# k8s应用配置详解

由 温兆钦创建, 最后修改于不到1分钟以前

## 1. 概述

### k8s主要通过Object定义各种部署任务(例如：部署应用、部署Ingress路由规则、部署service等等)，通过kubectl命令远程操作k8s集群。

Object的定义通常以Yaml格式进行描述。

> Yaml是一种直观的数据序列化格式，主要通过缩进的方式组织数据。

Yaml例子：

```
#  注释
house:
  family:
    name: Doe
    parents: # - 开始，表示数组元素
      - John
      - Jane
    children:
      - Paul
      - Mark
      - Simone
  address:
    number: 34
    street: Main Street
    city: Nowheretown
    zipcode: 12345
```

定义好object文件后就可以通过kubectl命令将object发送给k8s服务器执行。

Object格式：

```
apiVersion: apps/v1 # k8s api版本
kind: Deployment  # Object类型
metadata:   # Object元数据定义，例如定义名字
  name: nginx-deployment
spec:  # Object 内容定义
  .....
status: # Object 运行状态，通常不需要我们定义，k8s负责更新
  ....
```

## 2. kubectl 命令详解

## 2.1. kubectl安装&配置

安装kubectl命令

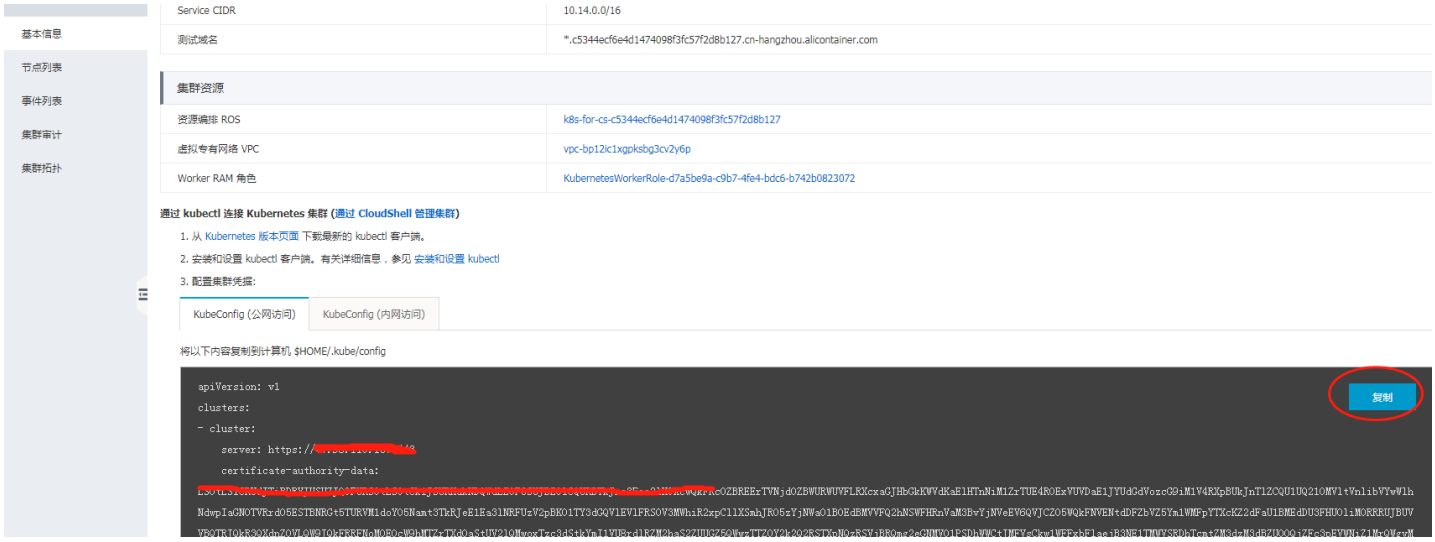
```
#LINUX
curl -LO https://storage.googleapis.com/kubernetes-release/release/v1.14.0/bin/linux/amd64/
chmod +x ./kubectl
mv ./kubectl /usr/bin/kubectl

#Windows，下载kubectl命令，然后配置下环境变量PATH，将kubectl的路径加入到PATH
https://storage.googleapis.com/kubernetes-release/release/v1.14.0/bin/windows/amd64/kubectl
```

配置k8s集群连接凭证

kubectl命令连接远程服务器的配置默认存放在$HOME/.kube/config文件中，也可以通过指定--kubeconfig参数执行配置文件，例子：kubectl --kubeconfig ./k8s.conf  get pods

k8s集群的凭证，以阿里云为例展示如何获取：

复制下图中的凭证保存到指定的位置即可




## 2.2. kubectl命令常用操作

```
#应用一个object配置，至于部署什么，就看yaml配置文件定义了什么，这里部署一个pod
kubectl apply -f ./pod.yaml

#kubectl get子命令用户展示k8s资源，k8s资源就是我们通过object部署的东西。
#展示所有pod
kubectl get pods

#展示kube-system名字空间下的所有pod， 默认kubectl命令只是操作default默认名字空间，可以通过-n参数指
kubectl get pods -n kube-system

#展示所有deployment
kubectl get deploy
```

```
#展示所有service（服务）
kubectl get svc

#展示所有ingress
kubectl get ingress

#以yaml格式，展示具体的ingress资源的配置， 展示任意资源具体的yaml配置命令格式：kubectl get 资源类
kubectl get ingress 资源名字 -o yaml

#删除指定的资源，具体删除内容由配置文件object定义
kubectl delete -f ./pod.yaml

#在容器中执行命令格式
kubectl exec 容器id  命令

#在admin-app-5f84f57f7-vthlf容器中，执行ls /alidata/www 命令
kubectl exec admin-app-5f84f57f7-vthlf ls /alidata/www

#在admin-app-5f84f57f7-vthlf容器中，打开shell交互窗口，需要加上-it参数
kubectl exec -it admin-app-5f84f57f7-vthlf bash
```

## 3. k8s常用对象（Object）类型

### 3.1. deployment

主要用于部署pod，支持滚动升级。

```
apiVersion: apps/v1
#对象类型
kind: Deployment
metadata:
  name: nginx-deployment  #deployment名字
  labels:
    app: nginx   #deployment标签，可以自由定义
spec:
  replicas: 3   #pod 副本数量
  selector:     #pod选择器定义，主要用于定义根据什么标签搜索需要管理的pod
    matchLabels:
      app: nginx  #pod标签
  template:  #pod模版定义
    metadata:
      labels:   #pod 标签定义
        app: nginx
    spec:
      containers: #容器数组定义
      - name: nginx  #容器名
        image: nginx:1.7.9  #镜像地址
        command: #容器启动命令，【可选】
            - /alidata/www/scripts/startWebServer.sh
```

```
        ports:  #定义容器需要暴露的端口
            - containerPort: 80
        env: #环境变量定义【可选】
            - name: ZHIPUZI_CONSOLE_URL #变量名
              value: https://cy.zhipuzi.com #变量值
```

### 3.2. service

服务定义，主要用于暴露pods容器中的服务。

```
apiVersion: v1
#对象类型
kind: Service
metadata:
  name: my-service #服务名
spec:
  selector: #pod选择器定义，由这里决定请求转发给那些pod处理
    app: nginx #pod 标签
  ports: #服务端口定义
  - protocol: TCP  #协议类型，主要就是TCP和UDP
    port: 80 # 服务端口
    targetPort: 80 #pod 容器暴露的端口
```

### 3.3. ingress

http路由规则定义，主要用于将service暴露到外网中

```
apiVersion: extensions/v1beta1
#对象类型
kind: Ingress
metadata:
  name: my-ingress #ingress应用名
spec:
  rules: #路由规则
    - host: cy.zhipuzi.com  #域名
      http:
        paths:  #访问路径定义
          - path: /   #代表所有请求路径
            backend:  #将请求转发至什么服务，什么端口
              serviceName: my-service #服务名
              servicePort: 80 #服务端口
```

### 3.4. ConfigMap

主要用于容器配置管理。

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: my-config #配置项名字
data:
    key1: value1
    key2: value2
```

定义完配置后，可以通过以下方式在容器中应用配置：

```
#通过环境变量注入配置
apiVersion: v1
kind: Pod
metadata:
   name: config-pod-1
spec:
   containers:
     - name: test-container
       image: busybox
       command: [ "/bin/sh", "-c", "env" ]
       env:
         - name: SPECIAL_LEVEL_KEY                ## 环境变量
           valueFrom:                             ##使用valueFrom来指定env引用配置项的value值
             configMapKeyRef:
               name: my-config           ##引用的配置文件名称
               key: key1                 ##引用的配置项key
   restartPolicy: Never
```

```
#通过数据卷注入配置
apiVersion: v1
kind: Pod
metadata:
   name: config-pod-4
spec:
   containers:
     - name: test-container
       image: busybox
       command: [ "/bin/sh", "-c", "ls /etc/config/" ]   ##列出该目录下的文件名
       volumeMounts:
       - name: config-volume    #配置项名字
         mountPath: /etc/config #容器中的挂载目录
   volumes:         #数据卷定义
     - name: config-volume  #数据卷名
       configMap:           #数据卷类型
         name: my-config    #配置项名字
   restartPolicy: Never
```

无标签