Day06 kubernetes笔记

今日课程主题：

1、helm

2、dashboard

3、prometheus

4、grafana

问题1：jenkins 执行 shell 脚本，发现执行不成功？？

原因：jenkins使用的docker容器化方式安装的，无法发现宿主机的路径，因此无法启动宿主机的jar文件。 如果你需要执行此脚本，实现使用jenkins本身执行build,install,deploy，使用直接启动war的方式进行运行此脚本。

#磁盘空间不足，导致启动失败，必须加 ‘#!/bin/bash’ ，否则会启动不起来

#!/bin/bash

#export BUILD\_ID=dontKillMe这一句很重要，这样指定了，项目启动之后才不会被Jenkins杀掉。

export BUILD\_ID=dontKillMe

#指定最后编译好的jar存放的位置

www\_path=/var/codespace/test/

#Jenkins中编译好的jar位置

jar\_path=/root/.jenkins/workspace/test/supergo-eureka/target

#Jenkins中编译好的jar名称

jar\_name=supergo-eureka-1.0-SNAPSHOT.jar

#获取运行编译好的进程ID，便于我们在重新部署项目的时候先杀掉以前的进程

pid=$(cat /var/codespace/test/test-web.pid)

#进入指定的编译好的jar的位置

cd ${jar\_path}

#将编译好的jar复制到最后指定的位置

cp ${jar\_path}/${jar\_name} ${www\_path}

#进入最后指定存放jar的位置

cd ${www\_path}

#杀掉以前可能启动的项目进程

kill -9 ${pid}

#启动jar，指定SpringBoot的profiles为test,后台启动

java -jar -Dspring.profiles.active=test ${jar\_name} &

#将进程ID存入到shaw-web.pid文件中

echo $! > /var/codespace/test/test-web.pid

问题2：Ingress时候，同一个域名，根据不同的路径，把请求转发给不同的服务？？  
  
apiVersion: extensions/v1beta1

kind: Ingress

metadata:

name: nginx

namespace: default

labels:

app: nginx

annotations:

nginx.ingress.kubernetes.io/rewrite-target: / # URL重写

spec:

rules:

- host: ingress.kaikeba.com

http:

paths:

- path: /nginx

backend:

serviceName: nginx

servicePort: 80

- path: /tomcat

backend:

serviceName: tomcat

servicePort: 8080

# 1、Ingress https

这里用给tomcat服务添加证书为例。

* 生成私钥

openssl genrsa -out tls.key 2048

* 自签发证书

openssl req -new -x509 -key tls.key -out tls.crt -subj /C=CN/ST=Shanghai/L=Shanghai/O=DevOps/CN=ingress.kaikeba.com

* 创建K8S使用的证书配置文件Secret

kubectl create secret tls tomcat-ingress-secret --cert=tls.crt --key=tls.key

* 查看Secret描述

kubectl describe secret tomcat-ingress-secret

* 创建带tls认证的tomcat后端服务

# 创建好后，访问服务，结果如下图所示，由于是自签发证书不受信任。HTTPS已可以成功访问。

apiVersion: extensions/v1beta1

kind: Ingress

metadata:

name: ingress-tomcat-tls

namespace: default

annotations:

kubernetes.io/ingress.class: "nginx"

labels:

app: tomcat

spec:

tls:

- hosts:

- ingress.kaikeba.com

secretName: tomcat-ingress-secret

rules:

- host: ingress.kaikeba.com

http:

paths:

- backend:

serviceName: tomcat

servicePort: 8080

访问地址： <https://ingress.kaikeba.com:30443/>

注意： 30443 必须是ingress-service他的https转发端口。

# 2、helm

思考： Kubernetes

1、etcd

2、apiserver

3、pod

4、daemonset

……………………

K8s有非常多资源对象，部署一个服务时候，往往需要操作多个服务资源对象，且资源对象相互依赖，非常不好管理。Helm帮助管理资源对象。

## 2.1、Helm是什么？

Helm类似于yum安装指令，统一对安装服务进行管理，使得用户不需要关系服务之间的依赖关系。  
 k8s编程非常麻烦，资源对象非常多，因此helm就是一种k8s项目开发的方式，且k8s项目开发有固定项目结构。

类似：

前端：vue

后端：java

Java项目

|-src/main/resources

|-src/main/java

Helm项目开发结构：

Charts --- 依赖包 类似于项目lib,类似前端package-json

Chart.yaml

Templates

|-deployment.yaml

|-service.yaml

Values.yaml --- 相当于configmap资源对象，用来存储公共资源

安装redis集群：

Helm install . redis-chart

Helm安装的项目下载：

1、<https://helm.sh/>

2、阿里云

3、七牛

4、中科大

## 2.2、Helm 原理

Helm 的几个关键组件 Helm（客户端）、Tiller（服务器）、Repository（Chart 软件仓库）

Helm部署：

1、helm客户端

2、tiller服务端

使用helm部署，必须使用部署chart包，chart从Repository获取，因此必须制定一个Chart 软件仓库。

## 2.3、helm部署

1、github下载 helm-v2.15.2-linux-amd64.tar.gz

2、把helm放入usr/local/bin

3、安装tiller

1)从阿里云下载tiller镜像

docker pull registry.cn-hangzhou.aliyuncs.com/google\_containers/tiller:v2.15.2

2) 打包（helm init 默认下载的镜像地址：gcr.io/kubernetes-helm/tiller:v2.15.2）

Docker tag registry.cn-hangzhou.aliyuncs.com/google\_containers/tiller:v2.15.2 gcr.io/kubernetes-helm/tiller:v2.15.2

3） 备份上传到其他 node节点

docker save -o tiller.tar gcr.io/kubernetes-helm/tiller:v2.15.2

查询： helm version

## 2.4、Helm工程开发

工程结构：

Chart.yaml

Templates

|-deployment.yaml

|-service.yaml

Values.yaml --- 相当于configmap资源对象，用来存储公共资源

1）创建chart.yaml

名称

版本

2）创建templates

Deployment.yaml

Service.yaml

……………………

3) 创建values.yaml

相当于configmap,存储镜像地址，版本。

# 3、dashbord

参考文档资源即可