HBASE-26220 Use P2P communicate between region servers to sync the list for bootstrap node

Background

In the current connection registry implementation, we use region servers as connection registry endpoints. There is a getBootstrapNodes method in the connection registry endpoint, where we will return a small set of region servers to client side and then client side will use these region servers as bootstrap nodes. Now, on every region server, we will set up a RegionServerAddressTracker to track the region server list, by setting up a watcher on ZooKeeper.
For large clusters, this could put big pressures on ZooKeeper, and considering the usage of the region server list, we neither need the accurate live region server list, nor need the whole region server list.
So here, we propose to use something like a P2P network between region servers to exchange the live region server list, to reduce the pressure on ZooKeeper.

Algorithm

1. On start up, every region server needs to get an initial list of live region servers in the cluster. Notice that it does not need to be all the live region servers, only a small set is enough. On how to get this list, could either from HMaster or ZooKeeper. I prefer HMaster because
    a. We want to reduce the dependency on ZooKeeper.
    b. To fetch from ZooKeeper, we need to call getChildren which will return all the live region servers, while on HMaster it is easy to control the number of the returned live region servers.
2. Every time after a reasonable interval (5 minutes plus a jitter maybe?), each region server will select a region server from its list randomly, send a request to the region server to get its live region server list, and then merge the returned list with its own.
3. Each region server will schedule a background task to verify the liveness of the region servers in its list and remove the dead servers from the list. Once the number of live region servers is too low, we will send a request to the still live region servers soon to get new live region servers. If there is none, we need to fetch the list from HMaster again.

Edge cases

Cold start

The region servers for a cluster can not start at the same time, so for the region servers which start earlier, they can not get enough live region servers from HMaster or ZooKeeper. This should be the same case described in the above #3, but we need to have some backoff policy to avoid DDoS the HMaster or other region servers.

local convergence

It is possible that after some round of exchange, the region servers in the clusters have been divided into some self connected groups, which means each group will not know there are still other live region servers outside the group.

In general, this is not a big deal in our use case, as long as we still have enough live region servers in the list for a region server. And if the number of the live region servers is too small, we could still fetch from HMaster since they have the full list.

Small Cluster

In the above algorithm, there are some descriptions like "if the number of live region servers in the list is too small, then blabla". The problem here is that, if the cluster itself is small, for example, we only have 1 live region server in the cluster, then no matter what we do, the number of live region servers is 0(excluding the live region server itself). So, still the same way to fix, going to HMaster.

When to go to HMaster

1. Startup
2. If the number of live region servers is 0
3. If the number of the live region servers is below the threshold, and after a configurable round of exchange from other region servers, we still can not increase the number

How to avoid DDoS HMaster

The HMaster could return the total number of region servers in the cluster to region servers, if we find out that our live region server list is already all the live region servers in the cluster, we will start to backoff. The reset of the backoff time should be that we finally increase the number of live region servers.

Final implementation in PR

The final logic has been simplified a lot, due to the following two tricks/reasons:

1. If we can not get enough live region servers from master, then we do not need to contact region servers to exchange the bootstrap nodes, we only need to request master. This is because that, master will return as many live region servers as possible, so if it can only return a small number of region servers, it means there are only a small number of region servers in the cluster, so it is not big deal to let them always request master, it will not be a big pressure for master.
2. A region server could just use the ‘request live region servers’ call to check whether the remote side region server is still alive, so we only need to schedule one background task for contacting other region servers.