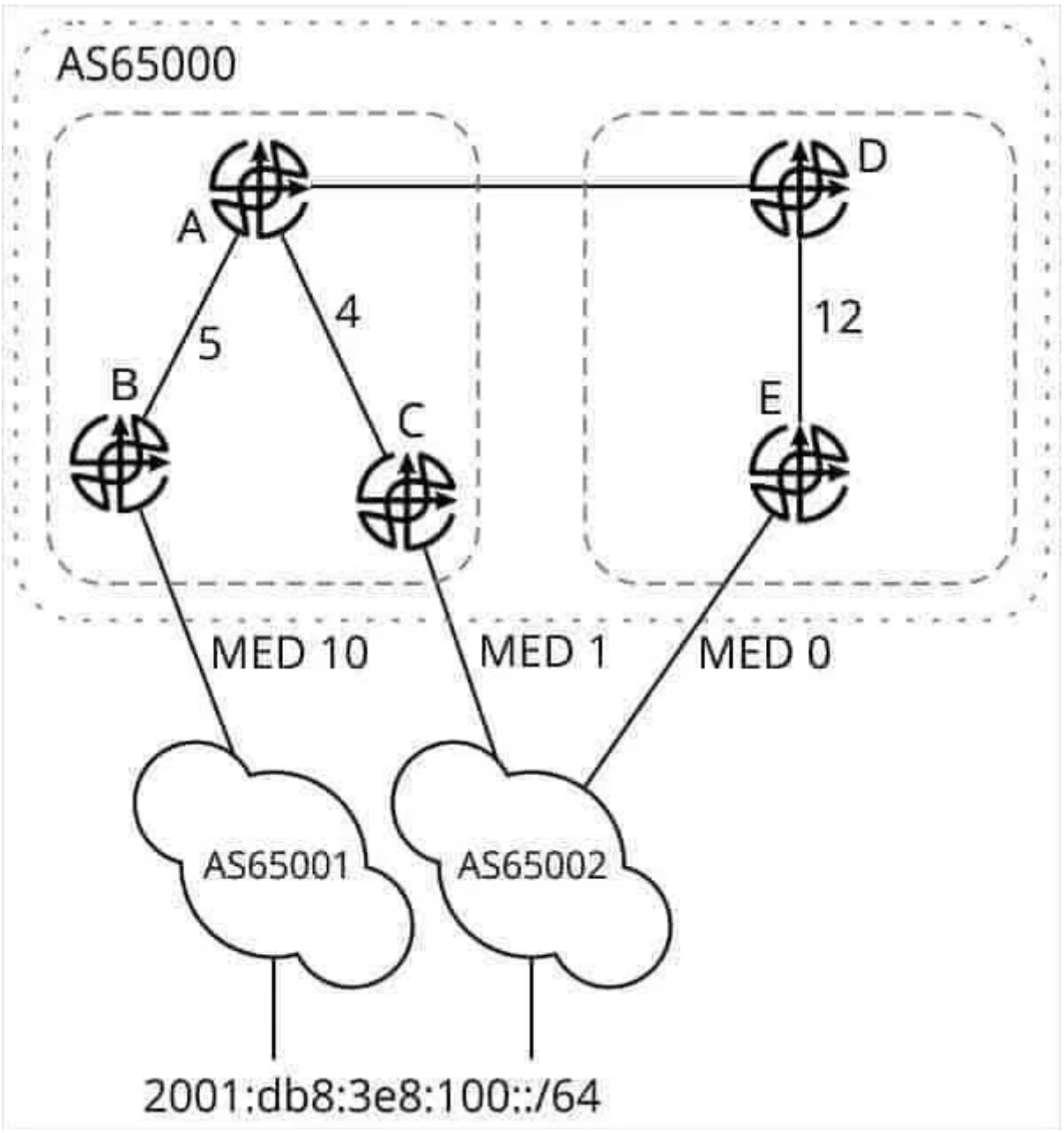
-this will take more than 4 minutes to read-



For those who are interested, I'm pretty much following RFC3345 in this explanation.

There are two BGP route reflectors here, in two different clusters, labeled A and D. The metric for each link is listed on the links between the RR clients, B, C, and E, and the RRs; the cost of the link between the RRs is 1. A single route, 2001:db8:3e8:100::/64 is being advertised in with an AS path of the same length from three different eBGP peering points, each with a different MED. E is receiving the route with a MED of 0, C with a MED of 1, and B with a MED of 10.

Starting with A, walk through one cycle of the persistent oscillation. At A there are two routes —

```
edge MED IGP Cost
C    1   4
B    10  5 (BEST)
```

When A runs the bestpath calculation, it will determine the best path should be through C, rather than B (because the IGP cost is lower; the MEDs are not compared due to different AS paths), so it will send an update to each of its peers about this change in its best path. This results in D having the following table—

```
edge MED IGP Cost
E    0   12 (BEST)
C    1   5
```

When D receives this update from A, it will calculate the best path towards 2001:db8:3e8:100::/64, and choose the path through E because this path has the lowest MED (the MEDs are compared here because the AS path is the same). On completing the best path calculation, E will send an update to its peers, letting them know about its new best path to the destination, which primarily means A. On receiving this update, A has three routes to the destination in its table—

```
edge MED IGP Cost
E    0   13
C    1   4
B    10  5 (BEST)
```

The key point to think through here is why the third route in the table is best. The BGP bestpath process compares each pair of routes, starting with the first pair. So the first two routes are compared, the best of those two is chosen and compared with the third, the best of these two is compared with the fourth, etc., until the end of the table is reached. Here the first two paths are compared, and the path through E because of the lower MED (the AS path is the same). When the path through E is compared to the path through B, however, the path through B wins because the IGP metric is lower (the AS path is different).

At this point, A will now send an update to its peers, specifically D, informing D of this change in its best path. Note this update removes any information about the path through C from D's table, so D only has a partial view of the available paths. As a result, D will have the following in its table—

```
edge MED IGP Cost
E    0   12
B    10  6 (BEST)
```

D will select the path through B because the IGP cost is lower, and send an update to A with this new information. This will result in A having the table this process started with—

```
edge MED IGP Cost
C    1   4
B    10  5 (BEST)
```

What is interesting about this is the removal of information about the path through C from D's view of the network. Essentially, what is happening here is D is switching between two different views of the network topology, one of which includes B, the other of which includes C. The reason the ADD_PATH extension solves this problem is that A and D both have a full view of every exit point once each BGP speaker sends every route to each destination, rather than just the best path.

This is, in effect, another instance of the inconsistency of a distributed database causing a persistent condition in a control plane. In (loosely!) CAP theorem terms, distributed routing protocol always choose accessibility (the local device can read the database to calculate loop free paths) and partitioning (the database is always copied to every device speaking the protocol) over consistency—eventually, or "not always," consistent databases will always be the result of such a situation. As A and D read their databases, each of which contain incomplete information about the real state of the network, they will make different decisions about what the best path to the destination in question is. As they each change their views of the topology, they will send updated information to one another, causing the other BGP speaker to recompute its view of the topology, and...

Persistent BGP oscillation is an interesting study in the way consistency impacts distributed routing protocol design and convergence.

Related

BGP and Suboptimal Route Reflection (https://rule11.tech/bgp-and-suboptimal-route-reflection/)
15 October 2018
In "BGP"

Section 10 Routing Loops (https://rule11.tech/section-10-routing-loops/)
9 January 2018
In "BGP"

Optimal Route Reflection (https://rule11.tech/optimal-route-reflection/)
17 April 2017
In "BGP"

← Ants and Leaves (https://rule11.tech/ants-and-leaves/)

Reflections → (https://rule11.tech/reflections/)

2 comments

HEMANTH RAJ on 30 August 2017 at 2:46 am

Hi Russ

BGP shouldnt be calculating the best path comparing the two prefixes and compare the rest as we lose end to end visibility on the best path as it has to calculate all in serial fashion to avoid using incosistent paths across the network with dissimilar metrics and nowadasys we dont use disimilar metrics in our IGP core anymore to avoid this issue.

Any thoughts.

Russ on 30 August 2017 at 8:15 am

Most providers strip MED at their edge, so this specific form of oscillation is not common. Another option is to *always compare MED,* which caused BGP to compare MEDs across inconsistent AS paths–but this has other bad implications from a best path perspective. Overall, the add paths solution provides the best solution with optimal paths, while the other solutions prevent the oscillation, but with (potentially) less than optimal routes being used.

🙂

Russ