

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.

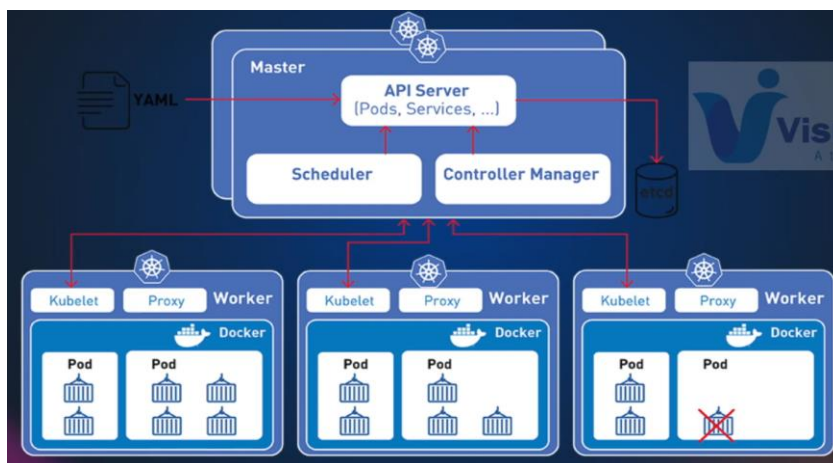
Orchestration Tools:

- 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

The screenshot displays the AWS Management Console interface for EC2 instances. At the top, there's a header for 'Instances (1/1)' with an 'Info' link. Below this is a search bar and a filter dropdown set to 'All states'. A table lists the instance 'kops' with ID 'i-0eb1e5567b25eb16e', state 'Running', type 't2.micro', and a 'View alarms' link. Below the table, a detailed view for instance 'i-0eb1e5567b25eb16e (kops)' is shown with tabs for Details, Status and alarms, Monitoring, Security, Networking, Storage, and Tags. The 'Details' tab is active, showing a summary of the instance's configuration: Instance ID 'i-0eb1e5567b25eb16e (kops)', Public IPv4 address '54.153.41.130' with a link to 'open address', Private IPv4 addresses '172.31.5.215', IPv6 address '-', Instance state 'Running', and Public IPv4 DNS 'ec2-54-153-41-130.us-west-1.compute.amazonaws.com'.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status
kops	i-0eb1e5567b25eb16e	Running	t2.micro	-	View alarms

i-0eb1e5567b25eb16e (kops)

Instance summary

Instance ID i-0eb1e5567b25eb16e (kops)	Public IPv4 address 54.153.41.130 open address	Private IPv4 addresses 172.31.5.215
IPv6 address -	Instance state Running	Public IPv4 DNS ec2-54-153-41-130.us-west-1.compute.amazonaws.com

2) Create a s3 bucket

The screenshot shows the AWS IAM console with a user named 'kopsbucketthere' selected. The user is located in the 'Asia Pacific (Singapore) ap-southeast-1' region. The console shows the user's ARN as 'arn:aws:iam::891376971929:user/kopsbucketthere' and their creation date as 'June 29, 2024, 19:46:36 (UTC+05:30)'. There is a link to 'View analyzer for ap-southeast-1'.

Name	Region	ARN	Created
kopsbucketthere	Asia Pacific (Singapore) ap-southeast-1	arn:aws:iam::891376971929:user/kopsbucketthere	June 29, 2024, 19:46:36 (UTC+05:30)

3) Create an IAM user

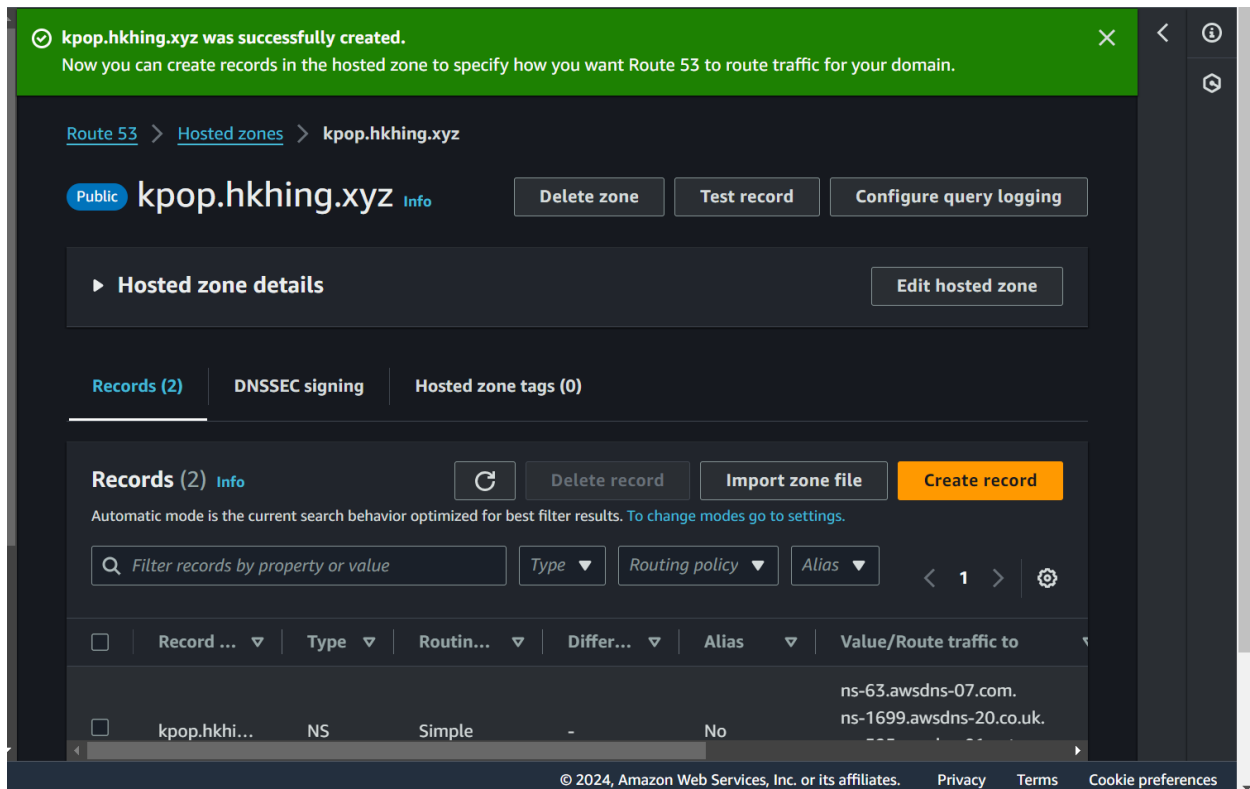
The screenshot displays the AWS IAM console for an IAM user named 'kopadmin'. The user is located in the 'Asia Pacific (Singapore) ap-southeast-1' region. The console shows the user's ARN as 'arn:aws:iam::891376971929:user/kopadmin' and their creation date as 'June 29, 2024, 19:52 (UTC+05:30)'. The user has console access disabled and no last console sign-in. There are two access keys: 'Access key 1' (AKIA47CRVGC7ARB3LW3 - Active) and 'Access key 2' (Create access key). The user is never used.

kopadmin

Summary

ARN arn:aws:iam::891376971929:user/kopadmin	Console access Disabled	Access key 1 AKIA47CRVGC7ARB3LW3 - Active Never used. Created today.
Created June 29, 2024, 19:52 (UTC+05:30)	Last console sign-in -	Access key 2 Create access key

4) Create a hosted zone



5) Save the nameservers in your GoDaddy DNS record.

<input type="checkbox"/>	NS	kpop	ns-1158.awsDNS-16.org.	1 Hour		
<input type="checkbox"/>	NS	kpop	ns-1699.awsDNS-20.co.uk.	1 Hour		
<input type="checkbox"/>	NS	kpop	ns-525.awsDNS-01.net.	1 Hour		
<input type="checkbox"/>	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:tyk10Qe5zwzVgASuejiLC22qIaqSF+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 [126 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 Packages [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 Components [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:~$ |

```

kubernetes.io/docs/tasks/tools/install-kubectl-linux/

kubernetes

Documentation Kubernetes Blog Training Partners Community Case Studies Versions English

Search this site

- Documentation
- Getting started
- Concepts
- Tasks
 - Install Tools
 - Install and Set Up kubectl on Linux**
 - Install and Set Up kubectl on macOS
 - Install and Set Up kubectl on Windows
 - Administer a Cluster
 - Configure Pods and Containers

- Install using native package management
- Install using other package management

Install kubectl binary with curl on Linux

1. Download the latest release with the command:

x86-64 ARM64

```
curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
```

Note:
To download a specific version, replace the `$(curl -L -s https://dl.k8s.io/release/stable.txt)` portion of the command with the specific version.
For example, to download version 1.30.0 on Linux x86-64, type:

Before you begin

- Install kubectl on Linux
- Install kubectl binary with curl on Linux
- Install using native package management
- Install using other package management
- Verify kubectl configuration
- Optional kubectl configurations and plugins
- Enable shell autocompletion
- Install kubectl convert plugin

Go to documentation and install kubectl for Linux

```
ubuntu@ip-172-31-5-94: ~
ubuntu@ip-172-31-5-94:~$ curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload   Total   Spent    Left     Speed
100 138    100 138    0    0  1191      0 --:--:-- --:--:-- --:--:-- 1200
100 49.0M 100 49.0M    0    0 84.2M      0 --:--:-- --:--:-- --:--:-- 147M
ubuntu@ip-172-31-5-94:~$
```

```
ubuntu@ip-172-31-5-94:~$ curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload   Total   Spent    Left     Speed
100 138    100 138    0    0  1191      0 --:--:-- --:--:-- --:--:-- 1200
100 49.0M 100 49.0M    0    0 84.2M      0 --:--:-- --:--:-- --:--:-- 147M
ubuntu@ip-172-31-5-94:~$ ls
kubectl
ubuntu@ip-172-31-5-94:~$ chmod +x ./kubectl
ubuntu@ip-172-31-5-94:~$ sudo mv kubectl /usr/local/bin/
ubuntu@ip-172-31-5-94:~$ kubectl --version
error: unknown flag: --version
See 'kubectl --help' for usage.
ubuntu@ip-172-31-5-94:~$
```

An error occurred let us go to documentation and explore

← → ↻ 🏠 kops.sigs.k8s.io/getting_started/install/ ☆ 📁 📄 🌐 All Bookmarks

Installing 🔍 Search kubernet.es/kops v1.29.0 15.7k 4.6k

Getting Started

- Installing
- Deploying to AWS
- Deploying to GCE
- Deploying to Digital Ocean - Beta
- Deploying to Hetzner - Beta
- Deploying to OpenStack - Beta
- Deploying to Azure - Alpha
- Deploying to Spot Ocean - Alpha
- kOps Commands
- kOps Arguments
- kubectl usage
- Production setup

kubectl is required, see [here](#).

macOS and Linux From Homebrew

```
brew update && brew install kops
```

The `kops` binary is also available via our [releases](#).

GitHub Releases

Linux

```
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/kops
```

macOS

```
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-darwin-amd64
chmod +x kops
sudo mv kops /usr/local/bin/kops
```

Table of contents

- Prerequisite
- macOS and Linux From Homebrew
- GitHub Releases
 - Linux
 - macOS
 - Windows

```
ubuntu@ip-172-31-5-94:~$ curl -Lo kops https://github.com/kubernetes/kops/releases/download/$(curl -s https://api.github.com/repos/kubernetes/kops/releases/latest | grep tag_name | cut -d '"' -f 4)/kops-linux-amd64
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload   Total   Spent    Left     Speed
100 184M  100 184M    0     0  53.7M      0  0:00:03  0:00:03 --:--:-- 68.6M
ubuntu@ip-172-31-5-94:~$ chmod +x kops
ubuntu@ip-172-31-5-94:~$ sudo mv kops /usr/local/bin/kops
ubuntu@ip-172-31-5-94:~$ kops version
Client version: 1.29.0 (git-v1.29.0)
ubuntu@ip-172-31-5-94:~$
```



```
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.hkHING.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 instead
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 subnet us-west-1b
I0630 06:43:39.490378 3290 subnets.go:224] Assigned CIDR 172.20.128.0/17 to subnet 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.

```
ubuntu@ip-172-31-5-94:~$ kops update cluster --name kpop.hkHING.xyz --state=s3://kopsbucketlist --yes --admin
I0630 06:46:52.559631 3294 executor.go:113] Tasks: 0 done / 113 total; 45 can run
I0630 06:46:52.620387 3294 vfs_keystorereader.go:143] CA private key was not found
I0630 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 --yes
```