# 云享家 | Kops on AWS 这也许是你最想要的Kubernetes部署攻略

王逸飞 BespinGlobal 1月3日

**王逸飞**

**资深解决方案交付工程师**

**Bespin Global China**

## What is kops?

kops给自己的定义是**The easiest way to get a production grade Kubernetes cluster up and running**，最简单途径来启动和运行Kubernetes。（但是可能不太符合广告法：））

kops 可以通过命令行创建，销毁，升级，维护生产级，高可用的Kubernetes集群。在AWS提供正式版的支持，GCE正在beta测试，VMware vSphere 在alpha测试，其他平台正在筹划中。

## What we need

**一个AWS账号，Access key ID和Secret access key，需要以下权限：**

```
AmazonEC2FullAccess
AmazonRoute53FullAccess
AmazonS3FullAccess
IAMFullAccess
AmazonVPCFullAccess
```

**一个域名，以便我们可以访问我们的应用和集群**

当然也可以使用gossip作为折中方案，这样的话，就可以免去域名的麻烦，但是在访问应用和集群的时

候，会有其他不便出现。

### 一台能使用命令行工具的终端

可以使macOS，可以使Linux，可以使任何平台的云主机，也可以是windows 的WSL。

## 环境准备

### 安装kops:

**On macOS:**

```
curl -OL https://github.com/kubernetes/kops/releases/download/1.10.0/kops-darwin-amd64
chmod +x kops-darwin-amd64
mv kops-darwin-amd64 /usr/local/bin/kops
# you can also install using Homebrew
brew update && brew install kops
```

**On Linux:**

```
wget https://github.com/kubernetes/kops/releases/download/1.10.0/kops-linux-amd64
chmod +x kops-linux-amd64
mv kops-linux-amd64 /usr/local/bin/kops
```

Release page: https://github.com/kubernetes/kops/releases

### 配置AWS CLI

```
pip install awscli
```

在终端输入aws configure，输入账号的Access key ID 與 Secret access key:

```
[root@ip-172-31-1-161 example-voting-app]# aws configure
AWS Access Key ID [****************CRLQ]:
AWS Secret Access Key [****************nFVN]:
Default region name [us-east-2]:
Default output format [None]:
```

### 安装Kubernetes的命令行工具，kubectl

```
  cat <<EOF > /etc/yum.repos.d/kubernetes.repo
  [kubernetes]
  name=Kubernetes
  baseurl=https://packages.cloud.google.com/yum/repos/kubernetes-el7-
x86_64
  enabled=1
  gpgcheck=1
  repo_gpgcheck=1
  gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg
https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg
  EOF
  yum install -y kubectl
```

## 创建集群

### 在 AWS S3 新增一個 bucket 供 Kubernetes 使用

在AWS控制台创建一个bucket，供Kubernetes使用。创建完毕之后，在终端配置环境变量：

```
export KOPS_STATE_STORE=s3://s3.k8s******
```

S3的地址可以通过AWS CLI查看

```
aws s3 ls
```

为了方便后续的操作，可以把export命令写进bashrc。

### 通过Route53配置域名

如果我们不想使用gossip 来配置我们的Kubernetes，我们需要使用AWS的route53配置我们的集群入口。在AWS控制台Route53页，有**"DNS management"**提供对现有域名的管理和解析，**"Domain registration"** 则提供从 AWS 上申请一个新域名的服务。由于目前有域名可以使用，就跳过了Domain registration这个步骤。

点选**Create Hosted zone**，Domain Name填写自己的域名，例如k8s.xxx.xxx，确认之后会看到AWS提示的NS值，在你域名管理网站下配置NS解析。本次使用的域名是万网管理的，所以去阿里云的控制台配置域名的解析：

| 记录类型 | 主机记录 | 解析线路(isp) | 记录值 |
| --- | --- | --- | --- |
| NS | k8s | 默认 | ns |
| NS | k8s | 默认 | ns- |
| NS | k8s | 默认 | ns- |
| NS | k8s | 默认 | n |

如果选择了gossip而不是通过域名访问，这步就可以跳过了。

为了后续操作的方便，将name添加到环境变量：

```
# 如果使用Route53，NAME是所配置的域名
export NAME=k8s.xxx.xxx
# 如果是gossip，NAME必须以k8s.local结尾，这里我使用了cluster.k8s.local
export NAME=cluster.k8s.local
```

同理，这里可以将环境变量的配置添加到bashrc，方便后续使用。

### 创建secret

创建SSH秘钥和kops secret，用于节点的访问。

```
ssh-keygen -t rsa -P ''ssh-keygen -t rsa -P ''
kops create secret --name ${NAME} sshpublickey admin -i ~/.ssh/id_rsa.pub
```

### 创建集群

**kops提供跨zones的高可用方案，在创建集群时可以配置master数量和所在zones。**

```
# 单master
kops create cluster --zones=us-east-2a ${NAME}
# 多master跨zones高可用
kops create cluster --zones=us-east-2a,us-east-2b --master-count 3 ${NAME}
```

**创建配置完毕之后，kops给出提示，在创建集群之前，可以检查集群的配置文件是否正确**：

```
Suggestions:
 * list clusters with: kops get cluster
 * edit this cluster with: kops edit cluster ${NAME}
 * edit your node instance group: kops edit ig --name=${NAME} nodes
 * edit your master instance group: kops edit ig --name=${NAME} master-us-east-2a

Finally configure your cluster with: kops update cluster cluster.k8s.local --yes
```

**更改和确认无误之后，Update cluster创建集群，kops会自动在AWS创建EC2和VPC等资源，等待集**

**群被拉起**:

```
Suggestions:
 * validate cluster: kops validate cluster
 * list nodes: kubectl get nodes --show-labels
 * ssh to the master: ssh -i ~/.ssh/id_rsa admin@api.cluster.k8s.local
 * the admin user is specific to Debian. If not using Debian please use the appropriate user based on your OS.
 * read about installing addons at: https://github.com/kubernetes/kops/blob/master/docs/addons.md.
```

**集群起来之后，可以验证集群状态**:

```
kops validate cluster
Using cluster from kubectl context: cluster.k8s.local

Validating cluster cluster.k8s.local

INSTANCE GROUPS
NAME                    ROLE    MACHINETYPE     MIN     MAX     SUBNETS
master-us-east-2a       Master  c4.large        1       1       us-east-2a
nodes                   Node    t2.medium       2       2       us-east-2a

NODE STATUS
NAME                                            ROLE    READY
ip-172-20-41-227.us-east-2.compute.internal     node    True
ip-172-20-52-202.us-east-2.compute.internal     node    True
ip-172-20-56-169.us-east-2.compute.internal     master  True
```

**kops会自动地为kubelet配置（~/.kube/config），可以直接使用kubectl检查和管理集群**:

```
kubectl get nodes
NAME                                            STATUS    ROLES     AGE
 VERSION
ip-172-20-41-227.us-east-2.compute.internal   Ready     node      40m
 v1.10.6
ip-172-20-52-202.us-east-2.compute.internal   Ready     node      39m
 v1.10.6
ip-172-20-56-169.us-east-2.compute.internal   Ready     master    40m
 v1.10.6
```

**如果使用域名，可以点开AWS控制台，Route53页面，会有看到kops自动配置的域名解析，可以直接访问api，etcd，node等。**

例如访问api.k8s.xxx.xxx域名，可以查看Kubernetes所有的API。

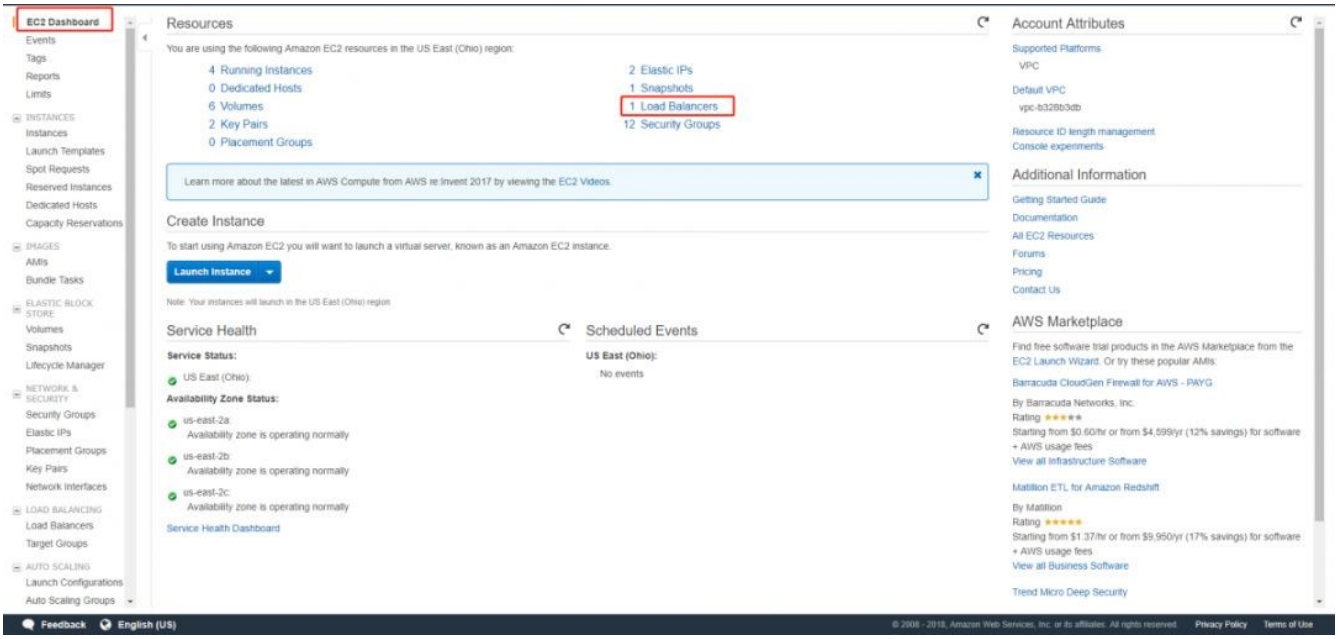
## 使用集群

**部署pod或deployment这件事不是本次的重点，这部分先略过。**

### 使用在AWS上使用Kubernetesservice的LoadBalancer

```
# 这里使用一个简单的docker demo: voting
git clone https://github.com/dockersamples/example-voting-app.git
# 为配合AWS，需要将./k8s-specifications/vote-service.yaml和
# result-service.yaml的type，从NodePort改为LoadBalancer
kubectl create -f k8s-specifications/
```

创建完毕之后，在EC2 控制面板里，可以看到自动生成的LoadBalancer：

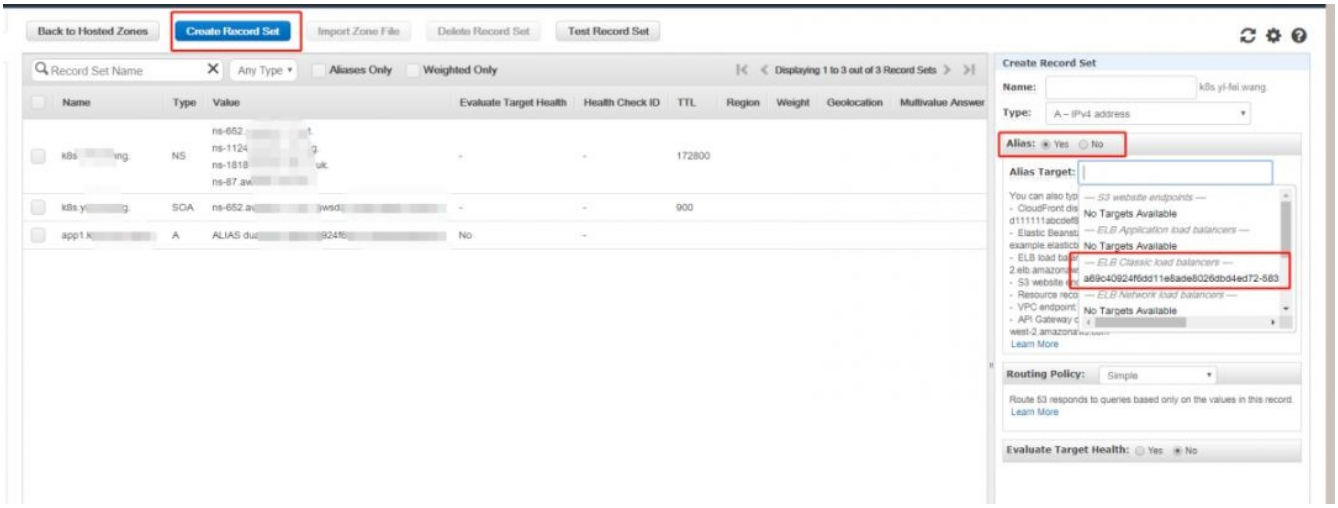



可以在AWS控制台查看Port和Instance等信息。

接着可以在Route53，为服务提供域名解析：

如果再之前配置了域名，可以在Route53页面配置服务的域名解析，便于应用发布；如果是使用gossip，只能通过amazon的域名访问服务了。

在Route53页面点击Create Record Set，配置域名的前缀，Alias勾选之后，Target选择ELB的LoadBalancers里，需要配置域名的服务即可。




删除集群

如果集群使用完毕，可以通过kops清理集群，节约费用。**kops会自动的将master，node以及Load Balancer等删除。**

```
kops delete cluster k8s.xxx.xxx
```

确认无误之后，再输入 --yes，删除集群。

```
kops delete cluster k8s.xxx.xxx --yes
```

**kops作为多种Kubernetes安装工具之一，极大的简化了Kubernetes使用难度，降低Kubernetes的使用门槛。**
与kops相似，还有很多其他工具同样地提供Kubernetes的安装功能，例如亲儿子kubeadm，kubespray。目前看来，没有，也不会存在一种工具完全地符合所有的需求。**kops作为官方推荐的一种部署工具，自动化程度极高，部署简单，配置清晰，可以创建机器，部署docker engine，配置域名，保存配置等，并且支持Kubernetes升级，更关键的是，他提供了AZ级别的高可用方案自动部署。**但是，与此同时，他严重的依赖了公有云的功能：VPC，S3等，有严重的Vendor Lock，不适用于更多场景。
- kubeadm的问题在于，对于高可用支持并非太友好，自身也不支持docker，kubectl的安装，他的目标是配置集群，而非安装。
- kubespray则需要依赖python和ansible，但他兼容了更多的系统，也解绑了平台。
- rke是rancher的部署工具，继承了rancher的一贯风格，所有组件docker化，包括kubelet，rke和kubespray相似，不支持安装docker，也需要ansible的支持。
下期，我们将接着分享其他的工具。

看都看完了，还不点这里试试

阅读原文