7. Autoscaling

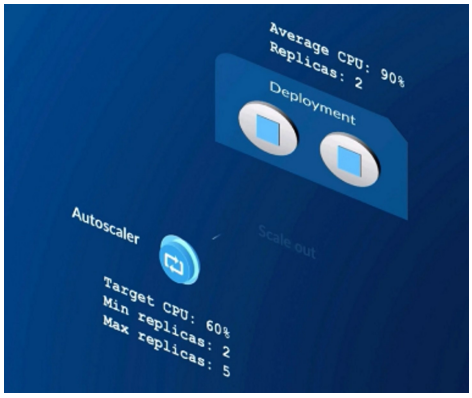
27 February 2022 15:29

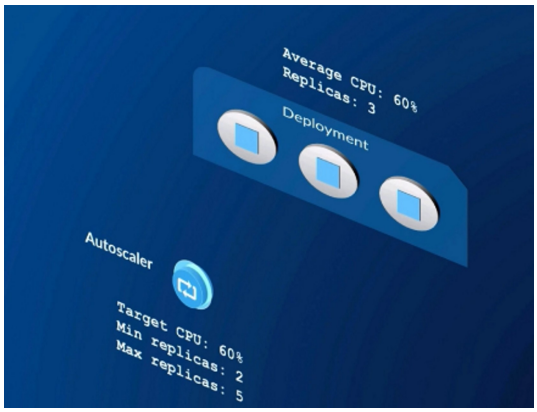
Autoscaling :

Autoscaling Deployments

- Scale automatically based on CPU utilization (or custom metrics)
- Set target CPU along with min and max replicas
- Target CPU is expressed as a percentage of the Pod's CPU request

→ If no cpu request is set autoscaling will not take any action.





Creates more replicas if average usage exceeds target.

→ Dont creat more replicas than max set

→ Dont decrease less than min set replicas.

Autoscaling depends on metrics being collected on the cluster.

Metrics

- Autoscaling depends on metrics being collected
- Metrics Server is one solution for collecting metrics
- Several manifest files are used to deploy Metrics Server (https://github.com/kubernetes-sigs/metrics-server)

→ we have to get metrics server up & running before using autoscaling.

```
admin@Jamess-MBP src % kubectl apply -f metrics-server/
clusterrole.rbac.authorization.k8s.io/system:aggregated-metrics-reader created
clusterrolebinding.rbac.authorization.k8s.io/metrics-server:system:auth-delegator created
rolebinding.rbac.authorization.k8s.io/metrics-server-auth-reader created
apiservice.apiregistration.k8s.io/v1beta1.metrics.k8s.io created
serviceaccount/metrics-server created
deployment.apps/metrics-server created
service/metrics-server created
clusterrole.rbac.authorization.k8s.io/system:metrics-server created
clusterrolebinding.rbac.authorization.k8s.io/system:metrics-server created
admin@Jamess-MBP src % _
```

→ all fully under metrics server dirs are being created.

```
admin@Jamess-MBP src % kubectl top pods -n deployments
NAME                          CPU(cores)   MEMORY(bytes)
app-tier-74cd4c68c9-h74qz     2m           45Mi
app-tier-74cd4c68c9-hcg26     1m           45Mi
app-tier-74cd4c68c9-p24sr     2m           45Mi
app-tier-74cd4c68c9-pstcr     2m           45Mi
app-tier-74cd4c68c9-sfxgm     2m           46Mi
data-tier-8646dd765b-7f457    2m           2Mi
support-tier-997bc57fb-2dtq2  10m          2Mi
support-tier-997bc57fb-dvlcv  11m          2Mi
support-tier-997bc57fb-gtd2d  12m          2Mi
support-tier-997bc57fb-kvvtr  11m          2Mi
support-tier-997bc57fb-mdlrp  12m          2Mi
admin@Jamess-MBP src %
admin@Jamess-MBP src % _
```

→ milli CPU

1000 Mi = 1 CPU

Previous

```
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: app-tier
  labels:
    app: microservices
    tier: app
spec:
  replicas: 1
  selector:
    matchLabels:
      tier: app
  template:
    metadata:
      labels:
        app: microservices
        tier: app
    spec:
      containers:
      - name: server
        image: lrakai/microservices:server-v1
        ports:
          - containerPort: 8080



        env:
          - name: REDIS_URL
            # Environment variable service discovery
            # Naming pattern:
            #   IP address: <all_caps_service_name>_SERVICE_H
            #   Port: <all_caps_service_name>_SERVICE_PORT
            #   Named Port: <all_caps_service_name>_SERVICE_P
            value: redis://$(DATA_TIER_SERVICE_HOST):$(DATA_T
            # In multi-container example value was
            # value: redis://localhost:6379
admin@Jamess-MBP src %
```

Updated

```
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: app-tier
  labels:
    app: microservices
    tier: app
spec:
  replicas: 5
  selector:
    matchLabels:
      tier: app
  template:
    metadata:
      labels:
        app: microservices
        tier: app
    spec:
      containers:
      - name: server
        image: lrakai/microservices:server-v1
        ports:
          - containerPort: 8080
        resources:
          requests:
            cpu: 20m # 20 milliCPU / 0.02 CPU
        env:
          - name: REDIS_URL
            # Environment variable service discovery
            # Naming pattern:
            #   IP address: <all_caps_service_name>_SERVICE_H
            #   Port: <all_caps_service_name>_SERVICE_PORT
            #   Named Port: <all_caps_service_name>_SERVICE_P
            value: redis://$(DATA_TIER_SERVICE_HOST):$(DATA_T
            # In multi-container example value was
            # value: redis://localhost:6379
```

→ K8s will deploy pods on to each node having min of 20 milli CPU

```
admin@Jamess-MBP src % kubectl create -f 6.1-app_tier_cpu_request.yaml -n deployments
Error from server (AlreadyExists): error when creating "6.1-app_tier_cpu_request.yaml": services "app-tier" already exists
Error from server (AlreadyExists): error when creating "6.1-app_tier_cpu_request.yaml": deployments.apps "app-tier" already exists
admin@Jamess-MBP src %
admin@Jamess-MBP src %
admin@Jamess-MBP src % _
```

→ we can delete & recreate

(or)

there apply command which updates the deployment.

(or)
→ use apply command which updates the deployment.

* CKAD → administrator course will have ideas on diff between apply & create

```
admin@Jamess-MBP src % kubectl get -n deployments deployments app-tier
NAME       READY   UP-TO-DATE   AVAILABLE   AGE
app-tier   5/5     5            5           13m
admin@Jamess-MBP src %
```

```
admin@Jamess-MBP src % cat 6.2-autoscale.yaml
apiVersion: autoscaling/v1
kind: HorizontalPodAutoscaler
metadata:
  name: app-tier
  labels:
    app: microservices
    tier: app
spec:
  maxReplicas: 5
  minReplicas: 1
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: app-tier
  targetCPUUtilizationPercentage: 70

# Equivalent to
# kubectl autoscale deployment app-tier --max=5 --min=1 --cpu-percent=70
admin@Jamess-MBP src % _
```

→ kind for autoscaling
→ lower & upper limit
70%
Version
command for above manifest

```
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
app-tier      1/1     1            1           19m
data-tier     1/1     1            1           19m
support-tier  5/5     5            5           19m
```

→ decreased.

```
admin@Jamess-MBP src % kubectl api-resources
```

→ gives all resources with their shortnames.

```
NAME                              SHORTNAMES   APIGROUP                       NAMESPACED   KIND
customresourcedefinitions         crd,crds     apiextensions.k8s.io           false        CustomResourceDefinition
apiservices                                    apiregistration.k8s.io         false        APIService
controllerrevisions                            apps                           true         ControllerRevision
daemonsets                        ds           apps                           true         DaemonSet
deployments                       deploy       apps                           true         Deployment
replicasets                       rs           apps                           true         ReplicaSet
statefulsets                      sts          apps                           true         StatefulSet
tokenreviews                                   authentication.k8s.io          false        TokenReview
localsubjectaccessreviews                      authorization.k8s.io           true         LocalSubjectAccessReview
selfsubjectaccessreviews                       authorization.k8s.io           false        SelfSubjectAccessReview
selfsubjectrulesreviews                        authorization.k8s.io           false        SelfSubjectRulesReview
subjectaccessreviews                           authorization.k8s.io           false        SubjectAccessReview
horizontalpodautoscalers          hpa          autoscaling                    true         HorizontalPodAutoscaler
cronjobs                          cj           batch                          true         CronJob
jobs                                           batch                          true         Job
certificatesigningrequests        csr          certificates.k8s.io            false        CertificateSigningRequest
leases                                         coordination.k8s.io            true         Lease
endpointslices                                 discovery.k8s.io               true         EndpointSlice
events                            ev           events.k8s.io                  true         Event
```

→ short name

```
admin@Jamess-MBP src % kubectl describe -n deployments hpa
Name:                                                  app-tier
Namespace:                                             deployments
Labels:                                                app=microservices
                                                       tier=app
Annotations:                                           <none>
CreationTimestamp:                                     Tue, 04 Aug 2020 12:10:16 +1200
Reference:                                             Deployment/app-tier
Metrics:                                               ( current / target )
  resource cpu on pods  (as a percentage of request):  40% (8m) / 70%
Min replicas:                                          1
Max replicas:                                          5
Deployment pods:                                       1 current / 1 desired
Conditions:
  Type            Status  Reason              Message
  ----            ------  ------              -------
  AbleToScale     True    ReadyForNewScale    recommended size matches current size
  ScalingActive   True    ValidMetricFound    the HPA was able to successfully calculate a replica count from cpu resource utilization (percent
age of request)
  ScalingLimited  False   DesiredWithinRange  the desired count is within the acceptable range
Events:
  Type    Reason             Age   From                       Message
  ----    ------             ----  ----                       -------
  Normal  SuccessfulRescale  107s  horizontal-pod-autoscaler  New size: 2; reason: All metrics below target
  Normal  SuccessfulRescale  77s   horizontal-pod-autoscaler  New size: 1; reason: All metrics below target
admin@Jamess-MBP src % _
```

```
admin@Jamess-MBP src % kubectl get -n deployments hpa
NAME       REFERENCE                TARGETS   MINPODS   MAXPODS   REPLICAS   AGE
app-tier   Deployment/app-tier      40%/70%   1         5         1          7m47s
admin@Jamess-MBP src %
```

```
admin@Jamess-MBP src %
admin@Jamess-MBP src % kubectl edit -n deployments hpa
```

(we can update manifest file & use apply)
(or) (use edit command).

Opens in VI & we can edit the config like replicas if we want.

```
        f:apiVersion: {}
        f:kind: {}
        f:name: {}
      f:targetCPUUtilizationPercentage: {}
    manager: kubectl
    operation: Update
    time: "2020-08-04T00:10:16Z"
  - apiVersion: autoscaling/v1
    fieldsType: FieldsV1
    fieldsV1:
      f:metadata:
        f:annotations:
          .: {}
          f:autoscaling.alpha.kubernetes.io/conditions: {}
          f:autoscaling.alpha.kubernetes.io/current-metrics: {}
      f:status:
        f:currentCPUUtilizationPercentage: {}
        f:currentReplicas: {}
        f:desiredReplicas: {}
        f:lastScaleTime: {}
    manager: kube-controller-manager
    operation: Update
    time: "2020-08-04T00:18:07Z"
  name: app-tier
  namespace: deployments
  resourceVersion: "4191"
  selfLink: /apis/autoscaling/v1/namespaces/deployments/horizontalpodautoscalers/app-tier
  uid: d8118aba-f4aa-432d-b5cc-67d6e8535e62
spec:
  maxReplicas: 5
  minReplicas: 2
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: app-tier
  targetCPUUtilizationPercentage: 70
status:
```

In VI

```
admin@Jamess-MBP src %
admin@Jamess-MBP src % kubectl edit -n deployments hpa
horizontalpodautoscaler.autoscaling/app-tier edited
```

```
Every 1.0s: kubectl get -n deployments deployment

NAME           READY   UP-TO-DATE   AVAILABLE   AGE
app-tier       2/2     2            2           24m
data-tier      1/1     1            1           24m
support-tier   5/5     5            5           24m
```

metrics server.
Prerequisites for autoscaling.