Escalonamento de Clusters Kubernetes

Escalonamento de Clusters Kubernetes no Google Cloud

Configuração de um Cluster no Google Kubernetes Engine (GKE)

O Google Kubernetes Engine (GKE) é uma plataforma gerenciada que facilita a implantação, o gerenciamento e a escalabilidade de aplicativos em contêineres usando Kubernetes.

Passos para Configurar um Cluster no GKE:

1. Criar um Projeto no Google Cloud:

- Acesse o Google Cloud Console e selecione ou crie um novo projeto.

2. Ativar a API do Kubernetes Engine:

- No painel do Google Cloud Console, vá para “Kubernetes Engine” e ative a API se ainda não estiver ativada.

3. Criar o Cluster:

- Navegue até “Clusters” no menu do Kubernetes Engine.

- Clique em “Criar Cluster” e configure os parâmetros:

Nome do Cluster: Escolha um nome identificável.

Localização: Escolha uma zona ou região específica.

Tipo de Máquina: Selecione o tipo de instância do Compute Engine que deseja usar.

Número de Nós: Defina o número de nós desejados no cluster.

4. Finalizar a Configuração:

- Após definir as configurações, clique em "Criar". O GKE provisionará automaticamente os recursos necessários e criará o cluster.

Imagine que você está desenvolvendo uma aplicação web de grande escala e precisa de uma infraestrutura escalável. O GKE facilita a configuração de clusters para gerenciar e escalar sua aplicação conforme a demanda.

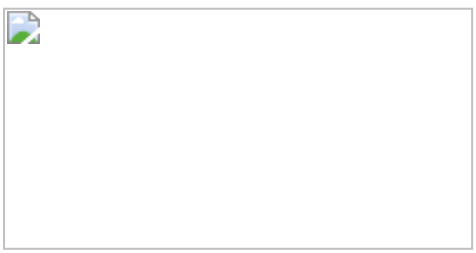
Criação de Contêiner Docker com Cloud Build

Cloud Build é um serviço do Google Cloud que permite compilar, testar e implantar aplicativos rapidamente. Ele pode ser usado para criar imagens Docker a partir do código-fonte.

Passos para Criar um Contêiner Docker:

1. Criação do Dockerfile:

- No repositório de código-fonte, crie um Dockerfile que define como sua imagem Docker será construída. Exemplo básico para uma aplicação Node.js:



2. Configuração do Cloud Build:

- No Google Cloud Console, vá para “Cloud Build” e clique em “Configurações”.

- Configure uma integração com o repositório de código-fonte (por exemplo, GitHub ou Cloud Source Repositories).

3. Construção da Imagem:

Crie um arquivo cloudbuild.yaml para definir o pipeline de build.

4. Iniciar o Build:

- Execute o Cloud Build para compilar a imagem Docker:
gcloud builds submit --config cloudbuild.yaml .

Se você está desenvolvendo um novo serviço microservice em Node.js, o Cloud Build pode automatizar a criação da imagem Docker e seu armazenamento no Google Container Registry, prontamente disponível para implantação.

Implantação de Contêineres no GKE

Depois de criar a imagem Docker, o próximo passo é implantá-la no cluster GKE.

Passos para Implantar Contêineres no GKE:

1. Preparar o Manifesto de Implantação:

- Crie um arquivo YAML de implantação (deployment.yaml) que define a aplicação e os parâmetros de escalonamento:

2. Implantar no Cluster:

- Use o comando kubectl apply para implantar kubectl apply -f deployment.yaml

Suponha que você tenha criado um microservice que deve ser escalável e resistente a falhas. Implantando-o no GKE com 3 réplicas garantirá alta disponibilidade e balanceamento de carga automático.

Exposição da Implantação no GKE

Para tornar a aplicação acessível externamente, é necessário criar um serviço que exponha os pods.

Passos para Expor a Implantação:

1. Criar um Serviço Kubernetes:

- Crie um arquivo service.yaml que define um serviço do tipo LoadBalancer:

2. Implantar o Serviço:

Use kubectl apply para criar o serviço: kubectl apply -f service.yaml

3. Acesso Externo:

- Após a criação do serviço, um endereço IP externo será provisionado, permitindo o acesso público à aplicação.

Se você está implantando uma aplicação web que deve ser acessível a usuários finais, configurar um serviço LoadBalancer no GKE é a maneira mais simples de expor a aplicação na internet.

Escalonamento de Implantações no GKE

O GKE oferece várias opções para escalonamento de implantações, tanto horizontal quanto vertical, para atender à demanda de tráfego.

Tipos de Escalonamento:

• **Escalonamento Horizontal (HPA):**

- Ajusta automaticamente o número de pods com base na utilização de CPU, memória ou outras métricas.

Configuração básica: kubectl autoscale deployment my-app --cpu-percent=80 --min=3 --max=10

• **Escalonamento Vertical:**

- Ajusta os recursos de CPU e memória de um contêiner.

- Usa o Vertical Pod Autoscaler para sugerir ou aplicar automaticamente as mudanças.

Uma loja online durante a Black Friday pode precisar escalar horizontalmente sua aplicação para suportar o aumento de tráfego. Com o

HPA configurado, o GKE gerencia automaticamente o aumento e a redução do número de réplicas conforme necessário.

Atualização de Sites Sem Tempo de Inatividade

Kubernetes permite a implantação de atualizações sem tempo de inatividade, garantindo que sua aplicação permaneça disponível durante o processo.

Passos para Atualização Sem Tempo de Inatividade:

1. Rolling Updates:

- Kubernetes realiza atualizações de maneira gradual, substituindo os pods antigos por novos. A configuração padrão de um deployment já usa rolling updates.

Para iniciar uma atualização:

kubectl set image deployment/my-app my-app=gcr.io/[PROJECT_ID]/my-app:v2

Verificação de Status:

Monitore o progresso da atualização:
kubectl rollout status deployment/my-app

2. Rollback em Caso de Falha:

Se algo der errado, é possível reverter a atualização:
kubectl rollout undo deployment/my-app

Se sua aplicação de e-commerce precisa ser atualizada com novos recursos ou correções de bugs, Kubernetes pode fazer isso de forma contínua, garantindo que os clientes não experimentem interrupções.

Conteúdo Bônus

Artigo: "Execute cargas de trabalho de pilha completa em grande escala no GKE" - Disponível em Cloud.Google
"Neste tutorial, mostramos como executar um aplicativo da Web apoiado por um banco de dados relacional altamente disponível em grande escala no Google Kubernetes Engine (GKE)."

Referências Bibliográficas

BASSO, D. E. **Administração de Redes de Computadores**. Contentus, 2020.
KUROSE, J. F.; ROSS, K. W. **Redes de Computadores e a Internet: Uma Abordagem Top-Down**. 8. ed. Pearson, 2021.
MARINHO, A. L.; CRUZ, J. L. da. (Orgs.). **Desenvolvimento de Aplicações para Internet**. 2. ed. Pearson, 2020.
PUGA, S.; RISSETTI, G. **Lógica de Programação e Estruturas de Dados, com Aplicações em Java**. 3. ed. Pearson, 2016.
ROHLING, L. J. **Segurança de Redes de Computadores**. Contentus, 2020.
SILVA, C. F. da. **Projeto Estruturado e Gerência de Redes**. Contentus, 2020.
TANENBAUM, A. S.; FEAMSTER, N.; WETHERALL, D. J. **Redes de Computadores**. 6. ed. Pearson, 2021.
TOCCI, R. J.; WIDMER, N. S.; MOSS, G. L. **Sistemas Digitais: Princípios e Aplicações**. 12. ed. Pearson, 2018.

Ir para exercício