系统环境与IP

- Master: Ubunut14.04 IP：192.168.1.158
- Minion1: Ubuntu14.04 IP：192.168.1.159
- Minion2: Ubuntu14.04 IP：192.168.1.157

> *整个环境在VM下完成，可用master克隆出两个子机*

MASTER的处理

1. 安装VIM： `sudo apt-get install vim -y`
2. 安装SSH： `sudo apt-get install openssh-server`
3. 启动SSH： `sudo service ssh start`
4. 安装GIT： `sudo apt-get install git -y`
5. 克隆kubernets： `git clone --depth 1 ttps://github.com/kubernetes/kubernetes.git`
6. 安装docker： `wget -qO- https://get.docker.com/ | sh`

随后便可进行对MASTER的克隆

MASTER上的kubernets的操作

1. 确定默认参数： `sudo vim /home/administrator/kubernetes/cluster/ubuntu/config-default.sh`
2. 修改内容 `export nodes="vcap@10.10.103.250 vcap@10.10.103.162 vcap@10.10.103.223"` 为 `export nodes=${nodes:-"administrator@192.168.1.158 administrator@192.168.1.159 administrator@192.168.1.157"}` 其中第一个node为同时为master和node的192.168.1.158；
3. roles按顺序定义机器角色， `ai` 表示master和node， `a` 表示master， `i` 表示node，*本例子使用两个node，所以是* `ai i i`；
4. NUM_NODES为定义的nodes的总个数；（其他参数暂不详细记载）

安装kubernets之前的准备：

由于安装kubernets需要master对其他nodes进行scp操作，则如果安装了ssh，仍然需要输入密码。

无密码通讯

例如master(192.168.1.158)对node(192.168.1.159)进行通讯： 在192.168.1.158上：

1. 进入root权限： `sudo su`
2. 进入root文件夹下： `cd ~`
3. 生成ssh秘钥： `ssh-keygen -t rsa -C "271802559@qq.com"` ，其中后面的邮箱可随意填写，随后的提示均回车跳过
4. 随后在 `/root/.ssh/` 下生成三个文件，其中的 `id_rsa.pub` 内容如下：

```
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDNFzzCok2AIBrKp8MNnS/g6JCKjbnbrFuQqk9riSWrJQyoEioPaXGJGh4h2g7/9DGILy7NeH
8qS4ZeQpxSAflr2U1HZ
3T4u06wyElO0jpudkV4TwiPUv0yNZyY5Bviz09Xute1k5SZ48ARC2EpnlV6Wp/
1tKTnZPwqEVO1rw82xLKy4xITlFLFif0C/NPwNO8Zp5/
cAQDzXSMC6OtxIJawgVwHrh2ZYOP23Gaq+Dl8znQeWbYQPAgJy8opmQh8e2AAp7lXsXzaI
w1FZAj1pINf+UH5V072f23MmJjPikx40WfH7NBDflBVlRWEtmRlixeZBGlai6zzrThWyKL
Kk4OX 271802559@qq.com
```

在192.168.1.159上：

1. 进入root权限： `sudo su`
2. 进入root文件夹下： `cd ~`
3. 生成ssh秘钥： `ssh-keygen -t rsa -C "271802559@qq.com"`
4. 进入.ssh文件夹下创建新的文件夹： `touch authorized_keys`

5. 加入192.168.1.158上的`id_rsa.pub`的内容： `sudo vim authorized_keys`

在192.168.1.158上测试：

1. 进入root权限： `sudo su`
2. 进行ssh链接： `ssh 192.168.1.159` 内容如下：

```
root@administrator158:/home/administrator# ssh 192.168.1.159
Welcome to Ubuntu 14.04.5 LTS (GNU/Linux 4.2.0-27-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

Last login: Wed Oct 19 11:48:56 2016 from 192.168.1.158
root@administrator159:~#
```

注：*本例子是两个root权限下的通讯， 在administrator用户下直接`ssh 192.168.1.159`是不成功的， 但是`sudo ssh 192.168.1.159`便可以*

修改Ubuntu主机名

由于在VM下进行master的克隆，则需要对新子机进行重新修改主机名：

1. `sudo vim /etc/hostname` 修改为想要的

   ```
   administrator158
   ```

2. `sudo vim /etc/hosts` ，改第二行`127.0.1.1 administrator158`

   ```
   127.0.0.1       localhost
   127.0.1.1       administrator158

   # The following lines are desirable for IPv6 capable hosts
   ::1     ip6-localhost ip6-loopback
   fe00::0 ip6-localnet
   ff00::0 ip6-mcastprefix
   ff02::1 ip6-allnodes
   ff02::2 ip6-allrouters
   ```

3. 重启计算机即可： `sudo reboot`

安装kubernets

1. `cd kubernetes/cluster`
2. `KUBERNETES_PROVIDER=ubuntu ./kube-up.sh` 结果：

```
....
Found 3 node(s).
NAME            STATUS    AGE
192.168.1.157   Ready     1m
192.168.1.158   Ready     2m
192.168.1.159   Ready     2m
Validate output:
NAME                 STATUS    MESSAGE              ERROR
controller-manager   Healthy   ok
scheduler            Healthy   ok
etcd-0               Healthy   {"health": "true"}
Cluster validation succeeded
Done, listing cluster services:

Kubernetes master is running at http://192.168.1.158:8080

To further debug and diagnose cluster problems, use 'kubectl cluster-
info dump'.
```

过程中需要输入很多密码，很是麻烦，不知为何

3. 访问 `http://192.168.1.158:8080` ，结果如下：

```
{
  "paths": [
    "/api",
    "/api/v1",
    "/apis",
    "/apis/apps",
    "/apis/apps/v1alpha1",
    "/apis/authentication.k8s.io",
    "/apis/authentication.k8s.io/v1beta1",
    "/apis/authorization.k8s.io",
    "/apis/authorization.k8s.io/v1beta1",
    "/apis/autoscaling",
    "/apis/autoscaling/v1",
    "/apis/batch",
    "/apis/batch/v1",
    "/apis/batch/v2alpha1",
    "/apis/certificates.k8s.io",
    "/apis/certificates.k8s.io/v1alpha1",
    "/apis/extensions",
    "/apis/extensions/v1beta1",
    "/apis/policy",
    "/apis/policy/v1alpha1",
    "/apis/rbac.authorization.k8s.io",
    "/apis/rbac.authorization.k8s.io/v1alpha1",
    "/apis/storage.k8s.io",
    "/apis/storage.k8s.io/v1beta1",
    "/healthz",
    "/healthz/ping",
    "/logs",
    "/metrics",
    "/swaggerapi/",
    "/ui/",
    "/version"
  ]
}
```

4. 测试

    1. `cd kubernetes/cluster/ubuntu/binaries`

    2. `./kubectl get nodes`

结果：

```
NAME            STATUS    AGE
192.168.1.157   Ready     41m
192.168.1.158   Ready     42m
192.168.1.159   Ready     42m
```

5. 设置kubectl：

   1. `sudo cp kubernets/cluster/ubuntu/binaries/kubectl /usr/local/bin/kubectl`
   2. `sudo chmod +x /usr/local/bin/kubectl`
   3. 输入： `kubectl cluster-info` 内容：

```
Kubernetes master is running at http://192.168.1.158:8080

To further debug and diagnose cluster problems, use 'kubectl cluster-
info dump'.
```

deployments的设置

1. 需要建立docker的私有库，这个在作者的另一篇博文中可以找到。

   > 由于kubernentes对于images的获取，默认是从go.io中下载，而目前在国内该网站已被屏蔽，所以需要建立docker的私有库，方便镜像的管理。同时上传与下载速度也很快。

   对于kubernetes来pull镜像，需要在registry私有库中添加一个镜像： `kubernetes/pause`

```
 root@administrator158:/home/administrator# docker search pause
NAME       DESCRIPTION        STARS     OFFICIAL   AUTOMATED
kubernetes/pause               12
```

   注意此处的pause的tag是latest

   将其放入registry中：

```
 Jians-MacBook-Pro:~ jianchan$ curl http://192.168.1.78:5000/v2/_catalog
 {"repositories":["docker-whale","kube-ui","llll","pause","test_docker"]}
```

2. 改变kubelet配置： 将 `/etc/default/kubelet` 修改成以下内容：

```
 KUBELET_OPTS="--address=0.0.0.0
 --port=10250
 --pod-infra-container-image=192.168.1.78:5000/pause:latest
 --hostname-override=192.168.1.158
 --api-servers=http://192.168.1.158:8080
 --logtostderr=true
 --cluster-dns=192.168.3.10
 --cluster-domain=cluster.local
 --config=
 --allow-privileged=false"
```

   其中的 `--pod-infra-container-image` 是连接私有库中的pause:latest镜像

3. 编写简单的kubetest.yaml文件：

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: kubetest
spec:
  replicas: 3
  template:
    metadata:
      labels:
        app: kubetest
    spec:
      containers:
      - name: kubetest
        image: 192.168.1.78:5000/kube_test
        ports:
        - containerPort: 80
```
 其中：
```

- apiVersion不能乱写
- replicas表示复制多少,该yaml文件中开启了3个
- image表示镜像所放的位置
- 文件里不要出现 `_` 等符号

4. 创建deployments：

   ```
   # kubectl create -f kubetest.yaml --record
   deployment "kubetest" created
   ```

   查看当前的deployments：

   ```
   # kubectl get deployments
   NAME       DESIRED    CURRENT    UP-TO-DATE    AVAILABLE    AGE
   kubetest   3          3          3             1            35s
   ```

   查看当前的pods：

   ```
   # kubectl get pods
   NAME                          READY     STATUS              RESTARTS    AGE
   kubetest-3705389421-aiwna     0/1       Running             0           1m
   kubetest-3705389421-e8rfo     1/1       Running             0           1m
   kubetest-3705389421-hort1     1/1       Running             0           1m
   ```

   由于yaml文件中的 `replicas` 为1

   查看当前pod的详细内容：

   ```
   # kubectl describe pod kubetest-3705389421-hort1
   Name:        kubetest-3705389421-hort1
   Namespace:   default
   Node:        192.168.1.158/192.168.1.158
   Start Time: Thu, 10 Nov 2016 10:01:07 +0800
   Labels:      app=kubetest
           pod-template-hash=3705389421
   Status:      Running
   IP:      172.16.34.2
   Controllers:     ReplicaSet/kubetest-3705389421

   ......
   ```

   可以看到此容器放在 `192.168.1.158` 中。
   ```

在 `192.168.1.158` 中查看容器运行情况：

```
# docker ps -a
CONTAINER ID        IMAGE                              COMMAND               CREATED             STATUS
      PORTS               NAMES
eeee65950de9        192.168.1.78:5000/kube_test        "python /sendlog.py"   4 minutes ago       Up 4 minutes
                          k8s_kubetest.479a355e_kubetest-3705389421-hort1_default_8b7222ec-a6e9-11e6-88ee-000c
29cc749c_7229d4ab
c89b1d0f26f0        192.168.1.78:5000/pause:latest     "/pause"               4 minutes ago       Up 4 minutes
                          k8s_POD.94edfd06_kubetest-3705389421-hort1_default_8b7222ec-a6e9-11e6-88ee-000c29cc7
49c_144f91bd
```

通过 `kebectl describe pod xxxx` ,可以看到其他两个容器分别在157，159两台nodes上。

删除deployments：

```
# kubectl delete deployments/docker-whale
deployment "docker-whale" deleted
```

kubernetes-Dashboard部署

1. dashboard部署文件在 `kubernetes/cluster/addons/dashboard` 中，将 `dashboard-controller.yaml` 中的image地址改为私有库中对应的image： `image: 192.168.1.78:5000/kubernetes-dashboard-amd64:v1.4.2`

2. 其中的 `kubernetes-dashboard-amd64:v1.4.2` 需要pull。 可以通过 `docker search kubernetes-dashboard` 来查找，或者到 `dockerhub` 上查找相对应地image。需要注意tag，以及image的版本。 **pull下来后将其push到私有库中，此时仍然要注意tag问题，要和yaml文件中内容对应**

3. 具体操作：

```
# cd kubernetes/cluster/addons/dashboard

# kubectl create -f ./
```

4. 查看信息：

```
# kubectl --namespace kube-system get po -o wide
NAME                                    READY     STATUS    RESTARTS   AGE       IP              NODE
kubernetes-dashboard-v1.4.2-3tl2n       1/1       Running   0          5m        172.16.34.3     192.168.1.158

# kubectl cluster-info
Kubernetes master is running at http://localhost:8080
kubernetes-dashboard is running at http://localhost:8080/api/v1/proxy/namespaces/kube-system/services/kubernet
es-dashboard

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.

# kubectl get pods --all-namespaces
NAMESPACE     NAME                                   READY     STATUS    RESTARTS   AGE
default       kubetest-3705389421-aiwna              1/1       Running   0          3h
default       kubetest-3705389421-e8rfo              1/1       Running   0          3h
default       kubetest-3705389421-hort1              1/1       Running   0          3h
kube-system   kubernetes-dashboard-v1.4.2-3tl2n      1/1       Running   0          9m
```

5. 通过浏览器 `http://192.168.1.158:8080/ui` 便可查看。

Troubleshooting

1. Node status is NotReady

   1.1 某些节点失败

```
kubectl get nodes

NAME            STATUS     AGE
192.168.1.157   NotReady   42d
192.168.1.158   Ready      42d
192.168.1.159   Ready      42d
```

1.2 查看详细信息

```
kubectl describe node 192.168.1.157

......
Conditions:
Type          Status          LastHeartbeatTime                       LastTransitionTime
Reason                  Message
----          ------          -----------------                       -----------------
------                  -------
OutOfDisk     Unknown         Sat, 28 May 2016 12:56:01 +0000         Sat, 28 May 2016 12:56:41 +0000
NodeStatusUnknown       Kubelet stopped posting node status.
Ready         Unknown         Sat, 28 May 2016 12:56:01 +0000         Sat, 28 May 2016 12:56:41 +0000
NodeStatusUnknown       Kubelet stopped posting node status.
......
```

从中可以看到节点unready的原因是**outofdisk**，从而导致**Kubelet stopped posting node status.** 所以可以查看下 `192.168.1.157` 的容量，其操作系统是ubuntu14.04，可通过 `df` 进行查看：

```
df

Filesystem     1K-blocks     Used Available Use% Mounted on
udev             2008212        4   2008208   1% /dev
tmpfs             403856     3784    400072   1% /run
/dev/sda1       12253360 10108744   1499140  88% /
none                   4        0         4   0% /sys/fs/cgroup
none                5120        0      5120   0% /run/lock
none             2019260      256   2019004   1% /run/shm
none              102400       40    102360   1% /run/user
```

然后通过 `docker rmi image` 来删除一些没用的镜像

1.3 重启kubelet

```
1. ssh administrator@192.168.1.157

2. sudo su

3. /etc/init.d/kubelet restart

stop: Unknown instance:
kubelet start/running, process 59261
```

1.4 查看节点

```
kubectl get nodes

NAME            STATUS    AGE
192.168.1.157   Ready     42d
192.168.1.158   Ready     42d
192.168.1.159   Ready     42d
```

恢复正常

参考文献:

1. kubernets官网关于Ubuntu上的搭建
2. ssh安装 和 scp命令 使用
3. Installing and setting up kubectl
4. 【kubernetes】ubuntu14.04 64位 搭建kubernetes过程
5. Dashboard部署
6. kubernets官网关于Web UI (Dashboard)
7. Application Introspection and Debugging
8.