MultinodeCluster10

Prior to the 0.7 release, Cassandra storage configuration is described by the *conf/storage-conf.xml* file. As of 0.7, it is described by the *conf/cassandra.yaml* file. Please refer to MultinodeCluster06 for about pre-0.7 configuration.

Creating a multinode cluster

The default cassandra.yaml provided with cassandra is great for getting up and running on a single node. However, it is inappropriate for use in a multi-node cluster. The configuration and process here are the *simplest* way to create a multi-node cluster, but may not be the *best* way in production deployments.

Preparing the first node

The default cassandra.yaml uses the local, loopback address as its listen (inter-node) and Thrift (client access) addresses:

```
listen_address: locahost

rpc_address: localhost
```

As the listen address is used for intra-cluster communication, it must be changed to a routable address so the other nodes can reach it. For example, assuming you have an Ethernet interface with address 192.168.1.1, you would change the listen address like so:

```
listen_address: 192.168.1.1
```

The Thrift interface can be configured using either a specified address, like the listen address, or using the wildcard 0.0.0.0, which causes cassandra to listen for clients on all available interfaces. Update it as either:

```
rpc_address: 192.168.1.1
```

Or perhaps this machine has a second NIC with ip 10.140.179.1 and so you split the traffic for the intra-cluster network traffic from the thrift traffic for better performance:

```
rpc_address: 10.140.179.1
```

If the DNS entry for your host is correct, it is safe to use a hostname instead of an IP address. Similarly, the seed information should be changed from the loopback address:

```
seeds:
  - 127.0.0.1
```

Becomes:

```
seeds:
  - 192.168.1.1
```

Once these changes are made, simply restart cassandra on this node. Use netstat (e.g. `netstat -ant | grep 7000`) to verify cassandra is listening on the right address. Look for a line like this:

```
tcp4 0 0 192.168.1.1.7000 *.* LISTEN
```

If netstat still shows cassandra listening on 127.0.0.1.7000, then either the previous cassandra process was not properly killed or you are not editing the cassandra.yaml file cassandra is actually using.

Preparing the rest of the nodes

The other nodes in the ring will use a cassandra.yaml almost identical to the one on your first node, so use that configuration as the base for these changes rather than the default cassandra.yaml. The first change is to turn on automatic bootstrapping. This will cause the node to join the ring and attempt to take control of a range of the token space:

```
auto_bootstrap: true
```

From 1.0, auto_bootstrap is true by default so you don't need to add it into cassandra.yaml.

The second change is to the listen address, as it must also not be the loopback and cannot be the same as any other node. Assuming your second node has an Ethernet interface with the address 192.168.2.1, set its listen address with:

```
listen_address: 192.168.2.1
```

Finally, update the the Thrift address to accept client connections, as with the first node, either with a specific address or the wildcard:

```
rpc_address: 192.168.2.1
```

Or:

```
rpc_address: 10.140.180.1
```

Note that you should leave the Seeds section of the configuration as is so the new nodes know to use the first node for bootstrapping. Once these changes are made, start cassandra on the new node and it will automatically join the ring, assign itself an initial token, and prepare itself to handle requests.

https://c.statcounter.com/9397521/0/fe557aad/1/|stats