**Kubernetes常用命令**

2019-08-30

目 录

[1 kubectl命令介绍 2](#_Toc18177919)

[1.1 基本命令 2](#_Toc18177920)

[1.2 资源对象 3](#_Toc18177921)

[1.3 输出选项 5](#_Toc18177922)

[2 kubectl常用命令 7](#_Toc18177923)

[2.1 集群相关 7](#_Toc18177924)

[2.2 Pod相关 7](#_Toc18177925)

[2.2.1 pod 7](#_Toc18177926)

[2.2.2 deployment 8](#_Toc18177927)

[2.2.3 service 9](#_Toc18177928)

[2.2.4 其它 9](#_Toc18177929)

[2.3 Docker相关 10](#_Toc18177930)

[3 容器命令 10](#_Toc18177931)

[4 镜像命令 10](#_Toc18177932)

[5 其它参考 12](#_Toc18177933)



# kubectl命令介绍

## 基本命令

Kubectl是kubernetes的命令行工具。职责是对集群中资源对象进行操作，这些操作包括对资源对象的增、删、改、查以及高级操作（滚动升级）等。下表中显示了kubectl支持的所有操作命令，以及这些命令的语法和描述信息。

Kubernetes中所有组件都是资源，以下命令适用于所有组件。

| **操作** | **语法** | **描述** |
| --- | --- | --- |
| annotate | kubectl annotate (-f FILENAME \| TYPE NAME \| TYPE/NAME) KEY\_1=VAL\_1 … KEY\_N=VAL\_N [–overwrite] [–all] [–resource-version=version] [flags] | 添加或更新一个或多个资源的注释 |
| api-versions | kubectl api-versions [flags] | 列出可用的API版本 |
| apply | kubectl apply -f FILENAME [flags] | 将来自于文件或stdin的配置变更应用到主要对象中。 |
| attach | kubectl attach POD -c CONTAINER [-i] [-t] [flags] | 连接到正在运行的容器上，以查看输出流或与容器交互（stdin）。 |
| autoscale | kubectl autoscale (-f FILENAME \| TYPE NAME \| TYPE/NAME) [–min=MINPODS] –max=MAXPODS [–cpu-percent=CPU] [flags] | 自动扩宿容由副本控制器管理的Pod。 |
| cluster-info | kubectl cluster-info [flags] | 显示群集中的主节点和服务的的端点信息。 |
| config | kubectl config SUBCOMMAND [flags] | 修改kubeconfig文件。 |
| create | kubectl create -f FILENAME [flags] | 从文件或stdin中创建一个或多个资源对象。 |
| delete | kubectl delete (-f FILENAME \| TYPE [NAME \| /NAME \| -l label \| –all]) [flags] | 删除资源对象。 |
| describe | kubectl describe (-f FILENAME \| TYPE [NAME\_PREFIX \| /NAME \| -l label]) [flags] | 显示一个或者多个资源对象的详细信息。 |
| edit | kubectl edit (-f FILENAME \| TYPE NAME \| TYPE/NAME) [flags] | 通过默认编辑器编辑和更新服务器上的一个或多个资源对象。 |
| exec | kubectl exec POD [-c CONTAINER] [-i] [-t] [flags] [– COMMAND [args…]] | 在Pod的容器中执行一个命令。 |
| explain | kubectl explain [–include-extended-apis=true] [–recursive=false] [flags] | 获取Pod、Node和服务等资源对象的文档。 |
| expose | kubectl expose (-f FILENAME \| TYPE NAME \| TYPE/NAME) [–port=port] [–protocol=TCP\|UDP] [–target-port=number-or-name] [–name=name] [—-external-ip=external-ip-of-service] [–type=type] [flags] | 为副本控制器、服务或Pod等暴露一个新的服务。 |
| get | kubectl get (-f FILENAME \| TYPE [NAME \| /NAME \| -l label]) [–watch] [–sort-by=FIELD] [[-o \| –output]=OUTPUT\_FORMAT] [flags] | 列出一个或多个资源。 |
| label | kubectl label (-f FILENAME \| TYPE NAME \| TYPE/NAME) KEY\_1=VAL\_1 … KEY\_N=VAL\_N [–overwrite] [–all] [–resource-version=version] [flags] | 添加或更新一个或者多个资源对象的标签。 |
| logs | kubectl logs POD [-c CONTAINER] [–follow] [flags] | 显示Pod中一个容器的日志。 |
| patch | kubectl patch (-f FILENAME \| TYPE NAME \| TYPE/NAME) –patch PATCH [flags] | 使用策略合并补丁过程更新资源对象中的一个或多个字段。 |
| port-forward | kubectl port-forward POD [LOCAL\_PORT:]REMOTE\_PORT […[LOCAL\_PORT\_N:]REMOTE\_PORT\_N] [flags] | 将一个或多个本地端口转发到Pod。 |
| proxy | kubectl proxy [–port=PORT] [–www=static-dir] [–www-prefix=prefix] [–api-prefix=prefix] [flags] | 为kubernetes API服务器运行一个代理。 |
| replace | kubectl replace -f FILENAME | 从文件或stdin中替换资源对象。 |
| rolling-update | kubectl rolling-update OLD\_CONTROLLER\_NAME ([NEW\_CONTROLLER\_NAME] –image=NEW\_CONTAINER\_IMAGE \| -f NEW\_CONTROLLER\_SPEC) [flags] | 通过逐步替换指定的副本控制器和Pod来执行滚动更新。 |
| run | kubectl run NAME –image=image [–env=”key=value”] [–port=port] [–replicas=replicas] [–dry-run=bool] [–overrides=inline-json] [flags] | 在集群上运行一个指定的镜像。 |
| scale | kubectl scale (-f FILENAME \| TYPE NAME \| TYPE/NAME) –replicas=COUNT [–resource-version=version] [–current-replicas=count] [flags] | 扩宿容副本集的数量。 |
| version | kubectl version [–client] [flags] | 显示运行在客户端和服务器端的Kubernetes版本。 |

## 资源对象

kubernetes提供了很多资源对象，开发和运维人员可以通过这些对象对容器进行编排操作。在下表中，是kubectl所支持的资源对象类型，以及它们的缩略别名。

|  |
| --- |
|  |
| **资源对象类型** | **缩略别名** |
| apiservices |  |
| certificatesigningrequests | csr |
| clusters |  |
| clusterrolebindings |  |
| clusterroles |  |
| componentstatuses | cs |
| configmaps | cm |
| controllerrevisions |  |
| cronjobs |  |
| customresourcedefinition | crd |
| daemonsets | ds |
| deployments | deploy |
| endpoints | ep |
| events | ev |
| horizontalpodautoscalers | hpa |
| ingresses | ing |
| jobs |  |
| limitranges | limits |
| namespaces | ns |
| networkpolicies | netpol |
| nodes | no |
| persistentvolumeclaims | pvc |
| persistentvolumes | pv |
| poddisruptionbudget | pdb |
| podpreset |  |
| pods | po |
| podsecuritypolicies | psp |
| podtemplates |  |
| replicasets | rs |
| replicationcontrollers | rc |
| resourcequotas | quota |
| rolebindings |  |
| roles |  |
| secrets |  |
| serviceaccounts | sa |
| services | svc |
| statefulsets |  |
| storageclasses |  |

## 输出选项

kubectl默认执行完命令后的输出格式为纯文本格式，可以通过-o或者–output字段指定命令的输出格式。

kubectl [COMMAND] [TYPE] [NAME] -o=<output\_format>

| **输出格式** | **描述** |
| --- | --- |
| -o=custom-columns=<spec> | 使用以逗号分隔的自定义列打印表格。 |
| -o=custom-columns-file=<filename> | 使用文件中自定义列打印表格。 |
| -o=json | 输出JSON格式的API对象 |
| -o=jsonpath=<template> | 打印在jsonpath表达式中定义的字段 |
| -o=jsonpath-file=<filename> | 打印文件中以jsonpath表达式定义的字段 |
| -o=name | 仅仅输出资源对象的名称。 |
| -o=wide | 输出带有附加信息的纯文本格式。对于Pod对象，将会包含Node名称。 |
| -o=yaml | 输出YAML格式的API对象 |

# kubectl常用命令

## 2.1 集群相关

* **查看集群组件状态**

kubectl get componentstatuses

* **查看kubelet日志**

journalctl -xeu kubelet

journalctl -f -u kubelet

* **获取集群信息**

kubectl cluster-info

kubectl get cs

* **新建/删除namespace**

kubectl create namespace {namespace-name}

kubectl delete namespace {namespace-name}

## Pod相关

### pod

* **查看pod列表**

#指定命名空间的pod列表

kubectl get pods -n {namespace-name} -o wide

#所有命名空间的pod列表

kubectl get pods --all-namespaces -o wide

* **查看pod日志**

kubectl logs {pod-name} -n {namespace-name}

* **查看pod详情**

kubectl describe pod {pod-name} -n {namespace-name}

* **将容器镜像运行在pod中**

kubectl run {pod-name} --image={repository:tag} --replicas=1 --port=900

* **在pod中运行命令**

kubectl exec {pod-name} {cmd-name}

* **进入pod**

kubectl exec {pod-name} -it bash

* **查看deploy列表**

kubectl get deployments -n {namespace-name} -o wide

* **删除单个pod**

kubectl delete pods {pod-name}

### deployment

* **使用命令直接创建创建deployment**

kubectl run {deployment-name} --image={repository:tag} --replicas={replicas-num}

* **使用yaml创建deployment**

kubectl create -f {yaml-name}.yaml

* **删除deployment**

kubectl delete deployment {deployment-name}

* **查看所有deployment**

kubectl get deployment

* **查看deployment详细信息**

kubectl describe deployment {deployment-name}

* **查看replicaset**

kubectl get rs

* **查看replicaset详细信息**

kubectl describe rs {replicaset-name}

* **deployment扩容/缩容**

kubectl scale deployment nginx --replicas={replicas-num}

* **deployment升级/回滚**

kubectl set image deploy {deployment-name} {deployment-name}={repository:tag}

* **查看升级状态**

kubectl rollout status deployment {deployment-name}

* **查看升级历史**

kubectl rollout history deployment {deployment-name}

* **查看历史版本详情**

kubectl rollout history deployment {deployment-name} --revision=2

### 2.2.3 service

* **使用yaml创建service**

kubectl create -f {yaml-name}.yaml

* **使用yarm删除service**

kubectl delete -f {yaml-name}.yaml

* **直接删除某个service**

kubectl delete svc nginx

### 2.2.4 其它

* **编辑**

kubectl edit {resource-type}/{resource-name}

或者

kubectl edit –f {yaml-name}

* **查看endpoint**

kubectl get endpoints

* **在线查看Yaml语法**

kubectl explain pod

kubectl explain pod.spec | grep -i "containers" -5

* **检验Yarm语法**

kubectl create -f flask.yaml --validate

* **使用Yaml文件删除pod**

kubectl delete -f flask.yaml

## Docker相关

### 容器命令

* **查看容器进程**

docker ps|grep {container-name}

* **查看容器日志**

docker logs --tail=500 {container-id}

* **进入容器内部**

docker exec -it {container} /bin/bash

### 镜像命令

* **镜像列表**

docker images

* **导出镜像**

docker save -o {output-path} {repository:tag}

【说明】

output-path：导出到的本地目录

* **导入镜像**

docker load --input {image-tar}

* **修改cgroupdriver**

vi /etc/docker/daemon.json

#添加如下内容

{

"exec-opts": ["native.cgroupdriver=systemd"]

}

# 其它参考

参考网址：https://blog.csdn.net/andriy\_dangli/article/details/79269356









