**HPA控制器的工作流程（V1版本）**

1. 创建HPA资源对象，关联对应资源例如Deployment，设定目标CPU使用率阈值，最大，最小replica数量。

前提：pod一定要设置资源限制，参数request，HPA才会工作。

2. HPA控制器每隔15秒钟（可以通过设置controller manager的–horizontal-pod-autoscaler-sync-period参数设定，默认15s）通过观测metrics值获取资源使用信息

3. HPA控制器将获取资源使用信息与HPA设定值进行对比，计算出需要调整的副本数量

4. 根据计算结果调整副本数量，使得单个POD的CPU使用率尽量逼近期望值，但不能照顾设定的最大，最小值。

5. 以上2,3,4周期循环

1.HPA控制器观测资源使用率并作出决策是有周期的，执行是需要时间的，在执行自动伸缩过程中metrics不是静止不变的，可能降低或者升高，如果执行太频繁可能导致资源的使用快速抖动，因此控制器每次决策后的一段时间内不再进行新的决策。对于扩容这个时间是3分钟，缩容则是5分钟，对应调整参数

--horizontal-pod-autoscaler-downscale-delay

--horizontal-pod-autoscaler-upscale-delay

2.自动伸缩不是一次到位的，而是逐渐逼近计算值，每次调整不超过当前副本数量的2倍或者1/2

**配置HPA实现应用横向扩展**

配置 deployment：deploy-nginx.yaml

apiVersion: apps/v1
kind: Deployment
metadata:
name: nginx
spec:
replicas: 1
selector:
matchLabels:
app: nginx
template:
metadata:
labels:
app: nginx
spec:
containers:
- name: nginx
image: nginx:1.7.9
ports:
- containerPort: 80
resources:
# 500m 就是500豪核心，即0.5个逻辑CPU.
# 2核2线程的CPU，可被系统识别为4个逻辑CPU，在K8s中对CPU的分配限制是对逻辑CPU做分片限制的。
# 也就是说分配给容器一个CPU，实际是分配一个逻辑CPU
limits:
cpu: 200m
memory: 200Mi
requests:
cpu: 200m
memory: 200Mi

配置 service：service-nginx.yaml

apiVersion: v1
kind: Service
metadata:
labels:
app: nginx
name: nginx
spec:
ports:
- name: nginx
port: 80
protocol: TCP
targetPort: 80
selector:
app: nginx
type: NodePort

配置 hpa：hpa-nginx.yaml

apiVersion: autoscaling/v1
kind: HorizontalPodAutoscaler
metadata:
name: nginx
spec:
maxReplicas: 10
minReplicas: 2
scaleTargetRef:
apiVersion: apps/v1
kind: Deployment
name: nginx
# 目标对象 cpu 利用率百分比为 40%
targetCPUUtilizationPercentage: 40

运行程序

kubectl apply -f .

演示 hpa

# 发起百万次请求，每次并发10个，并保持长连接
ab -n 1000000 -c 10 -k http://10.10.102.140/

观察 pod 的运行情况

kubectl get hpa
kubectl get pod -o wide