

4.10 RIPv2 Filtering with Prefix-Lists

- Configure a prefix-list on R5 so that it does not advertise the two RIP summaries of the BB3 networks to SW2, and so that all other networks are allowed to be advertised.
- Configure a prefix-list on R5 so that it does not install any updates received from R4 on the Frame Relay network. This configuration should not affect the updates received from other neighbors on this segment.
- Ensure that route feedback does not occur for the summaries that R4 is generating to R5.

Configuration

R4:

```
ip route 30.0.0.0 255.252.0.0 Null0
ip route 31.0.0.0 255.252.0.0 Null0
```

R5:

```
router rip
  distribute-list prefix RIP_FILTER_TO_SW2 out FastEthernet0/0
  distribute-list prefix PERMIT_ALL gateway NOT_FROM_R4 in
!
ip prefix-list NOT_FROM_R4 seq 5 deny 155.1.0.4/32
ip prefix-list NOT_FROM_R4 seq 10 permit 0.0.0.0/0 le 32
!
ip prefix-list PERMIT_ALL seq 5 permit 0.0.0.0/0 le 32
!
ip prefix-list RIP_FILTER_TO_SW2 seq 5 deny 30.0.0.0/14
ip prefix-list RIP_FILTER_TO_SW2 seq 10 deny 31.0.0.0/14
ip prefix-list RIP_FILTER_TO_SW2 seq 15 permit 0.0.0.0/0 le 32
```

Verification

Rack1R5#show ip route rip

```
R    222.22.2.0/24 [120/8] via 155.1.0.2, 00:00:05, Serial0/0.1
R    204.12.1.0/24 [120/1] via 155.1.45.4, 00:00:01, Serial0/1
     155.1.0.0/24 is subnetted, 14 subnets
R       155.1.146.0 [120/1] via 155.1.0.1, 00:00:02, Serial0/0.1
R       155.1.10.0 [120/2] via 155.1.58.8, 00:00:02, FastEthernet0/0
R       155.1.8.0 [120/1] via 155.1.58.8, 00:00:02, FastEthernet0/0
R       155.1.9.0 [120/3] via 155.1.0.3, 00:00:08, Serial0/0.1
R       155.1.13.0 [120/1] via 155.1.0.3, 00:00:08, Serial0/0.1
          [120/1] via 155.1.0.1, 00:00:02, Serial0/0.1
R       155.1.7.0 [120/2] via 155.1.0.3, 00:00:08, Serial0/0.1
R       155.1.37.0 [120/1] via 155.1.0.3, 00:00:08, Serial0/0.1
R       155.1.79.0 [120/2] via 155.1.0.3, 00:00:08, Serial0/0.1
R       155.1.67.0 [120/2] via 155.1.0.3, 00:00:08, Serial0/0.1
R       155.1.108.0 [120/1] via 155.1.58.8, 00:00:02, FastEthernet0/0
R    220.20.3.0/24 [120/8] via 155.1.0.2, 00:00:05, Serial0/0.1
     54.0.0.0/24 is subnetted, 1 subnets
R       54.1.1.0 [120/3] via 155.1.0.3, 00:00:08, Serial0/0.1
R    212.18.1.0/24 [120/4] via 155.1.0.3, 00:00:08, Serial0/0.1
R    212.18.0.0/24 [120/4] via 155.1.0.3, 00:00:08, Serial0/0.1
```

```

R    212.18.3.0/24 [120/4] via 155.1.0.3, 00:00:08, Serial0/0.1
R    212.18.2.0/24 [120/4] via 155.1.0.3, 00:00:08, Serial0/0.1
R    192.10.1.0/24 [120/1] via 155.1.0.2, 00:00:05, Serial0/0.1
    31.0.0.0/14 is subnetted, 1 subnets
R      31.0.0.0 [120/2] via 155.1.45.4, 00:00:01, Serial0/1
    150.1.0.0/24 is subnetted, 9 subnets
R      150.1.7.0 [120/2] via 155.1.0.3, 00:00:08, Serial0/0.1
R      150.1.6.0 [120/3] via 155.1.0.3, 00:00:08, Serial0/0.1
R      150.1.4.0 [120/1] via 155.1.45.4, 00:00:01, Serial0/1
R      150.1.3.0 [120/1] via 155.1.0.3, 00:00:08, Serial0/0.1
R      150.1.2.0 [120/1] via 155.1.0.2, 00:00:05, Serial0/0.1
R      150.1.1.0 [120/1] via 155.1.0.1, 00:00:02, Serial0/0.1
R      150.1.9.0 [120/3] via 155.1.0.3, 00:00:08, Serial0/0.1
R      150.1.8.0 [120/1] via 155.1.58.8, 00:00:03, FastEthernet0/0
R    205.90.31.0/24 [120/8] via 155.1.0.2, 00:00:05, Serial0/0.1
    30.0.0.0/14 is subnetted, 1 subnets
R      30.0.0.0 [120/2] via 155.1.45.4, 00:00:01, Serial0/1

```

```
Rack1SW2#show ip route 30.0.0.0
```

```
% Network not in table
```

```
Rack1SW2#show ip route 31.0.0.0
```

```
% Network not in table
```

```
Rack1R2#show ip route 30.0.0.0
```

```
Routing entry for 30.0.0.0/14, 1 known subnets
  Redistributing via rip
```

```
R      30.0.0.0 [120/3] via 155.1.0.5, 00:00:04, Serial0/0.1
```

```
Rack1R2#show ip route 31.0.0.0
```

```
Routing entry for 31.0.0.0/14, 1 known subnets
  Redistributing via rip
```

```
R      31.0.0.0 [120/3] via 155.1.0.5, 00:00:08, Serial0/0.1
```

Note

The first filter on R5, out to SW2, denies the exact to prefixes that are the summaries coming from R4, and permits all other routes. The syntax `0.0.0.0/0 le 32` in a prefix-list means match all routes. The next filter is done based on both the routes learned and who they are learned from. This filter says match any route coming in any interface, per the `PERMIT_ALL` list, and allow them to come in as long as they were not learned from R4, per the `deny 155.1.0.4/32` syntax.

☠ Pitfall

Routing filters introduce the possibility of routing loops due to route feedback, or traffic black holes due to incomplete routing information. In this particular scenario route feedback is introduced on the segments between R4 and R5 with the two summaries R5 learns from R4.

Without our filtering configuration R5 learns the 30.0.0.0/14 and 31.0.0.0/14 summaries from both links to R4. Since split-horizon is enabled on the point-to-point link to R4, this information is not sent back out this link. However since split-horizon was disabled on the Frame Relay link these two routes are sent back out this link. When this update is generated, the next-hop value of the originating router on the subnet is used. In this case the value of the next-hop field is 155.1.0.4 for R4. This can be seen from the routing output on R1, R2, or R3:

```
Rack1R1#show ip route 30.0.0.1
Routing entry for 30.0.0.0/14
  Known via "rip", distance 120, metric 3
  Redistributing via rip
  Last update from 155.1.0.4 on Serial0/0.1, 00:00:00 ago
  Routing Descriptor Blocks:
  * 155.1.0.4, from 155.1.0.5, 00:00:00 ago, via Serial0/0.1
    Route metric is 3, traffic share count is 1
```

When R4 receives this update back in is it discarded, because its own local IP address is the next-hop value in the update. The problem in this scenario comes in when the update R5 receives from R4 in the Frame Relay link is filtered out. In this case, R5 receives two updates, one from 155.1.0.4 and one from 155.1.45.4. The update from 155.1.0.4 is filtered out, per the distribute-list gateway filter, and the route via 155.1.45.4 is installed. This route is then sent out the Frame Relay link to all neighbors on the segment, with R5 set as the next-hop.

When R4 receives this update back in the Frame Relay link it is installed as a valid route in the routing table. This is due to the fact that **RIP does not generate a route to Null0 when generating summaries**. Shortly after being installed R4 realizes that the route is not valid, since it is locally generating the summary, and the route is poisoned. This process results in an infinite loop of the route being advertised and withdrawn. The result of this can be seen on R2 as a periodic traffic drop.

```
Rack1R2#ping 30.0.0.1 repeat 100
```

Type escape sequence to abort.

Sending 100, 100-byte ICMP Echos to 30.0.0.1, timeout is 2 seconds:

```
..!!!!U.....!!!!U.U....!U.
```

Success rate is 39 percent (11/28), round-trip min/avg/max = 100/102/105 ms

Pitfall

The step by step troubleshooting and verification of this problem is as follows. Note that other interfaces in the topology connecting to R4 and R5 have been disabled to simplify debug output, NTP is configured, and timestamping to the millisecond has been enabled to correlate events on both devices simultaneously.

To start, R4 sends the initial update containing 30.0.0.0/14 and 31.0.0.0/14 to R5.

```
Rack1R4#
Jun 26 13:44:01.349: RIP: sending v2 update to 224.0.0.9 via Serial0/1
(155.1.45.4)
Jun 26 13:44:01.349: RIP: build update entries
Jun 26 13:44:01.349:   30.0.0.0/14 via 0.0.0.0, metric 2, tag 0
Jun 26 13:44:01.349:   31.0.0.0/14 via 0.0.0.0, metric 2, tag 0
Jun 26 13:44:01.349:   150.1.4.0/24 via 0.0.0.0, metric 1, tag 0
Jun 26 13:44:01.349:   155.1.0.0/24 via 0.0.0.0, metric 1, tag 0
Jun 26 13:44:01.353:   204.12.1.0/24 via 0.0.0.0, metric 1, tag 0
```

R5 receives this update and installs it in the RIP database and the routing table.

```
Rack1R5#
Jun 26 13:44:01.373: RIP: received v2 update from 155.1.45.4 on Serial0/1
Jun 26 13:44:01.373:   30.0.0.0/14 via 0.0.0.0 in 2 hops
Jun 26 13:44:01.373: RT: SET_LAST_RDB for 30.0.0.0/14
   NEW rdb: via 155.1.45.4

Jun 26 13:44:01.373: RT: add 30.0.0.0/14 via 155.1.45.4, rip metric [120/2]
Jun 26 13:44:01.373: RT: NET-RED 30.0.0.0/14
Jun 26 13:44:01.373: RIP-DB: network_update with 30.0.0.0/14 succeeds
Jun 26 13:44:01.377: RIP-DB: adding 30.0.0.0/14 (metric 2) via 155.1.45.4 on
Serial0/1 to RIP database
Jun 26 13:44:01.377: RIP-DB: add 30.0.0.0/14 (metric 2) via 155.1.45.4 on
Serial0/1
Jun 26 13:44:01.377:   31.0.0.0/14 via 0.0.0.0 in 2 hops
Jun 26 13:44:01.377: RT: SET_LAST_RDB for 31.0.0.0/14
   NEW rdb: via 155.1.45.4

Jun 26 13:44:01.381: RT: add 31.0.0.0/14 via 155.1.45.4, rip metric [120/2]
Jun 26 13:44:01.381: RT: NET-RED 31.0.0.0/14
Jun 26 13:44:01.381: RIP-DB: network_update with 31.0.0.0/14 succeeds
Jun 26 13:44:01.381: RIP-DB: adding 31.0.0.0/14 (metric 2) via 155.1.45.4 on
Serial0/1 to RIP database
Jun 26 13:44:01.381: RIP-DB: add 31.0.0.0/14 (metric 2) via 155.1.45.4 on
Serial0/1
Jun 26 13:44:01.381:   150.1.4.0/24 via 0.0.0.0 in 1 hops
```

R5 takes the update from Serial0/1 and sends it out Serial0/0.1

```
Rack1R5#
Jun 26 13:44:02.026: RIP: sending v2 update to 224.0.0.9 via Serial0/0.1
(155.1.0.5)
Jun 26 13:44:02.026: RIP: build update entries
Jun 26 13:44:02.026:   30.0.0.0/14 via 0.0.0.0, metric 3, tag 0
Jun 26 13:44:02.026:   31.0.0.0/14 via 0.0.0.0, metric 3, tag 0
Jun 26 13:44:02.026:   150.1.4.0/24 via 0.0.0.0, metric 2, tag 0
Jun 26 13:44:02.026:   155.1.0.0/24 via 0.0.0.0, metric 1, tag 0
Jun 26 13:44:02.030:   155.1.45.0/24 via 0.0.0.0, metric 1, tag 0
Jun 26 13:44:02.030:   204.12.1.0/24 via 0.0.0.0, metric 2, tag 0
```

R4 receives the update on Serial0/0.1 from R5. Since there is no Null0 route for the summary R4 mistakenly installs its own origination.

```
Rack1R4#
Jun 26 13:44:02.118: RIP: received v2 update from 155.1.0.5 on Serial0/0.1
Jun 26 13:44:02.118:   30.0.0.0/14 via 0.0.0.0 in 3 hops
Jun 26 13:44:02.118: RT: network 30.0.0.0 is now variably masked
Jun 26 13:44:02.118: RT: SET_LAST_RDB for 30.0.0.0/14
    NEW rdb: via 155.1.0.5

Jun 26 13:44:02.118: RT: add 30.0.0.0/14 via 155.1.0.5, rip metric [120/3]
Jun 26 13:44:02.122: RT: NET-RED 30.0.0.0/14
Jun 26 13:44:02.122: RIP-DB: network_update with 30.0.0.0/14 succeeds
Jun 26 13:44:02.122: RIP-DB: adding 30.0.0.0/14 (metric 3) via 155.1.0.5 on
Serial0/0.1 to RIP database
Jun 26 13:44:02.122: RIP-DB: add 30.0.0.0/14 (metric 3) via 155.1.0.5 on
Serial0/0.1
Jun 26 13:44:02.122:   31.0.0.0/14 via 0.0.0.0 in 3 hops
Jun 26 13:44:02.126: RT: network 31.0.0.0 is now variably masked
Jun 26 13:44:02.126: RT: SET_LAST_RDB for 31.0.0.0/14
    NEW rdb: via 155.1.0.5

Jun 26 13:44:02.126: RT: add 31.0.0.0/14 via 155.1.0.5, rip metric [120/3]
Jun 26 13:44:02.126: RT: NET-RED 31.0.0.0/14
Jun 26 13:44:02.126: RIP-DB: network_update with 31.0.0.0/14 succeeds
Jun 26 13:44:02.130: RIP-DB: adding 31.0.0.0/14 (metric 3) via 155.1.0.5 on
Serial0/0.1 to RIP database
Jun 26 13:44:02.130: RIP-DB: add 31.0.0.0/14 (metric 3) via 155.1.0.5 on
Serial0/0.1
```

Two seconds later R4 realizes that the route is not valid, and poisons it back to R5 with a metric of 16.

```
Jun 26 13:44:04.125: RIP: sending v2 flash update to 224.0.0.9 via Serial0/0.1
(155.1.0.4)
Jun 26 13:44:04.130: RIP: build flash update entries
Jun 26 13:44:04.130:   30.0.0.0/14 via 155.1.0.5, metric 16, tag 0
Jun 26 13:44:04.130:   31.0.0.0/14 via 155.1.0.5, metric 16, tag 0
Jun 26 13:44:04.130: RIP: sending v2 flash update to 224.0.0.9 via Serial0/1
(155.1.45.4)
Jun 26 13:44:04.130: RIP: build flash update entries
Jun 26 13:44:04.130:   30.0.0.0/14 via 0.0.0.0, metric 16, tag 0
Jun 26 13:44:04.134:   31.0.0.0/14 via 0.0.0.0, metric 16, tag 0
```

R5 receives the invalid route from R4, withdraws the route from the routing table and database, and poisons the route back to R4.

```
Jun 26 13:44:04.149: RIP: received v2 update from 155.1.45.4 on Serial0/1
Jun 26 13:44:04.149:      30.0.0.0/14 via 0.0.0.0 in 16 hops (inaccessible)
Jun 26 13:44:04.149: RT: del 30.0.0.0/14 via 155.1.45.4, rip metric [120/2]
Jun 26 13:44:04.149: RT: delete subnet route to 30.0.0.0/14
Jun 26 13:44:04.149: RT: NET-RED 30.0.0.0/14
Jun 26 13:44:04.153: RT: delete network route to 30.0.0.0
Jun 26 13:44:04.153: RT: NET-RED 30.0.0.0/8
Jun 26 13:44:04.153: RIP-DB: Remove 30.0.0.0/14, (metric 4294967295) via
155.1.45.4, Serial0/1
Jun 26 13:44:04.153:      31.0.0.0/14 via 0.0.0.0 in 16 hops (inaccessible)
Jun 26 13:44:04.153: RT: del 31.0.0.0/14 via 155.1.45.4, rip metric [120/2]
Jun 26 13:44:04.157: RT: delete subnet route to 31.0.0.0/14
Jun 26 13:44:04.157: RT: NET-RED 31.0.0.0/14
Jun 26 13:44:04.157: RT: delete network route to 31.0.0.0
Jun 26 13:44:04.157: RT: NET-RED 31.0.0.0/8
Jun 26 13:44:04.157: RIP-DB: Remove 31.0.0.0/14, (metric 4294967295) via
155.1.45.4, Serial0/1
Jun 26 13:44:04.217: RIP: received v2 update from 155.1.0.4 on Serial0/0.1
Jun 26 13:44:06.156: RIP: sending v2 flash update to 224.0.0.9 via Serial0/0.1
(155.1.0.5)
Jun 26 13:44:06.156: RIP: build flash update entries
Jun 26 13:44:06.156:      30.0.0.0/14 via 0.0.0.0, metric 16, tag 0
Jun 26 13:44:06.156:      31.0.0.0/14 via 0.0.0.0, metric 16, tag 0
Jun 26 13:44:06.156: RIP: sending v2 flash update to 224.0.0.9 via Serial0/1
(155.1.45.5)
Jun 26 13:44:06.156: RIP: build flash update entries
Jun 26 13:44:06.160:      30.0.0.0/14 via 0.0.0.0, metric 16, tag 0
Jun 26 13:44:06.160:      31.0.0.0/14 via 0.0.0.0, metric 16, tag 0
```

R4 receives the invalid route back from R5 again, and likewise withdraws the route from the routing table and database. The process then continues indefinitely.

```
Jun 26 13:44:06.173: RIP: received v2 update from 155.1.45.5 on Serial0/1
Jun 26 13:44:06.177:      30.0.0.0/14 via 0.0.0.0 in 16 hops (inaccessible)
Jun 26 13:44:06.177:      31.0.0.0/14 via 0.0.0.0 in 16 hops (inaccessible)
Jun 26 13:44:06.281: RIP: received v2 update from 155.1.0.5 on Serial0/0.1
Jun 26 13:44:06.285:      30.0.0.0/14 via 0.0.0.0 in 16 hops (inaccessible)
Jun 26 13:44:06.285: RT: del 30.0.0.0/14 via 155.1.0.5, rip metric [120/3]
Jun 26 13:44:06.285: RT: delete subnet route to 30.0.0.0/14
Jun 26 13:44:06.285: RT: NET-RED 30.0.0.0/14
Jun 26 13:44:06.285: RIP-DB: Remove 30.0.0.0/14, (metric 4294967295) via
155.1.0.5, Serial0/0.1
Jun 26 13:44:06.289:      31.0.0.0/14 via 0.0.0.0 in 16 hops (inaccessible)
Jun 26 13:44:06.289: RT: del 31.0.0.0/14 via 155.1.0.5, rip metric [120/3]
Jun 26 13:44:06.289: RT: delete subnet route to 31.0.0.0/14
Jun 26 13:44:06.289: RT: NET-RED 31.0.0.0/14
Jun 26 13:44:06.289: RIP-DB: Remove 31.0.0.0/14, (metric 4294967295) via
155.1.0.5, Serial0/0.1
```

The fix for this problem is to ensure that R4 does not install its own summary back into the routing table. OSPF, EIGRP, and BGP already prevent this problem by design through the usage of a route to Null0 being generated every time a summary is created. Multiple solutions are valid to solve this problem, as long as R4 does not install the route.

In the above solution the fix is to configure static routes to Null0, mimicking the behavior of the other protocols.

Another solution would be to configure distribute-list filtering inbound on R4 to prevent the route from being installed.

Another valid solution would be to offset the route to 16 out on R5, or enable split-horizon, however these solutions would stop the routes from being advertised to the rest of the neighbors on the Frame Relay cloud.

```
Rack1R4#conf t
Enter configuration commands, one per line. End with CNTL/Z.
Rack1R4(config)#ip route 30.0.0.0 255.252.0.0 Null0
Rack1R4(config)#ip route 31.0.0.0 255.252.0.0 Null0
Rack1R4(config)#end
Rack1R4#
```

```
Rack1R2#ping 30.0.0.1 repeat 100
```

```
Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 30.0.0.1, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (100/100), round-trip min/avg/max =
100/112/341 ms
```