Kubernetes

Para Quem Não Entende Nada

Aula 05

Operators

Porque?

Precisamos descobrir que tipo de trabalho um Operator evita, e para isso vamos configurar um Prometheus e um Grafana.

A Aplicação

A aplicação em Lua possui um endpoint /metrics, que será utilizado pelo Prometheus para coletar informações.

Instalando o Prometheus

A instalação do Prometheus e do Grafana será manual, haverão muitos detalhes com os quais se preocupar e muita coisa ficará faltando.

Configurando o Prometheus

O problema maior não é aplicar os YAMLs, mas fazer alterações nas configurações do Prometheus como novos alertas ou outras formas de monitoramento.
Vamos utilizar um exemplo pronto.

Configurando o Prometheus

Para os alertas, teremos de adicionar uma linha no arquivo de configuração incluindo o arquivo de rules:

```
rule_files:
  - /etc/prometheus/prometheus-rules.yml
```

Configurando o Prometheus

Precisamos criar e montar mais um arquivo como ConfigMap, o arquivo de alertas, dentro do Prometheus.

```
containers:
  volumeMounts:
  - mountPath: /etc/prometheus/prometheus-rules.yml
    name: prometheus-rules
    subPath: prometheus-rules.yml
volumes:
- name: prometheus-rules
  configMap:
    name: prometheus-rules
```

Configurando o Prometheus

Como trabalhamos com um único configMap teremos muitos atritos entre a equipe e uma configuração errada pode comprometer toda a pilha de monitoramento.

Operator

Um operator estende o Kubernetes a fim de automatizar o gerenciamento de todo o ciclo de vida de uma aplicação em particular.

- *Operadores do Kubernetes*
Jason Dobies, Joshua Wood

Operator

O operator utilizado será o kube-prometheus. Este operator nos permite criar mais objetos do Kubernetes como Prometheus, ServiceMonitor e PrometheusRule.

CRD

Estes tipos novos recebem o nome de CRD – CustomResourceDefinition.

```
kubectl get prometheus -A
kubectl get servicemonitor -A
kubectl get prometheusrule -A
```

CRD

Agora podemos ignorar os detalhes do arquivo de configuração e definir diretamente objetos do Kubernetes com os quais já estamos habituados, o operator fará o restante para nós. Não haverá colisão entre os membros da equipe.

ServiceMonitor

ServiceMonitor corresponde aos alvos a serem monitorados. A porta precisa ter uma nome.

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  name: lua-app
  namespace: default
spec:
  selector:
    matchLabels:
      app: lua
  endpoints:
  - port: http
```

PrometheusRule

```
apiVersion: monitoring.coreos.com/v1
kind: PrometheusRule
metadata:
  name: openresty-failed-requests
  namespace: default
spec:
  groups:
  - name: OpenResty
    rules:
    - alert: OpenRestyFailedRequests
      expr: increase(nginx_http_requests_total{status="500"}[15m]) > 0
```

PrometheusRule corresponde as regras de alerta.

ServiceMonitor e PrometheusRule

Estamos utilizando o namespace default para nossas aplicações, caso utilizemos outro namespace precisaremos configurar a Role e a RoleBinding para a SA do prometheus naquele namespace.