Important Kubernetes Terminologies

A list of Kubernetes components and features related to containerization, security, and networking, including Cgroups, RBAC, DaemonSet, Ingress, and Network Plugin CNI, among others.

General

- **Cgroups**: Limits and isolates the resource usage of a collection of processes.
- **Namespace**: An abstract environment that can contain a set of pods and provides isolation between different objects in the cluster.
- **RBAC (Role-Based Access Control)**: Manages authorization decisions, allowing admins to dynamically configure access policies through the Kubernetes API.
- **Service Account**: Provides an identity for processes that run in a pod.
- **CustomResourceDefinition**: Custom code that defines a resource to add to your Kubernetes API server without building a complete custom server.
- **Label**: Tags objects with identifying attributes that are meaningful and relevant to users.
- **Selector**: Allows users to filter a list of resources based on labels.
- **Taint**: A key-value pair and an effect to prevent the scheduling of pods on nodes or node groups.
- **Toleration**: A key-value pair and an effect to enable the scheduling of pods on nodes or node groups that have a matching taint.

Pods

- **Deployment**: An API object that manages a replicated application.
- **DaemonSet**: Ensures a copy of a Pod is running across a set of nodes in a cluster.
- **Stateful Set**: Manages the deployment and scaling of a set of Pods, and provides guarantees about the ordering and uniqueness of these Pods.
- **Init Container**: One or more initialization containers that must run to completion before any app containers run.
- **ReplicaSet**: ReplicaSet is the next-generation Replication Controller.

Networking

- **Network Plugin CNI**: Network plugin that provides networking and security for containerized workloads, for example Cilium.
- **Ingress**: Provides external access to the services in a cluster.
- **Load Balancer**: Distributes incoming network traffic to a set of backend pods.
- **Node Port**: Exposes a service on a static port on each worker node in a cluster.
- **Cluster IP**: Exposes a service on a cluster-internal IP address.

Storage Overview - PV, PVC and Storage Class

- **Persistent Volume (PV)** It is a storage resource that can be provisioned by an administrator in a Kubernetes cluster, and made accessible to users. It is a Kubernetes object that represents a storage element in the cluster.
- **Persistent Volume Claim (PVC)** It is a request made by a user or a pod for storage resources in a Kubernetes cluster.
- **Storage Class** It is a way to categorize different storage types in a Kubernetes cluster, based on properties such as storage type, performance, and availability.
- **PV Provisioning** A storageclass is a Kubernetes object that stores information about creating a persistent volume for your pod. With a storageclass, there is no need to create a persistent volume separately before claiming it. When storage class is used to create a PVC, the kubernetes apiserver uses OpenEBS's 'openebs.io/local' provisioner plugin to create the Persistent Volume and claim it.
- **PVC Binding** When a PVC is created and matched with an available PV that meets its storage requirements. Once the PVC and PV are bound, the storage resources specified in the PVC become accessible to the application or pod that requested them.
- **Dynamic Provisioning**: Dynamic volume provisioning allows storage volumes to be created on-demand. Without dynamic provisioning, cluster administrators have to manually make calls to the cloud or storage provider to create new storage volumes, and then create PersistentVolume objects to represent it in Kubernetes.

Kubernetes helps to keep the things you save in a safe place by creating special boxes called "persistent volumes", and when you need to use something from the box, you ask for it by name (persistent volume claim), and Kubernetes gives it to you. A storage class tells Kubernetes how to make these boxes and where to put them.
