**How to create a multi-node Cassandra 2 cluster on Docker containers**

First of all, you will need a Linux server with root access (in my case it’s a CentOS 7.4). Start by installing Docker, which is easily done from the standard repositories.

Make sure that the Docker daemon is running, if not start it manually and enable it at startup:

systemctl start docker  
systemctl enable docker

In order to simplify the configuration of the cluster, the containers will need static IPs, so create a Docker network:

docker network create --subnet=172.18.0.0/16 mynet123

Cassandra needs Java to run and installing it is time-consuming, so download a Docker image that already has it installed without other additional packages. In this example, it is an Ubuntu 17 image with jdk 8:

docker pull docker.io/dongjoon/ubuntu17.04-jdk8

Check if the new image is available by executing the command:

docker images



Starting from this basic image, you will have to prepare the containers that will run Cassandra nodes. This can be done in two ways. The first method is to update the basic image and save it as a new one, which will serve as a base for the containers. The second is to use a Dockerfile to update and create a new image automatically. In this article, we will explore both ways, starting with the first.

Run the first container from the basic image, assigning it a static IP in the network that you have created:

docker run --net mynet123 --ip 172.18.0.22 -it docker.io/dongjoon/ubuntu17.04-jdk8 /bin/bash

You should be in a bash shell inside the container:



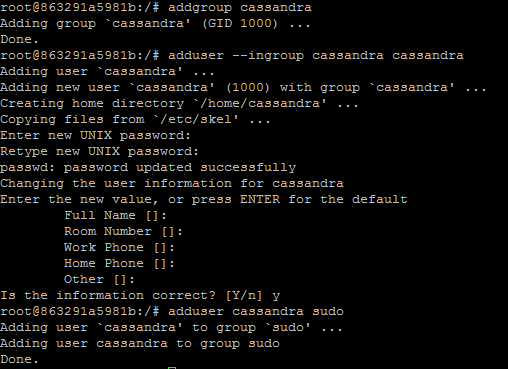
Add the Cassandra user and group (you might have to install the sudo program first):

sudo addgroup cassandra

sudo adduser --ingroup cassandra cassandra

sudo adduser cassandra sudo

You don’t have to setup a password for the user cassandra.

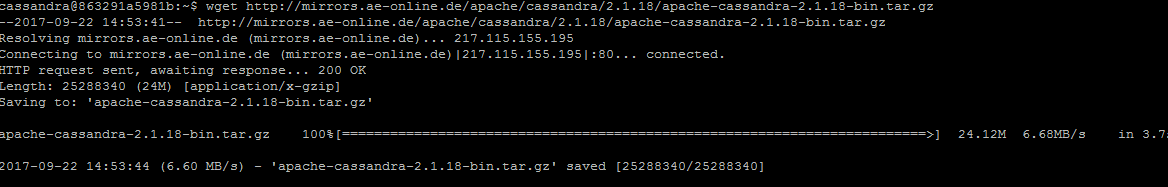


Switch to cassandra user:

su – cassandra

Then download Cassandra:

wget http://mirrors.ae-online.de/apache/cassandra/2.1.18/apache-cassandra-2.1.18-bin.tar.gz



Unpack and install Cassandra:

sudo tar -xvf /home/cassandra/apache-cassandra-2.1.18-bin.tar.gz -C /usr/local

cd /usr/local

sudo mv apache-cassandra-2.1.18/ cassandra-2.1.18

sudo ln -s /usr/local/cassandra-2.1.18/ /usr/local/Cassandra

sudo chown -R cassandra:cassandra /usr/local/cassandra-2.1.18

cd

Add Cassandra environment variables to .profile (you might have to install vim first):

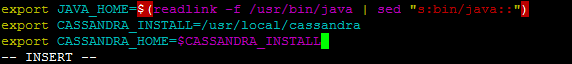
vim ~/.profile

Add the following lines at the end of the file then save it:

export JAVA\_HOME=$(readlink -f /usr/bin/java | sed "s:bin/java::")

export CASSANDRA\_INSTALL=/usr/local/cassandra

export CASSANDRA\_HOME=$CASSANDRA\_INSTALL



Reload these environment variables:

source ~/.profile

The final step is to edit Cassandra’s configuration file:

vim $CASSANDRA\_HOME/conf/cassandra.yaml

You will have to find and modify the following lines:

cluster\_name: 'MyFirstCluster'

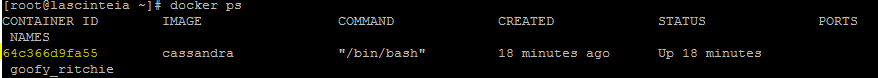
seeds: "172.18.0.22, 172.18.0.23"

listen\_address: 172.18.0.22

rpc\_address: 172.18.0.22

The basic image that you’ll use for all of the Cassandra containers is now ready. Press CTRL+P then CTRL+Q to exit the container without stopping it. You should be back to your system, you can run the command bellow to make sure that your container is still running:

docker ps

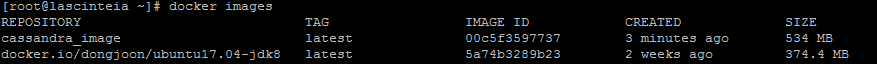


You will need the container ID for the next step. It is now the time to save the custom container as a separate image, to use it in the future (let’s name it “cassandra\_image”:

sudo docker commit CONTAINER\_ID cassandra\_image

Check if the new image is available for use:

docker images



You can now stop and delete the container:

docker stop CONTAINER\_ID

docker rm CONTAINER\_ID

It is time to start the three nodes and configure them:

docker run --net mynet123 --ip 172.18.0.22 -it cassandra\_image /bin/bash

The first one is already configured, so we only have to run the following command to start the service:

$CASSANDRA\_HOME/bin/cassandra -f

The final lines should look like this:



Exit the container with CTRL+P then CTRL+Q and start the next one:

docker run --net mynet123 --ip 172.18.0.23 -it cassandra\_image /bin/bash

You will have to edit two lines in the configuration file:

vim $CASSANDRA\_HOME/conf/cassandra.yaml

Modify these two lines to point to the static IP of this container:

listen\_address: 172.18.0.23

rpc\_address: 172.18.0.23

Then you can start the service here as well:

$CASSANDRA\_HOME/bin/cassandra -f

Exit the container with CTRL+P then CTRL+Q and start the last node:

docker run --net mynet123 --ip 172.18.0.24 -it cassandra\_image /bin/bash

Again, modify these two lines to point to the static IP of this container:

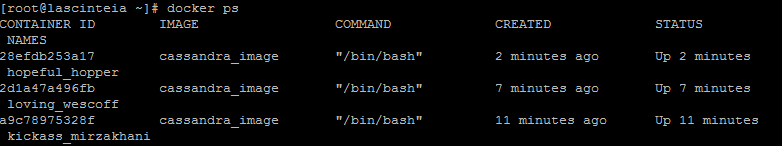
listen\_address: 172.18.0.24

rpc\_address: 172.18.0.24

Then you can start the service here as well:

$CASSANDRA\_HOME/bin/cassandra -f

Exit the container with CTRL+P then CTRL+Q and check if all of them are up:



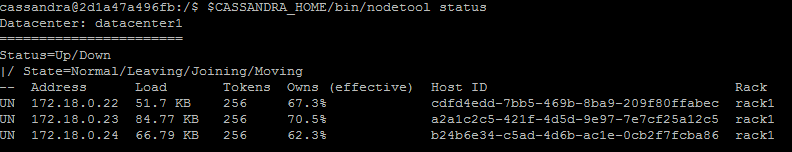
You can now enter any of the three containers to test the status of the nodes:

docker exec -it [container-id] bash

Run the following command:

$CASSANDRA\_HOME/bin/nodetool status

The result should look like this:



Congratulations, your Cassandra cluster is up and running.

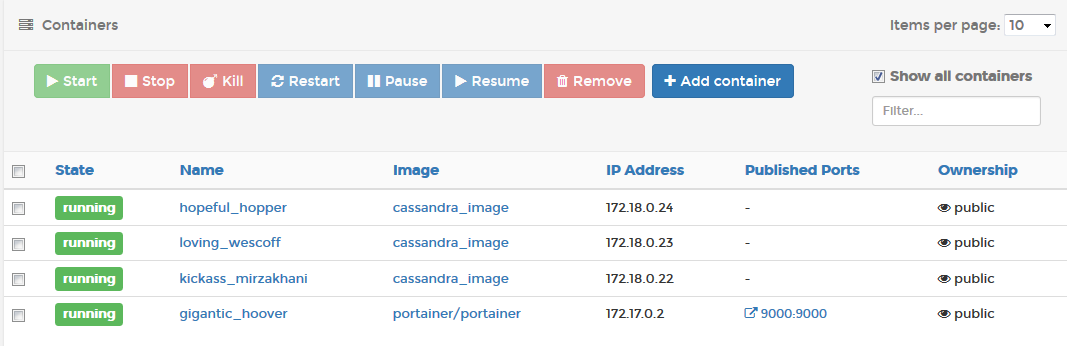
For easier management of Docker containers, you can install a simple tool named Portainer. It runs straight from a container, so it’s very fast and convenient. You only have to run two commands:

sudo docker pull portainer/portainer

sudo docker run -d -p 9000:9000 -v /var/run/docker.sock:/var/run/docker.sock portainer/portainer

That’s it, now you can access the following link and setup your admin password:  
<http://IP_Address:9000/>

You will see the 3 Cassandra containers running, as well as the one that hosts Portainer. Feel free to explore the options of this very powerful tool.



That was all, folks! I hope someone will enjoy my very first tutorial on Github ☺