SER CHALLENGE

quarta-feira, 15 de janeiro de 2025 16:58

Configure o ambiente do desafio

Instale qualquer cluster K8s local (ex: Minikube) em sua máquina e documente sua configuração, para que possamos executar sua solução.

RESPOSTA:

Teste realizado no seguinte ambiente:

VirtualBox v7
ubuntu-24.04.1-live-server-amd64
minikube version: v1.34.0
Máquina Virtual 2gb ram - 2cpu - 40gb

Parte 1 - Configure os aplicativos

Gostariamos que essa aplicação sre-challenge-app e seu banco de dados fossem executados em um cluster K8s.
Requisitos

1. A aplicação deve ser acessível de fora do cluster.

   Resposta:
   Fiz a alteração no script, tirei o type:ClusterIP para type: NodePort, foi acrescentado a porta 30006.
   nodePort: 30006 ***# Um número de porta específico para acesso externo***
   type: NodePort

2. Manifestos de implantação do kubernetes para executar com limitação de requests e usando HPA.

   Resposta:

   ## Explicação das Alterações:

   Limitações de Recursos (resources.requests e resources.limits):

   - **Requests: A quantidade de recursos garantidos que o Kubernetes alocará para o pod.**
   - **Limits: A quantidade máxima de recursos que o pod pode consumir.**

   No caso do MySQL, os valores são:

   - **Requests: 256 Mi de memória e 250m de CPU.**
   - **Limits: 512 Mi de memória e 500m de CPU.**

   No caso da aplicação Web, os valores são:

   - **Requests: 128 Mi de memória e 100m de CPU.**
   - **Limits: 256 Mi de memória e 250m de CPU.**

   Horizontal Pod Autoscaler (HPA):

   - **MySQL HPA: Escala o número de réplicas da implantação db-deployment com base na utilização de CPU. O HPA ajusta o número de réplicas entre 1 e 5, com um objetivo de 80% de utilização de CPU.**
   - **Web HPA: Escala o número de réplicas da implantação web-deployment com base na utilização de CPU. O HPA ajusta o número de réplicas entre 1 e 10, com um objetivo de 80% de utilização de CPU.**

   Configuração de Recursos:

   - **Com essa configuração, o Kubernetes garante que cada pod tenha uma quantidade**

mínima de recursos (requests) e não consuma mais do que o limite especificado (limits).

- **O HPA ajustará automaticamente o número de réplicas dos pods para atender à demanda de carga, com base na utilização de recursos (neste caso, a CPU).**

Parte 2 - Corrigir o problema

A aplicação tem um problema. Encontre e corrija! Você saberá que corrigiu o problema quando o estado dos pods no namespaces for semelhante a este:

```
NAME                                READY   STATUS          RESTARTS   AGE
db-5877fd4d4d-qmngl                 1/1     Running   0        6m50s
sre-challenge-app-59fd5ffc57-lm2xs  1/1     Running   0        7s
```

Consegui fazer a correção no pod db, segue a explicação:

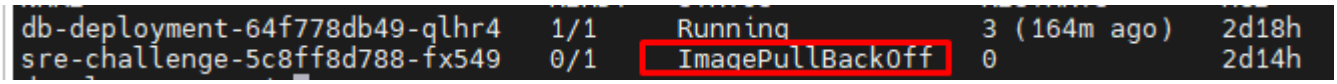
```
# MySQL Deployment
apiVersion: apps/v1
kind: Deployment
metadata:
  name: db-deployment
  labels:
    app: database
spec:
  replicas: 1
  selector:
    matchLabels:
      app: database
  template:
    metadata:
      labels:
        app: database
    spec:
      containers:
      - name: mysql
        image: mysql/mysql-server:8.0.23
        ports:
        - containerPort: 3306
        env:
        - name: MYSQL_ROOT_PASSWORD
          value: "12345678"
        - name: MYSQL_DATABASE
          value: "emp"
        - name: MYSQL_USER
          value: "root"
        - name: MYSQL_PASSWORD
          value: "12345678"

---

# MySQL Service
apiVersion: v1
```

```
kind: Service
metadata:
  name: db-service
spec:
  selector:
    app: database
  ports:
  - protocol: TCP
    port: 3306
    targetPort: 3306
  type: ClusterIP

---

# Web Deployment
apiVersion: apps/v1
kind: Deployment
metadata:
  name: web-deployment
  labels:
    app: web
spec:
  replicas: 1
  selector:
    matchLabels:
      app: web
  template:
    metadata:
      labels:
        app: web
    spec:
      containers:
      - name: web
        image: demo/sre-challenge
        ports:
        - containerPort: 8080
        env:
        - name: DATABASE_ROOT_PASSWORD
          value: "12345678"

---

# Web Service
apiVersion: v1
kind: Service
metadata:
  name: web-service
spec:
  selector:
    app: web
  ports:
  - protocol: TCP
    port: 8080
    targetPort: 8080
  type: NodePort

sre-challenge-app-59fd5ffc57-lm2xs   1/1       Running   0          7s
```

Sobre o pod SER:

Não foi possível executar, pois toda a vez que tentava fazer o pull da imagem, apresentou mensagem de não autorizado.
Acredito que essa imagem seja privada e não publica.

VIDE imagem abaixo:



Acredito que talvez seria necessário rodar essa imagem local e jogar ela para o k8s com o comando:

kubectl apply -f e o nome do pod

Parte 3 - Melhores práticas

Essa aplicação tem uma falha de segurança e gostariamos que as credenciais do MYSQL fossem armazenadas em uma secret do Kubernetes.
Requisitos

Manifesto do kubernetes usando a API de secret com as credenciais do Banco para implantação.

Resposta:

- **API de Secret: Criamos um Secret chamado mysql-secret para armazenar os valores sensíveis do banco de dados. As credenciais foram codificadas em Base64.**
- **Uso de valueFrom: Substituímos os valores explícitos das variáveis de ambiente nos deployments por referências ao Secret usando valueFrom.secretKeyRef.**
- **Segurança: Agora, as credenciais estão armazenadas em um Secret, que pode ser gerenciado separadamente e é protegido por permissões no cluster.**

Manifesto do kunernetes da aplicação com as informações da secret criada anteriormente.

Resposta:
Secret Referenciado:

- **As variáveis de ambiente DATABASE_ROOT_PASSWORD, DATABASE_NAME, DATABASE_USER e DATABASE_PASSWORD utilizam a Secret previamente criada (mysql-secret) para injetar os valores correspondentes.**

Deployment da Aplicação Web:

- **Configurado para utilizar a imagem demo/sre-challenge.**
- **Alocado recursos de CPU e memória para garantir desempenho controlado.**

Service para Exposição:

- **Um Service do tipo NodePort expõe a aplicação na porta 8080 para ser acessível no cluster.**

Horizontal Pod Autoscaler:

- **Configurado para escalar a aplicação web dinamicamente, ajustando entre 1 e 10 réplicas com base no uso médio de CPU.**

Com isso, a aplicação pode acessar o banco de dados utilizando as credenciais seguras gerenciadas pelo Kubernetes Secret.

Configuração do código da aplicação utilizando uma variável que foi referenciada no secrets do K8s (Application Properties do Java)

Resposta:
Para configurar uma aplicação Java (por exemplo, usando Spring Boot) para usar as variáveis de ambiente referenciadas no Kubernetes Secret, você pode configurar o arquivo application.properties ou application.yml para referenciar essas variáveis. Aqui está como fazer isso:

Configuração em application.properties

```
# Banco de Dados
spring.datasource.url=jdbc:mysql://db-service:3306/${DATABASE_NAME}
spring.datasource.username=${DATABASE_USER}
spring.datasource.password=${DATABASE_PASSWORD}
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
# Configurações Adicionais (opcional)
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.jpa.database-platform=org.hibernate.dialect.MySQL8Dialect
```

Configuração em application.yml

```
spring:
  datasource:
    url: jdbc:mysql://db-service:3306/${DATABASE_NAME}
    username: ${DATABASE_USER}
    password: ${DATABASE_PASSWORD}
    driver-class-name: com.mysql.cj.jdbc.Driver
  jpa:
    hibernate:
      ddl-auto: update
    show-sql: true
    database-platform: org.hibernate.dialect.MySQL8Dialect
```

Parte 4 - Perguntas

Sinta-se à vontade para expressar seus pensamentos e compartilhar suas experiências com exemplos do mundo real com os quais você trabalhou no passado.

Requisitos O que você faria para melhorar essa configuração e torná-la “pronta para produção”? Existem 2 microsserviços mantidos por 2 equipes diferentes. Cada equipe deve ter acesso apenas ao seu serviço dentro do cluster. Como você abordaria isso? Como você evitaria que outros serviços em execução no cluster se comunicassem com o sre-challenge-app?

Resposta:

1. Melhorias na Configuração para Produção

Segurança:

1. **Utilizar Secrets Gerenciados Externamente:**
   - **Em vez de gerenciar Secrets diretamente no cluster, considere o uso de ferramentas como HashiCorp Vault, AWS Secrets Manager, ou Azure Key Vault.**
   - **Configure o Kubernetes para integrar-se com essas ferramentas para maior segurança.**
2. **Habilitar TLS para Comunicação Interna:**
   - **Configure a comunicação interna entre serviços para usar TLS. Você pode usar ferramentas como Istio ou cert-manager para gerenciar certificados.**
3. **Autenticação e Autorização Entre Serviços:**
   - **Implemente autenticação baseada em tokens, como JSON Web Tokens (JWT), para validar requisições entre serviços.**
   - **Use um Identity Provider (IDP) como Keycloak ou Okta.**
4. **Configurar NetworkPolicies:**
   - **Restringir a comunicação entre pods utilizando NetworkPolicies. Por exemplo:**
     - **Permitir que apenas os microsserviços necessários (ou namespaces específicos) se comuniquem com o sre-challenge-app.**
     - **Bloquear acessos desnecessários de outros pods.**

Performance e Escalabilidade:

1. **HorizontalPodAutoscaler (HPA):**
   - **Ajustar as configurações do HPA para considerar métricas como utilização de memória, em vez de apenas CPU, usando o custom-metrics API se necessário.**
2. **Configurar Readiness e Liveness Probes:**
   - **Certifique-se de que os pods têm probes configuradas adequadamente para detectar falhas e reiniciar contêineres problemáticos automaticamente.**
3. **Configurar Limites de Recursos Precisos:**
   - **Basear limites de CPU/memória em análises reais de carga (benchmark) para evitar desperdício ou sobrecarga de recursos.**
4. **Usar Persistent Volumes (PV):**
   - **Para bancos de dados, configure Persistent Volumes com armazenamento provisionado dinamicamente (por exemplo, usando StorageClass apropriado para o ambiente).**

Logs e Monitoramento:

1. **Centralização de Logs:**
   - **Use ferramentas como ELK Stack, Fluentd, ou Loki para capturar e centralizar logs.**
2. **Monitoramento com Alertas:**
   - **Use Prometheus e Grafana para métricas e monitoramento.**
   - **Configure alertas para eventos como falhas no banco de dados ou utilização anormal de recursos.**

2. Isolamento de Microsserviços por Equipe

Separação de Namespaces:

- **Criar um Namespace por Equipe:**
  - **Por exemplo, team-a e team-b.**
  - **Configure os deployments, services e outros recursos de cada equipe no respectivo namespace.**

Gerenciamento de Acesso:

1. **RBAC (Role-Based Access Control):**
   - **Configure papéis (Roles) específicos para cada equipe usando Role e RoleBinding:**
   - **apiVersion: rbac.authorization.k8s.io/v1**
     kind: Role

```
metadata:
  namespace: team-a
  name: team-a-role
rules:
- apiGroups: [""]
  resources: ["pods", "services", "secrets"]
  verbs: ["get", "list", "create", "update", "delete"]
---
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: team-a-binding
  namespace: team-a
subjects:
- kind: User
  name: team-a-user
roleRef:
  kind: Role
  name: team-a-role
  apiGroup: rbac.authorization.k8s.io
```

2. **Restrição de Acesso ao Cluster:**
    - **Configure ClusterRole e ClusterRoleBinding apenas para administradores.**
    - **Usuários da equipe só podem acessar seu namespace.**

3. Restringindo Acesso ao sre-challenge-app

Configurar NetworkPolicies:

- **Use NetworkPolicy para restringir o tráfego de rede. Exemplo:**

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: restrict-access
  namespace: default
spec:
  podSelector:
    matchLabels:
      app: sre-challenge-app
  ingress:
  - from:
    - podSelector:
        matchLabels:
          app: web
    ports:
    - protocol: TCP
      port: 8080
  policyTypes:
  - Ingress
```

Isolamento com ServiceAccount:

- **Configure ServiceAccount para cada aplicação e restrinja os privilégios.**

Implementar Sidecar Proxy (Opcional):

- **Use um service mesh (como Istio ou Linkerd) para controlar e rastrear a comunicação entre serviços.**

Resumo

Com essas práticas:

1. **Cada equipe terá um namespace isolado e acesso restrito por RBAC.**

2. **O sre-challenge-app só será acessível por serviços autorizados via NetworkPolicies.**
3. **A infraestrutura será mais resiliente, segura e pronta para escalabilidade.**

Se precisar de ajuda em implementar algo específico, posso detalhar ainda mais!