In Kubernetes, cordon and drain are two commands that are used to prepare a node for maintenance.

- **Cordon** marks a node as unschedulable, which means that no new pods will be scheduled on the node. This is done to prevent new pods from being created on the node while it is being maintained.
- **Drain** evicts all pods from a node, including pods that are running and pods that are in a pending state. This is done to ensure that the node is completely empty before it is taken offline for maintenance.

The cordon and drain commands are often used together to prepare a node for maintenance. First, the node is cordoned to prevent new pods from being scheduled on it. Then, the node is drained to evict all existing pods. Once the node is drained, it is safe to perform maintenance on it.

Here is an example of how to use the cordon and drain commands together:

```
kubectl cordon node-1
kubectl drain node-1
```

This will cordon the node named `node-1` and then drain it. All pods on the node will be evicted, and the node will be marked as unschedulable. Once the drain command is complete, the node is safe to perform maintenance on.

It is important to note that the cordon and drain commands can have a significant impact on the availability of your applications. If you are cordoning and draining a node that is hosting critical applications, you should make sure that you have a plan to ensure that those applications are still available while the node is being maintained.

Here are some additional things to keep in mind when using the cordon and drain commands:

- The cordon and drain commands can be used on any node in your cluster, regardless of whether the node is running critical applications.
- The cordon and drain commands can be used to prepare a node for any type of maintenance, including software updates, hardware upgrades, and troubleshooting.
- The cordon and drain commands can be used to prepare a node for removal from the cluster

K8s: cordon & drain

t.me/vishalk17 github.com/vishalk17

Q. I want to evaluate the drain command in MicroK8s. How will it behave? Will it create the pod on another node when it drains it from the first one?

→ we have two nodes:

```
vagrant@k8s-master:~$ kubectl get nodes
NAME           STATUS   ROLES    AGE   VERSION
k8s-master     Ready    <none>   65m   v1.27.4
k8s-worker-1   Ready    <none>   58m   v1.27.4
vagrant@k8s-master:~$
```

- Check no. of pods we have in current namespace

```
vagrant@k8s-master:~$ kubectl get nodes
NAME           STATUS   ROLES    AGE   VERSION
k8s-master     Ready    <none>   65m   v1.27.4
k8s-worker-1   Ready    <none>   58m   v1.27.4
vagrant@k8s-master:~$ ls
httpd.yml  snap
vagrant@k8s-master:~$ kubectl get pods
No resources found in default namespace.
vagrant@k8s-master:~$
```

- Crete pods with 4 replica
  - vi httpd.yml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: apache-deployment
  labels:
    app: apache
spec:
  replicas: 4
  selector:
    matchLabels:
      app: apache
  template:
    metadata:
      labels:
        app: apache
    spec:
      containers:
      - name: apache
        image: httpd:latest
        ports:
        - containerPort: 80
```

- Save, apply

```
vagrant@k8s-master:~$ kubectl apply -f httpd.yml
deployment.apps/apache-deployment created
vagrant@k8s-master:~$
```

- Check where pod has been scheduled

```
vagrant@k8s-master:~$ kubectl get pods -o wide
NAME                                READY   STATUS    RESTARTS   AGE   IP              NODE
apache-deployment-6696dd9569-wssm2  1/1     Running   0          94s   10.1.235.197    k8s-master
apache-deployment-6696dd9569-xzd9r  1/1     Running   0          94s   10.1.230.10     k8s-worker-1
apache-deployment-6696dd9569-zbvf7  1/1     Running   0          94s   10.1.230.11     k8s-worker-1
apache-deployment-6696dd9569-7gjlb  1/1     Running   0          94s   10.1.235.198    k8s-master
vagrant@k8s-master:~$
```

- 2 scheduled on worker node and 2 on master node

- Q. what if I apply cordon on worker node ? Let see

- **Cordon** marks a node as unschedulable, which means that no new pods will be scheduled on the node. This is done to prevent new pods from being created on the node while it is being maintained.

kubectl cordon k8s-worker-1

```
vagrant@k8s-master:~$
vagrant@k8s-master:~$ kubectl get nodes
NAME           STATUS   ROLES    AGE   VERSION
k8s-master     Ready    <none>   75m   v1.27.4
k8s-worker-1   Ready    <none>   68m   v1.27.4
vagrant@k8s-master:~$ kubectl cordon k8s-worker-1
node/k8s-worker-1 cordoned
vagrant@k8s-master:~$ kubectl get nodes
NAME           STATUS                     ROLES    AGE   VERSION
k8s-master     Ready                      <none>   75m   v1.27.4
k8s-worker-1   Ready,SchedulingDisabled   <none>   68m   v1.27.4
vagrant@k8s-master:~$
```

```
11. vagrant master   13. vagrant master
Every 2.0s: microk8s kubectl get pods -o wide                                  k8s

NAME                                READY   STATUS    RESTARTS   AGE     IP              NODE
apache-deployment-6696dd9569-wssm2  1/1     Running   0          8m35s   10.1.235.197    k8s-master
apache-deployment-6696dd9569-xzd9r  1/1     Running   0          8m35s   10.1.230.10     k8s-worker-1
apache-deployment-6696dd9569-zbvf7  1/1     Running   0          8m35s   10.1.230.11     k8s-worker-1
apache-deployment-6696dd9569-7gjlb  1/1     Running   0          8m35s   10.1.235.198    k8s-master
```

```
11. vagrant master   13. vagrant master
Every 2.0s: microk8s kubectl get pods -o wide                                  k8s

NAME                                READY   STATUS    RESTARTS   AGE     IP              NODE
apache-deployment-6696dd9569-wssm2  1/1     Running   0          8m35s   10.1.235.197    k8s-master
apache-deployment-6696dd9569-xzd9r  1/1     Running   0          8m35s   10.1.230.10     k8s-worker-1
apache-deployment-6696dd9569-zbvf7  1/1     Running   0          8m35s   10.1.230.11     k8s-worker-1
apache-deployment-6696dd9569-7gjlb  1/1     Running   0          8m35s   10.1.235.198    k8s-master
```

Increase total no of replication to 6 previously was 4

- Edit httpd.yml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: apache-deployment
  labels:
    app: apache
spec:
  replicas: 6
  selector:
    matchLabels:
      app: apache
  template:
    metadata:
      labels:
        app: apache
    spec:
      containers:
      - name: apache
        image: httpd:latest
        ports:
        - containerPort: 80
```

- save , apply

```
vagrant@k8s-master:~$ vi httpd.yml
vagrant@k8s-master:~$ kubectl apply -f httpd.yml
deployment.apps/apache-deployment configured
vagrant@k8s-master:~$
```

```
Every 2.0s: microk8s kubectl get pods -o wide

NAME                                READY   STATUS    RESTARTS   AGE   IP             NODE
apache-deployment-6696dd9569-wssm2  1/1     Running   0          14m   10.1.235.197   k8s-master
apache-deployment-6696dd9569-xzd9r  1/1     Running   0          14m   10.1.230.10    k8s-worker-1
apache-deployment-6696dd9569-zbvf7  1/1     Running   0          14m   10.1.230.11    k8s-worker-1
apache-deployment-6696dd9569-7gjlb  1/1     Running   0          14m   10.1.235.198   k8s-master
apache-deployment-6696dd9569-98f7k  1/1     Running   0          15s   10.1.235.199   k8s-master
apache-deployment-6696dd9569-6qmlc  1/1     Running   0          15s   10.1.235.200   k8s-master
```

-
- Noticeable changes are,
    - Scheduling got disabled on worker node thus new pod got scheduled on master node
    - No noticeable changed in existing pods behaviour

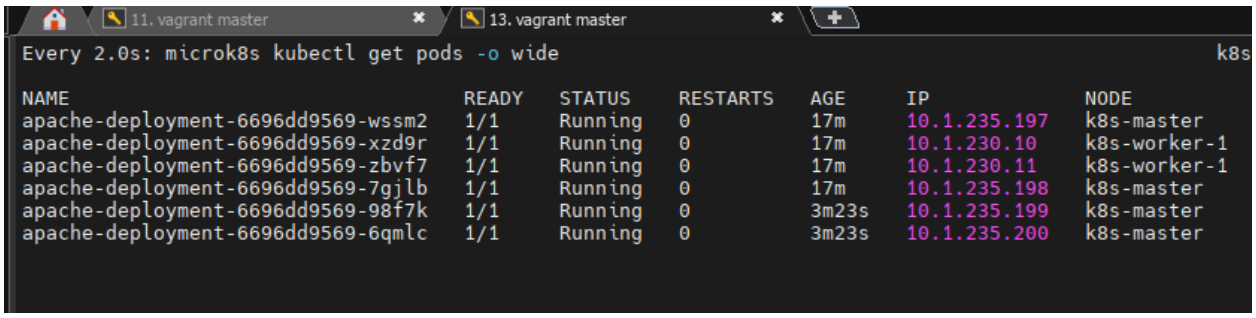
K8s: cordon & drain

t.me/vishalk17 github.com/vishalk17

Q. I want to evaluate drain in microk8s. how it will behave. Will it create the pod on another node, when it drains it from first one?

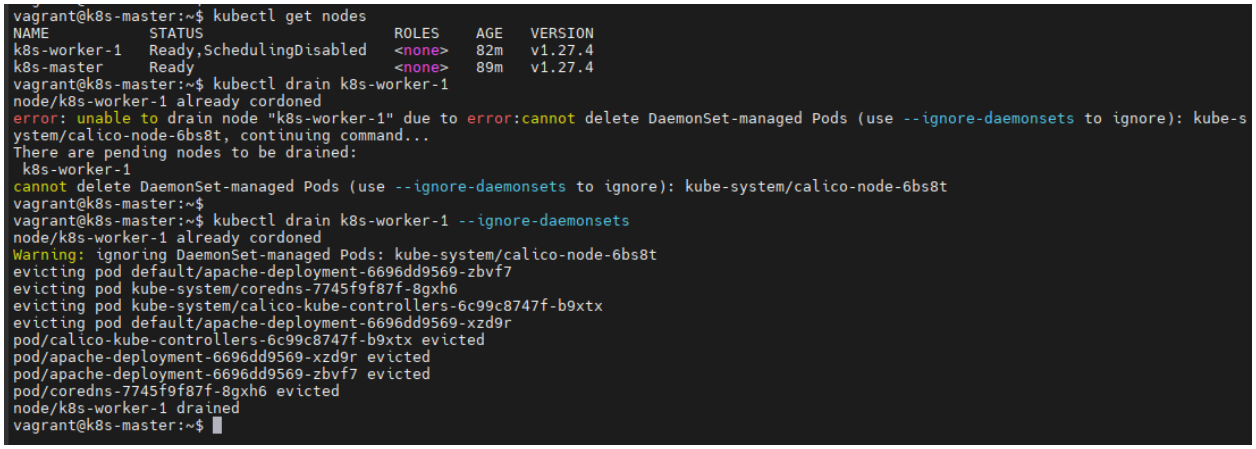
- **Drain** evicts all pods from a node, including pods that are running and pods that are in a pending state. This is done to ensure that the node is completely empty before it is taken offline for maintenance.

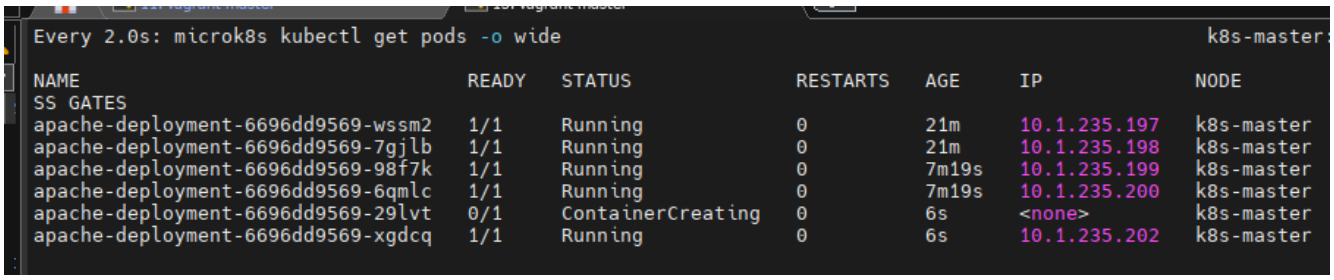
Lets check current state of pod scheduled ,



Lets apply drain on worker node,

- kubectl drain k8s-worker-1





Noticeable changes are as follows:

- Existing pod evicted from worker node and scheduled on or created on another available node , here it is master node.

How to remove Cordon :

Lets check on which node we have applied cordon:

```
vagrant@k8s-master:~$ kubectl get nodes
NAME           STATUS                     ROLES    AGE   VERSION
k8s-master     Ready                      <none>   95m   v1.27.4
k8s-worker-1   Ready,SchedulingDisabled   <none>   88m   v1.27.4
vagrant@k8s-master:~$
vagrant@k8s-master:~$
```

- Remove cordon from worker node

kubectl uncordon k8s-worker-1

```
vagrant@k8s-master:~$ kubectl get nodes
NAME           STATUS                     ROLES    AGE   VERSION
k8s-master     Ready                      <none>   96m   v1.27.4
k8s-worker-1   Ready,SchedulingDisabled   <none>   89m   v1.27.4
vagrant@k8s-master:~$
vagrant@k8s-master:~$
vagrant@k8s-master:~$ kubectl get nodes
NAME           STATUS                     ROLES    AGE   VERSION
k8s-master     Ready                      <none>   96m   v1.27.4
k8s-worker-1   Ready,SchedulingDisabled   <none>   89m   v1.27.4
vagrant@k8s-master:~$
vagrant@k8s-master:~$ kubectl uncordon k8s-worker-1
node/k8s-worker-1 uncordoned
vagrant@k8s-master:~$
vagrant@k8s-master:~$ kubectl get nodes
NAME           STATUS   ROLES    AGE   VERSION
k8s-master     Ready    <none>   96m   v1.27.4
k8s-worker-1   Ready    <none>   89m   v1.27.4
vagrant@k8s-master:~$
```

My devops repo :

- https://github.com/vishalk17/devops

My telegram channel:

- https://t.me/vishalk17_devops

Contact:

Telegram : t.me/vishalk17

vishalk17 My youtube Channel :

- YouTube https://www.youtube.com/@vishalk17

Ref:

DevOps Pro

- YouTube https://youtu.be/EKDmz49BhX8