CLOUD NATIVE
COMPUTING FOUNDATION

Cloud Native Community Groups

New Delhi

April 20, 09:00 AM – 3:00 PM

In-person event

CLOUD NATIVE
COMPUTING FOUNDATION

Navigating through Kubernetes

Krish Aryan

April 20, 2024

- GSoC mentee @OpenSUSE
- CKA and CKAD
- Terraform Associate
- AWS Cloud Practitioner
- CNCG New Delhi volunteer
- Love everything Cloud Native

AGENDA

- Pods
- ReplicaSets
- Deployments
- DaemonSets
- StatefulSets
- Services
- Ingress
- Persistent Volumes
- Persistent Volume Claims
- Storage Classes
- ConfigMaps
- Secrets
- Jobs
- CronJobs

Pods

- ➔ Smallest Unit of work in Kubernetes
- ➔ Pods contain one or more containers
- ➔ Pods are Ephemeral

ReplicaSets

- ➔ Primary method of managing pod replicas and their lifecycle.
- ➔ Includes their scheduling, scaling, and deletion.
- ➔ Their job is simple: **Always ensure the desired number of pods are running.**

Replication Controller (deprecated)

- ➔ original mechanism in Kubernetes for ensuring a desired number of Pod replicas were always running
- ➔ Deprecated because of inflexibility and limited functionality

Deployments

- ➔ Deployments manage Pod lifecycles, guaranteeing app availability.
- ➔ They handle rolling updates and rollbacks for smooth version transitions.
- ➔ Scaling up/down is simplified to meet application demands.
- ➔ Declarative configuration ensures easy management of desired application state.

DaemonSets

- ➔ Deployments manage Pod lifecycles, guaranteeing app availability.
- ➔ They handle rolling updates and rollbacks for smooth version transitions.
- ➔ Scaling up/down is simplified to meet application demands.
- ➔ Declarative configuration ensures easy management of desired application state.

StatefulSets

- ➔ StatefulSets manage stateful applications that require persistent storage and unique identity.
- ➔ They guarantee Pods are created in a specific order and maintain their identity even across restarts.
- ➔ Ideal for databases and other applications that rely on persistent data.

Services

- ➔ Services act as gateways for your applications, providing a single entry point for external users.
- ➔ Services come in different types to handle traffic distribution for your application.
- ➔ They decouple your application logic from network details, simplifying deployment and scaling.

Types of Services

ClusterIP	NodePort	Load Balancer	ExternalName
● Service is accessible only from within the Kubernetes cluster. ● Ideal for internal communication between Pods within the cluster.	● Service is accessible only from within the Kubernetes cluster. ● Ideal for internal communication between Pods within the cluster.	● Service is accessible only from within the Kubernetes cluster. ● Ideal for internal communication between Pods within the cluster.	● Acts as a DNS alias (CNAME record) for an external resource. ● Useful for integrating with existing external services.

Ingress

- ➔ Ingress acts as an API gateway for external traffic entering your Kubernetes cluster.
- ➔ It routes incoming requests to specific services based on defined rules.
- ➔ Ingress simplifies external service exposure, eliminating the need to manage individual IP addresses or ports.
- ➔ It offers features like load balancing, path-based routing, and TLS termination for secure communication.
- ➔ Ingress controllers like NGINX or HAProxy are deployed to manage Ingress resources.

Persistent Storage

- ➔ **Persistent Volume (PV):** Provides raw storage capacity for Pods.
- ➔ **Persistent Volume Claim (PVC):** Represents an application's request for storage, specifying size, access mode, and desired storage class.
- ➔ **Storage Class (SC):** Defines a category of storage with specific characteristics like performance or durability.

Configuration Management

➔ **ConfigMaps:** Store non-sensitive application configuration data like environment variables or configuration files, accessible to all Pods that need them.

➔ **Secrets:** Store sensitive data like passwords, tokens, or API keys that require strict access control. Only authorized Pods can access Secrets.

Jobs and CronJobs

- ➔ **Jobs:** Manage non-repeating tasks within the cluster. They run a defined Pod template to completion and then terminate, ideal for one-time data processing or batch jobs.
- ➔ **CronJobs:** Schedule tasks to run on a recurring basis. They leverage cron expressions to define the execution frequency and trigger the creation of Jobs that perform the desired task. This is perfect for nightly backups, automated reports, or any task requiring repetitive execution.

Custom Resource Definition (CRDs)

- ➔ It extends Kubernetes functionality by defining new resource types.
- ➔ This allows managing custom applications without building a separate API server.
- ➔ CRDs promote modularity and flexibility within the Kubernetes ecosystem.

Benefits and Considerations of CRDs

- **CRD Benefits**:
    - Manage diverse resources
    - Simplified application deployment
- **CRD Considerations**:
    - Requires controller development
    - Advanced Kubernetes concept

THANK YOU

TIME FOR SOME AMA