Docker-搭建Hadoop集群

基本步骤

- 准备Linux虚拟机，作为主服务器
- 准备基础images，CentOS7
- 基础images配置好网络，open-ssh，下载好各组件（Java、Scala、Hadoop、zookeeper、HIve、Flume），编写各种启动文件和hosts，准备环境变量，打包成images-env-ready
- 启动三个images-env-ready容器，注意端口和映射，container name，ip
- 先进入slave容器，并启动zookeeper，最后进入master，start-all，如果namenode没有启动，检查日志并单独启动

镜像准备(CentOS和Hadoop集群)

- 拉取CentOS镜像

```
docker pull centos # 尽量选择CentOS7，命令有不同
```

- 基于CentOS构建带有SSH功能的CentOS-SSH

```
1，编辑Dockerfile文件
FROM centos
MAINTAINER dys

RUN yum install -y openssh-server sudo
RUN sed -i 's/UsePAM yes/UsePAM no/g' /etc/ssh/sshd_config
RUN yum  install -y openssh-clients

RUN echo "root:111111" | chpasswd
RUN echo "root   ALL=(ALL)      ALL" >> /etc/sudoers
RUN ssh-keygen -t dsa -f /etc/ssh/ssh_host_dsa_key
RUN ssh-keygen -t rsa -f /etc/ssh/ssh_host_rsa_key

RUN mkdir /var/run/sshd
EXPOSE 22
CMD ["/usr/sbin/sshd", "-D"]
```

```
2，根据Dockerfile构建镜像
```

```
docker build -t='CentOS-SSH' .
```

- 基于CentOS-SSH构建Hadoop镜像

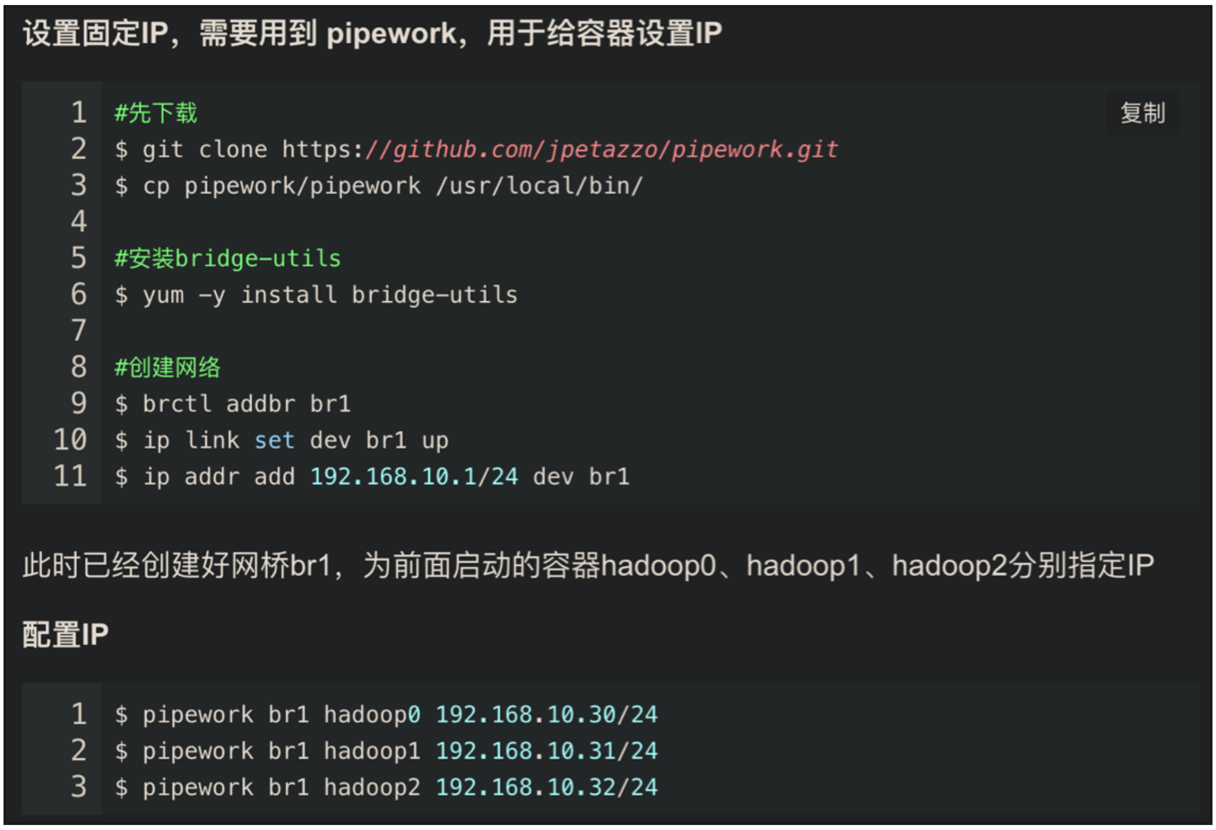
```
1，编辑Dockerfile，添加Java和hadoop压缩包（自动解压到指定目录）
FROM centos7-ssh
ADD jdk-8u101-linux-x64.tar.gz /usr/local/
RUN mv /usr/local/jdk1.8.0_101 /usr/local/jdk1.8
ENV JAVA_HOME /usr/local/jdk1.8
ENV PATH $JAVA_HOME/bin:$PATH

ADD hadoop-2.7.3.tar.gz /usr/local
RUN mv /usr/local/hadoop-2.7.3 /usr/local/hadoop
ENV HADOOP_HOME /usr/local/hadoop
ENV PATH $HADOOP_HOME/bin:$PATH

RUN yum install -y which sudo
```

```
2，构建Hadoop镜像
docker build -t='hadoop' .
```

```
3，开启镜像，构建集群
docker run --name hadoop0 --hostname hadoop0 -d -P -p 50070:50070 -p 8088:8088
hadoop
docker run --name hadoop1 --hostname hadoop1 -d -P hadoop
docker run --name hadoop2 --hostname hadoop2 -d -P hadoop
```

为容器配置网络

- 编辑如果是单机虚拟环境，配置虚拟网桥（Linux和Win）
-


设置固定IP，需要用到 pipework，用于给容器设置IP

```
#先下载
$ git clone https://github.com/jpetazzo/pipework.git
$ cp pipework/pipework /usr/local/bin/

#安装bridge-utils
$ yum -y install bridge-utils

#创建网络
$ brctl addbr br1
$ ip link set dev br1 up
$ ip addr add 192.168.10.1/24 dev br1
```

此时已经创建好网桥br1，为前面启动的容器hadoop0、hadoop1、hadoop2分别指定IP

配置IP

```
$ pipework br1 hadoop0 192.168.10.30/24
$ pipework br1 hadoop1 192.168.10.31/24
$ pipework br1 hadoop2 192.168.10.32/24
```

- 编辑各容器中的 '/etc/hosts'，确保IP和hosts对应
- 设置SSH-KEY，并分发至集群每个容器

```
ssh-keygen
（执行后会有多个输入提示，不用输入任何内容，全部直接回车即可）
ssh-copy-id -i /root/.ssh/id_rsa -p 22 root@hadoop0
ssh-copy-id -i /root/.ssh/id_rsa -p 22 root@hadoop1
ssh-copy-id -i /root/.ssh/id_rsa -p 22 root@hadoop2
```

- 测试SSH

Hadoop配置

- 进入容器的Shell，测试Java和Hadoop环境变量是否正确

`docker exec -it hadoop0 /bin/bash`

- 配置主容器的Hadoop各文件

```
1,core-site
2,HDFS-site
3,mapred-site
4,Yarn-site
5,hadoop-env.sh
```

```
6, slaves
```

- 将Hadoop文件分发到集群各容器，替换原来的文件

集群测试

- 编辑本地文件
- 上传到HDFS
`hadoop fs -put localpath hdfspath`
- 运行mapreduce-demo
- `hadoop jar $HADOOP_HOME/mapreduce/hadoop-mapreduce-examples-2.7.3.jar wordcount hdfspath hdfs-outputpath`

关闭容器

```
docker ps -a
docker stop contianerID
```

zookeeper配置

- master和slave的myid不同，需要在启动文件里写明

Spark配置

```
export SPARK_MASTER_IP=master
export SPARK_WORKER_MEMORY=128m
export JAVA_HOME=your JAVA_HOME
export SCALA_HOME= your SCALA_HOME
export SPARK_HOME= your SPARK_HOME
export HADOOP_CONF_DIR=
export SPARK_LIBRARY_PATH=$SPARK_HOME/lib
export SCALA_LIBRARY_PATH=$SPARK_LIBRARY_PATH
export SPARK_WORKER_CORES=1
export SPARK_WORKER_INSTANCES=1
export SPARK_MASTER_PORT=7077
```

- 配置slaves

```
master
slave1
slave2
```

启动脚本

- 在主节点内编写四个启动脚本，分别启动master，slave1，slave2，stop-master
- run_master.sh：启动RM、Spark-master、worker，NN，JN，zookeeper-QuorumPeerMain，配置myid和hosts「docker容器启动时，hosts清除，即便是在提交images时定义也不行」

```
#!/bin/bash
#清空hosts文件信息
echo> /etc/hosts
#配置主机的host
echo 172.17.0.1 host >> /etc/hosts
echo 172.17.0.2 master >> /etc/hosts
echo 172.17.0.3 slave1 >> /etc/hosts
echo 172.17.0.4 slave2 >> /etc/hosts

#配置 master 节点的 zookeeper 的 server id
echo 1 > /root/soft/apache/zookeeper/zookeeper-3.4.9/tmp/myid

zkServer.sh start

hadoop-daemons.sh start journalnode
hdfs namenode -format
hdfs zkfc -formatZK

start-dfs.sh
start-yarn.sh
start-all.sh
```

- run-slaves：启动zookeeper-QuorumPeerMain、DN、JN、NM

```
#!/bin/bash
```

```
#清空hosts文件信息
echo> /etc/hosts
#配置主机的host
echo 172.17.0.1 host >> /etc/hosts
echo 172.17.0.2 master >> /etc/hosts
echo 172.17.0.3 slave1 >> /etc/hosts
echo 172.17.0.4 slave2 >> /etc/hosts

#配置 slave 节点的 zookeeper 的 server id
# 数字每个slave需要不同
echo 3 > /root/soft/apache/zookeeper/zookeeper-3.4.9/tmp/myid

zkServer.sh start
```

- stop-master.sh

```
#!/bin/bash
zkServer.sh stop
hadoop-daemons.sh stop journalnode
stop-dfs.sh
stop-yarn.sh
stop-all.sh
```

- 集群启动需要先slave再master，slave先启动了zookeeper