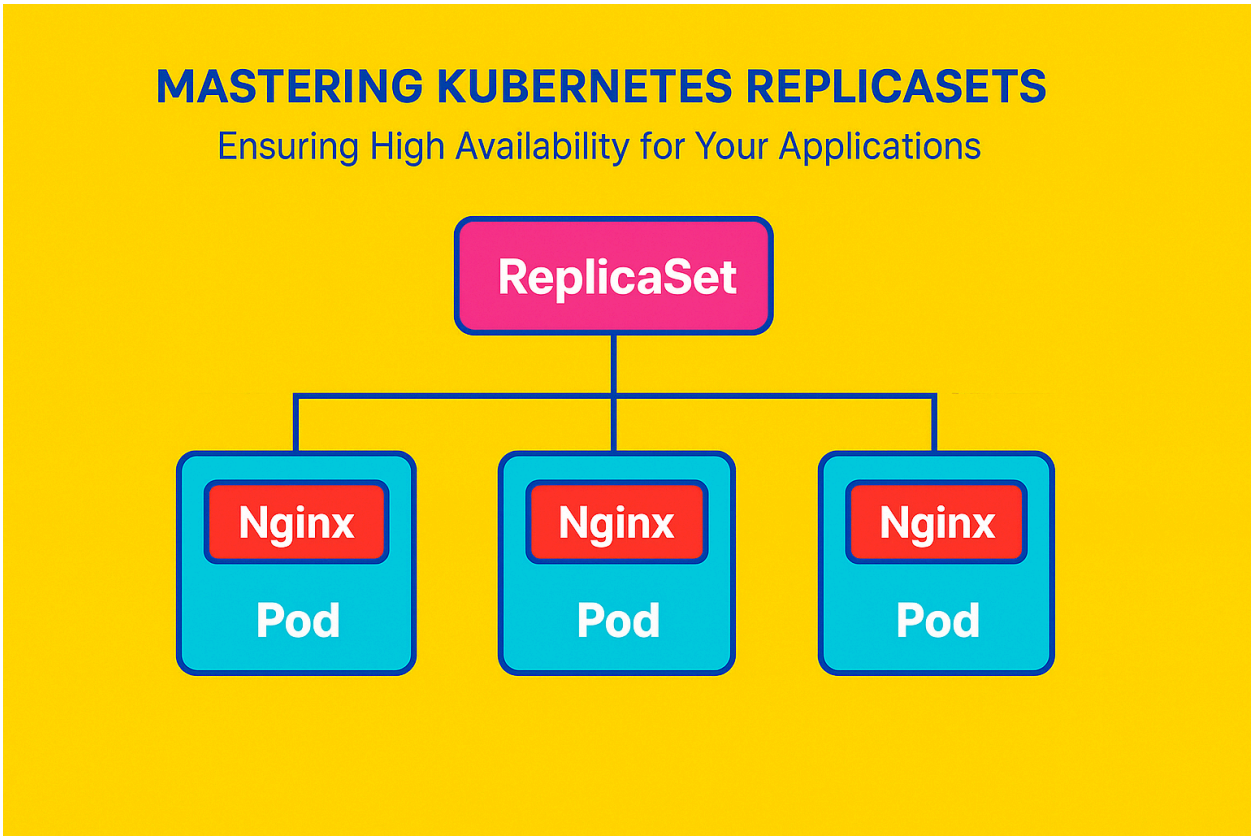


# Creating and Managing a ReplicaSet in Kubernetes

A ReplicaSet in Kubernetes ensures a specified number of Pod replicas are running at any given time. This exercise will guide you through creating a ReplicaSet to maintain the desired

## Prerequisites

· Kubernetes Cluster: Have a running Kubernetes cluster (locally using Minikube or kind, or a cloud-based service).

· kubectl: Install and configure kubectl to interact with your Kubernetes cluster.

· Basic Knowledge of YAML: Familiarity with YAML format will be helpful for understanding Kubernetes resource definitions.

**Step-by-Step Guide**

**Step 1: Understanding ReplicaSet**

A ReplicaSet ensures a specified number of Pod replicas are running at any given time. If a Pod crashes or is deleted, the ReplicaSet creates a new one to meet the defined number of replicas. This helps maintain application availability and ensures that your application can handle increased load by distributing traffic among multiple Pods.

**Step 2:**

```
controlplane ~ ➜ kubectl get nodes
NAME           STATUS   ROLES           AGE   VERSION
controlplane   Ready    control-plane   69m   v1.33.0
node01         Ready    <none>          68m   v1.33.0
node02         Ready    <none>          68m   v1.33.0
```

**Step 3: Create a ReplicaSet**

We'll define a ReplicaSet to maintain three replicas of a simple Nginx web server Pod.

Create a YAML file named replicaset.yaml

```
controlplane ~ ➜ vi replicaset.yaml
```

**Step 4: Display the replicaset yaml file**

```
controlplane ~ ➜ cat replicaset.yaml
apiVersion: apps/v1       # Specifies the API version used.
kind: ReplicaSet          # The type of resource being defined; here, it's a ReplicaSet.
metadata:
  name: nginx-replicaset  # The name of the ReplicaSet.
spec:
  replicas: 3             # The desired number of Pod replicas.
  selector:
    matchLabels:          # Criteria to identify Pods managed by this ReplicaSet.
      app: nginx          # The label that should match Pods.
  template:               # The Pod template for creating new Pods.
    metadata:
      labels:
        app: nginx        # Labels applied to Pods created by this ReplicaSet.
    spec:
      containers:
      - name: nginx       # Name of the container within the Pod.
        image: nginx:latest # Docker image to use for the container.
        ports:
        - containerPort: 80 # The port the container exposes.
```

**Step 5: Apply the YAML to Create the ReplicaSet**

Use the kubectl apply command to create the ReplicaSet based on the YAML file.

```
controlplane ~ ➜ kubectl apply -f replicaset.yaml
replicaset.apps/nginx-replicaset created
```

**Verify the ReplicaSet is running and maintaining the desired number of replicas:**

```
controlplane ~ ➜ kubectl get replicaset
NAME               DESIRED   CURRENT   READY   AGE
nginx-replicaset   3         3         3       27s
```

**To check the Pods created by the ReplicaSet:**

```
controlplane ~ ➜ kubectl get pods -l app=nginx
NAME                      READY   STATUS    RESTARTS   AGE
nginx-replicaset-7tq2z    1/1     Running   0          107s
nginx-replicaset-ftwfb    1/1     Running   0          107s
nginx-replicaset-zpv24    1/1     Running   0          107s
```

**Scaling the ReplicaSet**

**You can scale the number of replicas managed by the ReplicaSet using the kubectl scale command.**

```
controlplane ~ ✖ kubectl scale --replicas=5 replicaset/nginx-replicaset
replicaset.apps/nginx-replicaset scaled
```

```
controlplane ~ ➜ kubectl get replicaset
NAME               DESIRED   CURRENT   READY   AGE
nginx-replicaset   5         5         5       8m38s
```

```
controlplane ~ ➜ kubectl get pods -l app=nginx
NAME                      READY   STATUS    RESTARTS   AGE
nginx-replicaset-7tq2z    1/1     Running   0          3m40s
nginx-replicaset-ftwfb    1/1     Running   0          3m40s
nginx-replicaset-pnf5h    1/1     Running   0          22s
nginx-replicaset-zk9x6    1/1     Running   0          22s
nginx-replicaset-zpv24    1/1     Running   0          3m40s
```