Pods in Kubernetes

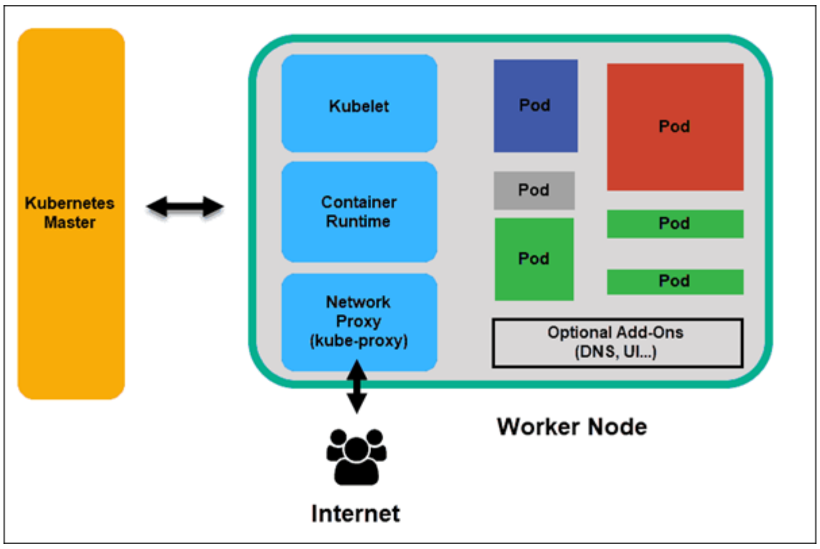
Pods are the smallest deployable units of computing that you can create and manage in **Kubernetes**.

Each Pod contains one or more little boxes called containers, and they all share storage and network resources. Pods are always run together and share a common environment. This environment includes things like different areas for each container, groups for managing resources, and other ways to keep things separate, just like how containers are kept separate, just like how containers are kept separate.

In Kubernetes cluster, we use Pods in two main ways:

Pods that run a single container: The "one-container-per-Pod" model is the most common Kubernetes use case; in this case, you can think of a Pod as a wrapper around a single container; Kubernetes manages Pods rather than managing the containers directly.

Pods that run multiple containers that need to work together: A Pod can encapsulate an application composed of multiple co-located containers that are tightly coupled and need to share resources. These co-located containers form a single cohesive unit.



Sumeha Jakate

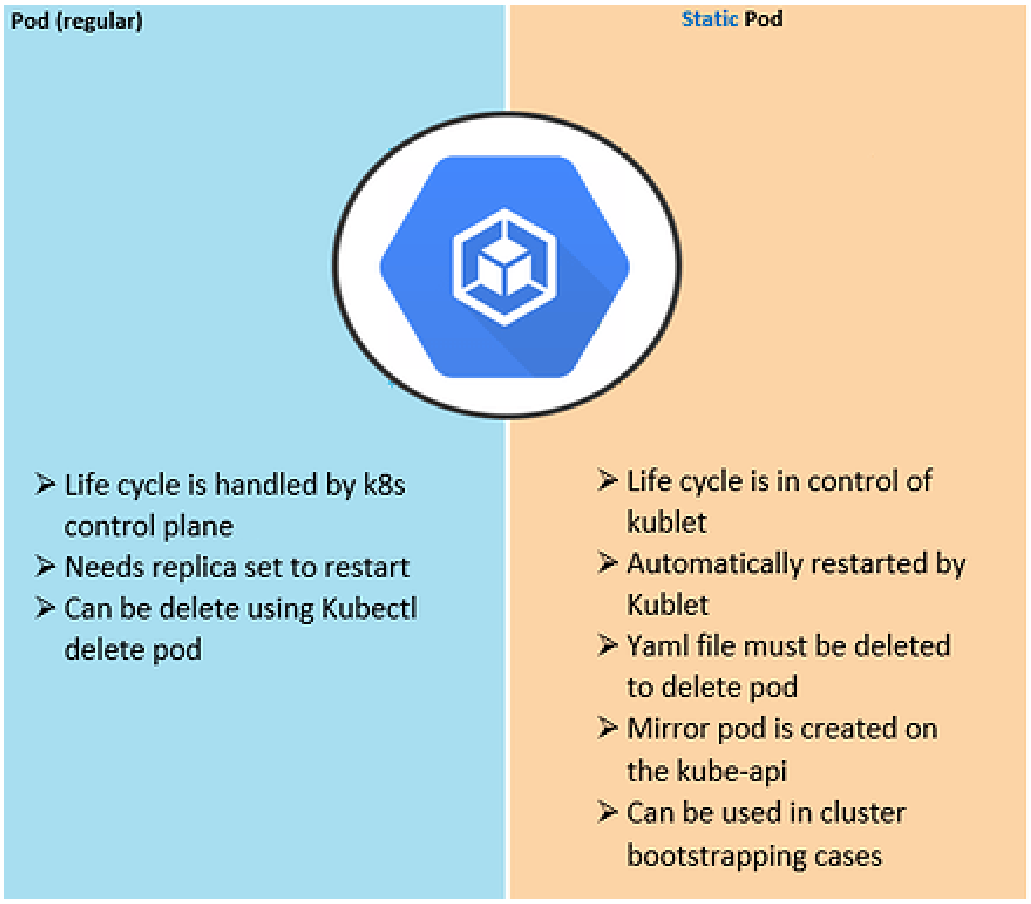
Types of Pods

There are two types of Pods:

1) Normal Pods
2) Static Pods

1) Normal Pods

These Pods are managed by the Kubernetes control plane, like Deployments or StatefulSets. You create them using commands or YAML files, and Kubernetes takes care of scheduling them onto nodes and making sure they stay healthy.



Pod (regular)

- Life cycle is handled by k8s control plane
- Needs replica set to restart
- Can be delete using Kubectl delete pod

Static Pod

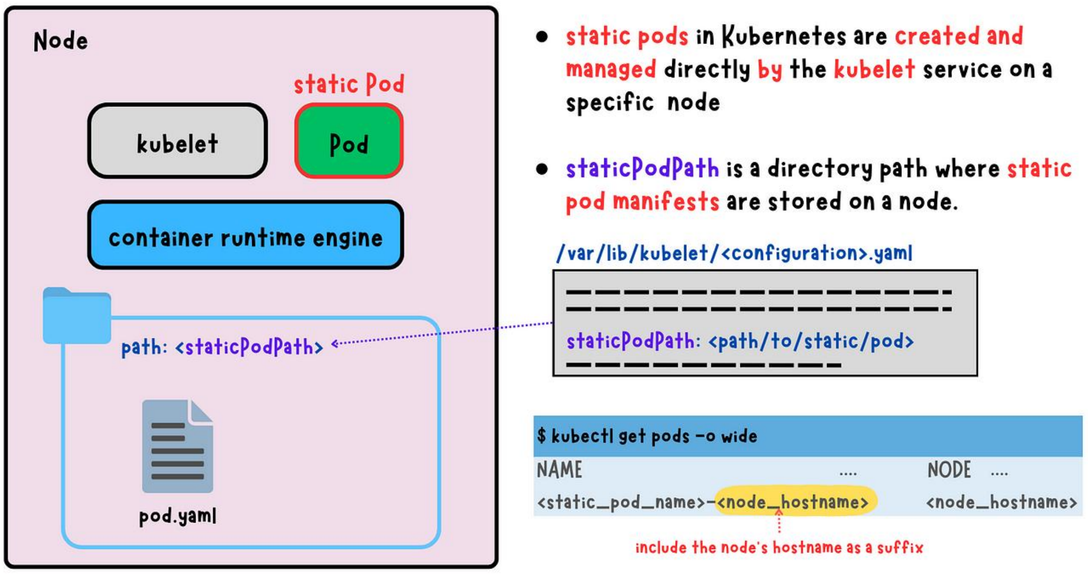
- Life cycle is in control of kublet
- Automatically restarted by Kublet
- Yaml file must be deleted to delete pod
- Mirror pod is created on the kube-api
- Can be used in cluster bootstrapping cases

2) Static Pods

Static Pods are managed directly by the kubelet daemon on a specific node, without the API server observing them. Whereas most Pods are managed by the control plane (for example, a Deployment), for static Pods, the kubelet directly supervises each static Pod (and restarts it if it fails).

Static Pods are always bound to one Kubelet on a specific node. The main use for static Pods is to run a self-hosted control plane: in other words, using the kubelet to supervise the individual control plane components.

The kubelet automatically tries to create a mirror Pod on the Kubernetes API server for each static Pod. This means that the Pods running on a node are visible on the API server but cannot be controlled from there.



- static pods in Kubernetes are created and managed directly by the kubelet service on a specific node
- staticPodPath is a directory path where static pod manifests are stored on a node.

```
staticPodPath: <path/to/static/pod>
```

```
$ kubectl get pods -o wide
NAME                                   ....        NODE     ....
<static_pod_name>-<node_hostname>                  <node_hostname>
```

Basic commands

Basic Pod Management

1. **Create a Pod**:

```
kubectl apply -f pod.yaml
```

2. **List Pods**:
   - All pods in the current namespace:
     ```
     kubectl get pods
     ```
   - Pods across all namespaces:
     ```
     kubectl get pods --all-namespaces
     ```
   - Pods with detailed output:
     ```
     kubectl get pods -o wide
     ```

3. **Describe a Pod**:

```
kubectl describe pod <pod-name>
```

4. **Delete a Pod**:

```
kubectl delete pod <pod-name>
```

   - Delete pods with a label:
     ```
     kubectl delete pods -l <label-key>=<label-value>
     ```

5. **Delete All Pods** in a Namespace:

```
kubectl delete pods --all
```

Interacting with Running Pods

6. **View Logs**:
   - Logs of a pod's main container:

     ```
     kubectl logs <pod-name>
     ```

   - Logs of a specific container in a multi-container pod:

     ```
     kubectl logs <pod-name> -c <container-name>
     ```

   - Stream logs in real-time:

     ```
     kubectl logs -f <pod-name>
     ```

7. **Access a Pod (Exec Command)**:
   - Open a shell session inside a pod:

     ```
     kubectl exec -it <pod-name> -- /bin/bash
     ```

   - Run a specific command:

     ```
     kubectl exec -it <pod-name> -- <command>
     ```