Blog Summary

Kubernetes vs. Docker Swarm

Kubernetes:

Advanced orchestration for complex, large-scale applications.

Supports advanced networking, scaling, storage, and multi-cloud deployments.

Steep learning curve and higher resource requirements.

Commands include kubectl for managing resources and services.

Best for production environments requiring fault tolerance, auto-scaling, and customization.

Docker Swarm:

Lightweight, simple orchestration for small-scale deployments.

Tight integration with Docker CLI and Docker Compose.

Commands include docker swarm for initializing, scaling, and managing services.

Ideal for quick setups, development, and single-cloud environments.

Docker vs. Podman

Docker:

Popular containerization platform with a daemon-based architecture.

Integrated orchestration via Docker Swarm and Kubernetes support.

Best for teams already using Docker or needing a robust community.

Podman:

Daemonless container engine with rootless containers for enhanced security.

Compatible with Docker commands and Kubernetes from the outset.

Better suited for security-focused and Kubernetes-heavy workflows.

DevOps Tools and Commands

Here are commands for some key DevOps tools used with Kubernetes, Docker, and containerized environments:

1. Kubernetes Commands

# Start a Kubernetes cluster (using Minikube)

minikube start

# Deploy a Pod

kubectl run nginx --image=nginx

# List Pods

kubectl get pods

# Expose a deployment as a service

kubectl expose deployment nginx --port=80 --type=LoadBalancer

# Scale a deployment

kubectl scale deployment nginx --replicas=5

# Delete a Pod

kubectl delete pod nginx

2. Docker Swarm Commands

# Initialize a Docker Swarm

docker swarm init

# Join a worker node to the Swarm

docker swarm join --token <token> <manager-ip>:2377

# Deploy a service

docker service create --name nginx --replicas 3 -p 80:80 nginx

# Scale a service

docker service scale nginx=5

# Remove a service

docker service rm nginx

# List Swarm nodes

docker node ls

3. Git Commands for CI/CD

# Clone a repository

git clone <repository\_url>

# Create and switch to a new branch

git checkout -b feature-branch

# Stage and commit changes

git add .

git commit -m "Commit message"

# Push changes to a remote branch

git push origin feature-branch

# Merge a branch into the main branch

git checkout main

git merge feature-branch

4. Jenkins Commands

# Start Jenkins (via Docker)

docker run -p 8080:8080 -p 50000:50000 jenkins/jenkins:lts

# Access Jenkins CLI

java -jar jenkins-cli.jar -s http://<jenkins-url> <command>

# List all Jenkins jobs

java -jar jenkins-cli.jar -s http://<jenkins-url> list-jobs

# Build a job

java -jar jenkins-cli.jar -s http://<jenkins-url> build <job\_name>

5. Terraform Commands

# Initialize Terraform in a directory

terraform init

# Validate Terraform configuration files

terraform validate

# Create and plan infrastructure changes

terraform plan

# Apply the infrastructure changes

terraform apply

# Destroy the infrastructure

terraform destroy

6. Ansible Commands

# Run a playbook

ansible-playbook playbook.yml

# Check inventory hosts

ansible all -i inventory.ini --list-hosts

# Run an ad-hoc command

ansible all -i inventory.ini -m ping

# Check a configuration file

ansible-config dump --only-changed

7. Helm Commands (for Kubernetes)

# Install a Helm chart

helm install my-app stable/nginx

# List installed releases

helm list

# Upgrade a release

helm upgrade my-app stable/nginx

# Uninstall a release

helm uninstall my-app

8. Prometheus and Grafana (Monitoring Tools)

# Run Prometheus (via Docker)

docker run -p 9090:9090 prom/prometheus

# Run Grafana (via Docker)

docker run -d -p 3000:3000 grafana/grafana

# Add Prometheus as a Grafana data source (via Grafana UI)