链滴

Mesos+ZooKeeper+Docker+Marathon 集群部署

作者：tutouxiaozhang

原文链接：https://ld246.com/article/1600166704046

来源网站：链滴

<img src="https://ld246.com/images/img-loading.svg" alt="" data-src="https://b3logfile
com/bing/20180224.jpg?imageView2/1/w/960/h/540/interlace/1/q/100">
<h2 id="Mesos介绍">Mesos 介绍</h2>
Mesos 是 apache 旗下的一个项目，是一个分布式资源管理系统，类似于一个操作系统，来管
一台机器的硬件，但是他是分布式的，所以也可以称作分布式操作系统。
官网：<a href="https://ld246.com/forward?goto=http%3A%2F%2Fmesos.apache.org%2F
target="_blank" rel="nofollow ugc">http://mesos.apache.org/
mesos 支持各种各样，上百种任务调度框架，官网推荐 Framework：<a href="https://ld246.c
m/forward?goto=http%3A%2F%2Fmesos.apache.org%2Fdocumentation%2Flatest%2Fframe
orks%2F" target="_blank" rel="nofollow ugc">http://mesos.apache.org/documentation/latest
frameworks/
这些框架包含了 长服务、大数据、分布式定时任务等等
他的两层架构设计很灵活，你也可以自己开发一个属于自己的任务调度系统，k8s 也可以作为 me
os 的一个 framework。
<h4 id="基本术语解释">基本术语解释</h4>

Mesos master：负责管理各个 Framework 和 Slave，各个节点的 mesos slave 将资源上报给
esos-master，并将资源分配个 Framework
Mesos slave ： 负责管理本节点上的各个 Mesos Task，为 Executor 分配资源
Framework ： 计算框架，如 Marathon 等，可以通过 MesosSchedulerDiver 接入 Mesos
Executor：执行器，在 Mesos Slave 上安装，用于启动计算框架中的 Task。
zookeeper： 注册中心，用于整个集群的高可用，存储 mesos、marathon 的配置信息
Docker： 一种容器解决方案，mesos 本身也自带了 container 解决方案，我们这里建议用 dock
r
Marathon：一种长服务任务调度系统，mesos 可支持多种任务调度框架，marathon 只是其中
种。

<h4 id="Mesos功能">Mesos 功能</h4>

使用 ZooKeeper 的容错复制主服务器
可扩展到数千个节点
使用 Linux 容器隔离任务
多资源调度（内存和 CPU 感知）
用于开发新并行应用程序的 Java，Python 和 C ++ API
用于查看群集状态的 Web UI

<h2 id="环境规划-">环境规划：</h2>
192.168.149.129 mesos1
192.168.149.130 mesos2
192.168.149.131 mesos3
192.168.149.132 mesos4 作为 slave 存在
<h4 id="高可用">高可用</h4>

zookeeper 作为注册配置中心，来存储 mesos，marathon 的配置，以及保证整个集群的高可
。一般来说部署奇数个节点，只要集群数量过半，集群还是可用的，比如 3 个节点可挂掉 1 个，5 个
点可挂掉 2 个。
mesos-master 至少部署 3 个节点，并配置 quorum 参数，保证 master 的高可用
mesos-slave 是整个计算和部署资源池，可部署成千上万个节点
marathon 部署至少三个节点，通过注册 zookeeper 选举，保证高可用

<h4 id="基础环境准备-">基础环境准备：</h4>
防火墙及 SELINUX 安全策略确认已经关闭
#静态解析配置
cat <> /etc/hosts

192.168.149.129 mesos1
192.168.149.130 mesos2
192.168.149.131 mesos3
192.168.149.132 mesos4
EOF
#yum 源准备，依赖更新
wget -O /etc/yum.repos.d/cetos-base.repo http://mirrors.aliyun.com/repo/Centos-7.repo
yum-config-manager --add-repo http://mirrors.aliyun.com/docker-ce/linux/centos/docker-ce.repo</>
yum install http://repos.mesosphere.io/el/7/noarch/RPMS/mesosphere-el-repo-7-1 noarch.rpm
yum install -y yum-utils device-mapper-persistent-data lvm2 lrzsz
<h2 id="mesos1-3-master操作">mesos1-3 master 操作</h2>
yum -y install mesos docker-ce marathon java-1.8.0-openjdk.x86_64
wget https://mirrors.tuna.tsinghua.edu.cn/apache/zookeeper/zookeeper-3.6.2/apace-zookeeper-3.6.2-bin.tar.gz
tar xf apache-zookeeper-3.6.2-bin.tar.gz
mv apache-zookeeper-3.6.2-bin /usr/local/zookeeper
cd /usr/local/zookeeper/
mkdir -p dataDir
mkdir dataLogDir
cd conf/
mv zoo_sample.cfg zoo.cfg
vim zoo.cfg
dataDir=/usr/local/zookeeper/dataDir
dataLogDir=/usr/local/zookeeper/dataLogDir
admin.serverPort=8888 #此处有坑，如果是 3.5 以上版本 zk，他默认会占用 marathon 的 8080 端口，在这里修改掉
#末尾添加，三个集群的 id 配置, 这里的主机名替换成真实的主机名
server.0=mesos1:2888:3888
server.1=mesos2:2888:3888
server.2=mesos3:2888:3888
<hr>
echo 0 > /usr/local/zookeeper/dataDir/myid #此处修改 myid 文件为自己的配置文件定义的 id
scp -r /usr/local/zookeeper/ root@mesos2:/usr/local/zookeeper
scp -r /usr/local/zookeeper/ root@mesos3:/usr/local/zookeeper
#分别修改 myid 文件为自己的配置文件中定义的 id
[root@mesos3 conf]# cd ..
[root@mesos3 zookeeper]# pwd
/usr/local/zookeeper #当前路径
[root@mesos1 zookeeper]# ./bin/zkServer.sh start #启动 zookeepe
[root@mesos1 zookeeper]# ./bin/zkServer.sh status #查看 leader 态
建议：生产环境下，建议 mesos-master、zookeeper 和 mesos-slave 分开部署，测试环境可用。

注意：zookeeper 需要部署成集群方式
<h4 id="zookeeper配置">zookeeper 配置</h4>
集群方式配置时，节点数量需要为奇数，比如 3 或者 5，测试环境建议 3 个节点，一般生产环境
议 5 个节点，节点过多反而影响整体的性能，因为 zookeeper 的数据多节点同步是同步进行的。</p

/var/lib/zookeeper/myid 三个节点的整个配置必须和如下配置中指定的 id 一致。这里分别为
、2、3
主配置文件：/etc/zookeeper/zoo.cfg
三个节点上都需要配置相同的配置文件
<h2 id="mesos4-slave操作">mesos4 slave 操作</h2>
yum -y install mesos docker-ce java-1.8.0-openjdk.x86_64
<h2 id="zk配置-">zk 配置：</h2>
mesos 主节点的服务器都需要配置编辑 /etc/mesos/zk 内容如下
<table>
<thead>
<tr>
<th>,mesos2:2181,mesos3:2181/mesos</th>
</tr>
</thead>
</table>
<h2 id="mesos-master配置">mesos master 配置</h2>
修改/etc/mesos-master/quorum 文件内容为 2，整个数字表示 不小于 master 的数量除以 2
来保证整个集群的高可用。
#编辑配置文件/etc/mesos-master/hostname

echo 192.168.149.129 >> /etc/mesos-master/hostname #分别输入本机 IP
<h2 id="指定docker作为容器方案mesos-slave配置">指定 docker 作为容器方案 mesos slave 配
</h2>
vim /etc/mesos-slave/containerizers
<table>
<thead>
<tr>
<th>docker #添加内容为 docker</th>
</tr>
</thead>
</table>
<h2 id="marathon配置-mesos主节点配置">marathon 配置；mesos 主节点配置</h2>
指定 mesos 在 zk 中的位置/mesos，以及 marathon 自己的 zk 位置/marathon
**vim /etc/default/marathon ** #编辑配置文件
<table>
<thead>
<tr>
<th>
MARATHON_MASTER="zk://mesos1:2181,mesos2:2181,mesos3:2181/mesos"MAR
THON_ZK=",mesos2:2181,mesos3:2181/marathon"</th>
</tr>
</thead>
</table>
<h2 id="Docker配置">Docker 配置</h2>
修改 systemctl 启动脚本， ExecStart 修改如下，主要是关闭 iptables，其他的配置可选
vim /usr/lib/systemd/system/docker.service
<table>
<thead>
<tr>
<th>
ExecStart=/usr/bin/dockerd -H --containerd=/run/containerd/containerd.sock --ip
ables=false --log-driver=syslog --log-opt syslog-facility=local6 --log-opt tag="{{.ImageName

}/{{.Name}}/{{.ID}}" --insecure-registry mesos1:5000</th>
</tr>
</thead>
</table>
systemctl daemon-reload #重载配置
<h4 id="启动docker">启动 docker</h4>
<table>
<thead>
<tr>
<th>systemctl start docker</th>
</tr>
</thead>
</table>
#启动 mesos-master 和 mesos-slave、marathon，需要先启动 docker
<h2 id="Mesos1-3---master机器启动服务">Mesos1-3 master 机器启动服务</h2>
<table>
<thead>
<tr>
<th>systemctl start mesos-master mesos-slave marathon</th>
</tr>
</thead>
</table>
<h2 id="Mesos4-slave启动服务">Mesos4 slave 启动服务</h2>
<table>
<thead>
<tr>
<th>systemctl start mesos-slave</th>
</tr>
</thead>
</table>
<h2 id="UI页面">UI 页面</h2>
marathon：<a href="https://ld246.com/forward?goto=http%3A%2F%2F192.168.149.12
%3A8080" target="_blank" rel="nofollow ugc">http://192.168.149.129
mesos：<a href="https://ld246.com/forward?goto=http%3A%2F%2F192.168.149.129%3
5050" target="_blank" rel="nofollow ugc">http://192.168.149.129:5050
<h2 id="mesos-slave-日志切割">mesos-slave 日志切割</h2>
在/etc/mesos-slave/下配置模块文件 container_logger 和 modules
<a href="https://ld246.com/forward?goto=mailto%3Aroot%40mesos4" target="_blank"
el="nofollow ugc">root@mesos4 /#touch /etc/mesos-slave/{container_logger
modules}
<a href="https://ld246.com/forward?goto=mailto%3Aroot%40mesos4" target="_blank"
el="nofollow ugc">root@mesos4 latest# vim /etc/mesos-slave/container_logg
r
org_apache_mesos_LogrotateContainerLogger #添加内容
<a href="https://ld246.com/forward?goto=mailto%3Aroot%40mesos4" target="_blank"
el="nofollow ugc">root@mesos4 latest# vim /etc/mesos-slave/modules</stro
g>
/etc/mesos-slave-modules.json #添加内容
<a href="https://ld246.com/forward?goto=mailto%3Aroot%40mesos4" target="_blank"
el="nofollow ugc">root@mesos4 latest# vim /etc/mesos-slave-modules.json<
strong> #以下为添加内容
{
"libraries": [
{

"file": "/usr/lib/liblogrotate_container_logger.so",
"modules": [
{
"name": "org_apache_mesos_LogrotateContainerLogger",
"parameters": [
{
"key": "launcher_dir",
"value": "/usr/libexec/mesos",
"key": "max_stdout_size",
"value": "5MB"
}
]
}
]
}
]
}
<h4 id="重启mesos-slave">重启 mesos-slave</h4>
<a href="https://ld246.com/forward?goto=mailto%3Aroot%40test41pub.tsht3.mc.ops" t
rget="_blank" rel="nofollow ugc">root@test41pub.tsht3.mc.ops latest# syste
ctl restart mesos-slave
在沙箱目录下生成这几个文件
<a href="https://ld246.com/forward?goto=mailto%3Aroot%40test41pub.tsht3.mc.ops" targe
="_blank" rel="nofollow ugc">root@test41pub.tsht3.mc.ops latest# ll</stron
>
<pre><code class="language-total highlight-chroma"><span c
ass="highlight-cl">-rw-r--r-- 1 root root   300 Jul 10 17:25 docker.tar.gz
-rw-r--r-- 1 root r
ot   2638 Jul 10 17:25 stderr
-rw-r--r-- 1 root r
ot   291 Jul 10 17:25 stderr.logrotate.conf
-rw-r--r-- 1 root r
ot 230253 Jul 10 18:11 stdout
-rw-r--r-- 1 root r
ot   290 Jul 10 17:25 stdout.logrotate.conf
-rw-r--r-- 1 root r
ot   321 Jul 10 17:47 stdout.logrotate.state
</code></pre>