

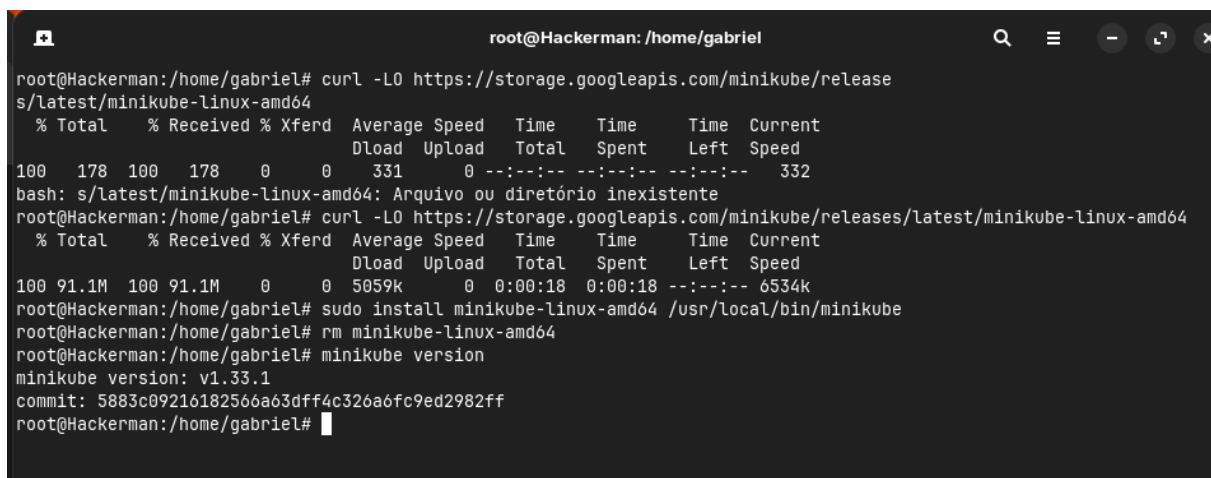
DevOps Módulo 15 - IT Talent

Escrito por: [Gabriel Oliveira dos Santos](#)

Github: <https://github.com/Hypothesis>

Instalação do Minikube

```
curl -LO https://storage.googleapis.com/minikube/release
```



```
root@Hackerman: /home/gabriel
root@Hackerman:/home/gabriel# curl -LO https://storage.googleapis.com/minikube/release
s/latest/minikube-linux-amd64
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload   Total   Spent    Left   Speed
100 178 100 178    0     0  331      0 --:--:-- --:--:-- --:--:--  332
bash: s/latest/minikube-linux-amd64: Arquivo ou diretório inexistente
root@Hackerman:/home/gabriel# curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload   Total   Spent    Left   Speed
100 91.1M 100 91.1M    0     0 5059k      0  0:00:18  0:00:18 --:--:-- 6534k
root@Hackerman:/home/gabriel# sudo install minikube-linux-amd64 /usr/local/bin/minikube
root@Hackerman:/home/gabriel# rm minikube-linux-amd64
root@Hackerman:/home/gabriel# minikube version
minikube version: v1.33.1
commit: 5883c09216182566a63dffa4c326a6fc9ed2982ff
root@Hackerman:/home/gabriel#
```

Instalação do Kubectl

-Instalando os Binários

```
curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io)
```

-Instalando o Kubectl

```
sudo install -o root -g root -m 0755 kubectl /usr/local/bin/k
```

```
root@Hackerman: /home/gabriel
root@Hackerman:/home/gabriel# curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left   Speed
100  138    100  138     0     0    275      0 --:--:-- --:--:-- --:--:--   275
100 49.0M  100 49.0M     0     0 5624k      0  0:00:08  0:00:08 --:--:-- 5701k
root@Hackerman:/home/gabriel# sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl
root@Hackerman:/home/gabriel#
```

Verificando a versao do Kubectl

```
kubectl version --client
```

```
root@Hackerman: /home/gabriel
root@Hackerman:/home/gabriel# curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left   Speed
100  138    100  138    0     0    275      0  --:--:-- --:--:-- --:--:--   275
100 49.0M  100 49.0M    0     0 5624k      0  0:00:08  0:00:08 --:--:-- 5701k
root@Hackerman:/home/gabriel# sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl
root@Hackerman:/home/gabriel# kubectl version --client
Client Version: v1.30.2
Kustomize Version: v5.0.4-0.20230601165947-6ce0bf390ce3
root@Hackerman:/home/gabriel#
```

Pods

Introdução ao Kubernetes

- Apresentação ao trabalho com clusters Kubernetes
- Foco em pods
- Padrões usados pelo kubectl
- Exemplos de arquivos de manifesto

Rede de Contêiner

- Todos os pods compartilham uma rede de contêiner
- Comunicação entre todos os pods
- Cada pod recebe um endereço IP único

Todos pods tem uma rede de containers, assim qualquer pod pode conversar com qualquer outro pod através da internet. A abstração do Kubernetes, permite que a gente não veja qual a complexidade ocorre por trás de cada construção de pod.

Declaração de Pods

- Inclui propriedades esperadas (imagem do contêiner, portas, política de reinício, limites de CPU e memória)
- Especificidades dos pods no Kubernetes

Declaração de Pods é semelhante ao Dockerfile, que especificam e declaram os Pods.

Arquivos de Manifesto

- Descrevem todos os tipos de recursos no Kubernetes
- Configurações específicas para cada tipo de recurso
- Enviados para o servidor API do Kubernetes usando kubectl

Ações do Cluster

- Seleção de nó com recursos disponíveis
- Agendamento do pod no nó
- Download das imagens do contêiner e execução

Pods, Manifesto, Criação, Descrição, Portas

Inicie o minikube:

```
minikube start
```

Vamos rodar um arquivo de manifesto, de nome basico.yml, so copiar isso para seu arquivo:

```
apiVersion: v1
kind: Pod
metadata:
  name: meu-pod
```

```
spec:
  containers:
  - name: my-container
    image: nginx:latest
    ports:
      - containerPort: 80
```

Vamos criar um pod apartir de um arquivo:

```
kubectl create -f "nome.yml"
```

para vermos o pod:

```
kubectl get pods
```

para vermos todas as especificações do nosso pod:

```
kubectl describes "nomepod"
```

para deletarmos nosso pod:

```
kubectl delete pod "nomepod"
```

Pods - Labels, Qualidade de Serviço, Gestão de Rec

Um label são como "etiquetas", que isso pode ser uma tag que diferencia varios pods rodando.

```
apiVersion: v1
kind: Pod
metadata:
  name: meu-pod
  labels:
    app: Backend
spec:
  containers:
  - name: my-container
```

```
image: nginx:latest
resource:
  requests:
    memory: "128Mi"
    cpu: "500m"
  limits:
    memory: "128Mi"
    cpu: "500m"
ports:
  - containerPort: 80
```

Essa foi um exemplo de como podemos colocar um label dentro de um .yaml para criarmos nossos pods.

O 128Mi, equivale a 128 Mega Bytes, pois o byte são 8 bits, e 1kb são 1024 bytes, o Mi é pra deixar certinho com os bits.

Já na cpu, 500m, significa que usa a metade de um cpu normal.

Serviço

Problemas de Rede

- Reagendamento de Pods em caso de falhas nos nós
- Novo endereço IP atribuído após reagendamento
- Uso de serviços para resolver problemas de rede

Serviço é que define as regras de rede, que tornam possível acessar nosso Pod pela internet.

Definição de Serviço

- Serviços definem regras de rede para acessar Pods no cluster e na internet
- Utilização de rótulos para acessar um grupo de Pods

Acesso Fixo

- Serviço proporciona um endereço fixo para os clientes
- Distribuição de solicitações entre Pods para balancear a carga

Serviços na prática

servico_web.yml que usaremos:

```
apiVersion: v1
kind: Service
metadata:
  labels:
    app: backend
  name: backend
spec:
  ports:
    - port: 80
  selector:
```



```
app: backend  
type: NodePort
```

agora vamos criar nosso serviço:

```
kubectl create -f servico_web.yml
```

para acessarmos nossa aplicação, usamos o ip do kubernetes com a porta da nossa aplicação rodando:

```
ip_kubernetes:porta_pod
```