Multicast replication in Percona XtraDB Cluster (PXC) and Galera

Jay Janssen (https://www.percona.com/blog/author/jay-janssen/) | June 5, 2013 | Posted In: Percona Software (https://www.percona.com/blog/category/percona-software/), Percona XtraDB Cluster (https://www.percona.com/blog/category/percona-xtradb-cluster/), XtraDB Cluster (https://www.percona.com/blog/category/xtradb-cluster/)

Bandwidth multiplication and synchronous clusters

I've seen a lot of people setting up clusters with 3-6+ nodes on 1 Gbps networks. 1 Gbps seems like a lot, doesn't it? Actually, maybe not as much as you think (https://www.google.com/search?q=1+Gbps+to+MBps). While the theoretical limit of 1Gbps is actually 120MBps, I start to get nervous around 100MBps. By default Galera uses unicast TCP for replication. Because synchronous replication needs to replicate to all nodes at once, this means 1 copy of your replication message is sent to other node in the cluster. The more nodes in your cluster, the more the bandwidth required for replication multiplies. Now, this isn't really much different from standard mysql replication. 1 master with 5 async slaves is going to send a separate replication stream to each, so your bandwidth requirements will be similar. However, with async replication you have the luxury of not blocking the master from taking writes if bandwidth is constrained and the slaves lag for a bit, not so in Galera (https://www.percona.com/blog/2013/05/02/galera-flow-control-in-percona-xtradb-cluster-for-mysql/). So, let's see this effect in action. I have a simple script that outputs the network throughput on an interface every second. I'm running a sysbench test on one node and measuring the outbound (UP) bandwidth on that same node:

```shell
# 2 nodes in the cluster
eth1 DOWN:24 KB/s UP:174 KB/s
eth1 DOWN:25 KB/s UP:172 KB/s
eth1 DOWN:27 KB/s UP:196 KB/s
eth1 DOWN:27 KB/s UP:195 KB/s
eth1 DOWN:27 KB/s UP:197 KB/s
eth1 DOWN:27 KB/s UP:200 KB/s

# 3 nodes in the cluster
eth1 DOWN:74 KB/s UP:346 KB/s
eth1 DOWN:79 KB/s UP:357 KB/s
eth1 DOWN:77 KB/s UP:342 KB/s
eth1 DOWN:79 KB/s UP:368 KB/s
eth1 DOWN:81 KB/s UP:368 KB/s
eth1 DOWN:78 KB/s UP:363 KB/s
```

This isn't much traffic in my puny local VMs, but you get the idea. We can clearly see some factor in play adding the extra nodes.

Multicast to the rescue!

One way to address this bandwidth constraint is to switch to multicast UDP replication in Galera. This is actually really easy to do. First, we need to make sure our environment will support multicast (https://gist.github.com/jayjanssen/5697813). This is a question for your network guys and beyond the scope of this post, but in my trivial VM environment, I just need to make sure that the multicast address space routes to my Galera replication interface, eth1:

```
Shell
[all nodes]# ip ro add dev eth1 224.0.0.0/4
[all nodes]# ip ro show | grep 224
224.0.0.0/4 dev eth1  scope link
```

In that space, we pick an unused mcast address (again, talk to your network guys). I'm using 239.192.0.11, so we'll add this to our my.cnf:

```
Shell
wsrep_provider_options          = "gmcast.mcast_addr=239.192.0.11"
```

If you already have wsrep_provider_options set, add it to the semicolon separated list instead of a separate line in your config. If we already have a running cluster, we need to shut it down, configure our mcast address and re-bootstrap it:

```
Shell
[root@node3 mysql]# service mysql stop
[root@node2 mysql]# service mysql stop
[root@node1 mysql]# service mysql stop
```

```
Shell
[root@node1 mysql]# service mysql start --wsrep_cluster_address=gcomm://
[root@node2 mysql]# service mysql start
[root@node3 mysql]# service mysql start
```

We can see that a multicast node still needs to bind to the Galera replication port, and of course that needs to be bound to the interface that the multicast will be received on.

```
Shell
[root@node3 mysql]# lsof -P +p 17493 | grep LISTEN
mysqld 17493 mysql 11u IPv4 39669 0t0 TCP *:4567 (LISTEN)
mysqld 17493 mysql 20u IPv4 39685 0t0 TCP *:3306 (LISTEN)
```

Now, let's re-do our above test:

```
Shell
# 2 nodes in the cluster
eth1 DOWN:15 KB/s UP:199 KB/s
eth1 DOWN:14 KB/s UP:195 KB/s
eth1 DOWN:15 KB/s UP:212 KB/s
eth1 DOWN:14 KB/s UP:204 KB/s
eth1 DOWN:13 KB/s UP:173 KB/s

# 3 nodes in the cluster
eth1 DOWN:62 KB/s UP:185 KB/s
eth1 DOWN:61 KB/s UP:187 KB/s
eth1 DOWN:52 KB/s UP:164 KB/s
eth1 DOWN:62 KB/s UP:187 KB/s
eth1 DOWN:64 KB/s UP:186 KB/s
eth1 DOWN:62 KB/s UP:193 KB/s
```

So, we can see our outbound bandwidth on our master node doesn't change as we add more nodes when we are using multicast.

Other multicast tips

We can also also bootstrap nodes using the mcast address:

```
Shell
#wsrep_cluster_address = gcomm://192.168.70.2,192.168.70.3,192.168.70.4
wsrep_cluster_address = gcomm://239.192.0.11
```

And this works fine. Pretty slick! Note that IST and SST will still use TCP unicast, so we still want to make sure those are configured to use the regular IP of the node. Typically I just set the wsrep_node_address setting on each node if this IP is not the default IP of the server. I could not find a way to migrate an existing unicast cluster to multicast with a rolling update. I believe (but could be proven wrong) that you must re-bootstrap your entire cluster to enable multicast.

Related

Typical misconceptions on Galera Replication for MySQL (https://www.percona.com/blog/…misconceptions-on-galera-for-mysql/)
November 17, 2014
In "High-availability"

Followup questions to 'What's new in Percona XtraDB Cluster 5.6' webinar (https://www.percona.com/blog/…questions-whats-new-percona-xtradb-cluster-5-6-webinar/)
February 7, 2014
In "MySQL"

Doing a rolling upgrade of Percona XtraDB Cluster from 5.5 to 5.6 (https://www.percona.com/blog/…upgrade-percona-xtradb-cluster-5-5-5-6/)
May 8, 2014
In "MySQL"

Jay Janssen (/blog/author/jay-janssen/)

Jay joined Percona in 2011 after 7 years at Yahoo working in a variety of fields including High Availability architectures, MySQL training, tool building, global server load balancing, multi-datacenter environments, operationalization, and monitoring. He holds a B.S. of Computer Science from Rochester Institute of Technology.