Kubernetes: Run the Container Orchestration Tool for Production

Overview: We have run numerous containers until now, all on a single Docker engine. However, if the Docker engine fails, all containers running on it will go down, making them inaccessible to users. To mitigate this risk in a production environment, we should cluster the Docker engine. This setup involves multiple Docker nodes, with a master node controlling all Docker nodes. The master node instructs Docker nodes on which containers to run and distributes containers across the nodes. If any Docker node fails, containers can run on a live Docker engine, ensuring continuity. This migration of containers to healthy nodes is called container orchestration. The master node, known as the orchestrator, manages a cluster of Docker nodes or worker nodes, creating a fault-tolerant pool of resources.

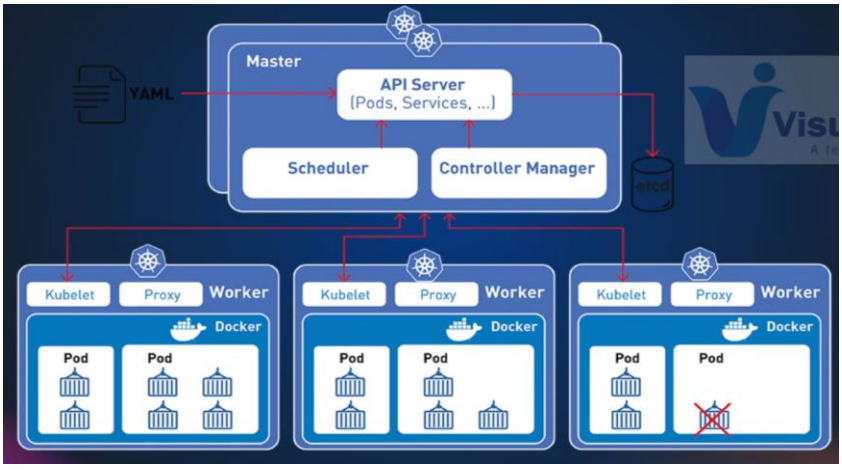
Orchestration Tools:

- Docker Swarm
- Kubernetes
- AWS ECS & EKS
- Azure Container Service
- Google Container Engine
- CoreOS Fleet
- OpenShift

Services Provided by Orchestration Tools:

- Load balancing
- Storage orchestration
- Automated rollouts and rollbacks
- Automatic bin packing
- Self-healing

Kubernetes Architecture:



Master Components:

1. **Kube API Server:**
    - Handles all requests and enables communication across stack services.
    - Component on the master that exposes the Kubernetes API.
    - Serves as the front end for the Kubernetes control plane.
    - Admins connect to it using the Kubectl CLI.
    - A web dashboard can be integrated with this API.
2. **ETCD Server:**
    - Stores all cluster information.
    - A consistent and highly available key-value store used as Kubernetes' backing store for all cluster data.
    - The Kube API stores and retrieves information from it.
    - Should be backed up regularly.
    - Stores the current state of everything in the cluster.
3. **Kube Scheduler:**
    - Watches newly created pods that have no node assigned and selects a node for them to run on.
    - Factors considered for scheduling decisions include:
        - Individual and collective resource requirements
        - Hardware/software/policy constraints
        - Affinity and anti-affinity specifications
        - Data locality
        - Inter-workload interference
        - Deadlines
4. **Controller Manager:**
    - Logically, each controller is a separate process, but to reduce complexity, they are compiled into a single binary and run in a single process.
    - These controllers include:
        - **Node Controller:** Responsible for noticing and responding when nodes go down.
        - **Replication Controller:** Maintains the correct number of pods for every replication controller object in the system.
        - **Endpoints Controller:** Populates the Endpoints object, joining Services & Pods.
        - **Service Account & Token Controllers:** Create default accounts and API access tokens for new namespaces.

Node Components:

1. **Kubelet:**
    - An agent that runs on each node in the cluster. It ensures that containers are running in a pod.
2. **Kube Proxy:**
    - A network proxy that runs on each node in the cluster.
    - Manages network rules allowing communication to Pods inside or outside of the cluster.
3. **Container Runtime:**
    - Kubernetes supports several container runtimes, including:
        - Docker
        - containerd
        - cri-o
        - rktlet
        - Kubernetes CRI (Container Runtime Interface)

Kubernetes Setup Tools

- Hard Way: Manual Setup

- Minikube:

  ○ One Node Kubernetes cluster on your computer

- Kubeadm:

  ○ Multi-node Kubernetes Cluster ○ Can be created on any Platforms vm's, ec2, physical machines etc

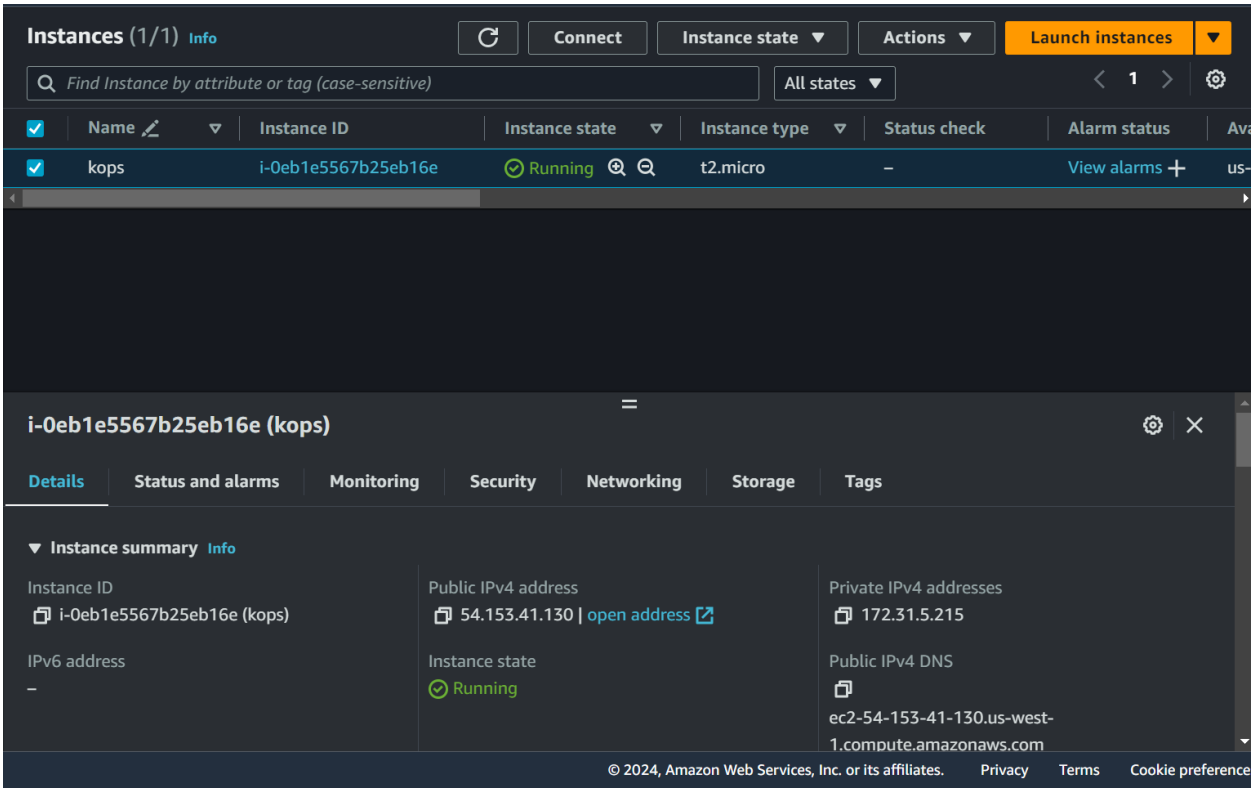
- Kops:

  ○ Multi node Kubernetes Cluster on AWS

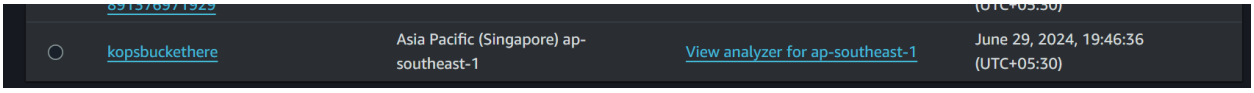
  NOTE - Kops is not part of the Kubernetes Cluster, it is just an instance to launch the Kubernetes cluster.

Setup with Kops

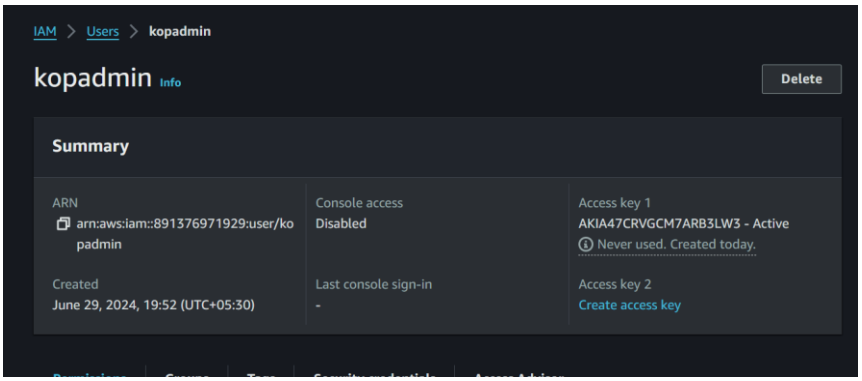
1) Launch an instance with Ubuntu AMI



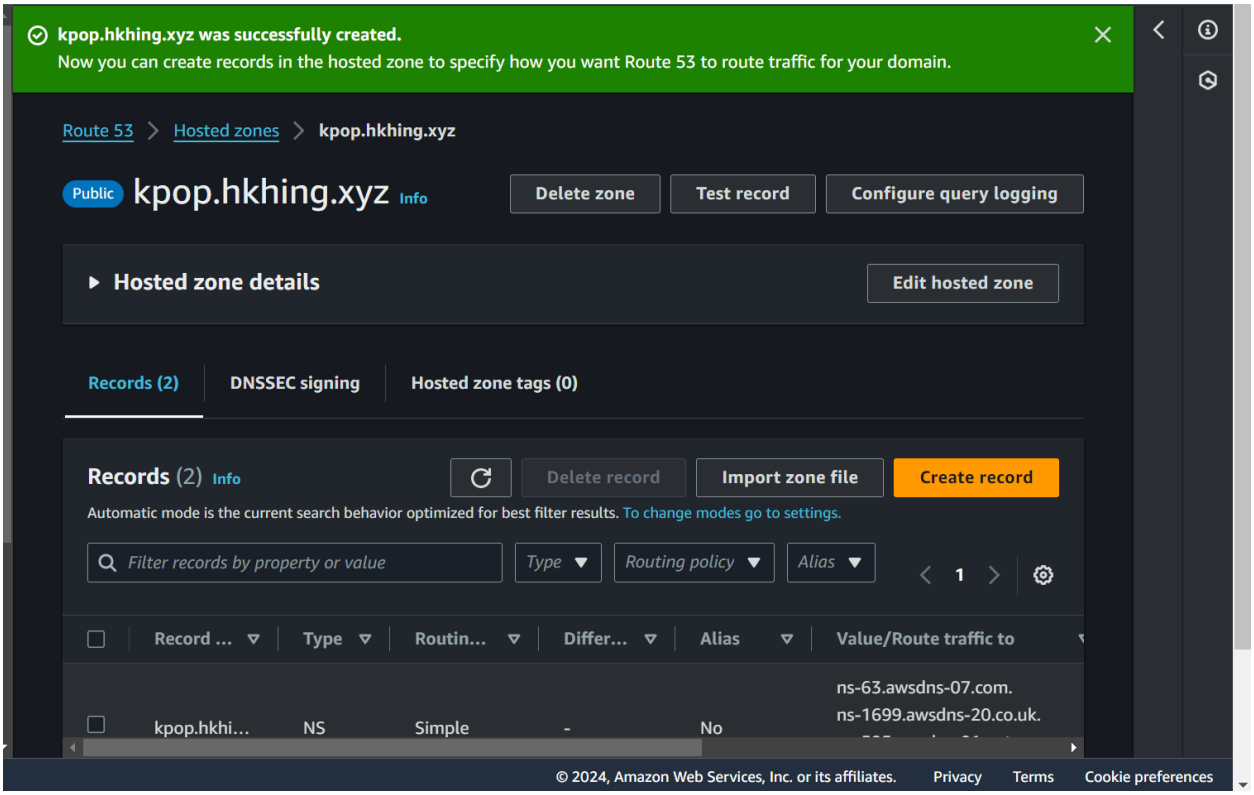
2) Create a s3 bucket



3) Create an IAM user



4) Create a hosted zone



5) Save the nameservers in your GoDaddy DNS record.

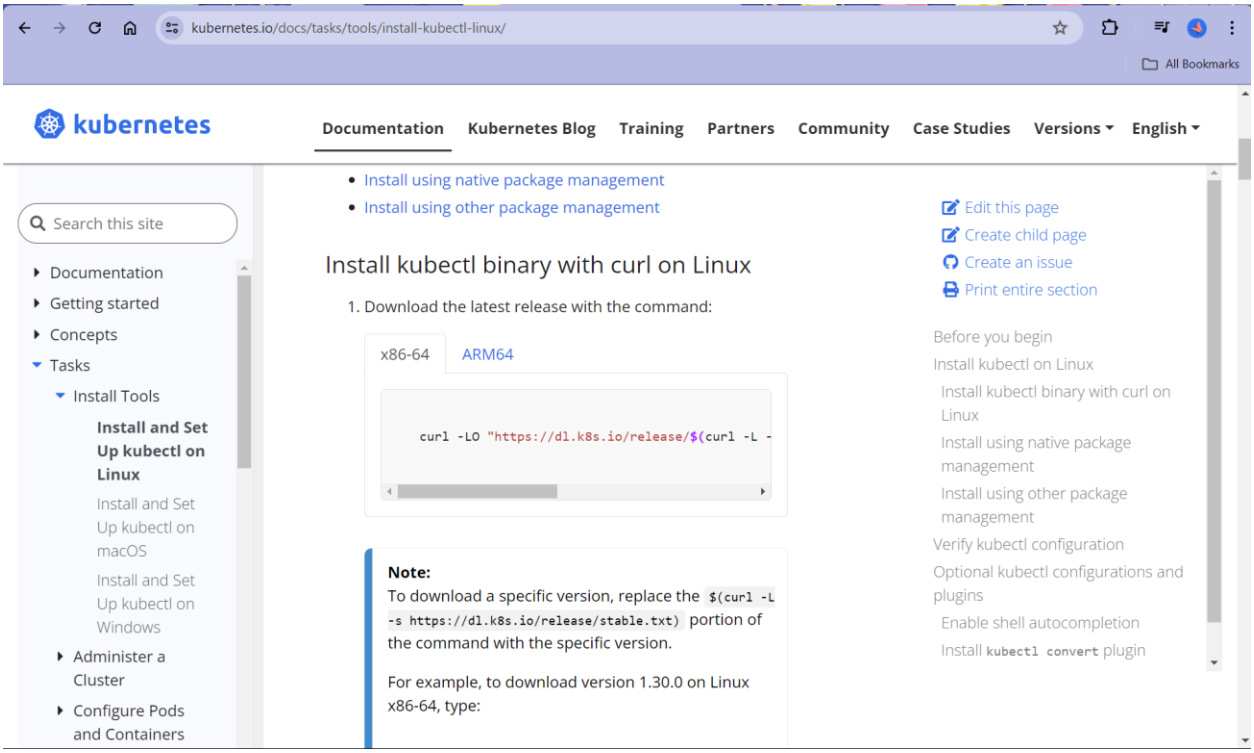
☐	NS	kpop	ns-1158.awsdns-16.org.	1 Hour	🗑	✎
☐	NS	kpop	ns-1699.awsdns-20.co.uk.	1 Hour	🗑	✎
☐	NS	kpop	ns-525.awsdns-01.net.	1 Hour	🗑	✎
☐	NS	kpop	ns-63.awsdns-07.com.	1 Hour	🗑	✎

6) Connect to your ec2 instance and setup everything

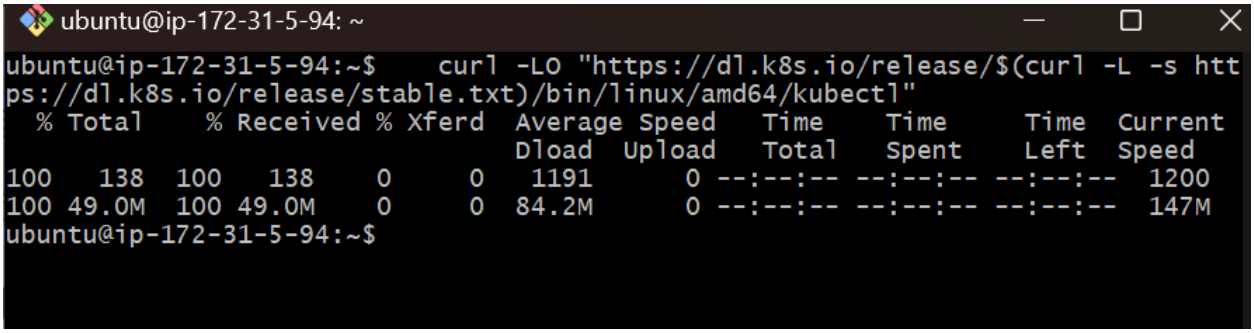
```
ubuntu@ip-172-31-5-215:~$ ssh-keygen
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/ubuntu/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ubuntu/.ssh/id_ed25519
Your public key has been saved in /home/ubuntu/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:tyklOQe5zwzVgASuejiLC22qIaqSF+aQrM51M5TsD8E ubuntu@ip-172-31-5-215
The key's randomart image is:
+--[ED25519 256]--+
|      .o...      |
|     .  .. o     |
|      . o . .    |
|    o..  =       |
|..  .E  S +      |
|ooooo .  @ o     |
|==*o.*   . *     |
|O=++. =  .       |
|%=.    .         |
+----[SHA256]-----+
ubuntu@ip-172-31-5-215:~$
```

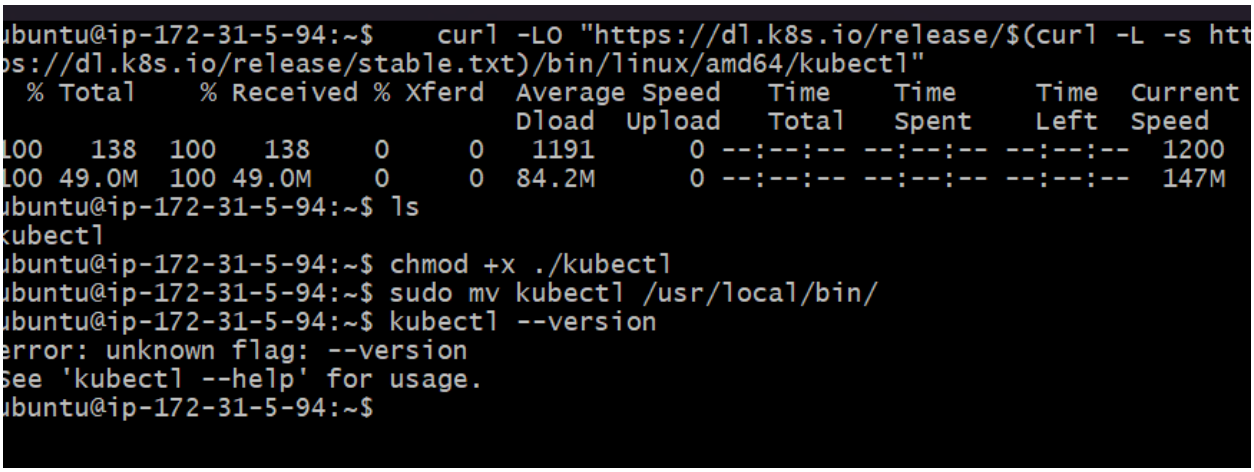
```
ubuntu@ip-172-31-5-215:~$ sudo apt update && sudo apt install awscli -y
Get:1 http://us-west-1.ec2.archive.ubuntu.com/ubuntu noble InRelease [256 kB]
Get:2 http://us-west-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [12
6 kB]
Get:3 http://us-west-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [
126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 http://us-west-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 Packages [
1401 kB]
Get:6 http://us-west-1.ec2.archive.ubuntu.com/ubuntu noble/main Translation-en [
513 kB]
Get:7 http://us-west-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packag
es [15.0 MB]
Get:8 http://us-west-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-
en [5982 kB]
Get:9 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [182
kB]
Get:10 http://us-west-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Compo
nents [3871 kB]
Get:11 http://us-west-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f
 Metadata [301 kB]
```

```
No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-5-94:~$ aws configure
AWS Access Key ID [None]: AKIA47CRVGCM7ARB3LW3
AWS Secret Access Key [None]: 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
Default region name [None]: us-west-1
Default output format [None]: json
ubuntu@ip-172-31-5-94:~$
```

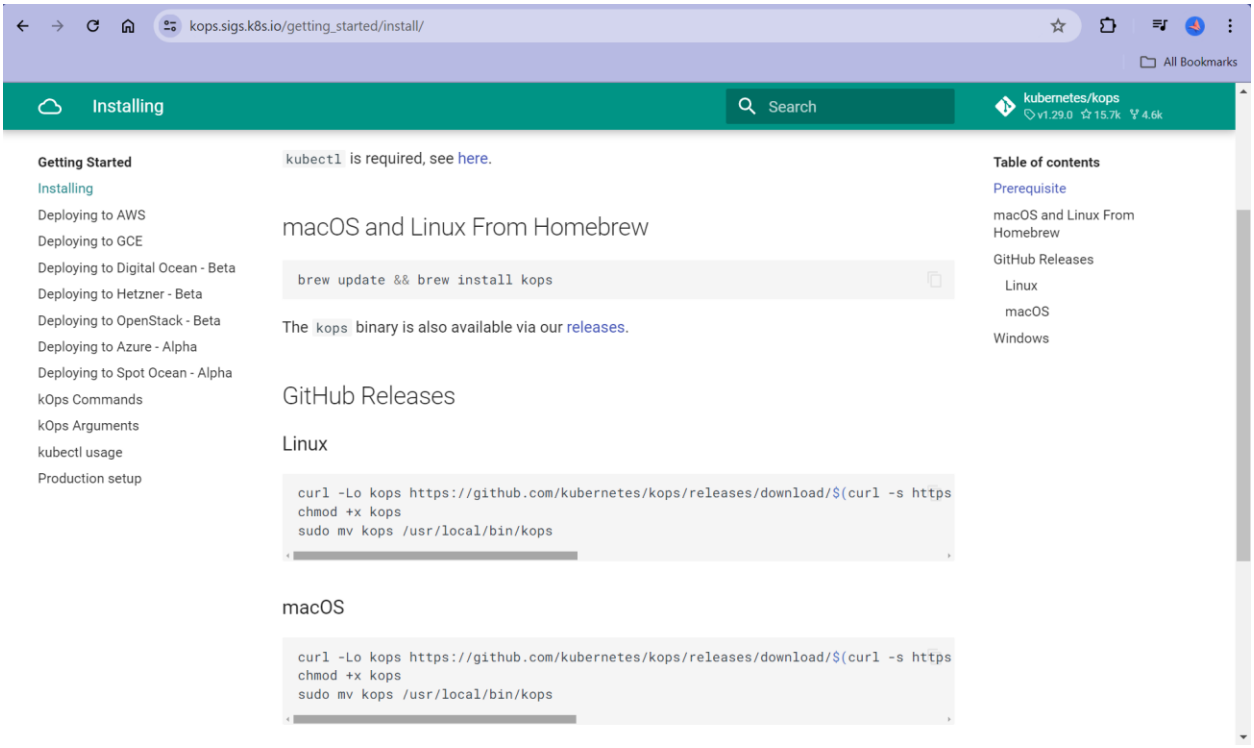


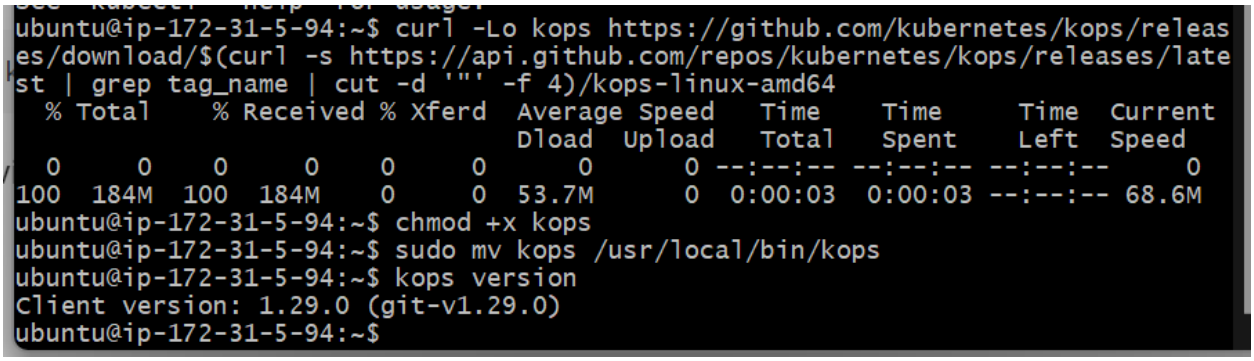
Go to documentation and install kubectl for Linux





An error occurred let us go to documentation and explore





```
ubuntu@ip-172-31-5-94:~$ nslookup -type=ns kpop.hkhing.xyz
Server:         127.0.0.53
Address:        127.0.0.53#53

Non-authoritative answer:
kpop.hkhing.xyz nameserver = ns-63.awsdns-07.com.
kpop.hkhing.xyz nameserver = ns-525.awsdns-01.net.
kpop.hkhing.xyz nameserver = ns-1158.awsdns-16.org.
kpop.hkhing.xyz nameserver = ns-1699.awsdns-20.co.uk.

Authoritative answers can be found from:

ubuntu@ip-172-31-5-94:~$
```

```
ubuntu@ip-172-31-5-94:~$ kops create cluster --name=kpop.hkhing.xyz \
> --state=s3://kopsbucketlist --zones=us-west-1b,us-west-1c \
> --node-count=2 --node-size=t3.small --master-size=t3.medium --dns-zone=kpop.hk
hing.xyz \
> --node-volume-size=8 --master-volume-size=8
Flag --master-size has been deprecated, use --control-plane-size instead
Flag --master-volume-size has been deprecated, use --control-plane-volume-size i
nstead
I0630 06:43:39.404836    3290 new_cluster.go:1445] Cloud Provider ID: "aws"
I0630 06:43:39.490254    3290 subnets.go:224] Assigned CIDR 172.20.0.0/17 to sub
net us-west-1b
I0630 06:43:39.490378    3290 subnets.go:224] Assigned CIDR 172.20.128.0/17 to s
ubnet us-west-1c
Previewing changes that will be made:
```

This command will not create a cluster but it will create a configuration for the cluster in the S3 bucket.

```
buntu@ip-172-31-5-94:~$  kops update cluster --name kpop.hkhing.xyz --state=s3:
/kopsbucketlist --yes --admin
0630 06:46:52.559631    3294 executor.go:113] Tasks: 0 done / 113 total; 45 can
run
0630 06:46:52.620387    3294 vfs_keystorereader.go:143] CA private key was not
ound
0630 06:46:52.631523    3294 keypair.go:226] Issuing new certificate: "etcd-man
```

```
ubuntu@ip-172-31-5-94:~$ kops validate cluster --state=s3://kopsbucketlist
Using cluster from kubectl context: kpop.hkhing.xyz

Validating cluster kpop.hkhing.xyz

INSTANCE GROUPS
NAME                            ROLE            MACHINETYPE     MIN     MAX     S
UBNETS
control-plane-us-west-1b        ControlPlane    t3.medium       1       1       u
s-west-1b
nodes-us-west-1b                Node            t3.small        1       1       u
s-west-1b
nodes-us-west-1c                Node            t3.small        1       1       u
s-west-1c

NODE STATUS
NAME    ROLE    READY
```

Delete the cluster

```
ubuntu@ip-172-31-12-44:~$ kops delete cluster  --name=kubevpro.groophy.in --state=s3://vprofile-kop-states --ye
s
```