DaemonSet

A DaemonSet in Kubernetes is a workload API object that:

- ❖ Ensures a specific Pod runs on all (or some) nodes in the cluster.
- ❖ Automatically manages Pods on new or removed nodes.
- ❖ Is commonly used for system-level services like logging or monitoring.

Key features of a DaemonSet in Kubernetes include:

1. **Node Coverage**: Ensures a Pod runs on all or specific nodes in the cluster.
2. **Automatic Updates**: Automatically adds or removes Pods when nodes are added or removed.
3. **Resource Consistency**: Ideal for deploying system-level services or monitoring agents consistently across nodes.

1. Create the yamlfile (vi daemon.yaml)

```yaml
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: gowtham-ak
spec:
  selector:
    matchLabels:
      name: gowtham-ak
  template:
    metadata:
      labels:
        name: gowtham-ak
    spec:
      containers:
      - name: gowtham-ak
        image: gowtham-ak-image
~
~
```

2. To verify the Pods:

Kubectl create –f daemon.yaml

```
controlplane $ kubectl create -f daemon.yaml
daemonset.apps/gowtham-ak created
```

3. To check the status:

Kubectl get daemonsets.apps

```
controlplane $ kubectl get daemonsets.apps
NAME         DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR   AGE
gowtham-ak   2         2         0       2            0           <none>          78s
```

4. To check the current pods status:

Kubectl get pod

```
controlplane $ kubectl get pod
NAME               READY   STATUS             RESTARTS   AGE
gowtham-ak-pbwlq   0/1     ImagePullBackOff   0          2m22s
gowtham-ak-szp5r   0/1     ImagePullBackOff   0          2m22s
```

5. To verify the overall status:

Kubectl describe daemonsets.apps

```
controlplane $ kubectl describe daemonsets.apps
Name:           gowtham-ak
Selector:       name=gowtham-ak
Node-Selector:  <none>
Labels:         <none>
Annotations:    deprecated.daemonset.template.generation: 1
Desired Number of Nodes Scheduled: 2
Current Number of Nodes Scheduled: 2
Number of Nodes Scheduled with Up-to-date Pods: 2
Number of Nodes Scheduled with Available Pods: 0
Number of Nodes Misscheduled: 0
Pods Status:  0 Running / 2 Waiting / 0 Succeeded / 0 Failed
Pod Template:
  Labels:  name=gowtham-ak
  Containers:
   gowtham-ak:
    Image:         gowtham-ak-image
    Port:          <none>
    Host Port:     <none>
    Environment:   <none>
    Mounts:        <none>
  Volumes:         <none>
  Node-Selectors:  <none>
  Tolerations:     <none>
```

```
Events:
  Type    Reason            Age    From                Message
  ----    ------            ----   ----                -------
  Normal  SuccessfulCreate  3m42s  daemonset-controller  Created pod: gowtham-ak-szp5r
  Normal  SuccessfulCreate  3m42s  daemonset-controller  Created pod: gowtham-ak-pbwlq
```

6. To again check the pods:

Kubectl get po

```
controlplane $ kubectl get po
NAME               READY   STATUS             RESTARTS   AGE
gowtham-ak-pbwlq   0/1     ImagePullBackOff   0          7m9s
gowtham-ak-szp5r   0/1     ImagePullBackOff   0          7m9s
```

7. And to delete the pods:

Kubectl delete pod gowtham-ak-pbwlq

```
controlplane $ kubectl delete pod gowtham-ak-pbwlq
pod "gowtham-ak-pbwlq" deleted
```

8. Again to check the pods:

Kubectl get po

```
controlplane $ kubectl get po
NAME               READY   STATUS             RESTARTS   AGE
gowtham-ak-bdp7b   0/1     ErrImagePull       0          6s
gowtham-ak-szp5r   0/1     ImagePullBackOff   0          8m12s
```

9. To check the nodes:

Kubectl get nodes

```
controlplane $ kubectl get nodes
NAME           STATUS   ROLES           AGE   VERSION
controlplane   Ready    control-plane   25d   v1.30.0
node01         Ready    <none>          25d   v1.30.0
```

10. To verify the kube system status:

Kubectl get pod –n kube-system

```
controlplane $ kubectl get pod -n kube-system
NAME                                       READY   STATUS    RESTARTS      AGE
calico-kube-controllers-75bdb5b75d-2b6mr   1/1     Running   2 (22m ago)   25d
canal-q652m                                2/2     Running   2 (22m ago)   25d
canal-wzjz6                                2/2     Running   2 (22m ago)   25d
coredns-5c69dbb7bd-6xvhl                   1/1     Running   1 (22m ago)   25d
coredns-5c69dbb7bd-xfk7l                   1/1     Running   1 (22m ago)   25d
etcd-controlplane                          1/1     Running   2 (22m ago)   25d
kube-apiserver-controlplane                1/1     Running   2 (22m ago)   25d
kube-controller-manager-controlplane       1/1     Running   2 (22m ago)   25d
kube-proxy-dp5fn                           1/1     Running   2 (22m ago)   25d
kube-proxy-nhmtq                           1/1     Running   1 (22m ago)   25d
kube-scheduler-controlplane                1/1     Running   2 (22m ago)   25d
```

11. To verify the canal and kube-proxy status:

Kubectl get daemonsets.apps –n kube-system

```
controlplane $ kubectl get daemonsets.apps -n kube-system
NAME         DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR            AGE
canal        2         2         2       2            2           kubernetes.io/os=linux   25d
kube-proxy   2         2         2       2            2           kubernetes.io/os=linux   25d
```