THIAGO ADRIANO

SOFTWARE ARCHITECTURE

KUBERNETES

AULA 04

SUMÁRIO

O QUE VEM POR AÍ?

Nesta aula, os professores apresentam um recurso muito utilizado no Kubernetes, o Service. Por meio de exemplos práticos, você vai aprender como criar os seus primeiros Services utilizando os tipos privados e públicos. Além disso, os docentes apresentarão um outro recurso, chamado ConfigMap, que pode ser utilizado para organizar os seus arquivos de configuração.

HANDS ON

Nesta aula, os docentes demonstrarão os tipos de Services por meio de exemplos práticos junto com o arquivo ConfigMap.

SAIBA MAIS

Nesta aula, aprenderemos sobre os Services e ConfigMaps, que são recursos importantes no Kubernetes para gerenciar e expor serviços e configurações de aplicativos de forma eficiente.

Services e ConfigMap

Os Services são recursos importantes do Kubernetes, que permitem que você exponha serviços dentro do cluster Kubernetes e gerencie o acesso a esses serviços de forma eficiente.

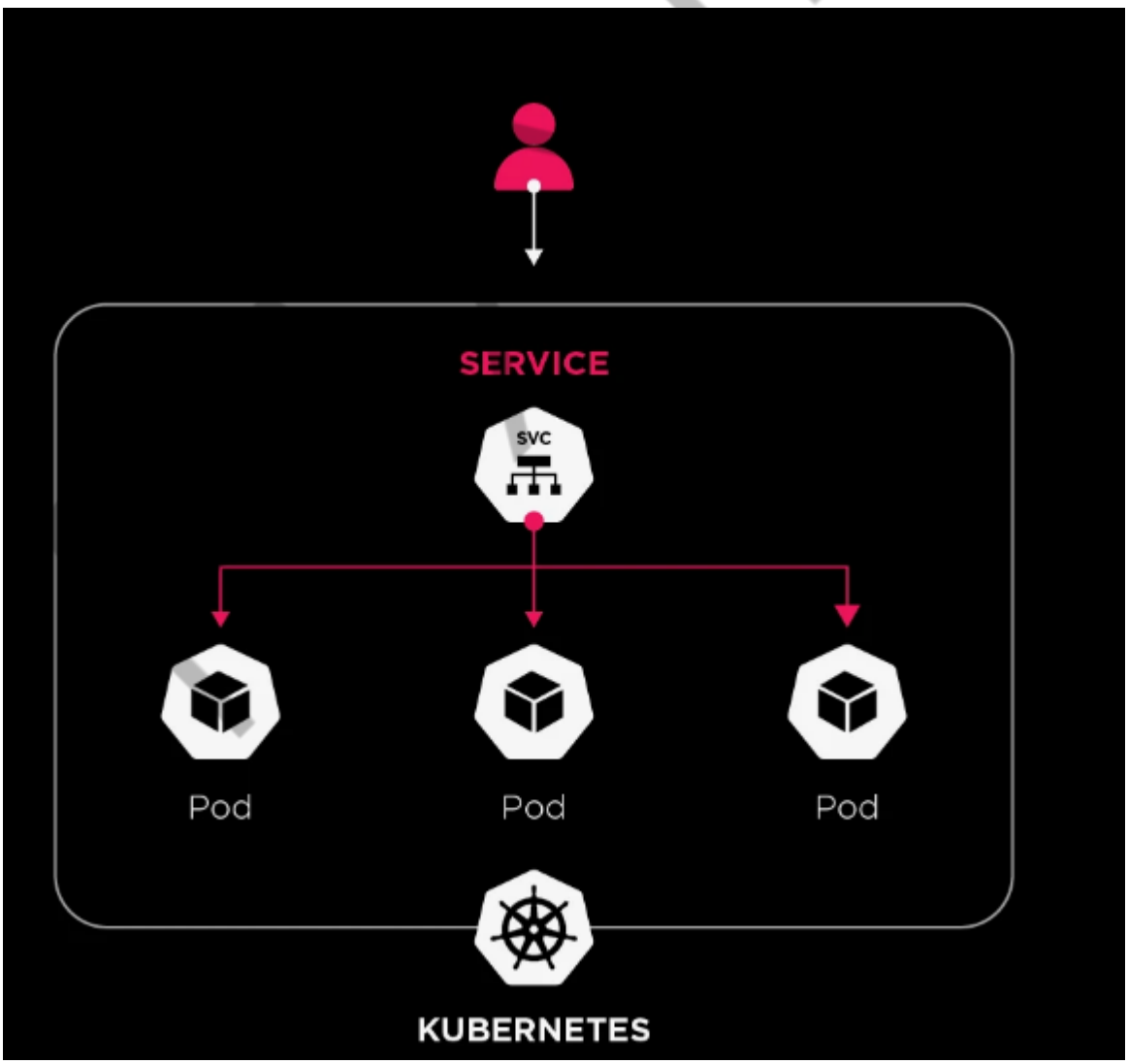

Figura 1 – Exemplo de services e Pods
Fonte: https://5i4i.short.gy/xMtNmk (2022), adaptado por FIAP (2023)

Um Service é um objeto do Kubernetes que permite que você exponha um conjunto de Pods como um serviço. Ele fornece um único ponto de entrada para acessar esses Pods, permitindo que você gerencie o acesso a esses serviços de forma eficiente. O Service é responsável por distribuir o tráfego entre os Pods do serviço, garantindo que o tráfego seja distribuído de forma justa e que a carga seja balanceada.

Existem três tipos de Services no Kubernetes:

- **ClusterIP**: este é o tipo padrão de Service no Kubernetes. Ele expõe o serviço em um endereço IP interno do cluster e é acessível apenas dentro do cluster Kubernetes. É útil para serviços que precisam se comunicar com outros serviços dentro do cluster Kubernetes.
- **NodePort**: este tipo de Service expõe o serviço em um número de porta no nó do cluster Kubernetes. Isso permite que o serviço seja acessível fora do cluster usando o endereço IP do nó e a porta atribuída. É útil para serviços que precisam ser acessados fora do cluster, como aplicativos da web.
- **LoadBalancer**: este tipo de Service expõe o serviço em um balanceador de carga externo. Isso permite que o serviço seja acessível fora do cluster usando um endereço IP externo e uma porta atribuída. É útil para serviços que precisam ser acessados de fora do cluster Kubernetes e exigem balanceamento de carga.

Nós já vimos como criar um Service do Kubernetes na prática, mas para que você tenha um melhor entendimento, vamos entender esta criação na parte teórica.

Conforme visto na videoaula onde criamos uma tela de login e uma API para receber os nossos dados, você pode criar um Service usando um arquivo de manifesto YAML.

O arquivo YAML deve especificar o tipo de Service, o seletor para os Pods que o Service deve direcionar e outras configurações, como portas e protocolos.

Na figura 2 – “Criação de um Service com ClusterIP”, há o exemplo de um arquivo YAML básico para criar um Pod no Kubernetes:

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: meu-pod
spec:
  containers:
  - name: meu-container
    image: nginx
    ports:
    - containerPort: 80
```

Figura 2 – Criação de um service com ClusterIP
Fonte: elaborado pelo autor (2023)

Neste exemplo, o arquivo YAML define um Pod chamado "meu-pod", que executa um container chamado "meu-container". O container usa a imagem do Nginx e expõe a porta 80.

Para ficar mais evidente, vamos entender os itens deste arquivo YAML:

- apiVersion: a versão da API do Kubernetes que está sendo usada. Neste caso, v1.
- kind: o tipo de objeto Kubernetes que estamos criando. Neste caso, um Pod.
- metadata: as informações de metadados sobre o Pod, incluindo o nome do Pod.
- spec: as especificações do Pod, incluindo informações sobre os containers que serão executados dentro do Pod.
- containers: uma lista de containers que serão executados dentro do Pod. Neste exemplo, há apenas um container.
- name: o nome do container. Neste exemplo, o nome é "meu-container".
- image: a imagem do container que será usada. Neste caso, a imagem do Nginx.
- ports: as portas que serão expostas pelo container. Neste exemplo, a porta 80 é exposta.

ConfigMaps

Em um ambiente de microsserviços, os serviços são geralmente configurados por meio de variáveis de ambiente. Essas variáveis podem ser definidas em arquivos YAML ou JSON e injetadas no container do serviço. No entanto, em alguns casos, as configurações podem ser muito extensas ou precisam ser compartilhadas entre vários serviços. É nesse cenário que entra em jogo o ConfigMap.

O ConfigMap é um recurso do Kubernetes que permite separar as configurações de um container de seus artefatos de implantação. Isso significa que é possível armazenar as configurações em um local centralizado e gerenciá-las separadamente do container.

As configurações podem ser armazenadas em um ConfigMap em um arquivo YAML. Essas configurações podem ser injetadas em um container de várias maneiras, incluindo como variáveis de ambiente, como arquivos montados ou como argumentos de linha de comando.

Para usar o ConfigMap para configurar um serviço, é necessário criar um objeto ConfigMap que contenha as configurações do serviço. Isso pode ser feito usando o seguinte comando:

```
kubectl create configmap my-service-config --from-file=config.yaml
```

Figura 3 – Criação de um service com ClusterIP
Fonte: elaborado pelo autor (2023)

E para verificar os dados do Configmap criado, podemos utilizar o seguinte comando:

```
PS D:\FIAP_POS\Kubernetes\poc> kubectl describe configmaps my-service-config
Name:         my-service-config
Namespace:    default
Labels:       <none>
Annotations:  <none>

Data
====
config.yaml:
----


BinaryData
====

Events:  <none>
PS D:\FIAP_POS\Kubernetes\poc>
```

Figura 4 – Criação de um service com ClusterIP
Fonte: elaborado pelo autor (2023)

Como este é um arquivo vazio, note que na descrição não temos nada.

Depois que o ConfigMap é criado, é possível injetar suas configurações em um container por meio de um objeto de implantação. Por exemplo, o seguinte objeto de implantação injeta as configurações do ConfigMap em um container como variáveis de ambiente:

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-service
spec:
  replicas: 1
  selector:
    matchLabels:
      app: my-service
  template:
    metadata:
      labels:
        app: my-service
    spec:
      containers:
        - name: my-service-container
          image: my-service-image
          envFrom:
            - configMapRef:
                name: my-service-config
```

Figura 5 – Criação de um service com ClusterIP

Fonte: elaborado pelo autor (2023)

Neste exemplo, o objeto de implantação contém um container chamado my-service-container, que usa a imagem my-service-image. As configurações do ConfigMap são injetadas como variáveis de ambiente no container, usando a seção envFrom.

Além disso, é possível injetar as configurações do ConfigMap como arquivos montados ou como argumentos de linha de comando, dependendo do formato em que as configurações são armazenadas.

O QUE VOCÊ VIU NESTA AULA?

Nesta aula, você aprendeu o que são os Services do Kubernetes, os tipos que você pode utilizar com ele e como utilizar o arquivo ConfigMap para organizar as suas configurações de ambiente.

O que achou do conteúdo? Conte-nos no Discord! Estamos disponíveis na comunidade para tirar dúvidas, fazer networking, enviar avisos e muito mais.

REFERÊNCIAS

DOBIES, J. **Operadores do Kubernetes: Automatizando a Plataforma de Orquestração de Contêineres.** [s.l.]: Novatec, 2020.

PALAVRAS-CHAVE

Arquitetura de Software. ConfigMap. Services.