Guide for Docker Image with Docker on Lab Cluster

Shangyu Xie

December 10, 2016

1 Prepration

1.1 Background

Docker image works to initialize container, which can be modified by the specific environment. There are two methods to modify or create the new image.

- Docker Commit, can be used to save the running container into a new image. As is known for docker starter, docker container consists of different level file layer on Docker Daemon running on the host system, while saving the container means snapshot the image, which can save the changes for the base image.
- Docker Build, this command builds an image from a Dockerfile and some contexts file, which usually some modified files for the base image. Dockerfile can include a series of command defined to execute to create an new image (file layer). I take this method.

1.2 Note

a. Docker Build Command. docker build -t REPOSITORY_ NAME .
Do not forget the DOT in the end, which indicates that the current directory is the working directory. So you must ender the Dockerfile's directory to build the image.

b. Dockerfile is a default name, so you cannot rename it or it goes wrong.

c. During the build process, you can see the flowing command output, which can help you to debug your image file.

2 Implemention

2.1 Idea

The goal is to enable the container can transfer the file (here is the system logs or hadoop logs) to the host system automatically. For simple and test, I choose to transfer a log file of Hadoop Datanode and netcat to do it.

The general idea is keeping the host system listening the specific port (for example, 1234), while container runs the nc command to transfer the file. Because the system keeps listening on the port all the time, so as long as the container transfers the file into the port, we can get the files.

For image, it is simple to add the command nc into the one of following directories or files.

- /etc/booststrap.sh
- /etc/rc.local
- /etc/profile.d/

All of above refer to 2.4, 2.5, 2.6.

2.2 Install the netcat

Since I did the test for hadoop docker container executor, so I took the sequenceiq/hadoop-docker:2.7.1 image file as the base. First run the image, and there is no netcat in the CentOS system.

Add the command $ *RUN yum install nc* into the Docker file.

2.3 Add Execution nc file

Edit the nc.sh file to run the nc command.
$ rm -r /usr/local/hadoop/logs/*-fb16*
$ nc 172.17.0.1 1234 < /usr/local/hadoop/logs/hadoop-root-datanode-*.log

Note

- first command is to delete the original log files left when creating the image, the container name is not changed.
- second command is the nc command to transfer the file to the port and IP address. Here for host system is 172.17.0.1 by default.

Add the following commands into the Dockerfile.
$ ADD nc.sh /usr/local/nc.sh
$ RUN chmod +x /usr/local/nc.sh

2.4 Add the /etc/bootstrap.sh file

Edit the bootstrap.sh.
$ /usr/local/nc.sh

2.5 Add(modify) rc.local file

Edit the rc.local file and add the
$ rm -r /usr/local/hadoop/logs/*-fb16*
$ nc 172.17.0.1 1234 < /usr/local/hadoop/logs/hadoop-root-datanode-*.log

Add the following commands into Dockerfile. $ ADD rc.local /etc/rc.local
$ RUN chown root:root /etc/rc.local

2.6 Add the /etc/profile.d

Add the following command into the Dockerfile.
$ ADD nc.sh /etc/profile.d/nc.sh
$ RUN chmod +x /etc/profile.d/nc.sh

2.7 Build the Image

After editing the following context files, while other files remain unchanged.

Run docker build command to create the file.
$ docker build -t shanxie/docker-base:v4 .
which could take a bit long. The name is my repository's name and tag is v4. This can enable to upload the image to the public repository.

You can choose to push the newly build image to the repository via *docker push* command.
First, login to the docker.

$ docker login
Input username: shanxie and password: xie690515
Then push the image.
$ docker push shanxie/docker-base:v4
You can also can create your own repository, while the image name need to be changed YOUR_ USERNAME/SOMTHING_ YOU_ LIKE.

3 Test

Test the new image by testing it on local and with hadoop respectively. Remember to run the netcat command on host first.

3.1 Edit the receive script

Edit the new script of test.sh.
# !/bin/bash
count=1
while [$count -lt 100];
do
echo -n $ count
echo -n ' '
date
nc -l 1234 > $ count
count=$ ((count+1))
sleep 1
done

Run the sh file.
$ ./test.sh

3.2 Single Container on local machine

Run $ docker run -it shanxie/docker-base:v4 /etc/bootstrap.sh -bash
If it does right, you can see the *1* file in the directory of test.sh on host.

3.3 with Hadoop

From previous result, we know that Hadoop YARN will initialize a bunch of containers for working as long as we configure the yarn-site.xml with docker and run the command with docker image.

Run command.

Teragen.
$ hadoop jar $ HADOOP_ HOME/share/hadoop/mapreduce/hadoop-mapreduce-examples-*.jar teragen -Dmapreduce.map.env="yarn.nodemanager.docker-container-executor.image-name=shanxie/docker-base:v4" -Dyarn.app.mapreduce.am.env= "yarn.nodemanager.docker-container-executor.image-name=shanxie/docker-base:v4" 10000000 teragen/1GB

Terasort.
$ hadoop jar $ HADOOP_ HOME/share/hadoop/mapreduce/hadoop-mapreduce-

examples-*.jar terasort -Dmapreduce.map.env="yarn.nodemanager.docker-container-executor.image-name=shanxie/docker-base:v4" -Dmapreduce.reduce.env="yarn.nodemanager.docker-container-executor.image-name=shanxie/docker-base:v4" -Dyarn.app.mapreduce.am.env="yarn.nodemanager.docker-container-executor.image-name=shanxie/docker-base:v4" terageninput/1GB terasort/1GB

If it does right, you should see the numbered name file in your work directory.