



Autoescalado en Kubernetes



■ Conceptos de escalado

- Escalado de PODs
- Escalado de Cluster



■ Tipos de autoescalado en Kubernetes

- Autoescalado horizontal de PODs (**HPA**) - Funcionalidad interna / estándar de Kubernetes
- Autoescalado horizontal de cluster (nodos) - Requiere integración con Cloud para provisionar nodos.
- Autoescalado vertical de PODs (**VPA**) - Funcionalidad externa
- Autoescalado vertical de cluster (nodos) - Funcionalidad externa



HPA (Horizontal POD Autoscaling)



■ Auto-escalado horizontal de Pods

- También llamado HPA
- Nos permite de forma dinámica aumentar y disminuir las réplicas de un Deployment.
- apiVersions
 - v1 solo soporta autoescalado en base a consumo de CPU.
 - v2 ha sido beta hasta 1.23. Ya es GA, permite cualquier métrica del metrics-server.
- Definimos umbral, y valores mínimo y máximos
- Es necesario definir resources en los pods.



Auto-escalado horizontal de Pods

- Demo disponible [aquí](#):

```
~$ kubectl autoscale deploy hpa-example --cpu-percent=20 --min=2 --max=20
~$ kubectl run -i --tty load-generator --rm --image=busybox /bin/sh

~# while true; do wget -q -O- http://hpa-example; done

~$ kubectl get hpa
```

```
apiVersion: autoscaling/v1
kind: HorizontalPodAutoscaler
metadata:
  name: hpa-example
spec:
  maxReplicas: 20
  minReplicas: 2
  scaleTargetRef:
    apiVersion: extensions/v1beta1
    kind: Deployment
    name: hpa-example
  targetCPUUtilizationPercentage: 20
```



Cluster Autoscaling



■ Autoescalado de Cluster

- Es un escalado horizontal de nodos del cluster (aumentar / disminuir nodos iguales).
- Cuando nos quedamos sin recursos, la manera de añadir más es añadir más nodos o añadir nodos más grandes.
- Mediante el componente “**cluster-autoscaler**” se notifica al API de gestión de infraestructura la creación de un nuevo nodo.
- Hay que instalarlo aparte, requiere integración con la cloud / infraestructura.
- Funcionamiento:
 - **Si un POD se va a crear y no quedan recursos disponibles, el scheduler notifica la necesidad de un nodo nuevo.**
 - **Cuando el nodo nuevo se crea, entonces el scheduler crea el POD en el nuevo nodo.**
 - **También funciona hacia abajo, si sobran recursos, el cluster-autoscaler se encarga de eliminar nodos.**



Autoescalado de Cluster

- En GKE

GRUPOS DE NODOS

- default-pool ^

- Nodos
- Redes
- Seguridad
- Metadatos

CLÚSTER

- Automatización
- Redes
- Seguridad

Los nombres de los grupos de nodos deben comenzar con una letra minúscula seguida por un máximo de 39 letras minúsculas, números o guiones. No pueden terminar con un guion. No se puede cambiar el nombre del grupo de nodos una vez creado.

Versión del plano de control: 1.24.8-gke.2000

☐ Posición de compactación ?

Tamaño

Cantidad de nodos *

3

El rango de direcciones del pod limita el tamaño máximo del clúster. [Más información](#)

☐ Habilitar el escalador automático de clústeres
Cluster autoscaler automatically creates or deletes nodes based on workload needs. [Learn more](#)

☐ Especificar las ubicaciones de los nodos ?



■ Node auto-provisioning (GKE)

- Si un POD requiere unas características de un nodo que no existe nos permite provisionar node-pools con nodos que las satisfaga.
- No está limitado sólo a CPU y RAM, si no a cualquier requisito hardware (GPU, Local SSD, TPU)
- Si un node-pool deja de ser utilizado, GKE lo eliminará.
- Podemos marcar como un node-pool como “manual” para que no lo elimine.
- Podemos marcar un node-pool creado por nosotros como “automático” para que si deja de usarse, GKE lo elimine.
- Con esto conseguimos una **experiencia de autoescalado vertical en cuanto a nodos.**



Node auto-provisioning

- En GKE

- Redes
- Seguridad
- Metadatos

CLÚSTER

- Automatización

- Redes
- Seguridad
- Metadatos
- Características

Ajuste de escala automático

☐ Habilitar el Ajuste de escala automático vertical de Pods

Habilitar el Ajuste de escala automático vertical de pods en un clúster te permite configurar un objeto de escalador automático vertical de pods para las cargas de trabajo del clúster. El Ajuste de escala automático vertical de Pods analiza y ajusta de manera automática las solicitudes de la CPU de los contenedores y las solicitudes de la memoria en función del uso real de los recursos de tus cargas de trabajo. [Más información](#)

☐ Habilitar el aprovisionamiento automático de nodos

El aprovisionamiento automático de nodos administra los grupos de nodos del clúster mediante la creación y eliminación de grupos de nodos según se requiera en función de las necesidades de carga de trabajo. Sin el aprovisionamiento automático de nodos, Kubernetes Engine solo iniciará nodos nuevos cuando crees grupos de nodos nuevos. [Más información](#)

Aprovisionamiento automático de etiquetas de red



Perfil de ajuste de escala automático

Equilibrado (predeterminado)



Vertical Pod Autoscaling



■ Auto-escalado vertical de PODs

- No es funcionalidad estándar en Kubernetes, hay que instalarla / configurarla de forma externa.
- Background
 - Nosotros normalmente estableceremos manualmente unos valores de requests y limits.
 - Puede que no “acertemos” con los valores adecuados de primeras.
 - Si queremos ajustar de forma óptima estos valores, es necesario ajustar correctamente los resources asignados, y puede ser costoso.
- VPA nos puede ayudar a encontrar estos valores y ajustarlos.
- Puede ayudar a reducir el mantenimiento y a usar los recursos de la manera más eficiente.



■ Funcionamiento de VPA

- En GKE se puede activar la funcionalidad.
 - [Guía / how-to](#)
 - [Referencia](#)
- Puede funcionar en 2 modos: “recomendación” y “automático”.
- Modo recomendación:
 - Únicamente nos hará un análisis de los consumos y nos dará una recomendación que podemos seguir.
- Modo automático:
 - Igualmente hará el análisis, pero adicionalmente se encargará de establecer los nuevos límites recreando los Pods (requiere reinicio)



VPA manual

```
~$ gcloud beta container clusters create [CLUSTER_NAME] --enable-vertical-pod-autoscaling
~$ gcloud beta container clusters update [CLUSTER-NAME] --enable-vertical-pod-autoscaling
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-rec-deployment
spec:
  replicas: 2
  template:
    metadata:
      labels:
        app: my-rec-deployment
    spec:
      containers:
        - name: my-rec-container
          image: nginx
```

```
apiVersion: autoscaling.k8s.io/v1beta2 # OUTDATED ;)
kind: VerticalPodAutoscaler
metadata:
  name: my-rec-vpa
spec:
  targetRef:
    apiVersion: "apps/v1"
    kind: Deployment
    name: my-rec-deployment
  updatePolicy:
    updateMode: "Off"
```

```
~$ kubectl get vpa my-rec-vpa --output yaml
```



VPA Automático

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-auto-deployment
spec:
  replicas: 2
  template:
    metadata:
      labels:
        app: my-auto-deployment
    spec:
      containers:
      - name: my-container
        image: k8s.gcr.io/ubuntu-slim:0.1
        resources:
          requests:
            cpu: 100m
            memory: 50Mi
        command: ["/bin/sh"]
        args: ["-c", "while true; do timeout 0.5s yes >/dev/null; sleep 0.5s; done"]
```

```
apiVersion: autoscaling.k8s.io/v1beta2
kind: VerticalPodAutoscaler
metadata:
  name: my-rec-vpa
spec:
  targetRef:
    apiVersion: "extensions/v1beta1"
    kind: Deployment
    name: my-rec-deployment
  updatePolicy:
    updateMode: "Auto"
```

```
~$ kubectl get vpa my-vpa --output yaml
```



Ejercicios autoescalado



■ Autoscaling - 1

Configurar un deployment que auto escale en número de PODs con un umbral, un número mínimo y un número máximo que tú elijas.



■ Autoscaling - 2

Configura un clúster con “cluster-autoscaler”.
Crea muchos PODs con resources y comprueba que funciona.



■ Autoscaling - 3

Crea un clúster con node auto provisioning.

Crea algún POD con más recursos disponibles de lo que hay en cualquier nodo actual.

Comprueba que se crea un nuevo nodo con las características requeridas.



■ Autoscaling - 4

Crea un POD de Nginx. Asigna 1 CPU y 1GBi de RAM.

Configura Vertical Pod Autoscaler

Comprueba que ajusta automáticamente los recursos del POD.





KEEPCODING

Tech School

Madrid | Barcelona | Bogotá