KUBERNETES DEPLOYMENT

I. Kubernetes rolling update

1. Update a Deployment

a. Updating a Deployment and checking our rollout status

https://github.com/hungtran84/k8s-cka/blob/master/d2_workloads/03_deploy_app/01-rollout-basic/deployment.yaml
kubectl apply -f deployment.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: hello-world
spec:
  replicas: 10
  selector:
    matchLabels:
      app: hello-world
  template:
    metadata:
      labels:
        app: hello-world
    spec:
      containers:
      - name: hello-world
        image: gcr.io/google-samples/hello-app:1.0
        ports:
        - containerPort: 8080
        resources:
          requests:
            memory: 128M
            cpu: 100m
          limits:
            memory: 128M
            cpu: 100m
---
apiVersion: v1
kind: Service
metadata:
  name: hello-world
```

```
spec:
  selector:
    app: hello-world
  ports:
  - port: 80
    protocol: TCP
    targetPort: 8080

deployment.apps/hello-world created
service/hello-world created
```

b. Check the status of the deployment

```
kubectl get deployment hello-world
NAME         READY   UP-TO-DATE   AVAILABLE   AGE
hello-world  10/10   10           10          99s
```

c. Update that deployment

```
https://github.com/hungtran84/k8s-cka/blob/master/d2_workloads/03_deploy_app/01-rollout-basic/deployment.v2.yaml
kubectl apply -f deployment.v2.yaml

apiVersion: apps/v1
kind: Deployment
metadata:
  name: hello-world
spec:
  replicas: 10
  selector:
    matchLabels:
      app: hello-world
  template:
    metadata:
      labels:
        app: hello-world
    spec:
      containers:
      - name: hello-world
        image: gcr.io/google-samples/hello-app:2.0
```

```
        ports:
        - containerPort: 8080
        resources:
          requests:
            memory: 64M
            cpu: 10m
          limits:
            memory: 64M
            cpu: 10m
---
apiVersion: v1
kind: Service
metadata:
  name: hello-world
spec:
  selector:
    app: hello-world
  ports:
  - port: 80
    protocol: TCP
    targetPort: 8080

deployment.apps/hello-world configured
service/hello-world unchanged
```

d. Check the status of that rollout

```
kubectl rollout status deployment hello-world
Waiting for deployment "hello-world" rollout to finish: 2 old replicas are pending termination...
Waiting for deployment "hello-world" rollout to finish: 2 old replicas are pending termination...
Waiting for deployment "hello-world" rollout to finish: 2 old replicas are pending termination...
Waiting for deployment "hello-world" rollout to finish: 1 old replicas are pending termination...
```

e. Check out Replicas, Conditions and Events

```
kubectl describe deployments hello-world

Name:                   hello-world
Namespace:              default
CreationTimestamp:      Wed, 22 Jul 2020 13:24:48 +0700
Labels:                 <none>
Annotations:            deployment.kubernetes.io/revision: 2
Selector:               app=hello-world
Replicas:               10 desired | 10 updated | 10 total | 10 available | 0 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=hello-world
  Containers:
   hello-world:
    Image:      gcr.io/google-samples/hello-app:2.0
    Port:       8080/TCP
    Host Port:  0/TCP
    Limits:
      cpu:     100m
      memory:  128M
    Requests:
      cpu:        100m
      memory:     128M
    Environment:  <none>
    Mounts:       <none>
  Volumes:        <none>
Conditions:
  Type          Status  Reason
  ----          ------  ------
  Available     True    MinimumReplicasAvailable
  Progressing   True    NewReplicaSetAvailable
OldReplicaSets:  <none>
NewReplicaSet:   hello-world-76d9cfd9c9 (10/10 replicas created)
Events:
  Type    Reason             Age              From                Message
```

```
  ----    ------            ----             ----                  -------
  Normal  ScalingReplicaSet  2m40s            deployment-controller  Scaled up
replica set hello-world-7bd56686bb to 10
  Normal  ScalingReplicaSet  35s              deployment-controller  Scaled up replica
set hello-world-76d9cfd9c9 to 3
  Normal  ScalingReplicaSet  35s              deployment-controller  Scaled down
replica set hello-world-7bd56686bb to 8
  Normal  ScalingReplicaSet  35s              deployment-controller  Scaled up replica
set hello-world-76d9cfd9c9 to 5
  Normal  ScalingReplicaSet  32s              deployment-controller  Scaled down
replica set hello-world-7bd56686bb to 7
  Normal  ScalingReplicaSet  32s              deployment-controller  Scaled up replica
set hello-world-76d9cfd9c9 to 6
  Normal  ScalingReplicaSet  32s              deployment-controller  Scaled down
replica set hello-world-7bd56686bb to 6
  Normal  ScalingReplicaSet  32s              deployment-controller  Scaled up replica
set hello-world-76d9cfd9c9 to 7
  Normal  ScalingReplicaSet  31s              deployment-controller  Scaled down
replica set hello-world-7bd56686bb to 5
  Normal  ScalingReplicaSet  16s (x8 over 31s)  deployment-controller  (combined
from similar events): Scaled down replica set hello-world-7bd56686bb to 0
```

Both replicasets remain, and that will become very useful shortly when we use a rollback

```
kubectl get replicaset
NAME                 DESIRED   CURRENT   READY   AGE
hello-world-76d9cfd9c9   10        10        10      82s
hello-world-7bd56686bb   0         0         0       3m27s

kubectl describe replicaset hello-world-76d9cfd9c9

Name:           hello-world-76d9cfd9c9
Namespace:      default
Selector:       app=hello-world,pod-template-hash=76d9cfd9c9
Labels:         app=hello-world
                pod-template-hash=76d9cfd9c9
Annotations:    deployment.kubernetes.io/desired-replicas: 10
                deployment.kubernetes.io/max-replicas: 13
```

```
                deployment.kubernetes.io/revision: 2
Controlled By:  Deployment/hello-world
Replicas:       10 current / 10 desired
Pods Status:    10 Running / 0 Waiting / 0 Succeeded / 0 Failed
Pod Template:
  Labels:  app=hello-world
           pod-template-hash=76d9cfd9c9
  Containers:
   hello-world:
    Image:      gcr.io/google-samples/hello-app:2.0
    Port:       8080/TCP
    Host Port:  0/TCP
    Limits:
      cpu:     100m
      memory:  128M
    Requests:
      cpu:        100m
      memory:     128M
    Environment:  <none>
    Mounts:       <none>
  Volumes:        <none>
Events:
  Type    Reason            Age   From                   Message
  ----    ------            ----  ----                   -------
  Normal  SuccessfulCreate  119s  replicaset-controller  Created pod: hello-world-
76d9cfd9c9-588qf
  Normal  SuccessfulCreate  119s  replicaset-controller  Created pod: hello-world-
76d9cfd9c9-w95rm
  Normal  SuccessfulCreate  119s  replicaset-controller  Created pod: hello-world-
76d9cfd9c9-f2l7v
  Normal  SuccessfulCreate  119s  replicaset-controller  Created pod: hello-world-
76d9cfd9c9-cczttt
  Normal  SuccessfulCreate  119s  replicaset-controller  Created pod: hello-world-
76d9cfd9c9-jhh67
  Normal  SuccessfulCreate  116s  replicaset-controller  Created pod: hello-world-
76d9cfd9c9-p959k
  Normal  SuccessfulCreate  116s  replicaset-controller  Created pod: hello-world-
76d9cfd9c9-895tv
```

```
  Normal  SuccessfulCreate  115s  replicaset-controller  Created pod: hello-world-76d9cfd9c9-q5vw8
  Normal  SuccessfulCreate  115s  replicaset-controller  Created pod: hello-world-76d9cfd9c9-9x9v4
  Normal  SuccessfulCreate  114s  replicaset-controller  (combined from similar events): Created pod: hello-world-76d9cfd9c9-hdmk2

kubectl describe replicaset hello-world-5646fcc96b
It may not found
Error from server (NotFound): replicasets.apps "hello-world-5646fcc96b" not found
```

2. Rollback

a. Observe behavior when rolling update a bad deployment

```
https://github.com/hungtran84/k8s-cka/blob/master/d2_workloads/03_deploy_app/01-rollout-basic/deployment.broken.yaml
kubectl apply -f deployment.broken.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: hello-world
spec:
  progressDeadlineSeconds: 10
  replicas: 10
  selector:
    matchLabels:
      app: hello-world
  template:
    metadata:
      labels:
        app: hello-world
    spec:
      containers:
      - name: hello-world
        image: gcr.io/google-samples/hello-ap:2.0
        ports:
        - containerPort: 8080
        resources:
```

```
        requests:
          memory: 64M
          cpu: 10m
        limits:
          memory: 64M
          cpu: 10m
---
apiVersion: v1
kind: Service
metadata:
  name: hello-world
spec:
  selector:
    app: hello-world
  ports:
  - port: 80
    protocol: TCP
    targetPort: 8080

deployment.apps/hello-world configured
service/hello-world unchanged
```

b. Why isn't this finishing...? after progressDeadlineSeconds which we set to 10 seconds

```
kubectl rollout status deployment hello-world
error: deployment "hello-world" exceeded its progress deadline
```

c. Let's check out Pods, ImagePullBackoff/ErrImagePull...ah an error in our image definition

```
kubectl get pods

NAME                     READY   STATUS       RESTARTS   AGE
hello-world-76d9cfd9c9-588qf   1/1     Running      0          4m30s
hello-world-76d9cfd9c9-895tv   1/1     Running      0          4m27s
hello-world-76d9cfd9c9-ccztt   1/1     Running      0          4m30s
hello-world-76d9cfd9c9-f2l7v   1/1     Running      0          4m30s
hello-world-76d9cfd9c9-hdmk2   1/1     Running      0          4m25s
hello-world-76d9cfd9c9-jhh67   1/1     Running      0          4m30s
```

```
hello-world-76d9cfd9c9-p959k   1/1     Running       0          4m27s
hello-world-76d9cfd9c9-w95rm   1/1     Running       0          4m30s
hello-world-8496cf4fcf-hsptk   0/1     ErrImagePull  0          66s
hello-world-8496cf4fcf-hsxfr   0/1     ErrImagePull  0          66s
hello-world-8496cf4fcf-lfdf6   0/1     ErrImagePull  0          66s
hello-world-8496cf4fcf-t6q6n   0/1     ErrImagePull  0          66s
hello-world-8496cf4fcf-tntjn   0/1     ErrImagePull  0          66s
```

d. 8 are online, let's look at why

```
kubectl describe deployments hello-world
Name:                   hello-world
Namespace:              default
CreationTimestamp:      Wed, 22 Jul 2020 13:24:48 +0700
Labels:                 <none>
Annotations:            deployment.kubernetes.io/revision: 3
Selector:               app=hello-world
Replicas:               10 desired | 5 updated | 13 total | 8 available | 5 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=hello-world
  Containers:
   hello-world:
    Image:      gcr.io/google-samples/hello-ap:2.0
    Port:       8080/TCP
    Host Port:  0/TCP
    Limits:
      cpu:     100m
      memory:  128M
    Requests:
      cpu:        100m
      memory:     128M
    Environment:  <none>
    Mounts:       <none>
  Volumes:        <none>
Conditions:
  Type          Status  Reason
```

```
  ----           ------  ------
  Available      True    MinimumReplicasAvailable
  Progressing    False   ProgressDeadlineExceeded
OldReplicaSets:  hello-world-76d9cfd9c9 (8/8 replicas created)
NewReplicaSet:   hello-world-8496cf4fcf (5/5 replicas created)
Events:
  Type    Reason             Age                From                   Message
  ----    ------             ----               ----                   -------
  Normal  ScalingReplicaSet  7m30s              deployment-controller  Scaled up
replica set hello-world-7bd56686bb to 10
  Normal  ScalingReplicaSet  5m25s              deployment-controller  Scaled up
replica set hello-world-76d9cfd9c9 to 3
  Normal  ScalingReplicaSet  5m25s              deployment-controller  Scaled down
replica set hello-world-7bd56686bb to 8
  Normal  ScalingReplicaSet  5m25s              deployment-controller  Scaled up
replica set hello-world-76d9cfd9c9 to 5
  Normal  ScalingReplicaSet  5m22s              deployment-controller  Scaled down
replica set hello-world-7bd56686bb to 7
  Normal  ScalingReplicaSet  5m22s              deployment-controller  Scaled up
replica set hello-world-76d9cfd9c9 to 6
  Normal  ScalingReplicaSet  5m22s              deployment-controller  Scaled down
replica set hello-world-7bd56686bb to 6
  Normal  ScalingReplicaSet  5m22s              deployment-controller  Scaled up
replica set hello-world-76d9cfd9c9 to 7
  Normal  ScalingReplicaSet  5m21s              deployment-controller  Scaled down
replica set hello-world-7bd56686bb to 5
  Normal  ScalingReplicaSet  2m1s (x11 over 5m21s)  deployment-controller
(combined from similar events): Scaled up replica set hello-world-8496cf4fcf to 5
```

e. check the rollout history, but which revision should we rollback to?

```
kubectl rollout history deployment hello-world
deployment.apps/hello-world
REVISION  CHANGE-CAUSE
1         <none>
2         <none>
3         <none>
kubectl describe deployments hello-world | head
Name:                   hello-world
```

```
Namespace:              default
CreationTimestamp:      Wed, 22 Jul 2020 13:24:48 +0700
Labels:                 <none>
Annotations:            deployment.kubernetes.io/revision: 3
Selector:               app=hello-world
Replicas:               10 desired | 5 updated | 13 total | 8 available | 5 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge

kubectl rollout history deployment hello-world --revision=2
deployment.apps/hello-world with revision #2
Pod Template:
  Labels:       app=hello-world
        pod-template-hash=76d9cfd9c9
  Containers:
   hello-world:
    Image:      gcr.io/google-samples/hello-app:2.0
    Port:       8080/TCP
    Host Port:  0/TCP
    Limits:
      cpu:      100m
      memory:   128M
    Requests:
      cpu:      100m
      memory:   128M
    Environment:        <none>
    Mounts:     <none>
  Volumes:      <none>
kubectl rollout history deployment hello-world --revision=2
deployment.apps/hello-world with revision #2
Pod Template:
  Labels:       app=hello-world
        pod-template-hash=76d9cfd9c9
  Containers:
   hello-world:
    Image:      gcr.io/google-samples/hello-app:2.0
    Port:       8080/TCP
```

```
    Host Port:  0/TCP
    Limits:
      cpu:      100m
      memory:   128M
    Requests:
      cpu:      100m
      memory:   128M
    Environment:        <none>
    Mounts:     <none>
  Volumes:      <none>
```

f. Undo our rollout to revision 2, which is our v2 container.

```
kubectl rollout undo deployment hello-world --to-revision=2
deployment.apps/hello-world rolled back

kubectl rollout status deployment hello-world
deployment "hello-world" successfully rolled out
```

II. Scheduler

1. Finding scheduling information

```
#Create a deployment with 3 replica
https://github.com/hungtran84/k8s-cka/blob/master/d2_workloads/03_deploy_app/02-scheduler/deployment.yaml
kubectl apply -f deployment.yaml
deployment.apps/hello-world created

#Pods spread out evenly across the Nodes due to our scoring functions
kubectl get pods -o wide
NAME                     READY   STATUS    RESTARTS   AGE   IP          NODE
NOMINATED NODE   READINESS GATES
hello-world-7bd56686bb-fdbvn   1/1     Running   0          48s   10.48.1.18   gke-
cluster-1-default-pool-990b49f7-dm0l   <none>         <none>
hello-world-7bd56686bb-hprpf   1/1     Running   0          48s   10.48.0.13   gke-
cluster-1-default-pool-990b49f7-scjx   <none>         <none>
hello-world-7bd56686bb-xkjxr   1/1     Running   0          48s   10.48.2.12   gke-
cluster-1-default-pool-990b49f7-bzft   <none>         <none>

#If we scale our deployment to 6...
kubectl scale deployment hello-world --replicas=6
deployment.apps/hello-world scaled


#We can see that the scheduler works to keep load even across the nodes.
kubectl get pods -o wide
NAME                     READY   STATUS    RESTARTS   AGE    IP
NODE                          NOMINATED NODE   READINESS GATES
hello-world-7bd56686bb-fdbvn   1/1     Running   0          2m14s   10.48.1.18   gke-
cluster-1-default-pool-990b49f7-dm0l   <none>         <none>
hello-world-7bd56686bb-gdqsz   1/1     Running   0          45s     10.48.0.14   gke-
cluster-1-default-pool-990b49f7-scjx   <none>         <none>
hello-world-7bd56686bb-hprpf   1/1     Running   0          2m14s   10.48.0.13   gke-
cluster-1-default-pool-990b49f7-scjx   <none>         <none>
hello-world-7bd56686bb-l5rbn   1/1     Running   0          45s     10.48.2.13   gke-
cluster-1-default-pool-990b49f7-bzft   <none>         <none>
hello-world-7bd56686bb-q52hl   1/1     Running   0          45s     10.48.1.19   gke-
```

```
cluster-1-default-pool-990b49f7-dm0l   <none>          <none>
hello-world-7bd56686bb-xkjxr   1/1     Running   0          2m14s   10.48.2.12   gke-
cluster-1-default-pool-990b49f7-bzft   <none>          <none>
#We can see the nodeName populated for this node
kubectl get pods hello-world-[tab][tab] -o yaml
....
  enableServiceLinks: true
  nodeName: gke-cluster-1-default-pool-990b49f7-dm0l
  priority: 0
  restartPolicy: Always
  schedulerName: default-scheduler
```

2. Scheduling Pods with resource requests

```
#Get resources allocable
kubectl top nodes
NAME                                        CPU(cores)   CPU%   MEMORY(bytes)
MEMORY%
gke-cluster-1-default-pool-990b49f7-bzft   75m          7%     730Mi          25%
gke-cluster-1-default-pool-990b49f7-dm0l   80m          8%     726Mi          25%
gke-cluster-1-default-pool-990b49f7-scjx   71m          7%     681Mi          24%


#Scheduling Pods with resource requests
https://github.com/hungtran84/k8s-
cka/blob/master/d2_workloads/03_deploy_app/02-scheduler/requests.yaml
kubectl apply -f requests.yaml
deployment.apps/hello-world-requests created

#We created three pods, one on each node
kubectl get pods -o wide
➜  scheduler git:(s07) ✗ kubectl get pods -o wide
NAME                                READY   STATUS    RESTARTS   AGE   IP
NODE                                NOMINATED NODE   READINESS GATES
hello-world-requests-77c54c869f-f27hh   1/1     Running   0          12s   10.48.0.15
gke-cluster-1-default-pool-990b49f7-scjx   <none>          <none>
hello-world-requests-77c54c869f-k2hwh   1/1     Running   0          12s   10.48.2.14
gke-cluster-1-default-pool-990b49f7-bzft   <none>          <none>
hello-world-requests-77c54c869f-mrcsp   1/1     Running   0          12s   10.48.1.20
```

```
gke-cluster-1-default-pool-990b49f7-dm0l   <none>          <none>


#Let's scale our deployment to 6 replica
kubectl scale deployment hello-world-requests --replicas=6
deployment.apps/hello-world-requests scaled


#We see that three Pods are pending...why?
kubectl get pods -o wide
NAME                                  READY   STATUS    RESTARTS   AGE    IP
NODE                                  NOMINATED NODE   READINESS GATES
hello-world-requests-77c54c869f-f27hh   1/1     Running   0          101s   10.48.0.15
gke-cluster-1-default-pool-990b49f7-scjx   <none>          <none>
hello-world-requests-77c54c869f-f974n   0/1     Pending   0          46s    <none>
<none>                                <none>          <none>
hello-world-requests-77c54c869f-jrq9z   0/1     Pending   0          46s    <none>
<none>                                <none>          <none>
hello-world-requests-77c54c869f-k2hwh   1/1     Running   0          101s   10.48.2.14
gke-cluster-1-default-pool-990b49f7-bzft   <none>          <none>
hello-world-requests-77c54c869f-mrcsp   1/1     Running   0          101s   10.48.1.20
gke-cluster-1-default-pool-990b49f7-dm0l   <none>          <none>
hello-world-requests-77c54c869f-t8q7q   0/1     Pending   0          46s    <none>
<none>                                <none>          <none>

kubectl get pods -o wide | grep Pending

hello-world-requests-77c54c869f-f974n   0/1     Pending   0          81s    <none>
<none>                                <none>          <none>
hello-world-requests-77c54c869f-jrq9z   0/1     Pending   0          81s    <none>
<none>                                <none>          <none>
hello-world-requests-77c54c869f-t8q7q   0/1     Pending   0          81s    <none>
<none>                                <none>          <none>




#Let's look at why the Pod is Pending...check out the Pod's events...
```

```
kubectl describe pods
Events:
  Type     Reason            Age                From               Message
  ----     ------            ----               ----               -------
  Warning  FailedScheduling  59s (x3 over 2m3s)  default-scheduler  0/3 nodes are
available: 3 Insufficient cpu.

#Let get list nodes
kubectl get nodes
NAME                                     STATUS   ROLES    AGE    VERSION
gke-cluster-1-default-pool-990b49f7-bzft   Ready    <none>   137m   v1.16.11-gke.5
gke-cluster-1-default-pool-990b49f7-dm0l   Ready    <none>   137m   v1.16.11-
gke.5
gke-cluster-1-default-pool-990b49f7-scjx   Ready    <none>   137m   v1.16.11-gke.5
kubectl describe node gke-cluster-1-default-pool-990b49f7-bzft
  Namespace                  Name                                         CPU Requests  CPU
Limits  Memory Requests  Memory Limits  AGE
  ---------                  ----                                         ------------  ----------
-----  -------------  ---
  default                    hello-world-requests-77c54c869f-k2hwh        300m
(31%)   300m (31%)  128M (4%)       128M (4%)      4m36s
  kube-system                fluentd-gke-c6gw9                            100m (10%)    1
(106%)   200Mi (7%)      500Mi (17%)    138m
  kube-system                fluentd-gke-scaler-cd4d654d7-68t7s           0 (0%)
0 (0%)      0 (0%)          0 (0%)         138m
  kube-system                gke-metrics-agent-lxknc                      3m (0%)       0
(0%)      50Mi (1%)       50Mi (1%)      138m
  kube-system                kube-dns-56d8cd994f-f89fb                    260m (27%)
0 (0%)      110Mi (3%)      170Mi (6%)     138m
  kube-system                kube-proxy-gke-cluster-1-default-pool-990b49f7-bzft
100m (10%)   0 (0%)      0 (0%)          0 (0%)         138m
  kube-system                prometheus-to-sd-hj22k                       0 (0%)        0
(0%)      0 (0%)          0 (0%)         138m
Allocated resources:
  (Total limits may be over 100 percent, i.e., overcommitted.)
  Resource                   Requests        Limits
  --------                   --------        ------
  cpu                        763m (81%)      1300m (138%)
```

```
  memory                      505487360 (17%)  882974720 (29%)
  ephemeral-storage           0 (0%)           0 (0%)
  hugepages-2Mi               0 (0%)           0 (0%)
  attachable-volumes-gce-pd  0                0
Events:                       <none>
```

3. Using Affinity and Anti-Affinity to schedule Pods to Nodes

```
https://github.com/hungtran84/k8s-cka/blob/master/d2_workloads/03_deploy_app/02-scheduler/deployment-affinity.yaml
kubectl apply -f deployment-affinity.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: hello-world-web
spec:
  replicas: 1
  selector:
    matchLabels:
      app: hello-world-web
  template:
    metadata:
      labels:
        app: hello-world-web
    spec:
      containers:
      - name: hello-world-web
        image: gcr.io/google-samples/hello-app:1.0
        ports:
        - containerPort: 8080
        resources:
          requests:
            memory: 128M
            cpu: 100m
          limits:
            memory: 128M
            cpu: 100m
```

```
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: hello-world-cache
spec:
  replicas: 1
  selector:
    matchLabels:
      app: hello-world-cache
  template:
    metadata:
      labels:
        app: hello-world-cache
    spec:
      containers:
      - name: hello-world-cache
        image: gcr.io/google-samples/hello-app:1.0
        ports:
        - containerPort: 8080
      affinity:
        podAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
          - labelSelector:
              matchExpressions:
              - key: app
                operator: In
                values:
                - hello-world-web
            topologyKey: "kubernetes.io/hostname"


deployment.apps/hello-world-web created
deployment.apps/hello-world-cache created

#Let's check out the labels on the nodes, look for kubernetes.io/hostname which
#we're using for our topologykey
kubectl describe nodes gke-cluster-1-default-pool-990b49f7-bzft | head
```

```
Name:               gke-cluster-1-default-pool-990b49f7-bzft
Roles:              <none>
Labels:             beta.kubernetes.io/arch=amd64
                    beta.kubernetes.io/instance-type=e2-medium
                    beta.kubernetes.io/os=linux
                    cloud.google.com/gke-nodepool=default-pool
                    cloud.google.com/gke-os-distribution=cos
                    failure-domain.beta.kubernetes.io/region=asia-southeast1
                    failure-domain.beta.kubernetes.io/zone=asia-southeast1-a
                    kubernetes.io/arch=amd64
kubectl get nodes --show-labels
NAME                                STATUS   ROLES    AGE    VERSION
LABELS
gke-cluster-1-default-pool-990b49f7-bzft   Ready    <none>   150m   v1.16.11-gke.5
beta.kubernetes.io/arch=amd64,beta.kubernetes.io/instance-type=e2-
medium,beta.kubernetes.io/os=linux,cloud.google.com/gke-nodepool=default-
pool,cloud.google.com/gke-os-distribution=cos,failure-
domain.beta.kubernetes.io/region=asia-southeast1,failure-
domain.beta.kubernetes.io/zone=asia-southeast1-
a,kubernetes.io/arch=amd64,kubernetes.io/hostname=gke-cluster-1-default-pool-
990b49f7-bzft,kubernetes.io/os=linux
gke-cluster-1-default-pool-990b49f7-dm0l   Ready    <none>   150m   v1.16.11-
gke.5   beta.kubernetes.io/arch=amd64,beta.kubernetes.io/instance-type=e2-
medium,beta.kubernetes.io/os=linux,cloud.google.com/gke-nodepool=default-
pool,cloud.google.com/gke-os-distribution=cos,failure-
domain.beta.kubernetes.io/region=asia-southeast1,failure-
domain.beta.kubernetes.io/zone=asia-southeast1-
a,kubernetes.io/arch=amd64,kubernetes.io/hostname=gke-cluster-1-default-pool-
990b49f7-dm0l,kubernetes.io/os=linux
gke-cluster-1-default-pool-990b49f7-scjx   Ready    <none>   150m   v1.16.11-gke.5
beta.kubernetes.io/arch=amd64,beta.kubernetes.io/instance-type=e2-
medium,beta.kubernetes.io/os=linux,cloud.google.com/gke-nodepool=default-
pool,cloud.google.com/gke-os-distribution=cos,failure-
domain.beta.kubernetes.io/region=asia-southeast1,failure-
domain.beta.kubernetes.io/zone=asia-southeast1-
a,kubernetes.io/arch=amd64,kubernetes.io/hostname=gke-cluster-1-default-pool-
990b49f7-scjx,kubernetes.io/os=linux
```

```
#We can see that web and cache are both on the name node
kubectl get pods -o wide
NAME                          READY   STATUS    RESTARTS   AGE    IP
NODE                                NOMINATED NODE   READINESS GATES
hello-world-cache-55d8d577db-zscz6   1/1     Running   0          3m14s   10.48.0.16
gke-cluster-1-default-pool-990b49f7-scjx   <none>          <none>
hello-world-web-8668c4858d-nnxqt     1/1     Running   0          3m14s   10.48.0.17
gke-cluster-1-default-pool-990b49f7-scjx   <none>          <none>


#If we scale the web deployment, it's still same node
kubectl scale deployment hello-world-web --replicas=2
deployment.apps/hello-world-web scaled

kubectl get pods -o wide
NAME                          READY   STATUS    RESTARTS   AGE    IP
NODE                                NOMINATED NODE   READINESS GATES
hello-world-cache-55d8d577db-zscz6   1/1     Running   0          4m26s   10.48.0.16
gke-cluster-1-default-pool-990b49f7-scjx   <none>          <none>
hello-world-web-8668c4858d-m9xqt     1/1     Running   0          33s     10.48.0.18
gke-cluster-1-default-pool-990b49f7-scjx   <none>          <none>
hello-world-web-8668c4858d-nnxqt     1/1     Running   0          4m26s   10.48.0.17
gke-cluster-1-default-pool-990b49f7-scjx   <none>          <none>



kubectl scale deployment hello-world-cache --replicas=2
kubectl get pods -o wide
NAME                          READY   STATUS    RESTARTS   AGE    IP
NODE                                NOMINATED NODE   READINESS GATES
hello-world-cache-55d8d577db-fzjv2   1/1     Running   0          53s     10.48.0.19
gke-cluster-1-default-pool-990b49f7-scjx   <none>          <none>
hello-world-cache-55d8d577db-zscz6   1/1     Running   0          6m5s    10.48.0.16
gke-cluster-1-default-pool-990b49f7-scjx   <none>          <none>
hello-world-web-8668c4858d-m9xqt     1/1     Running   0          2m12s   10.48.0.18
gke-cluster-1-default-pool-990b49f7-scjx   <none>          <none>
```

```
hello-world-web-8668c4858d-nnxqt   1/1   Running   0          6m5s   10.48.0.17
gke-cluster-1-default-pool-990b49f7-scjx   <none>           <none>
```

4. Node Cordoning

```
#Let's create a deployment with three replicas
https://github.com/hungtran84/k8s-cka/blob/master/d2_workloads/03_deploy_app/02-scheduler/deployment.yaml
kubectl apply -f deployment.yaml

apiVersion: apps/v1
kind: Deployment
metadata:
  name: hello-world
spec:
  replicas: 3
  selector:
    matchLabels:
      app: hello-world
  template:
    metadata:
      labels:
        app: hello-world
    spec:
      containers:
      - name: hello-world
        image: gcr.io/google-samples/hello-app:1.0
        ports:
        - containerPort: 8080
        resources:
          requests:
            memory: 128M
            cpu: 100m
          limits:
            memory: 128M

deployment.apps/hello-world created

kubectl get pods -o wide
```

```
NAME                     READY   STATUS    RESTARTS   AGE   IP          NODE
NOMINATED NODE   READINESS GATES
hello-world-7bd56686bb-4xq5b   1/1     Running   0          10s   10.48.0.20   gke-
cluster-1-default-pool-990b49f7-scjx   <none>           <none>
hello-world-7bd56686bb-mrwsw   1/1     Running   0          10s   10.48.1.21   gke-
cluster-1-default-pool-990b49f7-dm0l   <none>           <none>
hello-world-7bd56686bb-n4x6p   1/1     Running   0          10s   10.48.2.15   gke-
cluster-1-default-pool-990b49f7-bzft   <none>           <none>


#Let's drain (remove) the Pods from node3..
kubectl drain gke-cluster-1-default-pool-990b49f7-bzft
node/gke-cluster-1-default-pool-990b49f7-bzft cordoned
error: unable to drain node "gke-cluster-1-default-pool-990b49f7-bzft", aborting
command...

There are pending nodes to be drained:
 gke-cluster-1-default-pool-990b49f7-bzft
error: cannot delete DaemonSet-managed Pods (use --ignore-daemonsets to ignore):
kube-system/fluentd-gke-c6gw9, kube-system/gke-metrics-agent-lxknc, kube-
system/prometheus-to-sd-hj22k

#Let's try that again since daemonsets aren't scheduled we need to work around
them.
kubectl drain gke-cluster-1-default-pool-990b49f7-bzft --ignore-daemonsets
node/gke-cluster-1-default-pool-990b49f7-bzft already cordoned
WARNING: ignoring DaemonSet-managed Pods: kube-system/fluentd-gke-c6gw9,
kube-system/gke-metrics-agent-lxknc, kube-system/prometheus-to-sd-hj22k
evicting pod default/hello-world-7bd56686bb-n4x6p
evicting pod kube-system/fluentd-gke-scaler-cd4d654d7-68t7s
evicting pod kube-system/kube-dns-56d8cd994f-f89fb
pod/hello-world-7bd56686bb-n4x6p evicted


#Now all the workload is on node 1 and 2
kubectl get pods -o wide
NAME                     READY   STATUS    RESTARTS   AGE    IP
NODE                          NOMINATED NODE   READINESS GATES
```

```
hello-world-7bd56686bb-4xq5b  1/1   Running  0       5m20s  10.48.0.20  gke-
cluster-1-default-pool-990b49f7-scjx  <none>        <none>
hello-world-7bd56686bb-jvt2p  1/1   Running  0       69s    10.48.0.22  gke-
cluster-1-default-pool-990b49f7-scjx  <none>        <none>
hello-world-7bd56686bb-mrwsw  1/1   Running  0       5m20s  10.48.1.21  gke-
cluster-1-default-pool-990b49f7-dm0l  <none>        <none>

#We can uncordon node 3
kubectl uncordon gke-cluster-1-default-pool-990b49f7-bzft
kubectl get pods -o wide

#So let's scale that Deployment and see where they get scheduled...
kubectl scale deployment hello-world --replicas=4
kubectl get pods -o wide
NAME                  READY  STATUS   RESTARTS  AGE    IP
NODE                        NOMINATED NODE  READINESS GATES
hello-world-7bd56686bb-4xq5b  1/1   Running  0       8m40s  10.48.0.20  gke-
cluster-1-default-pool-990b49f7-scjx  <none>        <none>
hello-world-7bd56686bb-jvt2p  1/1   Running  0       4m29s  10.48.0.22  gke-
cluster-1-default-pool-990b49f7-scjx  <none>        <none>
hello-world-7bd56686bb-mrwsw  1/1   Running  0       8m40s  10.48.1.21  gke-
cluster-1-default-pool-990b49f7-dm0l  <none>        <none>
hello-world-7bd56686bb-v9qqh  1/1   Running  0       12s    10.48.2.16  gke-
cluster-1-default-pool-990b49f7-bzft  <none>        <none>
```