Replicaset

- A ReplicaSet in Kubernetes ensures that a specified number of pod replicas are running at any given time.
- It is responsible for maintaining the desired number of identical pod copies, scaling up or down as needed.
- This helps in achieving fault tolerance and load distribution across the cluster.

Step: 1

Create a Replicaset with three replicas and verify that the pods are running.

```yaml
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: my-replicaset
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
      - name: my-container
        image: nginx
~
```

Step: 2

Apply the YAML file to create the Replicaset

```
controlplane $ kubectl apply -f rc.yaml
replicaset.apps/my-replicaset created
```

Step: 3

Verify that the Replicaset and its pods are running

```
controlplane $ kubectl get pods -l app=my-app
NAME                  READY   STATUS    RESTARTS   AGE
my-replicaset-56k8w   1/1     Running   0          90s
my-replicaset-zr8g6   1/1     Running   0          90s
my-replicaset-zv95n   1/1     Running   0          90s
```

Scaling a Replicaset

- Scaling a ReplicaSet in Kubernetes adjusts the number of pod replicas it manages.
- You can scale a Replica Set by updating its spec.replicas field in the YAML file or by using the kubectl scale command.
- This allows for dynamic adjustment of application capacity based on demand.

Scale up a Replicaset to five replicas and verify that the pods are running.

Step: 1

Scale up the Replicaset to five replicas:

```
controlplane $ kubectl scale rs my-replicaset --replicas=5
replicaset.apps/my-replicaset scaled
```

Step: 2

Verify that the Replicaset and its pods are running:

```
controlplane $ kubectl get pods -l app=my-app
NAME                 READY   STATUS    RESTARTS   AGE
my-replicaset-56k8w  1/1     Running   0          35m
my-replicaset-gh59l  1/1     Running   0          104s
my-replicaset-w9mrd  1/1     Running   0          104s
my-replicaset-zr8g6  1/1     Running   0          35m
my-replicaset-zv95n  1/1     Running   0          35m
```

Updating a Replicaset

- Updating a Replica Set in Kubernetes is typically done by modifying its pod template spec, which necessitates creating a new ReplicaSet.
- To achieve a seamless update, it's recommended to use a Deployment, which manages Replica Sets and allows for rolling updates.
- Directly updating a ReplicaSet is not generally practiced as it can lead to downtime.

Update the image of a Replicaset and verify that the pods are updated.

Step: 1

Update the image of the Replicaset to httpd:latest

```
controlplane $ kubectl set image rs my-replicaset my-container=httpd:latest
replicaset.apps/my-replicaset image updated
```

Step: 2

Verify that the pods are updated:

Kubectl describe rs my-replicaset

```
controlplane $ kubectl describe rs my-replicaset
Name:         my-replicaset
Namespace:    default
Selector:     app=my-app
Labels:       <none>
Annotations:  <none>
Replicas:     5 current / 5 desired
Pods Status:  5 Running / 0 Waiting / 0 Succeeded / 0 Failed
Pod Template:
  Labels:  app=my-app
  Containers:
   my-container:
    Image:         httpd:latest
    Port:          <none>
    Host Port:     <none>
    Environment:   <none>
    Mounts:        <none>
  Volumes:         <none>
  Node-Selectors:  <none>
  Tolerations:     <none>
Events:
  Type    Reason            Age   From                   Message
  ----    ------            ----  ----                   -------
  Normal  SuccessfulCreate  49m   replicaset-controller  Created pod: my-replicaset-zv95n
  Normal  SuccessfulCreate  49m   replicaset-controller  Created pod: my-replicaset-zr8g6
  Normal  SuccessfulCreate  49m   replicaset-controller  Created pod: my-replicaset-56k8w
  Normal  SuccessfulCreate  15m   replicaset-controller  Created pod: my-replicaset-w9mrd
  Normal  SuccessfulCreate  15m   replicaset-controller  Created pod: my-replicaset-gh59l
```

Update the image of the Replicaset to nginx:latest

```
controlplane $ kubectl set image rs my-replicaset my-container=nginx:latest
replicaset.apps/my-replicaset image updated
```

```
controlplane $ kubectl describe rs my-replicaset
Name:         my-replicaset
Namespace:    default
Selector:     app=my-app
Labels:       <none>
Annotations:  <none>
Replicas:     5 current / 5 desired
Pods Status:  5 Running / 0 Waiting / 0 Succeeded / 0 Failed
Pod Template:
  Labels:  app=my-app
  Containers:
   my-container:
    Image:         nginx:latest
    Port:          <none>
    Host Port:     <none>
    Environment:   <none>
    Mounts:        <none>
  Volumes:         <none>
  Node-Selectors:  <none>
  Tolerations:     <none>
```

```
Events:
  Type    Reason            Age   From                   Message
  ----    ------            ----  ----                   -------
  Normal  SuccessfulCreate  51m   replicaset-controller  Created pod: my-replicaset-zv95n
  Normal  SuccessfulCreate  51m   replicaset-controller  Created pod: my-replicaset-zr8g6
  Normal  SuccessfulCreate  51m   replicaset-controller  Created pod: my-replicaset-56k8w
  Normal  SuccessfulCreate  17m   replicaset-controller  Created pod: my-replicaset-w9mrd
  Normal  SuccessfulCreate  17m   replicaset-controller  Created pod: my-replicaset-gh59l
```