链滴

Kubernetes- 资源管理

作者：mrchi

原文链接：https://ld246.com/article/1668421526900

来源网站：链滴

资源管理方式

- **命令式对象管理：直接使用命令去操作kubernetes资源**

```
kubectl run nginx-pod --image=nginx:1.17.1 --port=80
```

- **命令式对象配置：通过命令配置和配置文件去操作kubernetes资源**

```
kubectl create/patch -f nginx-pod.yaml
```

- **声明式对象配置：通过apply命令和配置文件去操作kubernetes资源**

用于创建和更新资源

```
kubectl apply -f nginx-pod.yaml
```

类型	操作对象	适用环境	优点	点
命令式对象管理	对象	测试	简单	能操作活动对象，无法审计、跟踪
命令式对象配置	文件	开发	可以审计、	踪 项目大时，配置文件多，操作麻烦
声明式对象配置	目录	开发	支持目录操作	外情况下难以调试

2.1 命令式对象管理

kubectl命令

kubectl是kubernetes集群的命令行工具，通过它能够对集群本身进行管理，并能够在集群上进行容化应用的安装部署。kubectl命令的语法如下：

```
kubectl [command] [type] [name] [flags]
```

comand：指定要对资源执行的操作，例如 create、get、delete

type：指定资源类型，比如 deployment、pod、service

name：指定资源的名称，名称大小写敏感

flags：指定额外的可选参数

```
# 查看所有pod
kubectl get pod

# 查看某个pod
kubectl get pod pod_name

# 查看某个pod,以yaml格式展示结果
kubectl get pod pod_name -o yaml
```

资源类型

kubernetes中所有的内容都抽象为资源，可以通过下面的命令进行查看:

```
kubectl api-resources
```

经常使用的资源有下面这些:

资源分类	资源名称	缩写	资源作用
集群级别资源	nodes	no	集群组成部分
	namespaces	ns	隔离Pod
pod资源	pods	po	装载器
pod资源控制器	replicationcontrollers	rc	控制pod资源
	replicasets	rs	控制pod资源
	deployments	deploy	控制pod资源
	daemonsets	ds	控制pd资源
	jobs		控制pod资源
	cronjobs	cj	控制pod资源
	horizontalpodautoscalers	hpa	控制pod资源
	statefulsets	sts	控制pd资源
服务发现资源	services	svc	一pod对外接口
	ingress	ing	统一pod外接口
存储资源	volumeattachments		储
	persistentvolumes	pv	储
	persistentvolumeclaims	pvc	储
配置资源	configmaps	cm	置
	secrets		配置

操作

kubernetes允许对资源进行多种操作，可以通过--help查看详细的操作命令

kubectl --help

经常使用的操作有下面这些：

命令分类	命令	翻译	命令作用
基本命令	create	创建	创建一个资源
	edit	编辑	编辑一个资源
	get	获取	获取一个资源
	patch	更新	更新一个资源
	delete	删除	删除一个资源
	explain	解释	展示资源文档
运行和调试	run	运行	在集群中运行一个指定的镜像
	expose	暴露	暴露资源为Service
	describe	描述	显示资源内部信息
	logs	日志输出容器在 pod 中的日志	打印Pod中容器在 pod 中的日志
	attach	缠绕进入运行中的容器	进入运行中的容器
	exec	执行容器中的一个命令	执行Pod中容器中的一个命令
	cp	复制	在Pod内外复制文件
	rollout	首次展示	管理资源的发布
	scale	规模	扩(缩)容Pod的数量
	autoscale	自动调整	自动调整Pod的数量
高级命令	apply	rc	通过文件对资源进行配置
	label	标签	更新资源上标签
其他命令	cluster-info	集群信息	显示集群信息
	version	版本	显示当前Server和Client的版本

下面以一个namespace / pod的创建和删除简单演示下命令的使用：

```
# 创建一个namespace
[root@k8s-master ~]# kubectl create namespace dev
namespace/dev created

# 获取namespace
[root@k8s-master ~]# kubectl get ns
NAME              STATUS   AGE
default           Active   21h
dev               Active   21s
kube-node-lease   Active   21h
kube-public       Active   21h
kube-system       Active   21h

# 在此namespace下创建并运行一个nginx的Pod
[root@k8s-master ~]# kubectl run pod --image=nginx:latest -n dev
kubectl run --generator=deployment/apps.v1 is DEPRECATED and will be removed in a futur
version. Use kubectl run --generator=run-pod/v1 or kubectl create instead.
deployment.apps/pod created

# 查看新创建的pod
[root@k8s-master ~]# kubectl get pod -n dev
NAME  READY   STATUS    RESTARTS   AGE
pod   1/1     Running   0          21s

# 删除指定的pod,删除时，再次查看发现还有pod 信息,不要疑惑，这个pod 已经删除
# 因为是控制器管理的,仔细查看会发现 pod 后面的内容是不同的
[root@k8s-master ~]# kubectl delete pod pod-864f9875b9-pcw7x
pod "pod" deleted

# 删除指定的namespace
[root@k8s-master ~]# kubectl delete ns dev
namespace "dev" deleted
```

2.2 命令式对象配置

命令式对象配置就是使用命令配合配置文件一起来操作kubernetes资源。

1） 创建一个 nginxpod.yaml，内容如下：

```
apiVersion: v1
kind: Namespace
metadata:
  name: dev

---

apiVersion: v1
kind: Pod
metadata:
  name: nginxpod
  namespace: dev
spec:
```

```
  containers:
  - name: nginx-containers
    image: nginx:latest
```

2）执行create命令，创建资源：

```
[root@k8s-master ~]# kubectl create -f nginxpod.yaml
namespace/dev created
pod/nginxpod created
```

此时发现创建了两个资源对象，分别是namespace和pod

3）执行get命令，查看资源：

```
[root@k8s-master ~]#  kubectl get -f nginxpod.yaml
NAME          STATUS   AGE
namespace/dev   Active   18s

NAME          READY   STATUS    RESTARTS   AGE
pod/nginxpod    1/1     Running   0          17s
```

这样就显示了两个资源对象的信息

4）执行delete命令，删除资源：

```
[root@k8s-master ~]# kubectl delete -f nginxpod.yaml
namespace "dev" deleted
pod "nginxpod" deleted
```

此时发现两个资源对象被删除了

总结:
    命令式对象配置的方式操作资源，可以简单的认为：命令 + yaml配置文件（里面是命令需要的种参数）

2.3 声明式对象配置

声明式对象配置跟命令式对象配置很相似，但是它只有一个命令apply。

```
# 首先执行一次kubectl apply -f yaml文件，发现创建了资源
[root@k8s-master ~]#  kubectl apply -f nginxpod.yaml
namespace/dev created
pod/nginxpod created

# 再次执行一次kubectl apply -f yaml文件，发现说资源没有变动
[root@k8s-master ~]#  kubectl apply -f nginxpod.yaml
namespace/dev unchanged
pod/nginxpod unchanged
```

总结:
    其实声明式对象配置就是使用apply描述一个资源最终的状态（在yaml中定义状态）
    使用apply操作资源：
        如果资源不存在，就创建，相当于 kubectl create
        如果资源已存在，就更新，相当于 kubectl patch

扩展：kubectl 可以在node节点上运行吗？

kubectl 的运行是需要进行配置的，它的配置文件是(master) $HOME/.kube，如果想要在 node 点运行此命令，需要将master上的.kube文件复制到node节点上，即在master节点上执行下面操作：

```
scp  -r  HOME/.kube   node1: HOME/
```

使用推荐: 三种方式应该怎么用？

创建/更新资源 使用声明式对象配置 kubectl apply -f XXX.yaml

删除资源 使用命令式对象配置 kubectl delete -f XXX.yaml

查询资源 使用命令式对象管理 kubectl get(describe) 资源名称