Google Kubernetes Engine

Despliegue y monitoreo de una aplicación escalable y resiliente.

- Oscar Hernández

oscar.hernandez@digitalonus.com

Ozrlz

- DevOps Engineer en Digital On Us (Sé algunas cosas de Kubernetes)

Conceptos para antes de empezar

- Deployment
- Ingress
- Métricas de Kubernetes
- Stackdriver
- Google Kubernetes Engine
- Cluster/Node Pool regional(es).
- Horizontal Pod Autoscaler

Para entender el repo

- Terraform
- Google Cloud Platform
- Nginx Ingress Controller

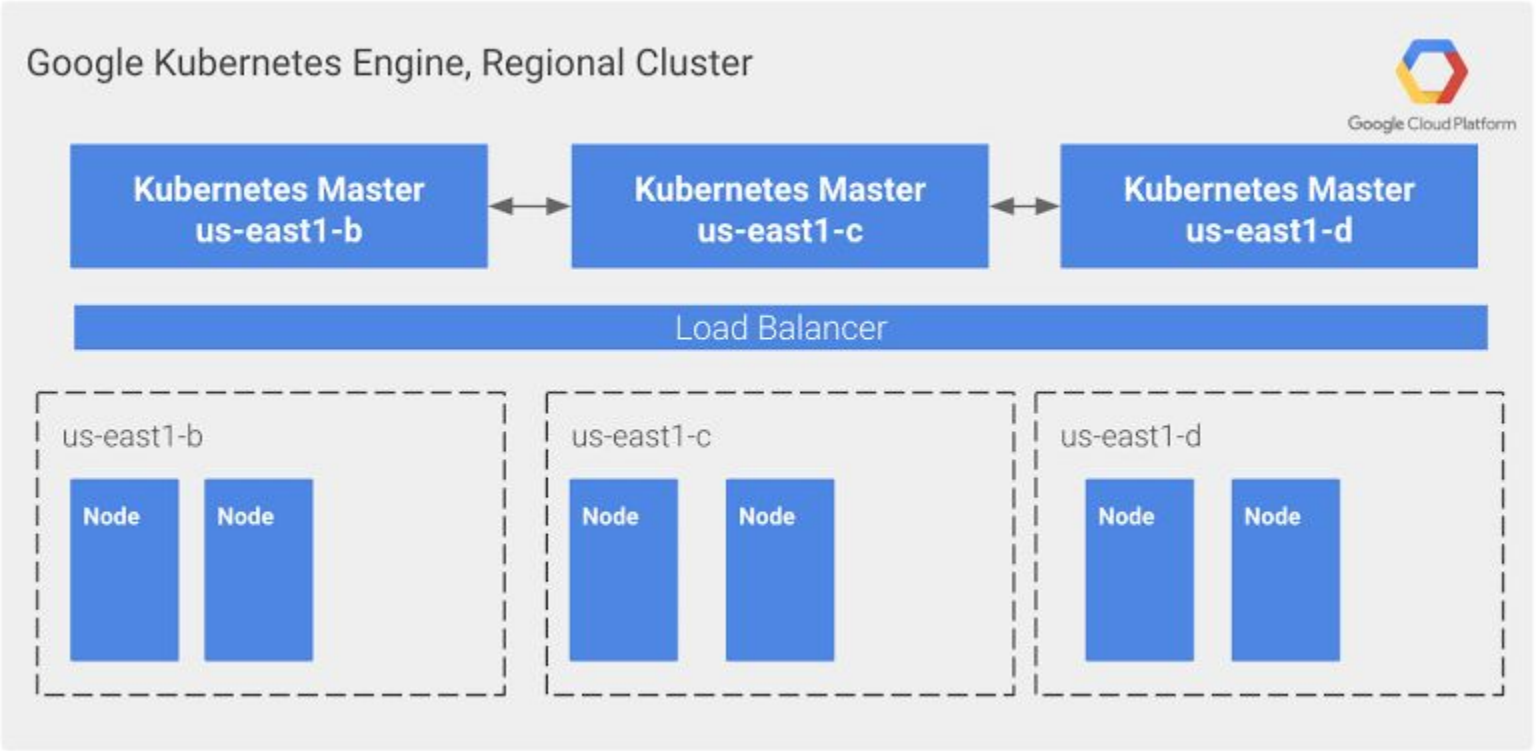
Cluster Regional

Masters y Node pools regionales

En lo que a tecnologías de la información respecta, la resiliencia es la capacidad de la infraestructura de proveer y mantener un funcionamiento aceptable a pesar de fallas y desafíos a la operación normal.

Un cluster regional ayuda (pero no cubre por completo) a que esto sea logrado, ya que los nodos maestros y trabajadores están disponibles en todas las zonas de una región, por lo que el cluster continúa funcional incluso cuando una zona completa está no disponible.

[1] - https://cloud.google.com/compute/sla



Google has an internal goal to keep the monthly uptime percentage at 99.5% for the Kubernetes API server for zonal clusters and 99.95% for regional clusters

Monitoreo

La solución actual

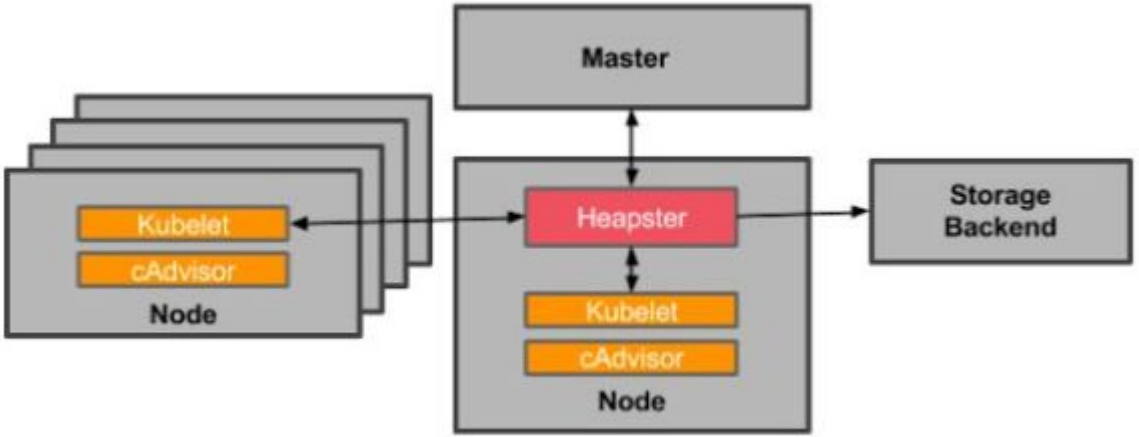
Google Kubernetes Engine cuenta ya con una solución para monitorear los recursos de tu cluster (nodos y pods) accesible a los usuarios con sólo marcar una casilla en los ajustes de tu cluster. Las métricas pueden verse rápidamente desde en los detalles del cluster, o se pueden gestionar desde Stackdriver y aprovecharlas al máximo.

Monitoreo:

- Heapster && Metrics-server

Logging:

- Fluentd



*Heapster actualmente se considera obsoleto**

Escalabilidad

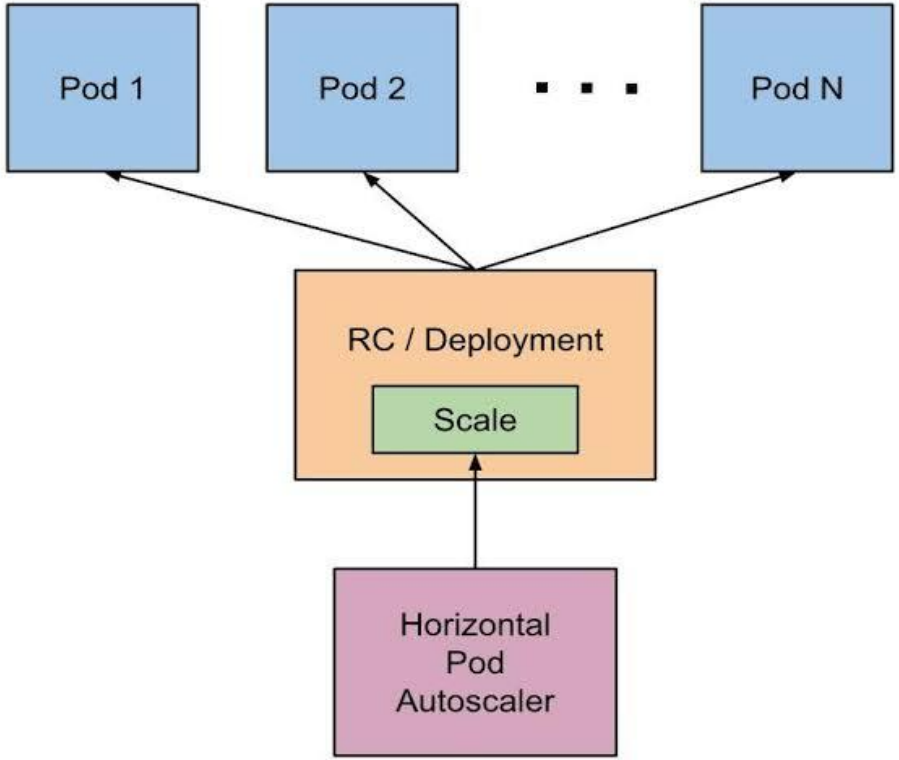
Escalado Horizontal de pods y nodos

Ajuste en tiempo real de pods y nodos usando dos componentes diferentes.

Horizontal Pod Autoscaler

Objeto de Kubernetes que ajusta el número de réplicas en un *Controlador de Replicación.*

Por defecto, sólo escalan respecto a métricas de recursos (Memoria y CPU). Tiene la habilidad de escalar respecto a métricas personalizadas o externas, pero se necesita implementar con adaptadores.



Implementations

API	metrics-server	k8s-prometheus-adapter	azure-metrics-adapter	custom-metrics-stackdriver-adapter	k8s-cloudwatch-adapter	zalando/kube-metrics-adapter	keda
metrics.k8s.io	X	X					
custom.metrics.k8s.io		X	X	X	X	X	X
external.metrics.k8s.io		X	X	X	X		X

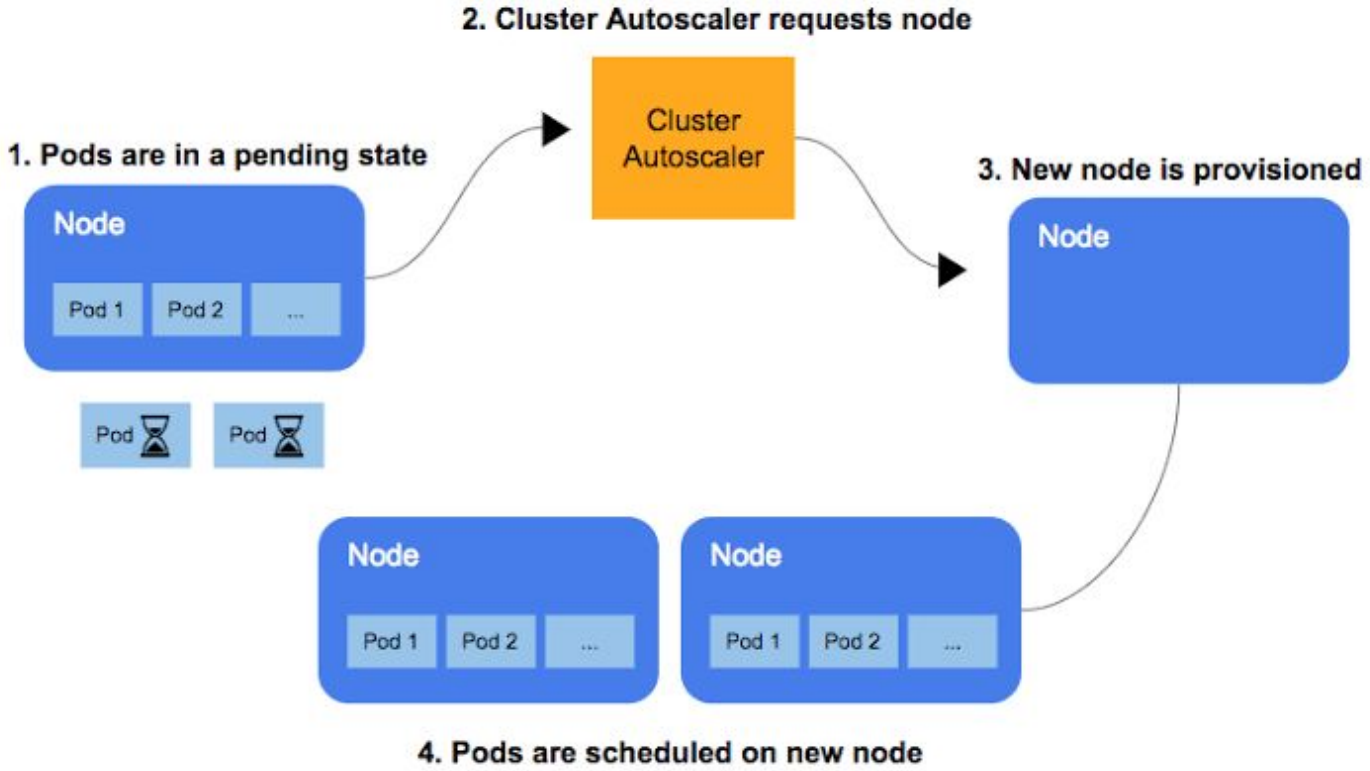
*Kubernetes te da la posibilidad de desarrollar tu propio adaptador**
KubeCon Europe 2019 <https://kccnceu19.sched.com/event/MPc1>

Cluster Autoscaler

- Proceso que corre en el master y sirve para ajustar el tamaño del cluster en tiempo real.

- Dado que tiene control sobre el cluster mismo, su funcionamiento es complicado.

- Agrega nodos al cluster cuando hay pods pendientes de agendar, tratando de mantener un balance entre zonas.
- Remueve nodos infrautilizados, agendando los pods de éstos en los nodos disponibles.



*Comportamiento esperado de un Cluster Autoscaler**

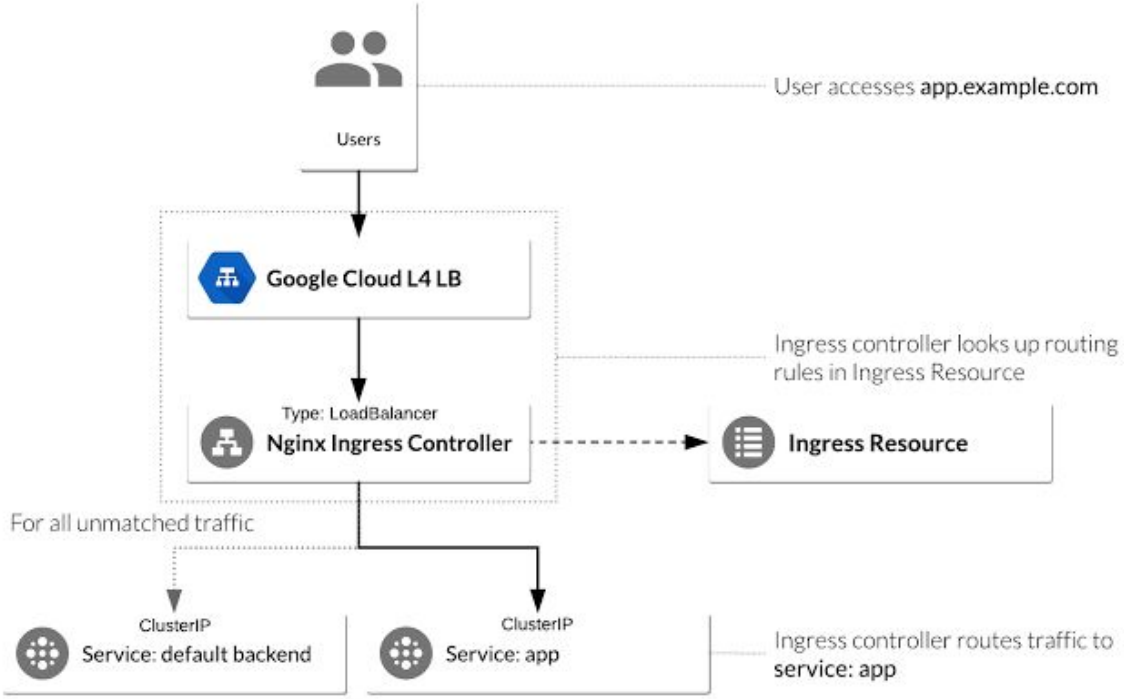
Nginx Ingress Controller

En qué consiste?

Una manera alternativa de implementar los *Ingress Resources* definidos y al mismo tiempo aprovechar features que sólo Nginx tiene.

Un despliegue de este Ingress Controller consta de:

- Un deployment que revisa los Ingress Resources definidos y se ajusta automáticamente.
- Un servicio de tipo LoadBalancer (L4) que expone a este Deployment.


Users
User accesses app.example.com
Google Cloud L4 LB
Ingress controller looks up routing rules in Ingress Resource
Type: LoadBalancer
Nginx Ingress Controller
Ingress Resource
For all unmatched traffic
ClusterIP
Service: default backend
ClusterIP
Service: app
Ingress controller routes traffic to service: app

Demo