Bringing Ideas to Life : Deployment and Real-World Operation

Presented by,
Naveenkumar Kumanan
SDE @AppViewX

DevOps – de mystifying

Picture credits:

https://blog.bytebytego.com/p/a-crash-course-in-cicd

Prerequisite software for the session

- Docker
- Kubectl
- K9s
- Kind

Download links for reference:

If using windows then better download wsl2 and install other software inside wsl2

Docker: https://docs.docker.com/engine/install/

Kubernetes: https://kubernetes.io/docs/tasks/tools/install-kubectl-linux/

K9s: https://webinstall.dev/k9s/

Kind: https://kind.sigs.k8s.io/docs/user/quick-start/

Agenda

Theory:

- **Docker and Containerization:** Introduction to containerization, Docker basics, and why it's crucial for modern development.
- **Manifest Files (Configuration-Driven Frameworks):** Understanding the structure and purpose of YAML and JSON files in Kubernetes.
- **Monolithic vs Microservice Architectures:** Comparison of monolithic and microservice architectures, and when to use each.
- **Kubernetes:** An overview of Kubernetes and its growing importance in the industry.

Practical:

- **Containerization:** Create Dockerfile , build Docker images for your application, and push them to Docker Hub.
- **Kubernetes Cluster Setup:** Set up a simple Kubernetes cluster using Kind.
- **Kubernetes Deployment:** Create Kubernetes manifest files for both frontend and backend services.

Scaling: Learn how to scale frontend and backend services by tweaking the deployment configurations.

SESSION ON DOCKERS

Virtualization vs Containerization

Picture credits: https://blog.bytebytego.com/p/what-are-the-differences-between

DOCKERS INTRODUCTION

- Docker is an open-source containerization tool. It allows developers to package their applications and dependencies into containers, which can be easily deployed across different environments.

- A container is a package containing all the necessary components required to run an application. It makes it super easy to move your app from one environment to another. You don't have to stress about compatibility issues or anything like that. For instance, a developer can quickly move the app from their laptop to a production server without any hassle

Reference: https://docs.docker.com/guides/docker-overview/

DOCKER HANDS ON

Picture credits: https://dev.to/mattqafouri/microservice-roadmap-4mci

Picture credits: https://blog.bytebytego.com/p/kubernetes-when-and-how-to-apply

kubernetes

SESSION ON KUBERNETES

KUBERENETES 101

1. What is Kubernetes?
2. Kubernetes Architecture
3. Main Kubernetes Components
   i. Pod
   ii. Service
   iii. Ingress
   iv. Config Map & Secret
   v. Volume – Data persistent
   vi. Deployment
   vii. Stateful Set
4. Summary of components
5. Kubernetes Configuration
6. How to write configuration for k8s?

   i. YAML

   ii. Helm package manager

1

WHAT IS KUBERNETES ?

Official definition of Kubernetes

- Open source container orchestration tool
- Developed by Google
- Helps manage containerized applications in different deployment environments

WHAT IS KUBERNETES ?

Need for container orchestration tool

- Trend from Monolith to Microservices
- Increased usage of containers
- Demand for a proper way of managing those hundreds of containers

WHAT IS KUBERNETES ?

What features do orchestration tools offer?

- High **Availability** or no downtime

- **Scalability** or high performance

- **Disaster recovery** - backup and restore

KUBERNETES ARCHITECTURE

KUBERNETES ARCHITECTURE

Control Plane Nodes

handful of master processes

much **more important**

Worker Nodes

higher workload

much bigger and more resources

KUBERNETES ARCHITECTURE

MAIN KUBERNETES COMPONENTS

3.1

PODS

3.1

PODS

- **Smallest** unit in Kubernetes
- **Abstraction** over container

3.1

PODS

- **Smallest** unit in Kubernetes
- **Abstraction** over container
- Usually **1 Application** per Pod
- Each Pod gets its **own IP address**
- **New IP address** on re-creation

3.2

SERVICE

- **Permanent** IP address
- Lifecycle of Pod and Service **not connected**

3.2

SERVICE

- You specify the **type of Service** on creation
- Internal Service is the **default** type

3.2

SERVICE

my-app
my-app
Service
Service
DB
Node
Node
SVC
permanent IP
load balancer

3.3

INGRESS

Ingress
my-app
Service
DB
Service
Node
ing
https://my-app.com
Ingress

3.4

CONFIGMAP & SECRET

3.5

VOLUME – DATA PERSISITENCE

- Storage on **local** machine
- Or **remote,** outside of the K8s cluster

VOLUME – DATA PERSISITENCE

! Kubernetes doesn't manage data persistance!

3.6

DEPLOYMENT

DEPLOYMENT

- **Blueprint** for "my-app" Pods
- You create **Deployments**
- Abstraction of Pods

3.6

DEPLOYMENT

❗ DB can't be replicated via Deployment!

3.7

STATEFUL SET

3.7

STATEFUL SET

Deployment =
for stateLESS Apps

StatefulSet =
for stateFUL Apps or Databases

3.7

STATEFUL SET

DB are often hosted outside of Kubernetes cluster

SUMMARY OF COMPONENTS

SUMMARY OF COMPONENTS

SUMMARY OF COMPONENTS

KUBERNETES CONFIGURATION

Kubernetes Configuration

Deployment =
a template for
creating pods

```yaml
kind: Deployment
metadata:
  name: my-app
  labels:
    app: my-app
spec:
  replicas: 2
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
        - name: my-app
          image: my-image
          env:
            - name: SOME_ENV
              value: $SOME_ENV
          ports:
            - containerPort: 8080
```

KUBERNETES CONFIGURATION

Kubernetes Configuration

- Declarative
- Is == Should

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
  labels:
    app: my-app
spec:
  replicas: 2
  selector:
    matchLabels:
      app: my-app
```

Controller Manager checks:

desired state == actual state ?

PARTS OF CONFIG FILE – METADATA

Each Configuration File has 3 Parts

1) metadata

Deployment

nginx-deployment.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels: ...
spec:
  replicas: 2
  selector: ...
  template: ...
```

Service

nginx-service.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  name: nginx-service
spec:
  selector: ...
  ports: ...
```

PARTS OF CONFIG FILE – SPECS

Each Configuration File has 3 Parts

1) metadata

2) specification

Deployment

nginx-deployment.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels: ...
spec:
  replicas: 2
  selector: ...
  template: ...
```

Service

nginx-service.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  name: nginx-service
spec:
  selector: ...
  ports: ...
```

PARTS OF CONFIG FILE - SPECS

nginx-deployment.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels: ...
spec:
  replicas: 2
  selector: ...
  template: ...
```

nginx-service.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  name: nginx-service
spec:
  selector: ...
  ports: ...
```

- Attributes of "spec" are **specific** to the **kind**

PARTS OF CONFIG FILE - STATUS

```yaml
status:
  availableReplicas: 1
  conditions:
  - lastTransitionTime: "2020-01-24T10:54:59Z"
    lastUpdateTime: "2020-01-24T10:54:59Z"
    message: Deployment has minimum availability
    reason: MinimumReplicasAvailable
    status: "True"
    type: Available
  - lastTransitionTime: "2020-01-24T10:54:56Z"
    lastUpdateTime: "2020-01-24T10:54:59Z"
    message: ReplicaSet "nginx-deployment-7d64f4b
    reason: NewReplicaSetAvailable
    status: "True"
    type: Progressing
  observedGeneration: 1
  readyReplicas: 1
  replicas: 1
  updatedReplicas: 1
```

≠

nginx-deployment.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels: ...
spec:
  replicas: 2
  selector: ...
  template: ...
```

PARTS OF CONFIG FILE - STATUS

HOW TO WRITE KUBE MANIFESTS - YAML

YAML Configuration Files

nginx-deployment.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels: ...
spec:
  replicas: 2
  selector: ...
  template: ...
```

- "**human friendly**" data serialization standard for all programming languages
- syntax: **strict indentation!**
- store the config file with your code

HELM PACKAGE MANAGER (ADVANCED)

- Helm is the best way to find, share, and use software built for Kubernetes.
- Helm helps you manage Kubernetes applications — Helm Charts help you define, install, and upgrade even the most complex Kubernetes application.
- Helm helps you write YAML configuration in a more generic and standard way so that it is production grade ready

HANDSON !!!

WHATS NEXT TO EXPLORE:

Other interesting topics in DevOps & microservices :

- **CI/CD Integration:**

  Introduction to continuous integration and deployment practices.

- **Monitoring and Logging:**

  Overview of tools and practices for monitoring and logging in Kubernetes.

- **Security Best Practices:**

  RBAC (Role-Based Access Control):
  Understanding and implementing RBAC to secure your Kubernetes environment.

- **Large eco system:**

  Kubernetes has very large eco system with new projects and opportunities to that are coming in every single day.

Q&A

USEFUL LINKS

- 1. Kubernetes crash course for absolute beginners (1 hour video) - https://youtu.be/s_o8dwzRlu4
- 2. Github repo link:

https://github.com/Naveen-oops/ToDo-App

https://github.com/Ajay2521/ToDo-App

You can reach out to me on linked In:

https://in.linkedin.com/in/naveenkumar-k-1122841a3

THANK YOU