/0
HOST23(config-if)#ip igmp join-group 229.29.29.29
```

```
HOST20(config)#do ping 229.29.29.29
Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 229.29.29.29, timeout is 2 seconds:

Reply to request 0 from 10.100.6.21, 2 ms
Reply to request 0 from 10.100.5.22, 15 ms
Reply to request 0 from 10.100.5.23, 15 ms
Reply to request 0 from 10.100.6.20, 2 ms



HOST21(config-if)#do ping 229.29.29.29
Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 229.29.29.29, timeout is 2 seconds:

Reply to request 0 from 10.100.6.20, 2 ms
Reply to request 0 from 10.100.5.23, 18 ms
Reply to request 0 from 10.100.5.22, 17 ms
Reply to request 0 from 10.100.6.21, 2 ms



HOST22(config-if)#do ping 229.29.29.29
Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 229.29.29.29, timeout is 2 seconds:

Reply to request 0 from 10.100.5.23, 1 ms
Reply to request 0 from 10.100.6.20, 16 ms
Reply to request 0 from 10.100.6.21, 14 ms
Reply to request 0 from 10.100.5.22, 1 ms

R5(config)#do sh ip mroute | begin \(
(*,229.0.0.0/8), 00:08:22/-, RP 10.255.255.5, flags: B
```

```
  Bidir-Upstream: Loopback0, RPF nbr: 10.255.255.5
  Incoming interface list:
    Serial2/0, Accepting/Sparse
    Ethernet0/0, Accepting/Sparse
    Loopback0, Accepting/Sparse

(*, 229.29.29.29), 00:02:58/00:03:27, RP 10.255.255.5, flags: B
  Bidir-Upstream: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Ethernet0/0, Forward/Sparse, 00:02:35/00:02:52
    Serial2/0, Forward/Sparse, 00:02:58/00:03:27

(*, 224.0.1.40), 00:45:30/00:02:32, RP 0.0.0.0, flags: DPL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list: Null
```

There are no specific sourced entries created for 229.29.29.29, only shared trees (*, Gs) are created.

And also the HOSTs are senders and receivers at the same time.

Task 08:

- Remove any RP configuration on all routers and also SW13 MLS
- Provide RP redundancy for PIM Bidirectional (Phantom RP)
- Configure PIM Sparse-Dense mode

Solution:

The method used to provide RP redundancy for PIM-Bidir is called Phantom RP, this RP address is actually a virtual RP. The IP address that is going to be advertised as the RP address is not defined on any router.

This RP address is going to be defined with different subnet masks on two of our routers with different IP addresses than the actual RP address!

For example our RP address is going to be: 10.254.254.1 but we define:

On R3: 10.254.254.3 255.255.255.248

On R4: 10.254.254.4 255.255.255.240

All of the multicast routers only does the RPF to the RP and they do not send any unicast packets to the RP. So we can use any IP address in those subnets.

The longest match route will be used for RPF checks (R3), if R3 fails the other longest match comes into play (R4).

```
R3 and R4:
no ip pim bsr-candidate Loopback0 0
no ip pim rp-candidate Loopback0



On all routers and also SW13 MLS:
no ip pim rp-address 10.255.255.5 GROUP-229 override bidir
ip pim rp-address 10.254.254.1 GROUP-229 bidir



R3:
interface Loopback254
```

```
 ip address 10.254.254.3 255.255.255.248
 ip pim sparse-mode
 ip ospf network point-to-point
 ip ospf 1 area 0
!

R4:
interface Loopback254
 ip address 10.254.254.4 255.255.255.240
 ip pim sparse-mode
 ip ospf network point-to-point
 ip ospf 1 area 0
!

R1:
interface Ethernet0/0
 ip pim sparse-dense-mode
!
interface Ethernet0/3
 ip pim sparse-dense-mode
!
R2:
interface Ethernet0/1
 ip pim sparse-dense-mode
!
interface Ethernet0/3
 ip pim sparse-dense-mode
!
R3:
interface Ethernet0/0
 ip pim sparse-dense-mode
!
interface Ethernet0/2
 ip pim sparse-dense-mode
!
```

```
R4:
interface Ethernet0/1
 ip pim sparse-dense-mode
!
interface Ethernet0/3
 ip pim sparse-dense-mode
!
interface Serial2/0
 ip pim sparse-dense-mode
!
interface Serial2/1
 ip pim sparse-dense-mode
!
R9:
interface Ethernet0/0
 ip pim sparse-dense-mode
!
interface Ethernet0/2
 ip pim sparse-dense-mode
!


R10:
interface Ethernet0/1
 ip pim sparse-dense-mode
!
interface Ethernet0/3
 ip pim sparse-dense-mode
!
R5:
interface Ethernet0/0
 ip pim sparse-dense-mode
!
interface Serial2/0
 ip pim sparse-dense-mode
```

```
!
R6:
interface Serial2/1
 ip pim sparse-dense-mode
!
interface Ethernet0/1
 ip pim sparse-dense-mode
!
interface Ethernet1/0
 ip pim sparse-dense-mode
!
interface Ethernet1/1
 ip pim sparse-dense-mode
!
SW13:
interface Vlan3
 ip pim sparse-dense-mode
!
interface Vlan4
 ip pim sparse-dense-mode
!

R7:
interface Ethernet0/0
 ip pim sparse-dense-mode
!
interface Ethernet0/2
 ip pim sparse-dense-mode
!
R8:
interface Ethernet0/1
 ip pim sparse-dense-mode
!
interface Ethernet0/3
```

```
 ip pim sparse-dense-mode
!
```

As you realized there is no interface that RP address set to. It is only a virtual RP-address

That RP is called Phantom RP, all routers are going to do RPF check towards the subnet that RP address resides:

```
HOST20(config)#do ping 229.29.29.29
Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 229.29.29.29, timeout is 2 seconds:

Reply to request 0 from 10.100.6.21, 2 ms
Reply to request 0 from 10.100.5.23, 16 ms
Reply to request 0 from 10.100.5.22, 16 ms
Reply to request 0 from 10.100.6.20, 2 ms



(*,229.0.0.0/8), 00:04:57/-, RP 10.254.254.1, flags: B
  Bidir-Upstream: Ethernet0/3, RPF nbr: 10.100.2.3
  Incoming interface list:
    Ethernet0/1, Accepting/Sparse-Dense
    Ethernet0/3, Accepting/Sparse-Dense

(*, 229.29.29.29), 00:09:59/00:02:59, RP 10.254.254.1, flags: BC
  Bidir-Upstream: Ethernet0/3, RPF nbr 10.100.2.3
  Outgoing interface list:
    Ethernet0/1, Forward/Sparse-Dense, 00:04:57/00:02:59
    Ethernet0/3, Bidir-Upstream/Sparse-Dense, 00:04:57/stopped

(*, 227.27.27.27), 00:10:00/00:02:59, RP 0.0.0.0, flags: DC
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Ethernet0/3, Forward/Sparse-Dense, 00:10:00/stopped
    Ethernet0/1, Forward/Sparse-Dense, 00:10:00/stopped
```

```
(*, 224.0.1.40), 00:15:24/00:02:58, RP 0.0.0.0, flags: DCL
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Ethernet0/1, Forward/Sparse-Dense, 00:15:24/stopped



SERVER14#ping 227.27.27.27
Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 227.27.27.27, timeout is 2 seconds:

Reply to request 0 from 10.100.6.21, 5 ms
Reply to request 0 from 10.100.5.23, 13 ms
Reply to request 0 from 10.100.5.17, 13 ms
Reply to request 0 from 10.100.5.22, 8 ms
Reply to request 0 from 10.6.24.24, 8 ms
Reply to request 0 from 10.6.25.25, 8 ms
Reply to request 0 from 10.100.6.20, 6 ms
Reply to request 0 from 10.100.6.16, 6 ms



R10(config)#do sh ip mroute 229.0.0.0/8
(*,229.0.0.0/8), 00:07:17/-, RP 10.254.254.1, flags: B
  Bidir-Upstream: Ethernet0/3, RPF nbr: 10.100.2.3
  Incoming interface list:
    Ethernet0/1, Accepting/Sparse-Dense
    Ethernet0/3, Accepting/Sparse-Dense

R10(config)#do sh ip mroute 227.27.27.27 | begin \(
(*, 227.27.27.27), 00:12:25/stopped, RP 0.0.0.0, flags: DC
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Ethernet0/3, Forward/Sparse-Dense, 00:12:25/stopped
    Ethernet0/1, Forward/Sparse-Dense, 00:12:25/stopped
```

```
(10.1.14.14, 227.27.27.27), 00:01:02/00:01:57, flags: T
  Incoming interface: Ethernet0/3, RPF nbr 10.100.2.4
  Outgoing interface list:
    Ethernet0/1, Forward/Sparse-Dense, 00:01:02/stopped, A
```

This time we have used Sparse-Dense mode, for 229/8 the RPF check towards the RP is performed, but for 227.27.27.27 group there is no RP, it is handled by dense mode.

Task 09:

- Remove any RP configuration on all routers and also SW13 MLS
- Remove PIM-Bidir configuration
- Implement Anycast RP (RP address: 10.250.250.1 on R4 and R5)

Solution:

In Anycast RP two ore more RPs are configured with the same IP addresses on loopback interfaces, this address must be configured with a /32 mask (a host address). This address must be advertised into IGPs, so the routers are going to interact with the closest RP. So by having RP redundancy with this method we can achieve load-sharing between them too (some sources register to one RP and the other ones register to another RP and so on…). But the problem is: a source may register with one RP and receivers may join to a different RP, there should be a method for RPs to exchange information about active sources. The MSDP is going to do this job.

Int the Anycast RP solution, all the RPs must be configured to be MSDP peers of each other. When a sender (source) registers with one RP, a message will be sent to the other RPs informing them of that source being active for a specific multicast group. If one the RPs fails, one of the other RPs would become the active RP (because they are using the same anycast address).

Anycast RP ensures that new sessions with source and receivers can begin at any time even if one RP is failed.

```
On all routers and also SW13 MLS:

no ip pim rp-address 10.254.254.1 GROUP-229 bidir

no ip pim bidir-enable


R4:

interface Loopback250

 ip address 10.250.250.1 255.255.255.255

 ip pim sparse-mode

 ip ospf 1 area 0

!

ip msdp peer 10.255.255.5 connect-source loopback 0

ip msdp originator-id loopback 0

ip pim bsr-candidate loopback 250
ip pim rp-candidate loopback 250


R5:

interface Loopback250

 ip address 10.250.250.1 255.255.255.255

 ip pim sparse-mode

 ip ospf 1 area 0

!

ip msdp peer 10.255.255.4 connect-source loopback 0

ip msdp originator-id loopback 0

ip pim bsr-candidate loopback 250

ip pim rp-candidate loopback 250
```

Verification:

```
R4(config)#

*Oct 21 21:19:32.820: %LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel0, changed state to up

*Oct 21 21:19:32.821: %LINEPROTO-5-UPDOWN: Line protocol on Interface Tunnel1, changed state to up

R4(config)#

*Oct 21 21:19:34.384: %MSDP-5-PEER_UPDOWN: Session to peer 10.255.255.5 going up



R4(config)#do sh ip msdp count

SA State per Peer Counters, <Peer>: <# SA learned>

    10.255.255.5: 0


SA State per ASN Counters, <asn>: <# sources>/<# groups>

    Total entries: 0



R5(config)#do sh ip msdp count

SA State per Peer Counters, <Peer>: <# SA learned>

    10.255.255.4: 0


SA State per ASN Counters, <asn>: <# sources>/<# groups>

    Total entries: 0
```

```
SERVER14#ping 227.27.27.27 re 1
Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 227.27.27.27, timeout is 2 seconds:

Reply to request 0 from 10.100.6.21, 5 ms
Reply to request 0 from 10.100.5.23, 12 ms
Reply to request 0 from 10.100.5.22, 12 ms
Reply to request 0 from 10.100.5.17, 8 ms
Reply to request 0 from 10.6.24.24, 8 ms
Reply to request 0 from 10.6.25.25, 8 ms
Reply to request 0 from 10.100.6.16, 5 ms
Reply to request 0 from 10.100.6.20, 5 ms



R5(config)#do sh ip msdp count
SA State per Peer Counters, <Peer>: <# SA learned>
    10.255.255.4: 1

SA State per ASN Counters, <asn>: <# sources>/<# groups>
    Total entries: 1
    65001: 1/1

R5(config)#do sh ip msdp sa-cache
MSDP Source-Active Cache - 1 entries
(10.1.14.14, 227.27.27.27), RP 10.255.255.4, BGP/AS 65001, 00:00:51/00:05:23, Peer 10.255.255.4
```

The moment SERVER14 tried to stream some data (in this example pings), it registered with the nearest RP (R4), that registered source has been sent to the other RP (R5) as an MSDP SA message. So R5 is in sync with R4:

```
R5(config)#do sh ip mroute 227.27.27.27 | begin \(
(*, 227.27.27.27), 00:08:46/stopped, RP 10.250.250.1, flags: S
  Incoming interface: Null, RPF nbr 0.0.0.0
  Outgoing interface list:
    Ethernet0/0, Forward/Sparse-Dense, 00:08:46/00:02:33

(10.1.14.14, 227.27.27.27), 00:00:19/00:02:40, flags: PMT   !!MSDP created entry!!
  Incoming interface: Serial2/0, RPF nbr 172.16.45.4
  Outgoing interface list: Null
```

Task 10:

- Enable IGMP Snooping for the 10.100.6.0/24 segment
- Only HOST20 and HOST21 must join 227.27.27.27 group
- HOST16 should not receive any multicast traffic destined to that group

Solution:

By default an L2 switch or an MLS which has a connection to the L2 domain treats with all the multicast traffic just like Unknown Unicast and Broadcast ones. It means BUM traffic will be forwarded everywhere. IGMP Snooping is a protocol that helps switch to listen to the IGMP traffic between the router and hosts. In this task HOST20 and 21 want to receive the 227.27.27.27 group traffic but HOST16 does not want that:

```
SW18:
ip igmp snooping
ip igmp snooping vlan 1
```

IGMP Snooping is on by default on the catalyst switches, we just wanted to show you the commands. But it only supports IGMP v2 with our virtual devices in the lab.

So let’s change the IGMP version on the routers and also hosts to version 2:

```
R9:
interface Ethernet0/0
 ip igmp version 2
!
R10:
interface Ethernet0/1
 ip igmp version 2
!
HOST20:
interface Ethernet0/0
 no  ip igmp join-group 227.27.27.27 source 10.2.15.15
 ip igmp version 2
 ip igmp join-group 227.27.27.27
!
HOST21:
```

```
interface Ethernet0/0
 no  ip igmp join-group 227.27.27.27 source 10.2.15.15
 ip igmp version 2
 ip igmp join-group 227.27.27.27
!
HOST16:
interface Ethernet0/2
 ip igmp join-group 227.27.27.27 source 10.2.15.15
 ip igmp version 2
!
```

Verification:

```
SW18(config)#do sh ip igmp snooping groups
Vlan      Group                    Version     Port List
---------------------------------------------------------
1         224.0.1.40               v2          Et0/0
1         227.27.27.27             v2          Et0/2, Et0/3
1         229.29.29.29             v2          Et0/3
SW18(config)#do sh ip igmp snooping mrouter
Vlan    ports
----    -----
   1    Et0/0(dynamic)
```

For our virtual device in this lab everything seems to be fine in the control plane, but not working well in the data plane, for only demonstration we just wanted to show you how it works but in action it will still forward the BUM traffic everywhere!:

```
HOST16(config)#access-list 100 permit ip any 227.27.27.27 0.0.0.0
HOST16(config)#do debug ip packet 100
IP packet debugging is on for access list 100



SERVER14#ping 227.27.27.27 re 1
Type escape sequence to abort.
Sending 1, 100-byte ICMP Echos to 227.27.27.27, timeout is 2 seconds:

Reply to request 0 from 10.100.6.16, 19 ms
Reply to request 0 from 10.100.5.23, 44 ms

Reply to request 0 from 10.100.5.22, 44 ms

Reply to request 0 from 10.100.5.17, 39 ms

Reply to request 0 from 10.6.25.25, 39 ms

Reply to request 0 from 10.6.24.24, 38 ms

Reply to request 0 from 10.100.5.23, 38 ms

Reply to request 0 from 10.100.5.22, 38 ms

Reply to request 0 from 10.100.5.17, 36 ms

Reply to request 0 from 10.6.24.24, 36 ms

Reply to request 0 from 10.6.25.25, 36 ms

Reply to request 0 from 10.100.6.21, 19 ms

Reply to request 0 from 10.100.6.20, 19 ms





HOST16(config-if)#

*Oct 21 22:36:50.917: IP: s=10.1.14.14 (Ethernet0/2), d=227.27.27.27, len 100, input feature, MCI Check(109),
rtype 0, forus FALSE, sendself FALSE, mtu 0, fwdchk FALSE

*Oct 21 22:36:50.917: IP: s=10.1.14.14 (Ethernet0/2), d=227.27.27.27, len 100, rcvd 2

*Oct 21 22:36:50.917: IP: s=10.1.14.14 (Ethernet0/2), d=227.27.27.27, len 100, stop process pak for forus
packet

HOST16(config-if)#

*Oct 21 22:37:17.695: IP: s=10.100.6.16 (local), d=227.27.27.27 (Ethernet0/2), len 32, sending broad/multicast

*Oct 21 22:37:17.695: IP: s=10.100.6.16 (local), d=227.27.27.27 (Ethernet0/2), len 32, sending full packet

HOST16(config-if)#

*Oct 21 22:38:17.627: IP: s=10.100.6.21 (Ethernet0/2), d=227.27.27.27, len 32, rcvd 0

*Oct 21 22:38:17.627: IP: s=10.100.6.21 (Ethernet0/2), d=227.27.27.27, len 32, input feature, packet consumed,
MCI Check(109), rtype 0, forus FALSE, sendself FALSE, mtu 0, fwdchk FALSE

HOST16(config-if)#exit

HOST16(config)#access-list 100 permit ip any 227.27.27.27 0.0.0.0

HOST16(config)#do debug ip packet 100

IP packet debugging is on for access list 100

HOST16(config)#

*Oct 21 22:39:09.268: IP: s=10.1.14.14 (Ethernet0/2), d=227.27.27.27, len 100, input feature, MCI Check(109),
rtype 0, forus FALSE, sendself FALSE, mtu 0, fwdchk FALSE
```

```
*Oct 21 22:39:09.268: IP: s=10.1.14.14 (Ethernet0/2), d=227.27.27.27, len 100, rcvd 2

*Oct 21 22:39:09.268: IP: s=10.1.14.14 (Ethernet0/2), d=227.27.27.27, len 100, stop process pak for forus
packet

HOST16(config)#

*Oct 21 22:39:21.728: IP: s=10.100.6.20 (Ethernet0/2), d=227.27.27.27, len 32, rcvd 0

*Oct 21 22:39:21.728: IP: s=10.100.6.20 (Ethernet0/2), d=227.27.27.27, len 32, input feature, packet consumed,
MCI Check(109), rtype 0, forus FALSE, sendself FALSE, mtu 0, fwdchk FALSE
```

Task 11:

- Configure IPv6 multicast routing
- Only use IPv6 enable interface command for inter-router interfaces (do not set any static IPv6 addresses)
- Router to Host or Server addressing scheme: 2001:IPv4SUBNET::R#/64
- Loopback 0 IPv6 scheme: 2001::R#/128
- Configure IPv6 PIM BSR with the RP of R3 loopback 0 interface
- HOSTs 20, 21 and 16 must join the group FF07::1 using IPv6 MLD

Solution:

IPv6 PIM configuration with BSR is very straight forward. Just like IPv4 PIM, we enable Multicast Routing for IPv6, you don't need to enable Sparse-mode PIM for each of the interfaces, IPv6 PIM is going to be enabled for all of the IPv6 enabled interfaces.

The RP configuration is the same as IPv4, they have only a little difference in the command syntax.

Instead of IGMP we have MLD (Multicast Listener Discovery) in IPv6, it is a component of the Internet Protocol Version 6 suite, it is used by IPv6 routers for discovering multicast listeners on a directly attached link. This protocol is embeded in ICMPv6 instead of using a separate protocol like IGMP.

```
R1:
ipv6 unicast-routing
ipv6 multicast-routing
interface range e0/0, e0/3
 ipv6 enable
!
interface e0/3
 ipv6 address 2001:10:1:14::1/64
interface loopback 0
 ipv6 address 2001::1/128
!
R2:
ipv6 unicast-routing
```

```
ipv6 multicast-routing
interface range e0/1, e0/3
 ipv6 enable
!
interface e0/3
 ipv6 address 2001:10:2:15::2/64
interface loopback 0
 ipv6 address 2001::2/128
!
```

R3:

```
ipv6 unicast-routing
ipv6 multicast-routing
interface range e0/2, e0/0
 ipv6 enable
!
interface loopback 0
 ipv6 address 2001::3/128
!
```

R4:

```
ipv6 unicast-routing
ipv6 multicast-routing
interface range e0/1, e0/3
 ipv6 enable
!
interface loopback 0
 ipv6 address 2001::4/128
!
```

R9:

```
ipv6 unicast-routing
ipv6 multicast-routing
interface range e0/2, e0/0
 ipv6 enable
!
interface loopback 0
```

```
 ipv6 address 2001::9/128
!
interface e0/0
 ipv6 address 2001:10:100:6::9/64
 standby version 2
 standby 6 ipv6 2001:10:100:6::254/64
!
```

R10:

```
ipv6 unicast-routing
ipv6 multicast-routing
interface range e0/1, e0/3
 ipv6 enable
!
interface loopback 0
 ipv6 address 2001::10/128
!
interface e0/1
 ipv6 address 2001:10:100:6::10/64
 standby version 2
 standby 6 ipv6 2001:10:100:6::254
!
```

HOST20:

```
ipv6 unicast-routing
interface e0/0
 ipv6 address 2001:10:100:6::20/64
 ipv6 mld join-group FF07::1
!
ipv6 route ::/0 ether 0/0 2001:10:100:6::254
```

HOST21:

```
ipv6 unicast-routing
interface e0/0
 ipv6 address 2001:10:100:6::21/64
 ipv6 mld join-group FF07::1
!
ipv6 route ::/0 ether 0/0 2001:10:100:6::254
```

```
HOST16:
ipv6 unicast-routing
interface e0/2
 ipv6 address 2001:10:100:6::16/64
 ipv6 mld join-group FF07::1
!
ipv6 route ::/0 ether 0/2 2001:10:100:6::254
SERVER14:
ipv6 unicast-routing
interface e0/3
 ipv6 address 2001:10:1:14::14/64
!
ipv6 route ::/0 ether 0/3 2001:10:1:14::1
SERVER15:
ipv6 unicast-routing
interface e0/3
 ipv6 address 2001:10:2:15::15/64
!
ipv6 route ::/0 ether 0/3 2001:10:2:15::2
R1, R2, R3, R4, R9, R10:
router eigrp IPv6-EIGRP
 !
 address-family ipv6 unicast autonomous-system 1
  !
  topology base
  exit-af-topology
 exit-address-family
!


R3:
ipv6 pim bsr candidate bsr 2001::3
ipv6 pim bsr candidate rp 2001::3
!
```

Verification:

```
SERVER14#ping FF07::1
Output Interface: ethernet0/3
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to FF07::1, timeout is 2 seconds:
Packet sent with a source address of 2001:10:1:14::14

Reply to request 0 received from 2001:10:100:6::21, 22 ms
Reply to request 0 received from 2001:10:100:6::16, 28 ms
Reply to request 0 received from 2001:10:100:6::20, 28 ms
Reply to request 1 received from 2001:10:100:6::16, 8 ms
Reply to request 1 received from 2001:10:100:6::21, 8 ms
Reply to request 1 received from 2001:10:100:6::21, 8 ms
Reply to request 1 received from 2001:10:100:6::20, 8 ms
Reply to request 1 received from 2001:10:100:6::16, 13 ms
Reply to request 1 received from 2001:10:100:6::20, 13 ms
Reply to request 2 received from 2001:10:100:6::20, 5 ms
Reply to request 2 received from 2001:10:100:6::21, 5 ms
Reply to request 2 received from 2001:10:100:6::16, 5 ms
Reply to request 3 received from 2001:10:100:6::21, 5 ms
Reply to request 3 received from 2001:10:100:6::16, 5 ms
Reply to request 3 received from 2001:10:100:6::20, 5 ms
Reply to request 4 received from 2001:10:100:6::20, 5 ms
Reply to request 4 received from 2001:10:100:6::21, 5 ms
Reply to request 4 received from 2001:10:100:6::16, 5 ms
Success rate is 100 percent (5/5), round-trip min/avg/max = 5/10/28 ms
18 multicast replies and 0 errors.

SERVER15#ping FF07::1
Output Interface: ethernet0/3
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to FF07::1, timeout is 2 seconds:
```

```
Packet sent with a source address of 2001:10:2:15::15


Reply to request 0 received from 2001:10:100:6::21, 32 ms

Reply to request 1 received from 2001:10:100:6::21, 10 ms

Reply to request 1 received from 2001:10:100:6::16, 10 ms

Reply to request 1 received from 2001:10:100:6::20, 10 ms

Reply to request 1 received from 2001:10:100:6::21, 17 ms

Reply to request 1 received from 2001:10:100:6::20, 17 ms

Reply to request 1 received from 2001:10:100:6::16, 17 ms

Reply to request 2 received from 2001:10:100:6::20, 5 ms

Reply to request 2 received from 2001:10:100:6::21, 5 ms

Reply to request 2 received from 2001:10:100:6::16, 5 ms

Reply to request 3 received from 2001:10:100:6::20, 4 ms

Reply to request 3 received from 2001:10:100:6::21, 4 ms

Reply to request 3 received from 2001:10:100:6::16, 4 ms

Reply to request 4 received from 2001:10:100:6::21, 5 ms

Reply to request 4 received from 2001:10:100:6::20, 5 ms

Reply to request 4 received from 2001:10:100:6::16, 10 ms

Success rate is 100 percent (5/5), round-trip min/avg/max = 4/10/32 ms

16 multicast replies and 0 errors.



R3(config)#do sh ipv6 pim bsr candidate-rp

PIMv2 C-RP information

    Candidate RP: 2001::3 SM

      Priority 192, Holdtime 150

      Advertisement interval 60 seconds

      Next advertisement in 00:00:26


R3(config)#do sh ipv6 pim bsr rp-cache

PIMv2 BSR C-RP Cache


BSR Candidate RP Cache
```

```
Group(s) FF00::/8, RP count 1
  RP 2001::3 SM
    Priority 192, Holdtime 150
    Uptime: 00:20:49, expires: 00:01:44



R3(config)#do sh ipv6 mroute | begin \(
(*, FF07::1), 00:18:02/00:02:36, RP 2001::3, flags: S
  Incoming interface: Tunnel3
  RPF nbr: 2001::3
  Immediate Outgoing interface list:
    Ethernet0/0, Forward, 00:18:02/00:02:36

(2001:10:1:14::14, FF07::1), 00:00:38/00:02:53, RP 2001::3, flags: SPR
  Incoming interface: Tunnel3
  RPF nbr: 2001::3
  Immediate Outgoing interface list:
    Ethernet0/0, Null, 00:00:38/never

(2001:10:1:14::14, FF07::1), 00:00:40/00:02:53, flags: S
  Incoming interface: Ethernet0/2
  RPF nbr: FE80::A8BB:CCFF:FE00:100
  Inherited Outgoing interface list:
    Ethernet0/0, Forward, 00:18:02/00:02:36

(2001:10:2:15::15, FF07::1), 00:00:17/00:03:16, flags: ST
  Incoming interface: Ethernet0/2
  RPF nbr: FE80::A8BB:CCFF:FE00:210
  Immediate Outgoing interface list:
    Ethernet0/0, Forward, 00:00:17/00:03:12
```