

Controllers are brain behind Kubernetes .
They are processes that monitor Kubernetes objects and respond accordingly .

Replication controller :

If we have single pod and pod crashes due to some reason . Users wont be able to access the application .
To prevent users from losing access to applications . we would like to have more than 1 instance or pod running at the same time .

If one fails we still have our application running on other one .

Replication controller helps us run multiple instances of single pod in kubernetes cluster . thus provides high availability .

Even if we have a single pod replication controller can help by automatically bringing up new pod when existing one fails .

Thus replication controller ensures that specified number of pod are running at all times .

We can create multiple pods to share load across them .

```
rc-definition.yml
apiVersion: v1
kind: ReplicationController
metadata:
  name: myapp-rc
  labels:
    app: myapp
    type: front-end
spec:
  template:
    POD
```

```
pod-definition.yml
apiVersion: v1
kind: Pod
metadata:
  name: myapp-pod
  labels:
    app: myapp
    type: front-end
spec:
  containers:
  - name: nginx-container
    image: nginx
```

```
rc-definition.yml
apiVersion: v1
kind: ReplicationController
metadata:
  name: myapp-rc
  labels:
    app: myapp
    type: front-end
spec:
  template:
    metadata:
      name: myapp-pod
      labels:
        app: myapp
        type: front-end
    spec:
      containers:
      - name: nginx-container
        image: nginx
```

```
pod-definition.yml
apiVersion: v1
kind: Pod
```

We have nested two definition files together – replication controller being the parent and pod definition being the child .

To specify how many replicas we need in replication controller .

rc-definition.yml

```
apiVersion: v1
kind: ReplicationController
metadata:
  name: myapp-rc
  labels:
    app: myapp
    type: front-end
spec:
  template:
    metadata:
      name: myapp-pod
      labels:
        app: myapp
        type: front-end
    spec:
      containers:
        - name: nginx-container
          image: nginx
  replicas: 3
```

pod-definition.yml

```
apiVersion: v1
kind: Pod
```

```
> kubectl create -f rc-definition.yml
```

```
replicationcontroller "myapp-rc" created
```

```
> kubectl get replicationcontroller
```

NAME	DESIRED	CURRENT	READY	AGE
myapp-rc	3	3	3	19s

Replicaset :

Replicaset requires a selector definition – selector section helps replica set identify what pod falls under it , Replicaset can also manage pod that were not created as part of replicaset creation .

replicaset-definition.yml

```
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: myapp-replicaset
  labels:
    app: myapp
    type: front-end
spec:
  template:
    metadata:
      name: myapp-pod
      labels:
        app: myapp
        type: front-end
    spec:
      containers:
        - name: nginx-container
          image: nginx
  replicas: 3
  selector:
    matchLabels:
      type: front-end
```

pod-definition.yml

```
apiVersion: v1
kind: Pod
```

```
> kubectl create -f replicaset-definition.yml
```

```
replicaset "myapp-replicaset" created
```

```
> kubectl get replicaset
```

NAME	DESIRED	CURRENT	READY	AGE
myapp-replicaset	3	3	3	19s

```
> kubectl get pods
```

NAME	READY	STATUS	RESTARTS	AGE
myapp-replicaset-9dd19	1/1	Running	0	45s
myapp-replicaset-9jtpx	1/1	Running	0	45s
myapp-replicaset-hq84m	1/1	Running	0	45s



Create replica set :

```
! replicaset.yaml ●
ts > ! replicaset.yaml > {} spec > {} template > {} metadata > {} labels
1  apiVersion: apps/v1
2  kind: ReplicaSet
3  metadata:
4    name: myapp-replicaset
5    labels:
6      app: myapp
7  spec:
8    selector:
9      matchLabels:
10       app: myapp
11    replicas: 3
12    template:
13      metadata:
14        name: nginx-2
15        labels:
16          app: myapp
17      spec:
18        containers:
19          - name: nginx
20            image: nginx

! nginx.yaml ×
pods > ! nginx.yaml > {} metadata
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    name: nginx-2
5    labels:
6      env: production
7  spec:
8    containers:
9      - name: nginx
10        image: nginx
11
12
```

```
admin@ubuntu-server replicasets # kubectl create -f replicaset.yaml
admin@ubuntu-server replicasets # kubectl get replicaset
NAME                DESIRED   CURRENT   READY   AGE
myapp-replicaset    3         3         3       8s
admin@ubuntu-server replicasets # kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
myapp-replicaset-8nxxl             1/1     Running   0          24s
myapp-replicaset-jlgr2             1/1     Running   0          24s
myapp-replicaset-pm4rl             1/1     Running   0          24s
```

If we delete any pod

```
admin@ubuntu-server replicasets # kubectl delete pod myapp-replicaset-8nxxl
pod "myapp-replicaset-8nxxl" deleted
admin@ubuntu-server replicasets # kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
myapp-replicaset-bvlst             1/1     Running   0          15s
myapp-replicaset-jlgr2             1/1     Running   0          76s
myapp-replicaset-pm4rl             1/1     Running   0          76s
```

New pod will be created again by replicaset .

```
admin@ubuntu-server replicasets #  
admin@ubuntu-server replicasets #  
admin@ubuntu-server replicasets # kubectl describe replicaset myapp-replicaset  
Name: myapp-replicaset  
Namespace: default  
Selector: app=myapp  
Labels: app=myapp  
Annotations: <none>  
Replicas: 3 current / 3 desired  
Pods Status: 3 Running / 0 Waiting / 0 Succeeded / 0 Failed  
Pod Template:  
  Labels: app=myapp  
  Containers:  
    nginx:  
      Image: nginx  
      Port: <none>  
      Host Port: <none>  
      Environment: <none>  
      Mounts: <none>  
      Volumes: <none>  
Events:  
  Type      Reason      Age   From      Message  
  ----      -  
Normal     SuccessfulCreate   116s   replicaset-controller   Created pod: myapp-replicaset
```

Replicaset ensures minimum number of replica are available all time .

What if there are more number of replicas then whats required ?

Lets create new pod with same label that replicaset selector uses .

```
admin@ubuntu-server pods # cat nginx.yaml  
apiVersion: v1  
kind: Pod  
metadata:  
  name: nginx-2  
  labels:  
    app: myapp  
spec:  
  containers:  
  - name: nginx  
    image: nginxadmin@ubuntu-server pods #
```

Now lets create pod directly not through replicaset .

```
admin@ubuntu-server pods # kubectl create -f nginx.yaml  
pod/nginx-2 created  
admin@ubuntu-server pods # kubectl get pods  
NAME                READY   STATUS    RESTARTS   AGE  
myapp-replicaset-bvlst 1/1     Running   0           2m56s  
myapp-replicaset-jlgr2 1/1     Running   0           3m57s  
myapp-replicaset-pm4rl 1/1     Running   0           3m57s  
nginx-2              0/1     Terminating 0           4s
```

Replicaset is terminating the new pods that we just created , its not allowing more pods with the same labels than the number of replicas configured on the replicaset

```

admin@ubuntu-server pods # kubectl describe replicaset myapp-replicaset
Name:          myapp-replicaset
Namespace:     default
Selector:      app=myapp
Labels:        app=myapp
Annotations:   <none>
Replicas:      3 current / 3 desired
Pods Status:   3 Running / 1 Waiting / 0 Succeeded / 0 Failed
Pod Template:
  Labels:  app=myapp
  Containers:
    nginx:
      Image:        nginx
      Port:         <none>
      Host Port:    <none>
      Environment:  <none>
      Mounts:       <none>
      Volumes:      <none>
Events:
  Type     Reason
  ----     -
Normal    SuccessfulCreate 4m3s replicaset-controller Created pod: m
Replicaset controller deletes new pod of nginx that we just created .

```

under the events section

of the output of describe command, ge

```

admin@ubuntu-server pods # kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
myapp-replicaset-bvlst             1/1     Running   0           3m33s
myapp-replicaset-jlgr2             1/1     Running   0           4m34s
myapp-replicaset-pm4rl             1/1     Running   0           4m34s
nginx-2                            0/1     Terminating 0           41s
admin@ubuntu-server pods # kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
myapp-replicaset-bvlst             1/1     Running   0           3m45s
myapp-replicaset-jlgr2             1/1     Running   0           4m46s
myapp-replicaset-pm4rl             1/1     Running   0           4m46s
admin@ubuntu-server pods #

```

Scale up application :

We must edit replicaset definition file and update its replica count to four .

Edit replicas to 4 .

```

admin@ubuntu-server replicasets # kubectl edit replicaset myapp-replicaset
replicaset.apps/myapp-replicaset edited
admin@ubuntu-server replicasets # et^C
admin@ubuntu-server replicasets # kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
myapp-replicaset-bvlst             1/1     Running   0           4m48s
myapp-replicaset-cssz8             1/1     Running   0           6s
myapp-replicaset-jlgr2             1/1     Running   0           5m49s
myapp-replicaset-pm4rl             1/1     Running   0           5m49s

```



```

admin@ubuntu-server replicasets # kubectl scale replicaset myapp-replicaset --replicas=2
replicaset.apps/myapp-replicaset scaled
admin@ubuntu-server replicasets #
admin@ubuntu-server replicasets #
admin@ubuntu-server replicasets #
admin@ubuntu-server replicasets # kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
myapp-replicaset-bvlst             0/1     Terminating    0          5m30s
myapp-replicaset-cssz8             0/1     Terminating    0          48s
myapp-replicaset-jlgr2             1/1     Running         0          6m31s
myapp-replicaset-pm4rl             1/1     Running         0          6m31s

```

It is scaling down to 2 replicas .

```

controlplane ~ → kubectl get pods
No resources found in default namespace.

```

How many PODs exist on the system? 0

```

controlplane ~ → kubectl get replicaset
No resources found in default namespace.

```

How many ReplicaSets exist on the system? 0

```

controlplane ~ → kubectl get replicaset
NAME                DESIRED   CURRENT   READY   AGE
new-replica-set     4         4         0       14s

```

How about now? How many ReplicaSets do you see? 1

How many PODs are DESIRED in the new-replica-set? 4

What is the image used to create the pods in the new-replica-set? Busybox777

```

controlplane ~ → kubectl describe replicaset
Name:                new-replica-set
Namespace:           default
Selector:             name=busybox-pod
Labels:               <none>
Annotations:          <none>
Replicas:             4 current / 4 desired
Pods Status:          0 Running / 4 Waiting / 0 Succeeded / 0 Failed
Pod Template:
  Labels:             name=busybox-pod
  Containers:
    busybox-container:
      Image:           busybox777
      Port:            <none>
      Host Port:       <none>
      Command:
        sh
        -c
        echo Hello Kubernetes! && sleep 3600
  Environment:        <none>
  Mounts:              <none>
  Volumes:             <none>
Events:
  Type    Reason             Age    From                      Message
  ----    -
  Normal  SuccessfulCreate   2m34s  replicaset-controller    Created pod: new-replica-set-5kfk5
  Normal  SuccessfulCreate   2m34s  replicaset-controller    Created pod: new-replica-set-8x5qn
  Normal  SuccessfulCreate   2m34s  replicaset-controller    Created pod: new-replica-set-vvdxl
  Normal  SuccessfulCreate   2m34s  replicaset-controller    Created pod: new-replica-set-lrcnw

```

How many PODs are READY in the new-replica-set? 0

Why do you think the PODs are not ready? Image doesn't exist

Kubectl describe pods

```
Events:
  Type      Reason      Age          From          Message
  ----      -
  Normal    Scheduled   4m23s        default-scheduler   Successfully assigned default/new-replica-set-lrcnw to controlplane
  Normal    Pulling     2m47s (x4 over 4m21s)  kubelet           Pulling image "busybox777"
  Warning   Failed      2m46s (x4 over 4m21s)  kubelet           Failed to pull image "busybox777": rpc error: code = Unknown desc = failed to pull and unpack image "docker.io/library/busybox777:latest": failed to resolve reference "docker.io/library/busybox777:latest": pull access denied, repository does not exist or may not be accessible
  Warning   Failed      2m46s (x4 over 4m21s)  kubelet           Error: ErrImagePull
  Warning   Failed      2m33s (x6 over 4m20s)  kubelet           Error: ImagePullBackOff
  Normal    BackOff     2m19s (x7 over 4m20s)  kubelet           Back-off pulling image "busybox777"
```

Delete any one of the 4 PODs.

```
controlplane ~ → kubectl get pods
NAME                                READY   STATUS              RESTARTS   AGE
new-replica-set-5kfk5              0/1     ImagePullBackOff    0           5m21s
new-replica-set-vvdx1              0/1     ImagePullBackOff    0           5m21s
new-replica-set-8x5qn              0/1     ImagePullBackOff    0           5m21s
new-replica-set-lrcnw              0/1     ImagePullBackOff    0           5m21s

controlplane ~ → kubectl delete pod new-replica-set-5kfk5
pod "new-replica-set-5kfk5" deleted
```

How many PODs exist now? 4

```
controlplane ~ → kubectl get pods
NAME                                READY   STATUS              RESTARTS   AGE
new-replica-set-hnztf              0/1     ErrImagePull        0           34s
new-replica-set-8x5qn              0/1     ErrImagePull        0           6m9s
new-replica-set-lrcnw              0/1     ErrImagePull        0           6m9s
new-replica-set-vvdx1              0/1     ImagePullBackOff    0           6m9s
```

Why are there still 4 PODs, even after you deleted one? Replicaset ensures desired number of pods are always available .

Create a ReplicaSet using the replicaset-definition-1.yaml file located at /root/.

```
controlplane ~ → cat replicaset-definition-1.yaml
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: replicaset-1
spec:
  replicas: 2
  selector:
    matchLabels:
      tier: frontend
  template:
    metadata:
      labels:
        tier: frontend
    spec:
      containers:
        - name: nginx
          image: nginx

controlplane ~ → kubectl create -f replicaset-definition-1.yaml
replicaset.apps/replicaset-1 created
```

Fix the issue in the replicaset-definition-2.yaml file and create a ReplicaSet using it.
This file is located at /root/.

```
controlplane ~ → cat replicaset-definition-2.yaml
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: replicaset-2
spec:
  replicas: 2
  selector:
    matchLabels:
      tier: nginx
  template:
    metadata:
      labels:
        tier: nginx
    spec:
      containers:
      - name: nginx
        image: nginx

controlplane ~ → kubectl create -f replicaset-definition-2.yaml
replicaset.apps/replicaset-2 created
```

Delete the two newly created ReplicaSets - replicaset-1 and replicaset-2

```
controlplane ~ → kubectl get pods
NAME                                READY   STATUS              RESTARTS   AGE
replicaset-1-h56jz                  1/1     Running             0           4m14s
replicaset-1-bw2qn                  1/1     Running             0           4m14s
new-replica-set-vvdx1               0/1     ImagePullBackOff    0           13m
new-replica-set-lrcnw               0/1     ImagePullBackOff    0           13m
new-replica-set-8x5qn               0/1     ImagePullBackOff    0           13m
new-replica-set-hnztf               0/1     ImagePullBackOff    0           7m52s
replicaset-2-tfjzf                  1/1     Running             0           37s
replicaset-2-kjssq                  1/1     Running             0           37s

controlplane ~ → kubectl get replicaset
NAME                DESIRED   CURRENT   READY   AGE
new-replica-set     4         4         0       13m
replicaset-1        2         2         2       4m30s
replicaset-2        2         2         2       53s

controlplane ~ → kubectl delete replicaset replicaset-1 replicaset-2
replicaset.apps "replicaset-1" deleted
replicaset.apps "replicaset-2" deleted

controlplane ~ → kubectl get replicaset
NAME                DESIRED   CURRENT   READY   AGE
new-replica-set     4         4         0       14m

controlplane ~ → kubectl get pods
NAME                                READY   STATUS              RESTARTS   AGE
new-replica-set-vvdx1               0/1     ImagePullBackOff    0           14m
new-replica-set-lrcnw               0/1     ImagePullBackOff    0           14m
new-replica-set-8x5qn               0/1     ImagePullBackOff    0           14m
new-replica-set-hnztf               0/1     ImagePullBackOff    0           8m41s
```


Fix the original replica set new-replica-set to use the correct busybox image.

Either delete and recreate the ReplicaSet or Update the existing ReplicaSet and then delete all PODs, so new ones with the correct image will be created.

```
controlplane ~ → kubectl get replicaset
NAME          DESIRED  CURRENT  READY  AGE
new-replica-set 4         4        0      14m

controlplane ~ → kubectl get pods
NAME          READY  STATUS             RESTARTS  AGE
new-replica-set-vvdxl 0/1    ImagePullBackOff    0          14m
new-replica-set-lrcnw 0/1    ImagePullBackOff    0          14m
new-replica-set-8x5qn 0/1    ImagePullBackOff    0          14m
new-replica-set-hnztf 0/1    ImagePullBackOff    0          8m41s

controlplane ~ → kubectl edit replicaset new-replica-set
replicaset.apps/new-replica-set edited

controlplane ~ → kubectl get pods
NAME          READY  STATUS             RESTARTS  AGE
new-replica-set-vvdxl 0/1    ImagePullBackOff    0          15m
new-replica-set-lrcnw 0/1    ImagePullBackOff    0          15m
new-replica-set-8x5qn 0/1    ImagePullBackOff    0          15m
new-replica-set-hnztf 0/1    ImagePullBackOff    0          10m

controlplane ~ → kubectl delete pods new-replica-set-vvdxl new-replica-set-lrcnw new-replica-set-8x5qn new-replica-set-hnztf
pod "new-replica-set-vvdxl" deleted
pod "new-replica-set-lrcnw" deleted
pod "new-replica-set-8x5qn" deleted
pod "new-replica-set-hnztf" deleted

controlplane ~ → kubectl get pods
NAME          READY  STATUS   RESTARTS  AGE
new-replica-set-h9nx8 1/1    Running  0          27s
new-replica-set-g9hh5 1/1    Running  0          27s
new-replica-set-qshcb 1/1    Running  0          27s
new-replica-set-zfkqn 1/1    Running  0          27s
```

Scale the ReplicaSet to 5 PODs.

Use kubectl scale command or edit the replicaset using kubectl edit replicaset.

```
controlplane ~ → kubectl edit replicaset new-replica-set
Edit cancelled, no changes made.

controlplane ~ → kubectl scale rs new-replica-set --replicas=5
replicaset.apps/new-replica-set scaled

controlplane ~ → kubectl get pods
NAME          READY  STATUS   RESTARTS  AGE
new-replica-set-h9nx8 1/1    Running  0          117s
new-replica-set-g9hh5 1/1    Running  0          117s
new-replica-set-qshcb 1/1    Running  0          117s
new-replica-set-zfkqn 1/1    Running  0          117s
new-replica-set-67g9s 1/1    Running  0           6s
```

Now scale the ReplicaSet down to 2 PODs.

Use the kubectl scale command or edit the replicaset using kubectl edit replicaset

```
controlplane ~ → kubectl scale rs new-replica-set --replicas=2
replicaset.apps/new-replica-set scaled

controlplane ~ → kubectl get pods
NAME          READY  STATUS   RESTARTS  AGE
new-replica-set-qshcb 1/1    Running  0          2m53s
new-replica-set-zfkqn 1/1    Running  0          2m53s
new-replica-set-h9nx8 1/1    Terminating 0          2m53s
new-replica-set-g9hh5 1/1    Terminating 0          2m53s
new-replica-set-67g9s 1/1    Terminating 0           62s
```