

Mesos

将服务器集群的资源抽象进行统一管理，例如任务在集群之间的分配调度和故障转移等，目标是想要实现一个分布式内核

安装步骤

- 64bit Linux
- kernel >=3.10
- gcc >=4.8.1
- 1-4步骤需要在集群中每台机器运行

1. 下载mesos

```
wget http://www.apache.org/dist/mesos/1.1.0/mesos-1.1.0.tar.gz
tar zxvf mesos-1.1.0.tar.gz
```

2. 安装依赖

- 获取maven的repo

```
wget http://repos.fedorapeople.org/repos/dchen/apache-maven/epel-apache-maven.repo -O /etc/yum.repos.d/epel-apache-maven.repo
```

- 安装epel repo

```
sudo yum install -y epel-release
```

- 获取svn repo

```
vim /etc/yum.repos.d/wandisco-svn.repo
```

添加如下内容：

[WANDiscoSVN]

name=WANDisco SVN Repo 1.9

enabled=1

baseurl=<http://opensource.wandisco.com/centos/7/svn-1.9/RPMS//>

gpgcheck=1

gpgkey=<http://opensource.wandisco.com/RPM-GPG-KEY-WANDisco>

- 更新systemd

```
sudo yum update systemd
```

- 安装development tools

```
sudo yum groupinstall -y "Development Tools"
```

- 安装依赖

```
sudo yum install -y apache-maven python-devel java-1.8.0-openjdk-devel z
lib-devel libcurl-devel openssl-devel cyrus-sasl-devel cyrus-sasl-md5 ap
r-devel subversion-devel apr-util-devel
```

3. 编译mesos

```
mkdir mesos
cd mesos-1.1.0
mkdir build
cd build
../configure --prefix=/home/null/mesos #此步进行依赖检测和相关配置
make -j10 #! ! 此过程比较费时!!
make install -j10
```

4. 集群间设置SSH互信,默认系统安装有ssh

ssh-keygen -t rsa #rsa加密方式生成公私钥，默认存放位置~/.ssh/

```
[null@master-cent7-2 mesos]$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/null/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/null/.ssh/id_rsa.
Your public key has been saved in /home/null/.ssh/id_rsa.pub.
The key fingerprint is:
19:00:25:62:5c:c4:05:38:29:25:9d:8c:d0:f8:33:01 null@master-cent7-2
The key's randomart image is:
+--[ RSA 2048]-----+
|EX+B*=+|
|+o@... |
| 0 0   |
| +     0|
| 0     S|
+-----+

```

将公钥追加到authorized_keys，并设置权限

```
cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
```

```
chmod 600 ~/.ssh/*
```

依次将各主机公钥复制到各个主机，并追加到authorized_keys既可。

注意，以下两步实际没有执行成功，请直接跳过

1. 相关配置，配置文件在-prefix指定的目录/etc/mesos下，官方默认带有4个实例文件:*.template

- master主机配置

```
cd /home/null/mesos
vim etc/mesos/masters #列出主结点的主机名或者IP地址，一个地址一行
vim etc/mesos/slaves #列出从结点的主机名或者IP地址，一个地址一行
mv etc/mesos/mesos-master-env.sh.template etc/mesos/mesos-master-env.sh
vim etc/mesos/mesos-master-env.sh #参照如下修改
```

- slave主机配置

```
cd /home/null/mesos
mv etc/mesos/mesos-slave-env.sh.template etc/mesos/mesos-slave-env.sh
vim etc/mesos/mesos-slave-env.sh # 参照如下修改
```

slave配置示例：

```
# This file contains environment variables that are passed to mesos-agent.
# To get a description of all options run mesos-agent --help; any option
# supported as a command-line option is also supported as an environment
# variable.

# You must at least set MESOS_master.

# The mesos master URL to contact. Should be host:port for
# non-ZooKeeper based masters, otherwise a zk:// or file:// URL.
export MESOS_master=192.168.125.171:8081 #设置master地址，端口和master设置的端口相对应

# Other options you're likely to want to set:
export MESOS_log_dir=/home/null/mesos/log #log存放位置，确保该文件夹存在
export MESOS_work_dir=/home/null/mesos/run #运行相关日志存放位置，确保该文件夹存在
# export MESOS_isolation=cgroup
```

2. 启动集群

在master主机上运行：

```
cd /home/null/mesos
./sbin/
```

• 启动mesos集群

master主机：

在编译的build目录下：

```
nohup bin/mesos-master.sh --ip=0.0.0.0 --port=8081 --work_dir=/home/null/mesos/work &
```

端口默认是5050，确保work_dir目录存在

agents主机：

在编译的build目录下：

```
nohup bin/mesos-agent.sh --master=192.168.125.171:8081 --work_dir=/home/null/mesos/work &
```

--master指定master主机，确保work_dir目录存在，可能出现错误：“Failed to initialize systemd: Failed to create systemd slice ‘mesos_executors.slice’.”

解决方法：添加参数--no-systemd_enable_support

启动之后可以通过浏览器访问master UI,效果如下：

The screenshot displays the Mesos Master web interface. At the top, there's a navigation bar with 'Mesos', 'Frameworks', 'Agents', and 'Offers'. Below this, a header bar shows 'Master' and a unique ID 'dd78106d-c10c-4072-a455-807687030247'. The main content area is divided into several sections:

- Cluster Information:** Located on the left, it shows 'Cluster: (Unnamed)', 'Server: 192.168.125.171:8081', 'Version: 1.1.0', and timestamps for 'Built', 'Started', and 'Elected'.
- Agents:** A table showing the status of agents. It lists 'Activated' as 2 and 'Deactivated' as 0.
- Tasks:** A table showing the status of tasks. It lists 'Staging' as 0, 'Starting' as 0, 'Running' as 2, 'Killing' as 0, 'Finished' as 0, 'Killed' as 0, 'Failed' as 0, 'Lost' as 0, and 'Orphan' as 0.
- Active Tasks:** A table with columns 'ID', 'Name', 'State', 'Started', and 'Host'. It shows two tasks: 'Task 1' (ID 1, RUNNING, started 2016-12-22T13:29:43+0800, host master-cent7-2) and 'Task 0' (ID 0, RUNNING, started 2016-12-22T13:29:43+0800, host master-cent7-3). Each task has a 'Sandbox' link.
- Completed Tasks:** A table with columns 'ID', 'Name', 'State', 'Started', 'Stopped', and 'Host'. It shows 'No completed tasks.'
- Orphan Tasks:** A table with columns 'ID', 'Name', 'State', 'Started', 'Stopped', and 'Host'. It shows 'No orphan tasks.'