Kubernetes

Para Quem Não Entende Nada

Aula 04

Probes, ServiceAccounts e RBAC

Probes

Os probres são uma forma mais específica de testar e identificar se um pod está em bom estado e pronto para receber requisições.

Probes

Existem alguns probes como Liveness e Readiness e alguns tipos de probes, os mais comuns são HTTP request e TCP.

Probes

Os probes funcionam em cascata, na seguinte ordem:

StartupProbe → LivenessProbe → ReadinessProbe

Liveness e Readiness

```
containers:
...
  livenessProbe:
    tcpSocket:
      port: 3306
    initialDelaySeconds: 30
    periodSeconds: 10
    timeoutSeconds: 1
    successThreshold: 1
    failureThreshold: 3
```

MySQL

```
containers:
...
  livenessProbe:
    httpGet:
      path: /
      port: 8080
    periodSeconds: 10
    failureThreshold: 6
  readinessProbe:
    httpGet:
      path: /
      port: 8080
    periodSeconds: 10
```

Lua

Analisando

Como já vimos anteriormente, para analisar os status de um pod podemos utilizar describe ou logs.
Um exemplo de mensagem:

Readiness probe failed: Get "http://10.0.0.1:81/": dial tcp 10.0.0.1:81: connect: connection refused

O kubeconfig

O arquivo ~/.kube/config ou kubeconfig é o arquivo que define a forma como um determinado ente se conecta ao Kubernetes.

O kubeconfig

O kubeconfig é dividido entre clusters, usuários e contextos.
Contexto é a junção entre um usuário e um cluster.

RBAC

RBAC – Role Base Access Control – é o mecanismo utilizado para autorização.
Utilizando o kubeconfig nos autenticamos com um certificado x509 mas a autorização está no “O” do certificado - system:masters

Roles e Rolebindings

As Roles são os conjuntos de permissões que um determinado ente pode possuir dentro do Kubernetes.

As RoleBindings são a conexão entre uma entidade e um conjunto de Roles.

Ambas possuem sua versão “cluster”.

ClusterRole do Kubeconfig

Sabemos que a autorização está na “Organization” do certificado – system:masters – e podemos encontrar a referência entre esta e a permissão procurando pelas ClusterRoleBindings.

SA

Uma SA – ServiceAccount – é um tipo de conta utilizado por entidades não humanas, como, aplicações.

Criamos SAs normalmente para automatizar alguma coisa dentro do Kubernetes.

SA e ClusterRoleBinding

Conectando uma sa com uma clusterrole:

```
$ kubectl create sa leitor

$ kubectl create clusterrolebinding \
--clusterrole=view \
--serviceaccount=default:leitor leitura
```

Deploy com SA específica

```
kubectl create deploy varredor \
--image=docker.io/hectorvido/kubectl:latest

serviceAccount: leitor
containers:
- name: kubectl
  tty: true
  stdin: true
  command:
    - sh
    - -c
    - while true; do kubectl get pods; sleep 5; done
```

Testar acesso da SA

Agora o pod consegue fazer consultas no cluster de Kubernetes, desde que sejam apenas leituras.

```
$ kubectl exec -ti varredor
```

Criando kubeconfig

```
$ kubectl create token leitor

$ kubectl --kubeconfig /tmp/kubeconfig config set-credentials varredor \
--token='<TOKEN>'

$ kubectl --kubeconfig /tmp/kubeconfig config set-cluster minikube \
--server=https://$(minikube ip):8443 –insecure-skip-tls-verify=true

$ kubectl --kubeconfig /tmp/kubeconfig config set-context default \
--user=varredor –cluster=minikube

$ kubectl --kubeconfig /tmp/kubeconfig config use-context default

$ kubectl --kubeconfig /tmp/kubeconfig get pods
$ kubectl --kubeconfig /tmp/kubeconfig delete pods --all
```