**CREATE KUBERNETES CLUSTER IN AWS USE KOPS**

**PREREQUIREMENTS "tag\_name": "v1.31.0" – 2024/aug**

1. linux machine (ubuntu)

2. AWS account

3. kops binary (kubernetes cluster initiate)

4. kubectl binary (kubernetes deployments)

**KOPS BINARY SETUP**

# 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/

**KUBECTL BINARY SETUP**

# curl -Lo kubectl https://storage.googleapis.com/kubernetes-release/release/$(curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt)/bin/linux/amd64/kubectl

# chmod +x ./kubectl

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

**SETUP IAM USER** (kops access aws resources)

This is awscli commands working methods. kindly configure aws-cli packages in your linux machines.

In order to build clusters within AWS we'll create a dedicated IAM user for kops. This user requires API credentials in order to use kops. Create the user, and credentials, using the AWS console.

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

1. AmazonEC2FullAccess

2. AmazonRoute53FullAccess

3. AmazonS3FullAccess

4. IAMFullAccess

5. AmazonVPCFullAccess

You can create the kops IAM user from the command line using the following:

# aws iam create-group --group-name kops

# aws iam attach-group-policy --policy-arn arn:aws:iam::aws:policy/AmazonEC2FullAccess --group-name kops

# aws iam attach-group-policy --policy-arn arn:aws:iam::aws:policy/AmazonRoute53FullAccess --group-name kops

# aws iam attach-group-policy --policy-arn arn:aws:iam::aws:policy/AmazonS3FullAccess --group-name kops

# aws iam attach-group-policy --policy-arn arn:aws:iam::aws:policy/IAMFullAccess --group-name kops

# aws iam attach-group-policy --policy-arn arn:aws:iam::aws:policy/AmazonVPCFullAccess --group-name kops

# aws iam create-user --user-name kops

# aws iam add-user-to-group --user-name kops --group-name kops

# aws iam create-access-key --user-name kops

You should record the SecretAccessKey and AccessKeyID in the returned JSON output, and then use them below:

**configure the aws client to use your new IAM user**

# aws configure # Use your new access and secret key here

# aws iam list-users # you should see a list of all your IAM users here

Notes:

Because "aws configure" doesn't export these vars for kops to use, we export them now

# export AWS\_ACCESS\_KEY\_ID=$(aws configure get aws\_access\_key\_id)

# export AWS\_SECRET\_ACCESS\_KEY=$(aws configure get aws\_secret\_access\_key)

**cluster type:**

1. single node cluster

2. Multinode cluster

**DNS kubernetes type:**

1. single node cluter means not need any dns entry its take default machine dns record. The only requirement to trigger this is to have the cluster name end with .k8s.local.
2. Multinode cluster means it act loadbalancer type. Loadbalancer means it's need single entry point to rechaed this cluster master nodes.we need to prepare somewhere to build the required DNS records. (use route 53 DNS record)

**Cluster State storage**

In order to store the state of your cluster, and the representation of your cluster, we need to create a dedicated S3 bucket for kops to use. This bucket will become the source of truth for our cluster configuration. In this guide we'll call this bucket awsdamo, but you should add a custom prefix as bucket names need to be unique.

# aws s3api create-bucket --bucket awssathish --region us-east-1

# aws s3api put-bucket-versioning --bucket awssathish --versioning-configuration Status=Enabled

**nodes authentication methods**

create sshkey for machines

# ssh-keygen

**Creating your first cluster**

**Prepare local environment (Multimaster types)**

We're ready to start creating our first cluster! Let's first set up a few environment variables to make this process easier.

# export NAME=aws.sathish.com

# export KOPS\_STATE\_STORE=s3://HDFC\_non\_prod

For a gossip-based cluster, make sure the name ends with k8s.local. For example:

# export NAME=sathish.k8s.local

# export KOPS\_STATE\_STORE=s3://HDFC\_non\_prod

Note:

You don’t have to use environmental variables here. You can always define the values using the –name and –state flags later.

# kops create cluster --zones us-east-1a ${NAME}

# kops create secret ${NAME} sshpublickey admin -i ~/.ssh/id\_rsa.pub (Optional)

**if choose multiple availability zones**

# kops create cluster --zones us-east-1a,us-east-1b ${NAME}

**kops cluster details and initiate machanisam:**

LIST CLUSTER DETAILS

# kops get cluster

EDIT CLUSTER

# kops edit cluster sathish.k8s.local

EDIT INSTANCE NODE GROUP

# kops edit ig --name= sathish.k8s.local nodes

EDIT MASTER INSTANCE GROUP

# kops edit ig --name= sathish.k8s.local master-us-east-1a

UPDATE CLUSTER

# kops update cluster --name sathish.k8s.local --yes

cluster validates:

-----------------

VALIDATE CLUSTER

# kops validate cluster

LIST NODES

# kubectl get nodes --show-labels

SSH CONNECTION ESTABLISHMENT

# ssh -i ~/.ssh/id\_rsa admin@public[master,nodes]

kops delete cluster --name=sathish.k8s.local --state=s3://awssathish --yes