**KOPS Cluster using WSL**

Step 1: Install Kops/ Kubectl/ Awscli

Step2: Set-up IAM Roles

Step3: Set-up DNS (R53)

Step4: Set-up S3 Bucket

Step5: Create Cluster

Step6: Clean-up

**Install KOPS:**

$ curl -Lo kops https://github.com/kubernetes/kops/releases/download/$(curl -s https://api.github.com/repos/kubernetes/kops/releases/latest | grep tag\_name | cut -d '"' -f 4)/kops-linux-amd64

$ chmod +x ./kops

$ sudo mv ./kops /usr/local/bin/

sudo mv ./kubectl /usr/local/bin/kubectl

**Install KUBECTL:**

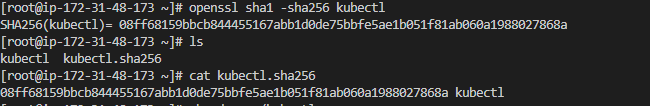
$ curl -o kubectl <https://amazon-eks.s3.us-west-2.amazonaws.com/1.19.6/2021-01-05/bin/linux/amd64/kubectl>

( Options: Verification of Kubectl: download SHA and comoare with SHA of out kubectl

$curl -o kubectl.sha256 <https://amazon-eks.s3.us-west-2.amazonaws.com/1.19.6/2021-01-05/bin/linux/amd64/kubectl.sha256>

$openssl sha1 -sha256 kubectl )

Verify the version is same:



# chmod +x ./kubectl

# mkdir -p $HOME/bin

# cp ./kubectl $HOME/bin/kubectl

# export PATH=$PATH:$HOME/bin

# kubectl version --short --client

Client Version: v1.19.6-eks-49a6c0

# whereis kubectl

kubectl: /root/bin/kubectl

**Setup IAM user and group**:

The kops user will require the following IAM permissions to function properly:

AmazonEC2FullAccess

AmazonRoute53FullAccess

AmazonS3FullAccess

IAMFullAccess

AmazonVPCFullAccess

**Recommended Approach**: Create a Group with the required policies and add user to this group

e.g. Group Name=kops

“ aws iam create-group --group-name kops”

**Create Group:**

$ aws iam create-group --group-name kops

{

"Group": {

"Path": "/",

"GroupName": "kops",

"GroupId": "AGPAZGKOKOINWJDNZA5VN",

"Arn": "arn:aws:iam::632061325851:group/kops",

"CreateDate": "2021-04-27T05:26:36Z"

}

}

**Attach IAM Roles to the group kops:**

$ aws iam attach-group-policy --policy-arn arn:aws:iam::aws:policy/AmazonEC2FullAccess --group-name test

$ aws iam attach-group-policy --policy-arn arn:aws:iam::aws:policy/AmazonRoute53FullAccess --group-name test

$ aws iam attach-group-policy --policy-arn arn:aws:iam::aws:policy/AmazonS3FullAccess --group-name test

$ aws iam attach-group-policy --policy-arn arn:aws:iam::aws:policy/IAMFullAccess --group-name test

$ aws iam attach-group-policy --policy-arn arn:aws:iam::aws:policy/AmazonVPCFullAccess --group-name test

**Create user:**

e.g. kops

“aws iam create-user --user-name kops”

$ aws iam create-user --user-name kops

{

"User": {

"Path": "/",

"UserName": "kops",

"UserId": "AIDAZGKOKOINXAWMJ3EMD",

"Arn": "arn:aws:iam::632061325851:user/kops",

"CreateDate": "2021-04-27T05:27:47Z"

}

}

**Add user to the group:**

$ aws iam add-user-to-group --user-name test --group-name test

To set AWS profile for this user “kops” generate access and secret keys.

$aws iam create-access-key --user-name test

{

"AccessKey": {

"UserName": "test",

"AccessKeyId": "AKIAZGKOKOIN5YNKZIAO",

"Status": "Active",

"SecretAccessKey": "kuz7rGWnIqC2c5SoBM3Mzw8HixBxXb/KlpTpBQh/",

"CreateDate": "2021-05-01T08:49:14+00:00"

}

}

$ aws configure

AWS Access Key ID [\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*]: AKIAZGKOKOINZFO3535I

AWS Secret Access Key [\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*FJG0]: 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

Default region name [us-east-1]:

Default output format [None]:

$ aws iam list-users

{

"Users": [

{

"Path": "/",

"UserName": "kops",

"UserId": "AIDAZGKOKOINXAWMJ3EMD",

"Arn": "arn:aws:iam::632061325851:user/kops",

"CreateDate": "2021-04-27T05:27:47Z"

},

{

"Path": "/",

"UserName": "mukki",

"UserId": "AIDAZGKOKOIN7L4WDG5CL",

"Arn": "arn:aws:iam::632061325851:user/mukki",

"CreateDate": "2021-03-07T05:05:52Z",

"PasswordLastUsed": "2021-04-27T04:17:10Z"

}

]

}

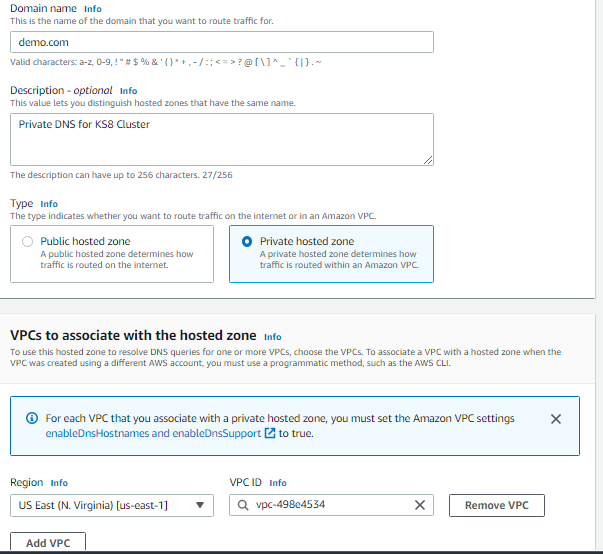
$ export AWS\_ACCESS\_KEY\_ID=$AKIAZGKOKOINZFO3535I

$ export AWS\_SECRET\_ACCESS\_KEY=$ZesIJCuEyzAhUlP04irN2NDJ4QUgtpAjej27DEOx

**R53 ( Private DNS) Configuration;**

Create Private Hosted Zone on R53 using Using Public/Private DNS (kOps 1.5+):

e.g. demo.com



Verify the DNS:

$ dig demo.com

; <<>> DiG 9.16.1-Ubuntu <<>> demo.com

;; global options: +cmd

;; Got answer:

;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 17862

;; flags: qr rd ad; QUERY: 1, ANSWER: 4, AUTHORITY: 0, ADDITIONAL: 0

;; WARNING: recursion requested but not available

;; QUESTION SECTION:

;demo.com. IN A

;; ANSWER SECTION:

demo.com. 0 IN A 151.101.2.165

demo.com. 0 IN A 151.101.130.165

demo.com. 0 IN A 151.101.66.165

demo.com. 0 IN A 151.101.194.165

;; Query time: 50 msec

;; SERVER: 172.30.176.1#53(172.30.176.1)

;; WHEN: Tue Apr 27 11:12:26 IST 2021

;; MSG SIZE rcvd: 98

$ nslookup demo.com

Server: 172.30.176.1

Address: 172.30.176.1#53

Non-authoritative answer:

Name: demo.com

Address: 151.101.2.165

Name: demo.com

Address: 151.101.130.165

Name: demo.com

Address: 151.101.66.165

Name: demo.com

Address: 151.101.194.165

**Create S3 Bucket:**

$ aws s3 ls

$ aws s3 mb s3://bucket.demo.com

$ aws s3 ls

2021-04-27 11:15:14 bucket.demo.com

Versioning Enabled:

$ aws s3api put-bucket-versioning --bucket bucket.demo.in --versioning-configuration Status=Enabled

$ export KOPS\_STATE\_STORE=s3://bucket.demo.com

**Cluster Set-up:**

e.g. mycluster.demo.com

export NAME=cluster.demo.com

export KOPS\_STATE\_STORE=s3://bucket.demo.com

export ZONES=us-east-1

**$ ssh-keygen**

kops create cluster --name= cluster.demo.com \

--zones=us-east-1a \

--node-count=2 \

--dns-zone=demo.com \

--dns private

It will create a cluster infra with option to edit and finally apply:

$ kops update cluster --name cluster.demo.com --yes --admin

Cluster validation:

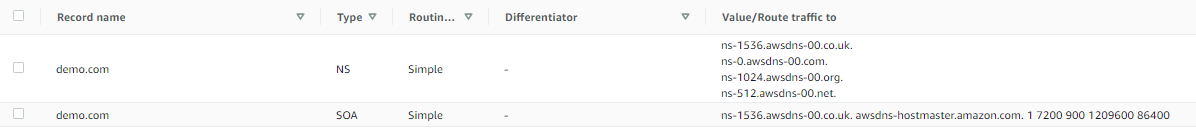
$ kops validate cluster

In a private DNS this throws error while connecting from Ubuntu machine probably access is within VPC which means resources inside VPC only can access cluster directly.

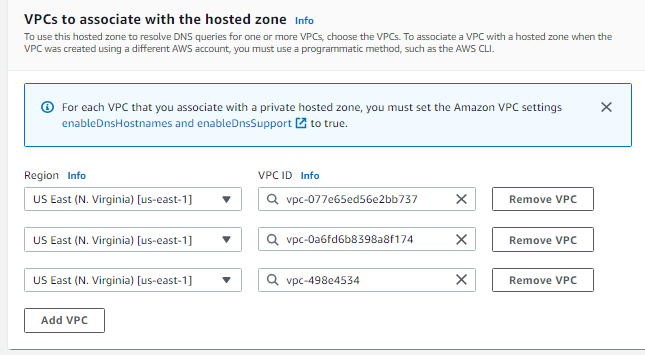
To validate cluster created properly.

R53 will have additional records created for our cluster.

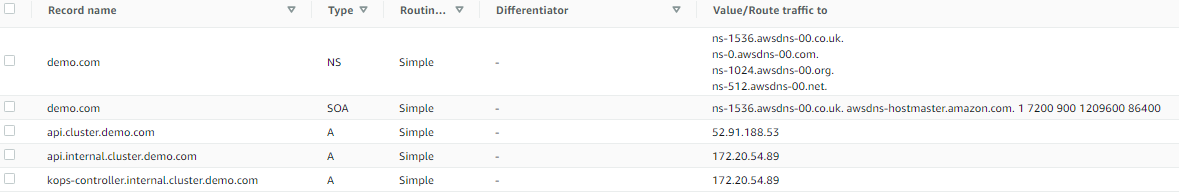
Original records while after Hosted Zone demo.com creation as private hosted zone under default VPC as well as Cluster will add new VPC alongwith the default



VPC-498…. Is default during the initial setup of private hosted zone creation:



Additonal Records created by Cluster:



Similarly S3 will have additional entry and IAM will have new roles for cluster.

**How to access Cluster in Private DNS:**

Since we have set-up cluster from a local WSL Ubuntu system using Private DNS and Cluster is hosted in a VPC so by default it can be accessed only through the resources inside the VPC. Typically a Public DNS will resolve this issue, however, we can still verify the cluster health by exploring SSH option.

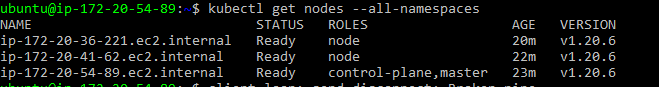
We can ssh into the Cluster-Api or Master Node

ssh -i ~/.ssh/id\_rsa [ubuntu@api.cluster.demo.com](mailto:ubuntu@api.cluster.demo.com)

or

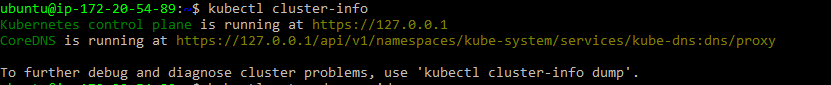
ssh to the master: ssh -i ~/.ssh/id\_rsa [ubuntu@Master](mailto:ubuntu@api.cluster.demo.com) Public IP

$ ssh -i ~/.ssh/id\_rsa [ubuntu@52.91.188.53](mailto:ubuntu@52.91.188.53)

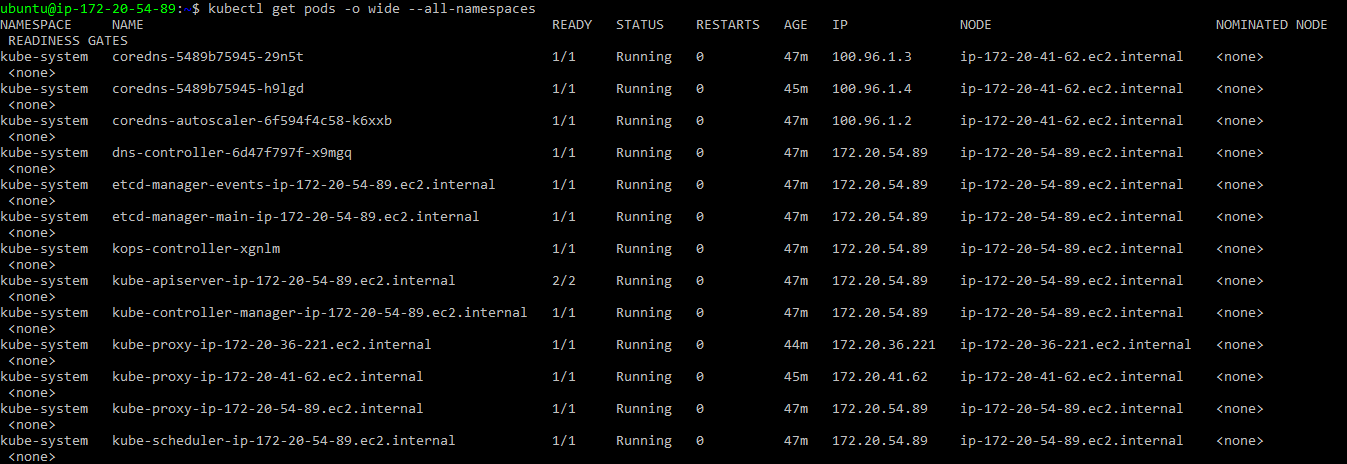








$ kubectl get pods -o wide --all-namespaces



**Cluster in HA Mode:**

# Create a cluster in AWS with HA masters. This cluster

# has also been configured for private networking in a kops-managed VPC.

# The bastion flag is set to create an entrypoint for admins to SSH.

export KOPS\_STATE\_STORE="s3://my-state-store"

export MASTER\_SIZE="c5.large"

export NODE\_SIZE="m5.large"

export ZONES="us-east-1a,us-east-1b,us-east-1c"

kops create cluster k8s-cluster.example.com \

--node-count 3 \

--zones $ZONES \

--node-size $NODE\_SIZE \

--master-size $MASTER\_SIZE \

--master-zones $ZONES \

--networking cilium \

--topology private \

--bastion="true" \

--yes

**Clean-up:**

**Delete Cluster:**

kops delete cluster --name=” “ --yes

$ kops delete cluster --name=cluster.demo.com –yes

Deleted cluster: "cluster.demo.com"

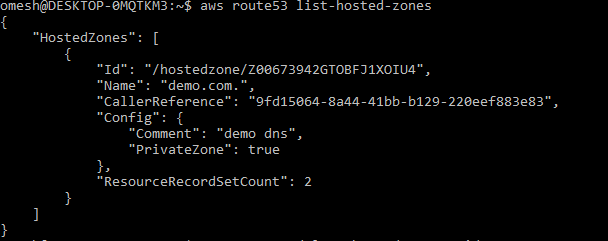
**Delete bucket:**

$ aws s3 rb s3://bucket.demo.com

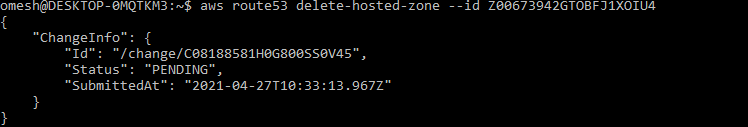
remove\_bucket: bucket.demo.com

**Delete Private DNS:**

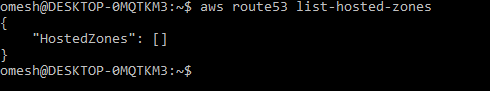
$ aws route53 list-hosted-zones



$ aws route53 delete-hosted-zone --id Z00673942GTOBFJ1XOIU4



$ aws route53 list-hosted-zones



Use case:

I have created the KOPS Cluster and successfully tested the deployment using the tested.yaml file

In this file the first part is Service and second part is Image Deployment.

Deployment:

Two replicas of image: containersol/k8s-deployment-strategies are deployed and container port is open at 8080.

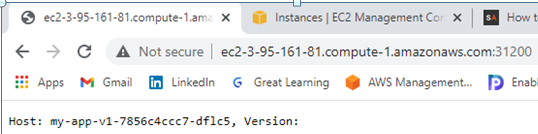
Service:

Load balancer is used to access the service exposed on the POD Port 8080 and node port 31200.

We accessed service using the Maser Node DNS and node port 31200.

<http://ec2-3-95-161-81.compute-1.amazonaws.com:31200/>

Testing of an app using single template for Deployment and Service:



Testing of Nodejs using separate templates for Deployment and Service:

