⬤Star	898	⬤Fork	152	👁Watch	52	⬤Follow @collabnix	0

Deployment 101

Kubernetes - Beginners | Intermediate | Advanced

View on GitHub   Join Slack

Deployment 101

We looked at ReplicaSets earlier. However, ReplicaSet have one major drawback: once you select the pods that are managed by a ReplicaSet, you cannot change their pod templates.

For example, if you are using a ReplicaSet to deploy four pods with NodeJS running and you want to change the NodeJS image to a newer version, you need to delete the ReplicaSet and recreate it. Restarting the pods causes downtime till the images are available and the pods are running again.

A Deployment resource uses a ReplicaSet to manage the pods. However, it handles updating them in a controlled way. Let's dig deeper into Deployment Controllers and patterns.

Creating Your First Deployment

The following Deployment definition deploys four pods with nginx as their hosted application:

```
git clone https://github.com/collabnix/dockerlabs
cd dockerlabs/kubernetes/workshop/Deployment101
kubectl create -f nginx-dep.yaml
deployment.apps/nginx-deployment created
```

Checking the list of application deployment

To list your deployments use the get deployments command:

```
$ kubectl get deployments
NAME              READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deployment  2/2     2            2           63s
```

```
[node1 Deployment101]$ kubectl describe deploy
Name:                   nginx-deployment
Namespace:              default
CreationTimestamp:      Mon, 30 Dec 2019 07:10:33 +0000
Labels:                 <none>
Annotations:            deployment.kubernetes.io/revision: 1
Selector:               app=nginx
Replicas:               2 desired | 2 updated | 2 total | 0 available | 2 un
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=nginx
  Containers:
   nginx:
    Image:         nginx:1.7.9
    Port:          80/TCP
    Host Port:     0/TCP
    Environment:   <none>
    Mounts:        <none>
  Volumes:         <none>
Conditions:
  Type            Status  Reason
```

```
     ————         ——————   ——————
  Available       False    MinimumReplicasUnavailable
  Progressing     True     ReplicaSetUpdated
OldReplicaSets:  <none>
NewReplicaSet:   nginx-deployment-6dd86d77d (2/2 replicas created)
Events:
  Type    Reason            Age    From                  Message
  ————    ——————            ————   ————                  ———————
  Normal  ScalingReplicaSet  90s    deployment-controller  Scaled up replica
```

We should have 1 Pod. If not, run the command again. This shows:

```
The DESIRED state is showing the configured number of replicas
The CURRENT state show how many replicas are running now
The UP-TO-DATE is the number of replicas that were updated to match the desi
The AVAILABLE state shows how many replicas are actually AVAILABLE to the us
```

```
[node1 Deployment101]$ kubectl get deployments
NAME              READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deployment  2/2     2            2           2m57s
```

```
[node1 Deployment101]$ kubectl get po
NAME                              READY   STATUS    RESTARTS   AGE
nginx-deployment-6dd86d77d-84fwp  1/1     Running   0          3m44s
nginx-deployment-6dd86d77d-xnrqp  1/1     Running   0          3m44s
```

Step #2. Scale up/down application deployment

Now let's scale the Deployment to 4 replicas. We are going to use the kubectl scale command, followed by the deployment type, name and desired number of instances:

```
Biradars-MacBook-Air-4:~ sangam$ kubectl scale deployments/nginx-deployment
deployment.extensions/nginx-deployment scaled
```

The change was applied, and we have 4 instances of the application available. Next, let's

check if the number of Pods changed:

Now There should be 4 pods running in the cluster

```
$ kubectl get deployments
NAME                READY    UP-TO-DATE    AVAILABLE    AGE
nginx-deployment    4/4      4             4            4m
```

There are 4 Pods now, with different IP addresses. The change was registered in the Deployment events log. To check that, use the describe command:

```
$ kubectl describe deployments/nginx-deployment
Name:                   nginx-deployment
Namespace:              default
CreationTimestamp:      Sat, 30 Nov 2019 20:04:34 +0530
Labels:                 <none>
Annotations:            deployment.kubernetes.io/revision: 1
Selector:               app=nginx
Replicas:               4 desired | 4 updated | 4 total | 4 available | 0 un
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=nginx
  Containers:
   nginx:
    Image:          nginx:1.7.9
    Port:           80/TCP
    Host Port:      0/TCP
    Environment:    <none>
    Mounts:         <none>
  Volumes:          <none>
Conditions:
  Type            Status   Reason
  ----            ------   ------
  Progressing     True     NewReplicaSetAvailable
  Available       True     MinimumReplicasAvailable
OldReplicaSets:   <none>
```

```
NewReplicaSet:   nginx-deployment-6dd86d77d (4/4 replicas created)
Events:
  Type     Reason            Age    From                   Message
  ----     ------            ----   ----                   -------
  Normal   ScalingReplicaSet  6m12s  deployment-controller  Scaled up replica
  Normal   ScalingReplicaSet  3m6s   deployment-controller  Scaled up replica
Biradars-MacBook-Air-4:~ sangam$
```

```
$ kubectl get pods -o wide
NAME                               READY    STATUS    RESTARTS    AGE      IP
nginx-deployment-6dd86d77d-b4v7k   1/1      Running   0           4m32s    10.1
nginx-deployment-6dd86d77d-bnc5m   1/1      Running   0           4m32s    10.1
nginx-deployment-6dd86d77d-bs6jr   1/1      Running   0           86s      10.1
nginx-deployment-6dd86d77d-wbdzv   1/1      Running   0           86s      10.1
Biradars-MacBook-Air-4:~ sangam$
```

You can also view in the output of this command that there are 4 replicas now.

Scaling the service to 2 Replicas

To scale down the Service to 2 replicas, run again the scale command:

```
$ kubectl scale deployments/nginx-deployment --replicas=2
deployment.extensions/nginx-deployment scaled
$ kubectl get deployments
NAME               READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deployment   2/2     2            2           7m23s
Biradars-MacBook-Air-4:~ sangam$
```

Step #3. Perform rolling updates to application deployment

So far, everything our Deployment did is no different than a typical ReplicaSet. The real power of a Deployment lies in its ability to update the pod templates without causing application outage.

Let's say that you have finished testing the nginx 1.7.9 , and you are ready to use it in

production. The current pods are using the older nginx version . The following command changes the deployment pod template to use the new image:

To update the image of the application to new version, use the set image command, followed by the deployment name and the new image version:

```
$ kubectl get deployments
NAME                  READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deployment      2/2     2            2           7m23s
$ kubectl describe pods
Name:              nginx-deployment-6dd86d77d-b4v7k
Namespace:         default
Priority:          0
PriorityClassName: <none>
Node:              docker-desktop/192.168.65.3
Start Time:        Sat, 30 Nov 2019 20:04:34 +0530
Labels:            app=nginx
                   pod-template-hash=6dd86d77d
Annotations:       <none>
Status:            Running
IP:                10.1.0.237
Controlled By:     ReplicaSet/nginx-deployment-6dd86d77d
Containers:
  nginx:
    Container ID:    docker://2c739cf9fe4dac53a4cc5c6097207da0c5edc2183f1f36f
    Image:           nginx:1.7.9
    Image ID:        docker-pullable://nginx@sha256:e3456c851a152494c3e4ff5fc
    Port:            80/TCP
    Host Port:       0/TCP
    State:           Running
      Started:       Sat, 30 Nov 2019 20:05:28 +0530
    Ready:           True
    Restart Count:   0
    Environment:     <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from default-token-ds5tg
Conditions:
  Type            Status
  Initialized     True
  Ready           True
```

```
    ContainersReady    True
    PodScheduled       True
Volumes:
  default-token-ds5tg:
    Type:        Secret (a volume populated by a Secret)
    SecretName:  default-token-ds5tg
    Optional:    false
QoS Class:        BestEffort
Node-Selectors:   <none>
Tolerations:      node.kubernetes.io/not-ready:NoExecute for 300s
                  node.kubernetes.io/unreachable:NoExecute for 300s
Events:
  Type    Reason     Age    From                     Message
  ----    ------     ----   ----                     -------
  Normal  Scheduled  10m    default-scheduler        Successfully assigned d
  Normal  Pulling    10m    kubelet, docker-desktop  Pulling image "nginx:1.
  Normal  Pulled     9m17s  kubelet, docker-desktop  Successfully pulled ima
  Normal  Created    9m17s  kubelet, docker-desktop  Created container nginx
  Normal  Started    9m17s  kubelet, docker-desktop  Started container nginx


Name:             nginx-deployment-6dd86d77d-bnc5m
Namespace:        default
Priority:         0
PriorityClassName:  <none>
Node:             docker-desktop/192.168.65.3
Start Time:       Sat, 30 Nov 2019 20:04:34 +0530
Labels:           app=nginx
                  pod-template-hash=6dd86d77d
Annotations:      <none>
Status:           Running
IP:               10.1.0.236
Controlled By:    ReplicaSet/nginx-deployment-6dd86d77d
Containers:
  nginx:
    Container ID:   docker://12ab35cbf4fdf78997b106b5eb27135f2fc37c890e723fe
    Image:          nginx:1.7.9
    Image ID:       docker-pullable://nginx@sha256:e3456c851a152494c3e4ff5fc
    Port:           80/TCP
    Host Port:      0/TCP
    State:          Running
```

```
        Started:        Sat, 30 Nov 2019 20:05:23 +0530
      Ready:            True
      Restart Count:    0
      Environment:      <none>
      Mounts:
        /var/run/secrets/kubernetes.io/serviceaccount from default-token-ds5tg
  Conditions:
    Type              Status
    Initialized       True
    Ready             True
    ContainersReady   True
    PodScheduled      True
  Volumes:
    default-token-ds5tg:
      Type:         Secret (a volume populated by a Secret)
      SecretName:   default-token-ds5tg
      Optional:     false
  QoS Class:        BestEffort
  Node-Selectors:   <none>
  Tolerations:      node.kubernetes.io/not-ready:NoExecute for 300s
                    node.kubernetes.io/unreachable:NoExecute for 300s
  Events:
    Type    Reason     Age     From                    Message
    ----    ------     ----    ----                    -------
    Normal  Scheduled  10m     default-scheduler       Successfully assigned d
    Normal  Pulling    10m     kubelet, docker-desktop Pulling image "nginx:1.
    Normal  Pulled     9m22s   kubelet, docker-desktop Successfully pulled ima
    Normal  Created    9m22s   kubelet, docker-desktop Created container nginx
    Normal  Started    9m22s   kubelet, docker-desktop Started container nginx
```

The command notified the Deployment to use a different image for your app and initiated a rolling update. Check the status of the new Pods, and view the old one terminating with the get pods command:

```
Biradars-MacBook-Air-4:~ sangam$ kubectl set image  deployments/nginx-deploy
deployment.extensions/nginx-deployment image updated
```

Checking description of pod again

```
$ kubectl describe pods
Name:              nginx-deployment-6dd86d77d-b4v7k
Namespace:         default
Priority:          0
PriorityClassName: <none>
Node:              docker-desktop/192.168.65.3
Start Time:        Sat, 30 Nov 2019 20:04:34 +0530
Labels:            app=nginx
                   pod-template-hash=6dd86d77d
Annotations:       <none>
Status:            Running
IP:                10.1.0.237
Controlled By:     ReplicaSet/nginx-deployment-6dd86d77d
Containers:
  nginx:
    Container ID:  docker://2c739cf9fe4dac53a4cc5c6097207da0c5edc2183f1f36f
    Image:         nginx:1.7.9
    Image ID:      docker-pullable://nginx@sha256:e3456c851a152494c3e4ff5fc
    Port:          80/TCP
    Host Port:     0/TCP
    State:         Running
      Started:     Sat, 30 Nov 2019 20:05:28 +0530
    Ready:         True
    Restart Count: 0
    Environment:   <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from default-token-ds5tg
Conditions:
  Type              Status
  Initialized       True
  Ready             True
  ContainersReady   True
  PodScheduled      True
Volumes:
  default-token-ds5tg:
    Type:       Secret (a volume populated by a Secret)
    SecretName: default-token-ds5tg
    Optional:   false
QoS Class:      BestEffort
Node-Selectors: <none>
```

```
Tolerations:        node.kubernetes.io/not-ready:NoExecute for 300s
                    node.kubernetes.io/unreachable:NoExecute for 300s
Events:
  Type    Reason     Age   From                    Message
  ----    ------     ----  ----                    -------
  Normal  Scheduled  16m   default-scheduler       Successfully assigned de
  Normal  Pulling    16m   kubelet, docker-desktop Pulling image "nginx:1.7
  Normal  Pulled     15m   kubelet, docker-desktop Successfully pulled imag
  Normal  Created    15m   kubelet, docker-desktop Created container nginx
  Normal  Started    15m   kubelet, docker-desktop Started container nginx


Name:              nginx-deployment-6dd86d77d-bnc5m
Namespace:         default
Priority:          0
PriorityClassName: <none>
Node:              docker-desktop/192.168.65.3
Start Time:        Sat, 30 Nov 2019 20:04:34 +0530
Labels:            app=nginx
                   pod-template-hash=6dd86d77d
Annotations:       <none>
Status:            Running
IP:                10.1.0.236
Controlled By:     ReplicaSet/nginx-deployment-6dd86d77d
Containers:
  nginx:
    Container ID:   docker://12ab35cbf4fdf78997b106b5eb27135f2fc37c890e723fe
    Image:          nginx:1.7.9
    Image ID:       docker-pullable://nginx@sha256:e3456c851a152494c3e4ff5fc
    Port:           80/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Sat, 30 Nov 2019 20:05:23 +0530
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from default-token-ds5tg
Conditions:
  Type            Status
  Initialized     True
```

```
  Ready             True
  ContainersReady   True
  PodScheduled      True
Volumes:
  default-token-ds5tg:
    Type:         Secret (a volume populated by a Secret)
    SecretName:   default-token-ds5tg
    Optional:     false
QoS Class:        BestEffort
Node-Selectors:   <none>
Tolerations:      node.kubernetes.io/not-ready:NoExecute for 300s
                  node.kubernetes.io/unreachable:NoExecute for 300s
Events:
  Type    Reason     Age   From                    Message
  ----    ------     ----  ----                    -------
  Normal  Scheduled  16m   default-scheduler       Successfully assigned de
  Normal  Pulling    16m   kubelet, docker-desktop Pulling image "nginx:1.7
  Normal  Pulled     15m   kubelet, docker-desktop Successfully pulled imag
  Normal  Created    15m   kubelet, docker-desktop Created container nginx
  Normal  Started    15m   kubelet, docker-desktop Started container nginx


Name:             nginx-deployment-784b7cc96d-kxc68
Namespace:        default
Priority:         0
PriorityClassName: <none>
Node:             docker-desktop/192.168.65.3
Start Time:       Sat, 30 Nov 2019 20:20:04 +0530
Labels:           app=nginx
                  pod-template-hash=784b7cc96d
Annotations:      <none>
Status:           Pending
IP:
Controlled By:    ReplicaSet/nginx-deployment-784b7cc96d
Containers:
  nginx:
    Container ID:
    Image:        nginx:1.9.1
    Image ID:
    Port:         80/TCP
    Host Port:    0/TCP
```

```
     State:          Waiting
       Reason:       ContainerCreating
     Ready:          False
     Restart Count:  0
     Environment:    <none>
     Mounts:
       /var/run/secrets/kubernetes.io/serviceaccount from default-token-ds5tg
Conditions:
  Type              Status
  Initialized       True
  Ready             False
  ContainersReady   False
  PodScheduled      True
Volumes:
  default-token-ds5tg:
    Type:          Secret (a volume populated by a Secret)
    SecretName:    default-token-ds5tg
    Optional:      false
QoS Class:         BestEffort
Node-Selectors:    <none>
Tolerations:       node.kubernetes.io/not-ready:NoExecute for 300s
                   node.kubernetes.io/unreachable:NoExecute for 300s
Events:
  Type    Reason     Age   From                   Message
  ----    ------     ----  ----                   -------
  Normal  Scheduled  36s   default-scheduler      Successfully assigned de
  Normal  Pulling    35s   kubelet, docker-desktop Pulling image "nginx:1.9
Biradars-MacBook-Air-4:~ sangam$
```

Step #4. Rollback updates to application deployment

The rollout command reverted the deployment to the previous known state. Updates are versioned and you can revert to any previously know state of a Deployment. List again the Pods:

```
$ kubectl rollout undo deployments/nginx-deployment
deployment.extensions/nginx-deployment rolled back

$ kubectl rollout status deployments/nginx-deployment
```

```
deployment "nginx-deployment" successfully rolled out
```

After the rollout succeeds, you may want to get the Deployment.

The output shows the update progress until all the pods use the new container image.

The algorithm that Kubernetes Deployments use when deciding how to roll updates is to keep at least 25% of the pods running. Accordingly, it doesn't kill old pods unless a sufficient number of new ones are up. In the same sense, it does not create new pods until enough pods are no longer running. Through this algorithm, the application is always available during updates.

You can use the following command to determine the update strategy that the Deployment is using:

```
Biradars-MacBook-Air-4:~ sangam$ kubectl describe deployments | grep Strateg
StrategyType:            RollingUpdate
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Biradars-MacBook-Air-4:~ sangam$
```

Step #5. Cleanup

Finally you can clean up the resources you created in your cluster:

```
kubectl delete service nginx-deployment
kubectl delete deployment nginx-deployment
```

Contributors

Sangam Biradar

Reviewers

Ajeet Singh Raina

Next »

Join KubeDaily

9 Members Online

Support

MEMBERS ONLINE

Aman Manapure
h3ll_boy
MEE6
Nitinkashyap   Apex Legends
ojaswa
Parmeshwar
prasad
trimankaur
vikas027

Free voice chat from Discord   Connect   Tweets by collabnix

kubelabs is maintained by **collabnix.**

This page was generated by GitHub Pages.