Lab: ReplicaSet

Introduction

A **ReplicaSet** purpose is to maintain a stable set of replica Pods running at any given time.

A ReplicaSet is defined with fields, including a selector that specifies how to identify Pods it can acquire, a number of replicas indicating how many Pods it should be maintaining, and a pod template specifying the data of new Pods it should create to meet the number of replicas criteria.

A ReplicaSet then fulfills its purpose by creating and deleting Pods as needed to reach the desired number. When a ReplicaSet needs to create new Pods, it uses its Pod template.

When to use a ReplicaSet

A ReplicaSet ensures that a specified number of pod replicas are running at any given time. However, a Deployment is a higher-level concept that manages ReplicaSets and provides declarative updates to Pods along with a lot of other useful features. Therefore, we recommend using Deployments instead of directly using ReplicaSets, unless you require custom update orchestration or don't require updates at all.

In this Lab, you will learn below items:

Objective:

- Create ReplicaSet
- Scale up/down ReplicaSet
- Cleanup

Note: Ensure you have running cluster deployed

1. Ensure that you have logged-in as **root** user with password as **linux** on **kube-master** node.

1.1 Let us clone the git repository which contains manifests required for this exercise, by executing the below command.

```
# git clone https://github.com/EyesOnCloud/k8s-rs.git
```

Output:

```
Cloning into 'k8s-rs'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
```

1.2 Let us view the manifest file

```
# cat -n ~/k8s-rs/rs-nginx.yaml
```

Output:

```
 1 apiVersion: apps/v1
 2 kind: ReplicaSet
 3 metadata:
 4   name: nginx
 5 spec:
 6   replicas: 3
 7   selector:
 8     matchLabels:
 9       tier: webserver
10   template:
11     metadata:
12       labels:
13         tier: webserver
14     spec:
15       containers:
16       - name: nginx
17         image: nginx
```

1.1 Let us create ReplicaSet by using the **rs-nginx.yaml** file.

```
# kubectl apply -f ~/k8s-rs/rs-nginx.yaml
```

Output:

```
[root@kube-master ~]# kubectl apply -f ~/k8s-rs/rs-nginx.yaml
replicaset.apps/nginx created
```

1.2 Let us list the replicaset, by executing the below command:

```
# kubectl get rs
```

Output:

```
[root@kube-master ~]# kubectl get rs
NAME    DESIRED   CURRENT   READY   AGE
nginx   3         3         3       5m32s
```

1.3 Let us check the details of the ReplicaSet by executing the command:

```
# kubectl describe replicaset nginx
```

Output:

```
[root@kube-master ~]# kubectl describe replicaset nginx
Name:         nginx
Namespace:    default
Selector:     tier=webserver
Labels:       <none>
Annotations:  <none>
Replicas:     3 current / 3 desired
Pods Status:  2 Running / 1 Waiting / 0 Succeeded / 0 Failed
Pod Template:
  Labels:  tier=webserver
  Containers:
   nginx:
    Image:        nginx
    Port:         <none>
    Host Port:    <none>
    Environment:  <none>
    Mounts:       <none>
  Volumes:        <none>
Events:
  Type    Reason            Age   From                   Message
  ----    ------            ----  ----                   -------
  Normal  SuccessfulCreate  10s   replicaset-controller  Created pod: nginx-wkxn6
  Normal  SuccessfulCreate  10s   replicaset-controller  Created pod: nginx-6759z
  Normal  SuccessfulCreate  10s   replicaset-controller  Created pod: nginx-fk7bq
```

1.4 Let us list the resources, by executing the below command:

```
# kubectl get all
```

Output:

```
[root@kube-master ~]# kubectl get all
NAME                READY    STATUS     RESTARTS    AGE
pod/nginx-6759z     1/1      Running    0           21s
pod/nginx-fk7bq     1/1      Running    0           21s
pod/nginx-wkxn6     1/1      Running    0           21s

NAME                    TYPE         CLUSTER-IP    EXTERNAL-IP    PORT(S)    AGE
service/kubernetes      ClusterIP    10.96.0.1     <none>         443/TCP    114m

NAME                          DESIRED    CURRENT    READY    AGE
replicaset.apps/nginx         3          3          3        21s
```

Scaling ReplicaSets

You can easily change the number of pods a particular ReplicaSet manages in one of two ways:

a. Edit the controller's configuration by using **kubectl edit rs ReplicaSet_name** and change the replicas count up or down as you desire.
b. Use kubectl directly. For example, **kubectl scale --replicas=2 rs/name**.

1.5 Let us list the replicaset, by executing the below command:

```
# kubectl get rs nginx
```

Output:

```
[root@kube-master ~]# kubectl get rs nginx
NAME     DESIRED    CURRENT    READY    AGE
nginx    3          3          3        35s
```

1.6 Let us verify the pod labels to understand how RS is managing the replicas, by executing the below command.

```
# kubectl get pods --show-labels
```

Output:

```
[root@kube-master ~]# kubectl get pods --show-labels
NAME           READY    STATUS     RESTARTS    AGE      LABELS
nginx-bpjjq    1/1      Running    0           7m24s    tier=webserver
nginx-nw85v    1/1      Running    0           7m24s    tier=webserver
nginx-pjxzp    1/1      Running    0           7m24s    tier=webserver
```

1.7 Let us scale up the nginx app to 4 replicas, by executing below command

```
# kubectl scale --replicas=4 rs/nginx
```

Output:

```
[root@kube-master ~]# kubectl scale --replicas=4 rs/nginx
replicaset.apps/nginx scaled
```

1.8 Let us list the details, by executing the below command.

```
# kubectl get all
```

Output:

```
[root@kube-master ~]# kubectl get all
NAME              READY   STATUS    RESTARTS   AGE
pod/nginx-6759z   1/1     Running   0          84s
pod/nginx-fk7bq   1/1     Running   0          84s
pod/nginx-wkxn6   1/1     Running   0          84s
pod/nginx-xmzn4   1/1     Running   0          36s

NAME                 TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
service/kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP   115m

NAME                    DESIRED   CURRENT   READY   AGE
replicaset.apps/nginx   4         4         4       84s
```

1.9 Let us scale down the nginx app to 2 replicas, by executing below command

```
# kubectl scale --replicas=2 rs/nginx
```

Output:

```
[root@kube-master ~]# kubectl scale --replicas=2 rs/nginx
replicaset.apps/nginx scaled
```

1.10 Let us list the details, by executing the below command.

```
# kubectl get all
```

Output:

```
[root@kube-master ~]# kubectl get all
NAME                 READY   STATUS    RESTARTS   AGE
pod/nginx-65gg8      1/1     Running   0          4m36s
pod/nginx-9rlz8      1/1     Running   0          4m36s

NAME                 TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
service/kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP   129m

NAME                       DESIRED   CURRENT   READY   AGE
replicaset.apps/nginx      2         2         2       4m36s
```

1.11 Let us clean up by deleting the replica set and notice that it deletes the pods as well

```
# kubectl delete rs/nginx
```

Output:

```
[root@kube-master ~]# kubectl delete rs/nginx
replicaset.apps "nginx" deleted
```