How to Safely Drain a Kubernetes Node?

Draining a node means **safely evicting all Pods** from a node before maintenance, scaling down, or decommissioning. To ensure **zero downtime** and minimal disruption, follow these steps:

📌 Step 1: Cordon the Node

Before draining, **cordon** the node to prevent new Pods from being scheduled on it.

```
sh

kubectl cordon <node-name>
```

🔷 What does this do?

- Marks the node as **unschedulable**
- Existing Pods keep running, but **no new Pods** will be scheduled here

---

📌 Step 2: Drain the Node

Now, safely evict Pods using:

```
sh

kubectl drain <node-name> --ignore-daemonsets --delete-emptydir-data
```

🔷 Breakdown of options:

- `--ignore-daemonsets`: Skips **DaemonSet Pods** (e.g., logging, monitoring Pods like `fluentd`, `kube-proxy`).
- `--delete-emptydir-data`: Ensures `emptyDir` volumes don't block eviction.

- `--force` *(Optional)*: Use only if Pods without controllers block eviction.

⚠️ **Common Errors & Fixes:**

- **Error: "Cannot delete Pods that declare no controller"**
  🔷 Use `--force` (not recommended unless necessary).

```
sh
-------
kubectl drain <node-name> --ignore-daemonsets --delete-emptydir-data --force
```

- **Error: "PDB prevents eviction"**
  🔷 Check the Pod Disruption Budget (PDB):

```
sh
-------
kubectl get pdb -A
```

If `MIN AVAILABLE` isn't met, Kubernetes won't drain Pods until a replacement is ready.

📌 Step 3: Verify the Node is Drained

Check that the node has **no more scheduled Pods** (except DaemonSets):

```
sh
-------
kubectl get pods --all-namespaces --field-selector spec.nodeName=<node-name>
```

If empty, the node is successfully drained.

📌 Step 4: Uncordon the Node (After Maintenance)

If you want to **bring the node back**, uncordon it so new Pods can be scheduled:

```
sh
-------
kubectl uncordon <node-name>
```

What is the main purpose of draining a node in Kubernetes?

A) To permanently delete all Pods running on the node
B) To safely evict all Pods so that the node can be **rebooted, upgraded, or removed**
C) To delete the node from the cluster
D) To restart the Kubernetes cluster

✅ **Correct Answer: B**

Command to Drain a Node

Which command is used to safely drain a node before maintenance?

A) `kubectl delete node <node-name>`
B) `kubectl drain <node-name> --ignore-daemonsets`
C) `kubectl stop node <node-name>`
D) `kubectl cordon <node-name>`

✅ **Correct Answer: B**

Preventing New Pods from Scheduling

Before draining a node, how can you **prevent new Pods** from being scheduled on it?

A) `kubectl drain <node-name>`
B) `kubectl delete node <node-name>`
C) `kubectl cordon <node-name>`
D) `kubectl stop <node-name>`

✅ **Correct Answer: C**