Route reflector design in an mpls network

Due to the need of having the same knowledge of the network and routing information, PE routers on an mpls network would require a full mesh of ibgp sessions. To avoid this impact and huge amount of configurations, Route Reflectors are used to better scale the network. They behave slightly differently from the standard bgp approach, breaking the ibgp split horizon rule: when they receive an ibgp route from a so called “route reflector client”, they propagate it to other ibgp clients and non clients (together with eBgp neighbors). A configuration parameter for route reflectors, which is very often ignored, is the “cluster id”. It has something to do with RR redundancy, and should be configured having in mind the consequences. If you don’t do nothing, the highest loopback address (at least on Cisco devices) is used (such as for the router id in many routing protocols).

So the questions are:

- to which cluster-id should the RR belong ?
- how should ibgp sessions be established between clients and RR ?

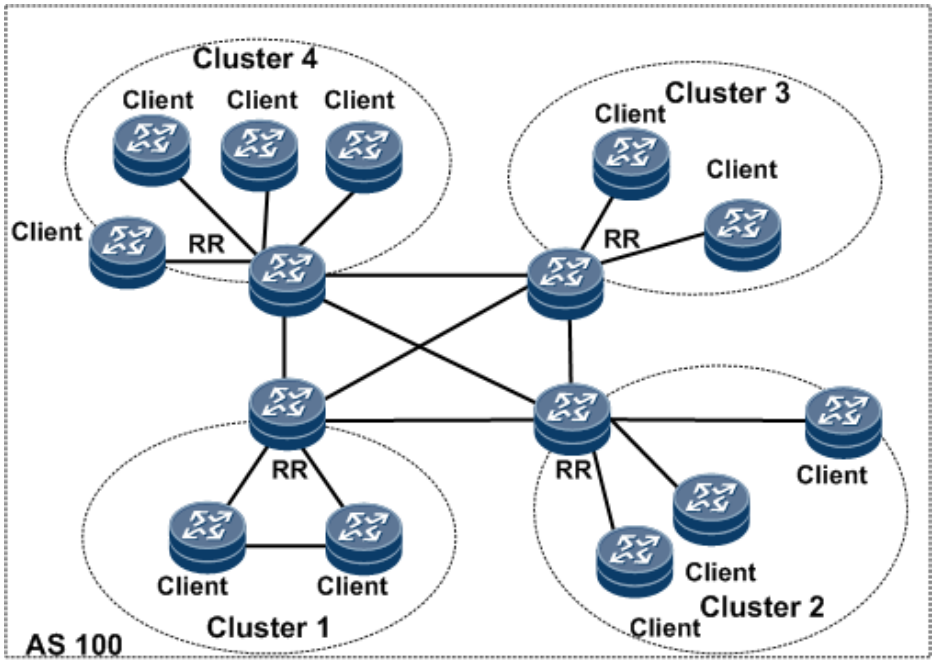

Figure 1 Core network example diagram.

In the above example diagram that has been found on the internet, the above questions can be easily answered: the network is divided into 4 different clusters, and the clients have an ibgp session with the RR belonging to the same cluster. This is because those 4 RR are also a SPOF (Single Point Of Failure) for the network … hopefully, there is no real network made like this. But what happens in case core P routers are redundant (as they always should be) ? Things should work like depicted in Figure 2: RR have a full mesh of ibgp session between them, and they also peer with all their clients. Every client has an ibgp session with the RR belonging to the same cluster. Things can scale even to multiple RR layers, as depicted in Figure 3. To avoid loops in case of normal operation or misconfigurations, two parameters are used:

- **originator ID** (added by the first RR, it’s the bgp router id of the node that originated the route)
- **cluster-id list** (added by every RR which propagates an i-bgp route)

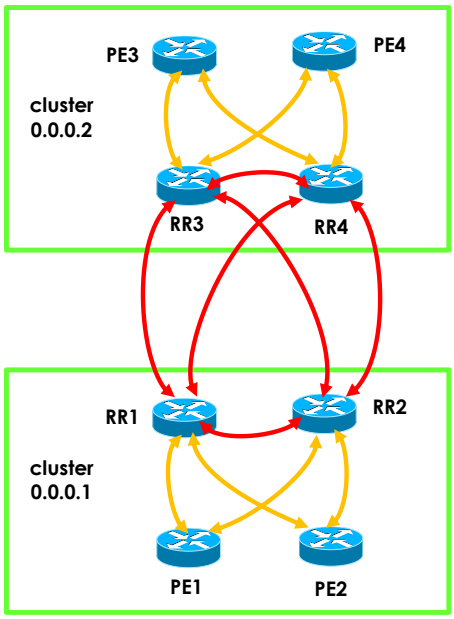

Figure 2 Ibgp sessions.

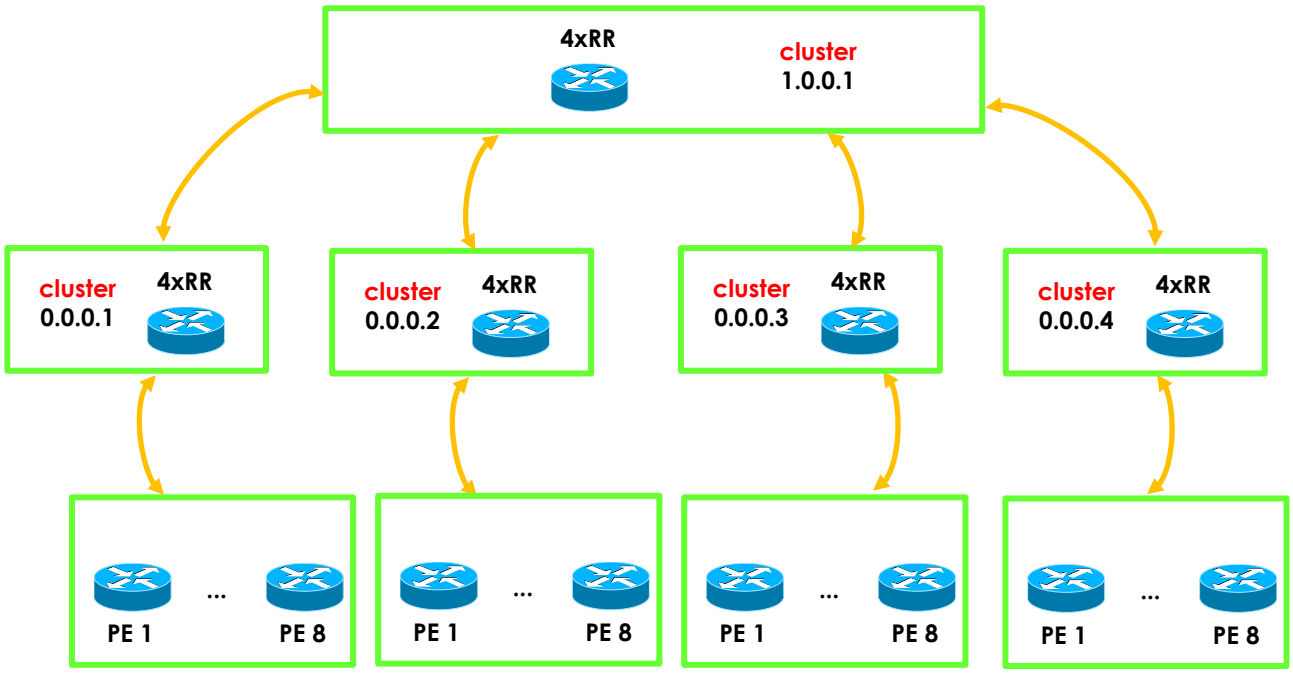

Figure 3 Multiple RR layers for even better scalability for the biggest networks.

In case a router receives a bgp route with its own bgp router-id as the 'originator', the route is discarded. In case a RR receives a bgp route with its own cluster-id in the cluster-list, the route is discarded. The cluster-list works exactly in the same way as the as-path bgp mandatory attribute for routes propagated outside the AS. This is what happens in Figure 4. RR do not ALWAYS add the cluster-id to reflected routes: in case routes are local or received by an eBgp neighbor, this doesn't happen. The reason is not just redundancy, but also the network that should work in normal conditions: problems could arise in case RR are not dedicated (as it should be as a best practice and safe approach), but they have multiple roles in the network and work also as PEs.

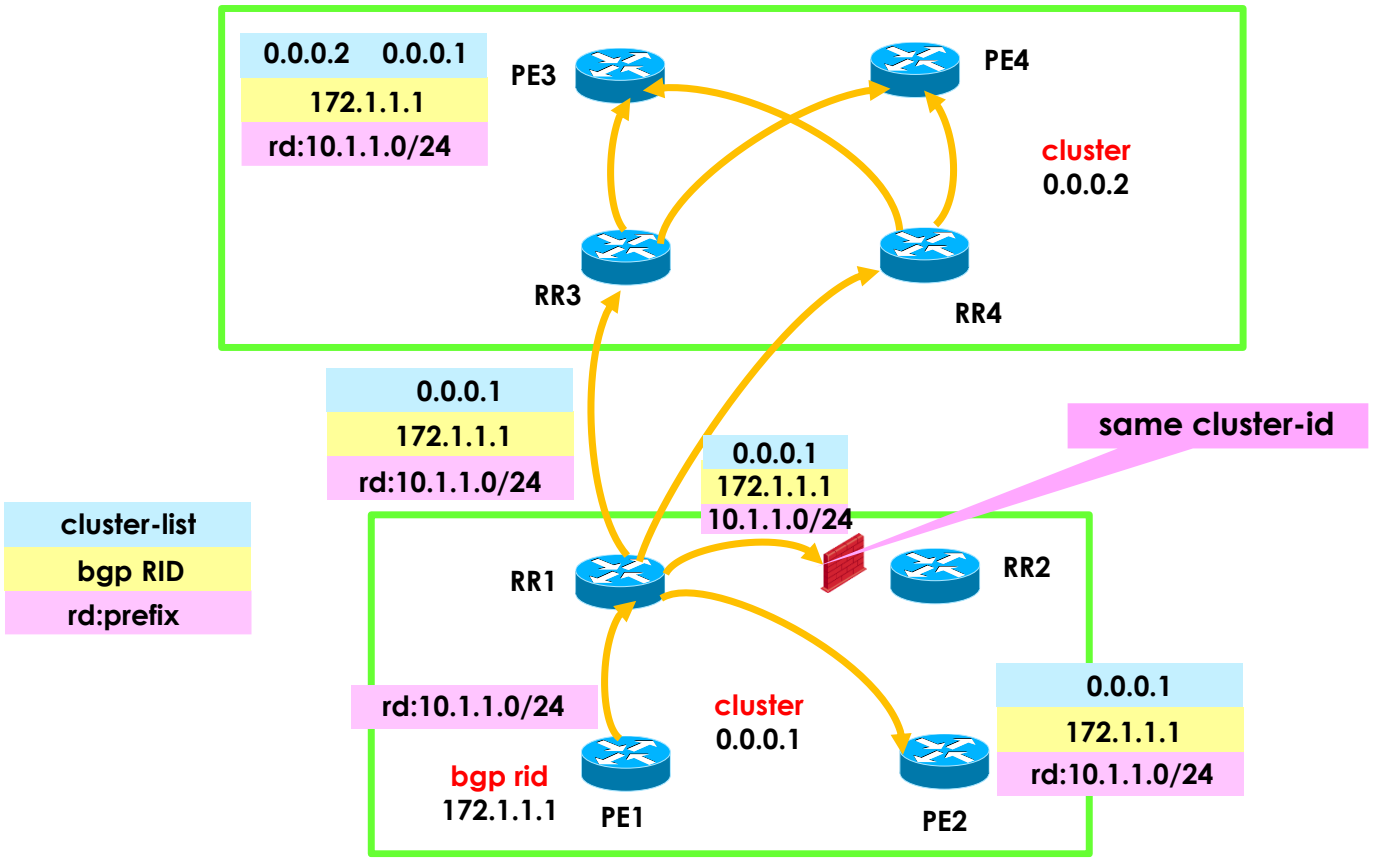

Figure 4 Routes propagated between RR belonging to the same cluster are discarded.

As depicted in Figure 5, should RR1 add its cluster-id to the route received via eBgp, RR2 would discard it because it belongs by design to the same cluster, thus CE2 would never receive it. This cannot be acceptable.

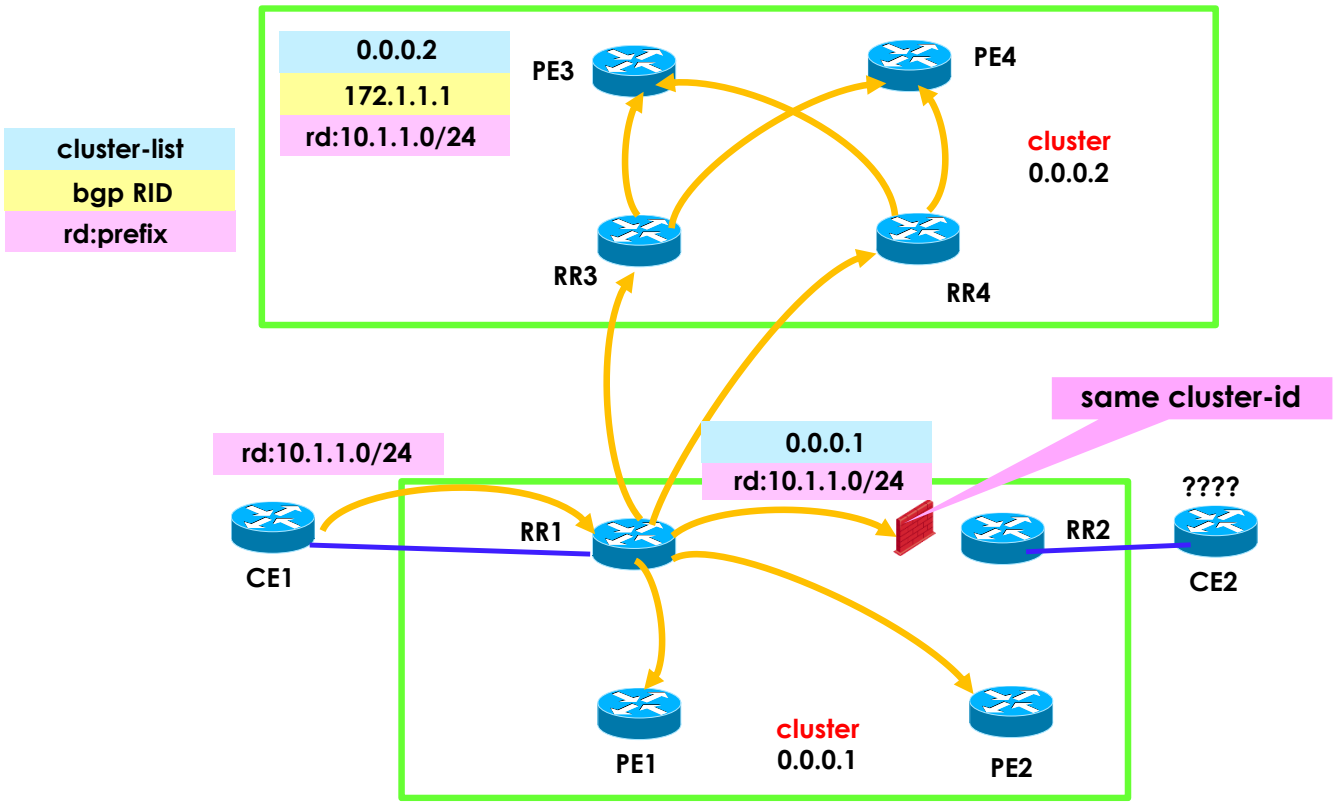

Figure 5 eBgp routes can't be treated as routes received by rr-clients.

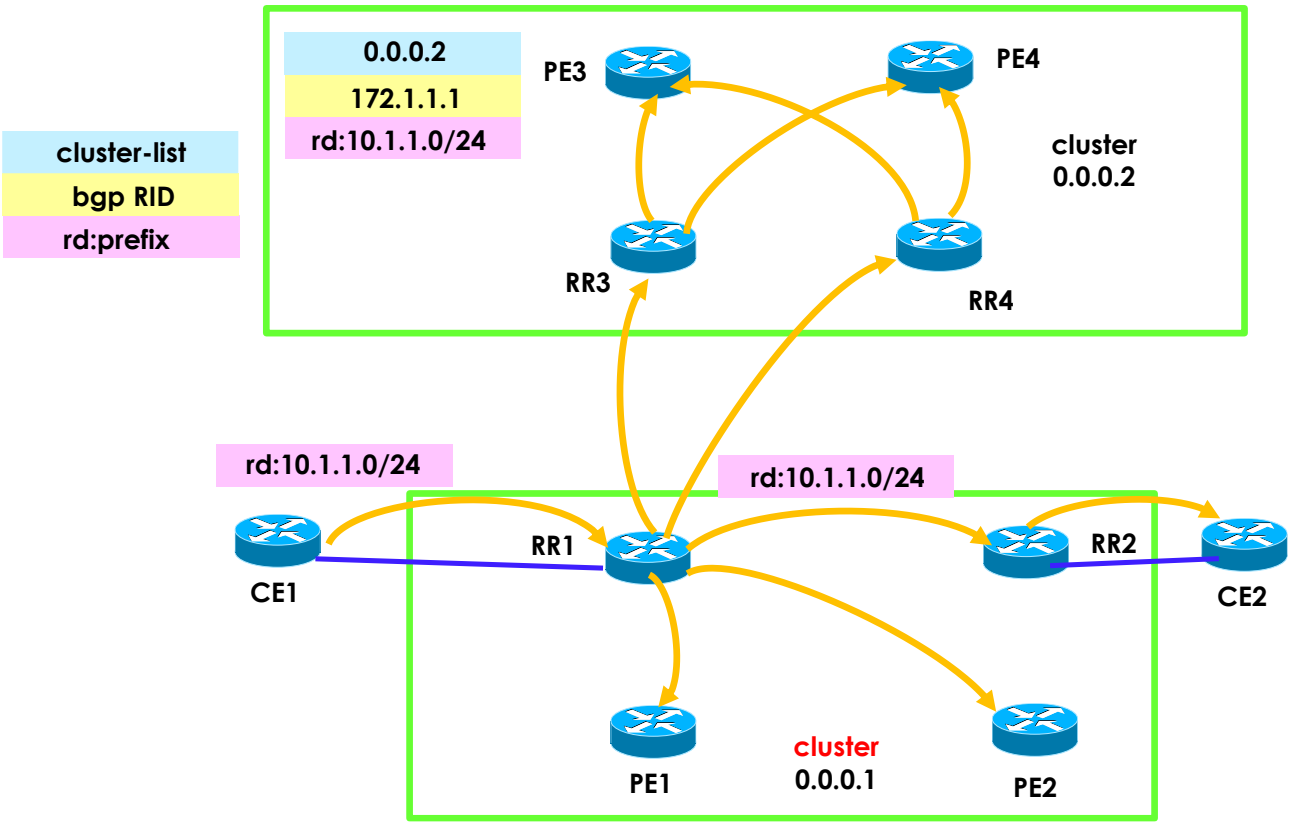

Figure 6 eBgp routes are propagated by RR without adding its cluster-id to the cluster list.

As depicted in Figure 7, the following things would happen in case there are only two RR in the network, they are configured in two different clusters, and all PEs have ibgp sessions toward both of them:

- PE1 announces the route to RR1
- RR1 propagates the route to PE2 and RR2
- RR2 propagates the route to PE2 and PE1 (discarded by PE1 thanks to originator's ID)

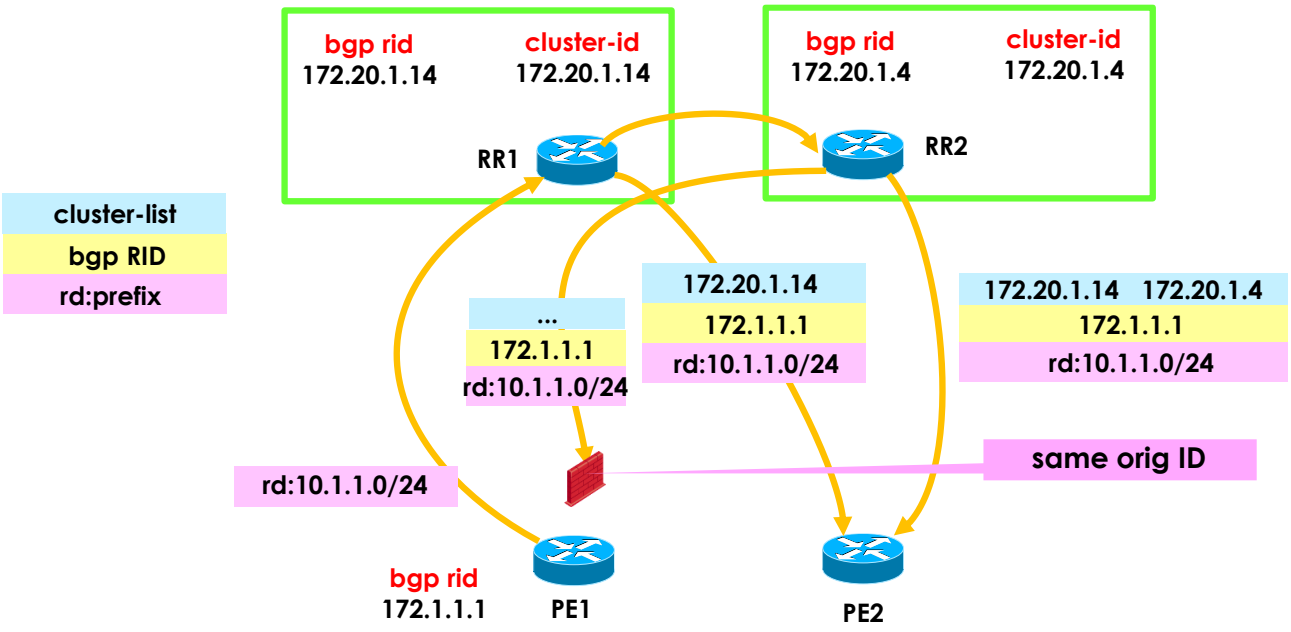

Figure 7 Bgp route propagation example.

The same happens toward RR2 (not depicted above, but described here):

- PE1 announces the route to RR2
- RR2 propagates the route to PE2 and RR1
- RR1 propagates the route to PE2 and PE1 (discarded by PE1 thanks to originator's ID)

For this reason, every PE receives back and rejects (at least) a couple of bgp routes because they are reflected back two times by both RR, since they belong to different clusters.

Here comes another interesting approach by Cisco: this doesn't seem to depend on the switch/router family, nor on the software release. Probably to simplify the software code, Cisco announces 'back' all the routes received via iBgp or eBgp, **even to the neighbor from which they are received**. From **RFC4456**:

''When an RR receives a route from an IBGP peer, it selects the best path based on its path selection rule. After the best path is selected, it must do the following depending on the type of peer it is receiving the best path from:

1) A route from a Non-Client IBGP peer:

Reflect to all the Clients.

2) A route from a Client peer:

Reflect to all the Non-Client peers and also **to the Client peers**. (Hence the Client peers are not required to be fully meshed.)"

… RFC doesn't specify something like "except from the neighbor from which the route was received". Cisco has extended this approach also to eBgp neighbors, even though in this case routes are discarded because of the bgp loop prevention mechanism. Things would change in an mpls network in case "as-override" is configured on the PE side toward the CE: in this case, routes would be seen on the CE (which is a quite dirty thing …).

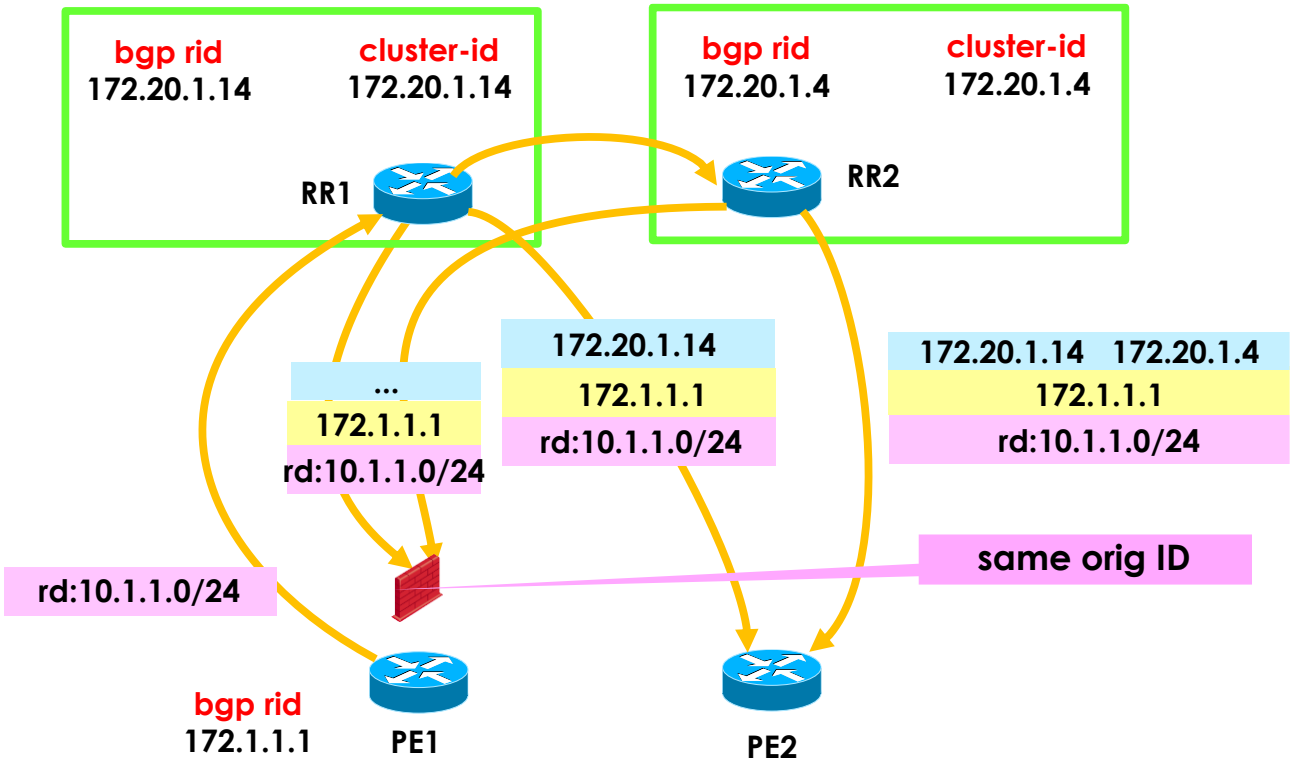

Figure 8 Bgp route propagation in an mpls network.

Given this behavior of Cisco routers, **every PE receives back its announce 4 times**. More generally speaking, if there are NxRR in the network, all configured in different clusters, every route is reflected back to the PE that has announced it for N square times. This is probably not a problem on little networks, but if you have 800k internet routes and say 4 RR, this would lead to 12,8 million routes ...

1.1 Dedicated RR vs shared RR

The question is … what changes between the two approaches ? to simplify things, let's consider only two RR devices. What changes if we configure them on the same cluster, or on different cluster ? from the above explained behavior, it would look like configuring them in the same cluster is better because it avoids having a number of 'spurious' bgp routes reflected back to the PEs. But is it really true ? is there anything else left to consider ?

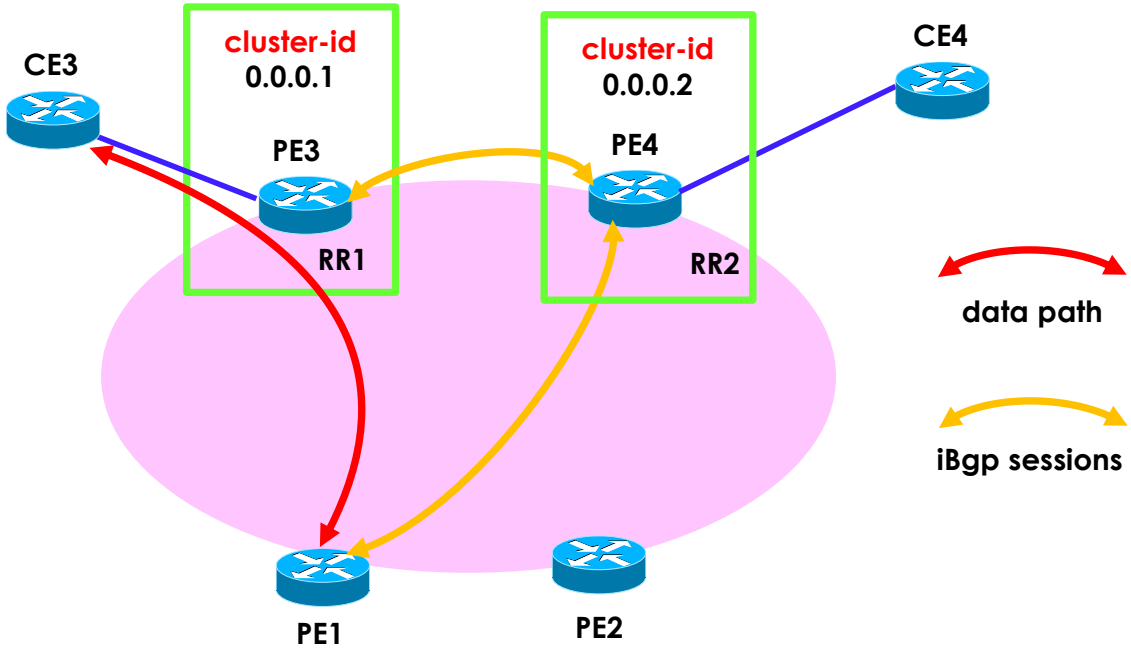

Figure 9 Bgp route propagation in case of PE1-RR ibgp failure.

Let's consider the above diagram, in which there are two RR also being PEs (RR1 is also PE3, RR2 is also PE4). For some strange reasons, the iBgp session between PE1 and RR1/PE3 is down, even though the two routers are still up and running. In this case, configuring RR in two different clusters, nothing changes respect to before: the traffic path (highlighted in red) remains bi-directionally the same. Routes propagated from PE1 are still received by CE3, routes announced by CE3 are still received by PE1.

What happens instead in case **a single cluster is configured** ? routes announced by CE3 would still be received by PE1, but in the opposite direction PE3 would discard PE1 routes, because of the previously explained loop prevention mechanism based on the cluster list. In case there are firewalls behind CE3 and CE4, and the two paths backup each other, asymmetrical routing would potentially arise.

Beware that even in case DEDICATED route reflectors are used, the above described problem would not arise, because a single ibgp session failure would have no effects on the traffic data path.

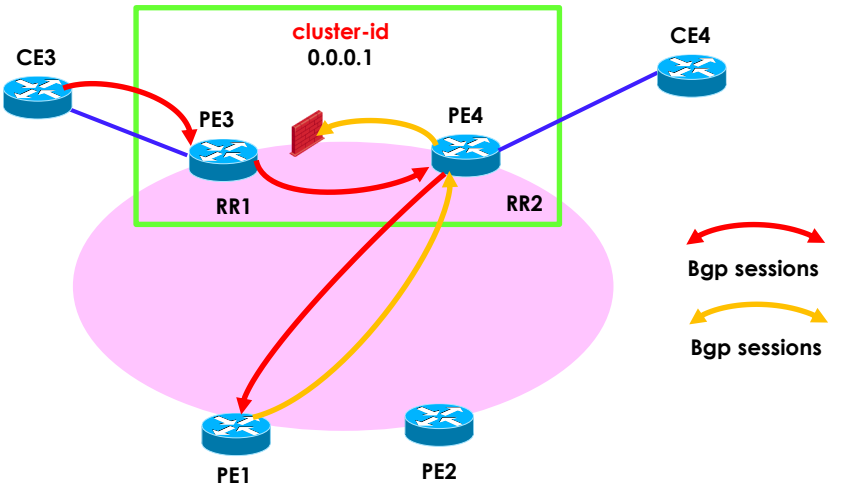

Figure 10 Double role PE/RR bgp propagation problems.

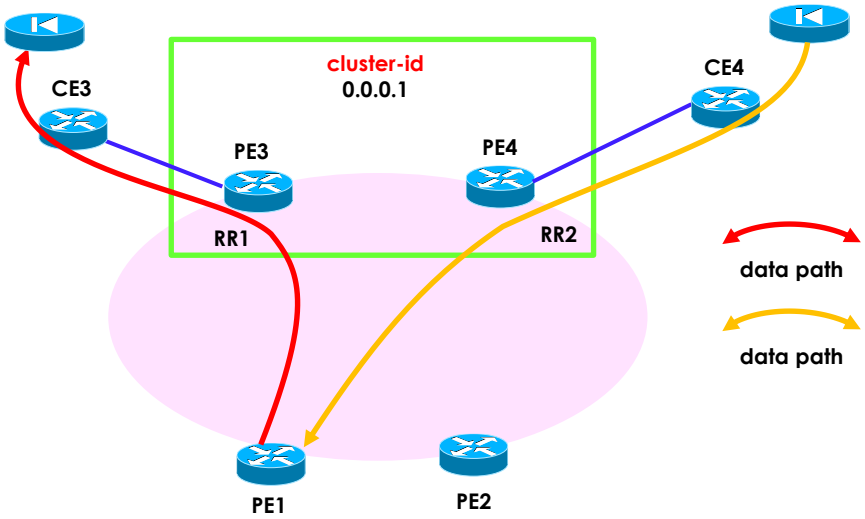

Figure 11 Potential asymmetrical routing issue.

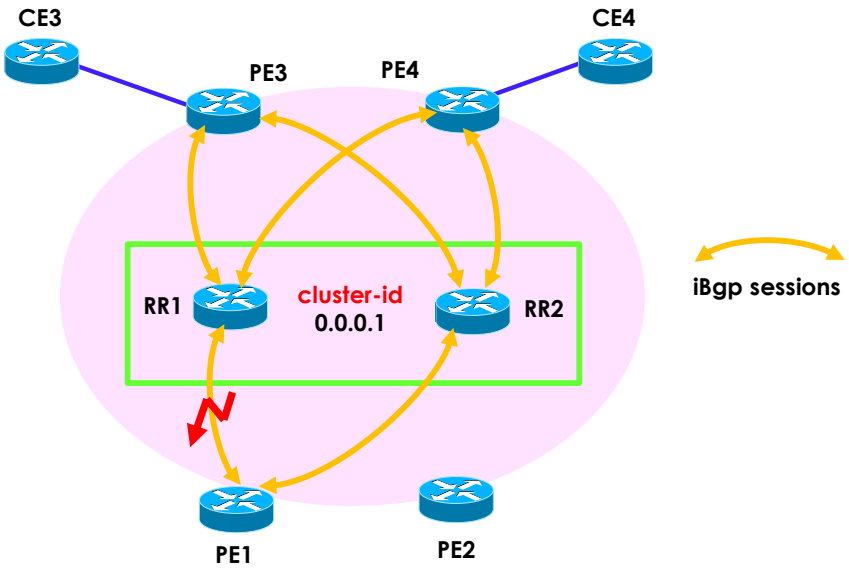

Figure 12 Same architecture with DEDICATED route reflectors.

2.1 More on cluster design and redundancy

Let's now suppose that there are TWO ibgp sessions failures, with all PE/RR routers still up and running. This is quite an **absurd failure scenario**, which would unlikely or never happen in a real network, in any case there would be different consequences, as depicted in Figure 13 and Figure 14.

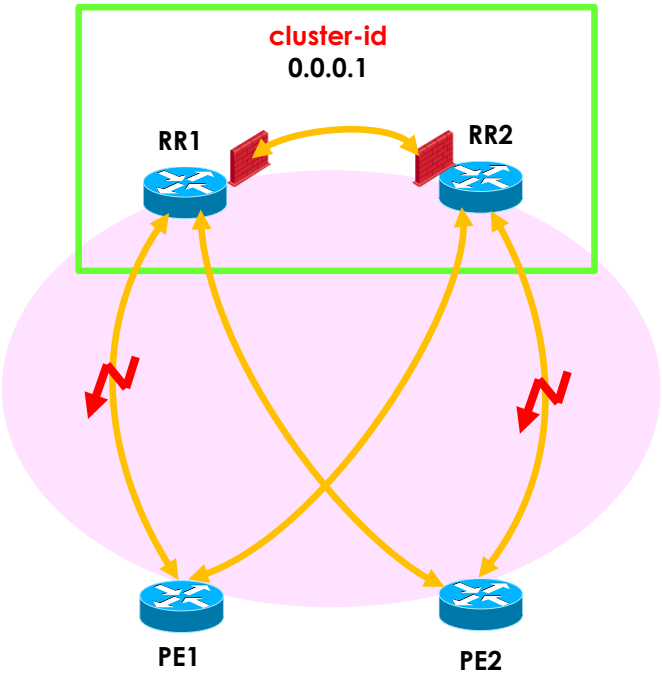

Figure 13 Same cluster for both RR, two ibgp sessions failures.

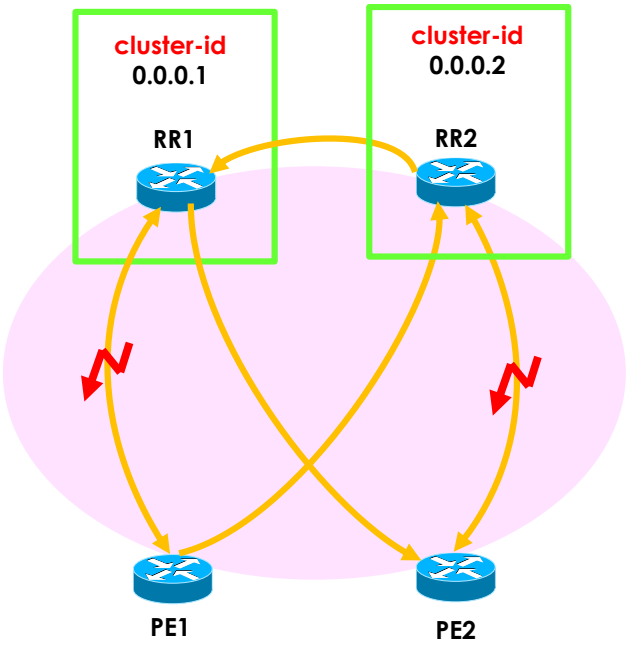

Figure 14 Different cluster for both RR, two ibgp sessions failures.

When could such an ibgp session failure happen ?

- PE got isolated from the network (multiple failures or planned activity)
- RR got isolated from the network (multiple failures or planned activity)

- the ip backbone is partitioned, due to multiple link/node failures
- traffic ‘black holing’: the control plane is up, but traffic is dropped

In some of the above examples, we wouldn’t fall in the described scenario (an isolated PE would simply stop receiving/sending traffic), a network ‘black holing’ is a rare failure scenario, that at least in my opinion shouldn’t drive or condition design decisions.

3.1 Conclusions

After all the above explanations, the following rules should be followed in a real life mpls network:

- use **DEDICATED route reflectors** whenever possible, this increases reliability and redundancy. Too often customers save money on this, which doesn’t make sense considering what you save respect to the total amount of expenses for the whole mpls network
- there is a ‘trade off’ between the number of spurious announces on the network, and the network reliability. If you have up to 3 RR in the network, you could still consider configuring them in 3 different clusters, otherwise go for dedicated RR and pair them in clusters, made up of at least two RR for each one. For example 4 RR could be divided into 2 clusters, 8 RR could be divided into 4 or 2 clusters.

When connecting RR to the network, also consider the topology. One other good trick could be that of configuring the isis overload bit permanently, or in case you have more than two RR in every cluster you could even connect them with a single link. This would avoid traffic flowing in the mpls backbone from transiting through RR low bandwidth links, due to errors in link’s cost configuration when configuring new nodes, or isolating nodes during a planned activity.