容器服务

Kubectl 操作集群

产品文档

腾讯云

版权所有：腾讯云计算（北京）有限责任公司

【版权声明】

©2013-2018 腾讯云版权所有

本文档著作权归腾讯云单独所有，未经腾讯云事先书面许可，任何主体不得以任何形式复制、修改、抄袭、传播全部或部分本文档内容。

【商标声明】

腾讯云

及其它腾讯云服务相关的商标均为腾讯云计算（北京）有限责任公司及其关联公司所有。本文档涉及的第三方主体的商标，依法由权利人所有。

【服务声明】

本文档意在向客户介绍腾讯云全部或部分产品、服务的当时的整体概况，部分产品、服务的内容可能有所调整。您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或模式的承诺或保证。

文档目录

Kubectl 操作集群
- Kubectl 命令行介绍
- 通过 Kubectl 连接集群
- 创建Service
- 创建 Ingress
- 创建 PV&PVC
- 开源组件

版权所有：腾讯云计算（北京）有限责任公司

Kubectl 操作集群
Kubectl 命令行介绍

最近更新时间：2018-07-10 11:10:27

kubectl 命令行介绍

kubectl 是一个用于对 Kubernetes 集群操作命令行工具。 本文涵盖 kubectl 语法，常见命令操作，并提供常见示例。 有关每个命令（包括所有主命令和子命令）的详细信息，请参阅 kubectl 参考文档 或使用 kubectl help 命令查看详细帮助，有关安装说明，请参阅 安装 kubectl。

通过 kubectl 创建 Nginx

以下示例帮助您熟悉运行常用 kubectl 操作：

kubectl create

由于 Nginx 名称容易冲突，从文件或标准输入创建资源。

```
$ kubectl create -f nginx.yaml # nginx-test.yaml文件见下
```

nginx-test.yaml 文件如下：

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
name: nginx-test
labels:
qcloud-app: nginx-test
spec:
replicas: 1
revisionHistoryLimit: 5
selector:
matchLabels:
qcloud-app: nginx-test
strategy: {}
template:
metadata:
creationTimestamp: null
labels:
```

版权所有：腾讯云计算（北京）有限责任公司

```
qcloud-app: nginx-test
spec:
containers:
- image: nginx:latest
imagePullPolicy: Always
name: nginx-test
resources:
limits:
cpu: 200m
memory: 128Mi
requests:
cpu: 200m
memory: 128Mi
securityContext:
privileged: false
serviceAccountName: ""
volumes: null
imagePullSecrets:
- name: qcloudregistrykey
status: {}
---
apiVersion: v1
kind: Service
metadata:
name: nginx-test
labels:
qcloud-app: nginx-test
spec:
ports:
- name: tcp-80-80-ogxxh
nodePort: 0
port: 80
protocol: TCP
targetPort: 80
selector:
qcloud-app: nginx-test
type: LoadBalancer
status:
loadBalancer: {}
```

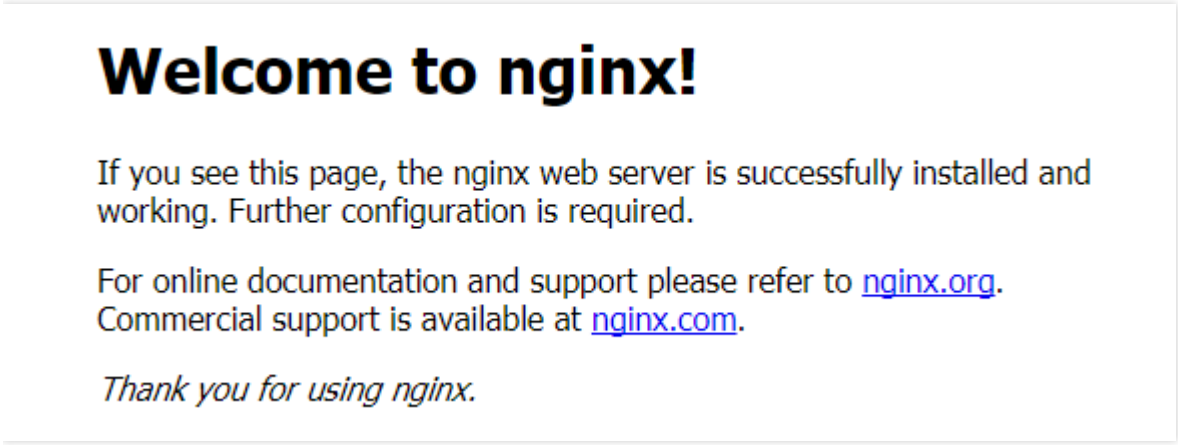
```
root@kubectld-1180979449-j2qtb:/# kubectl create -f nginx-test.yaml
deployment "nginx-test" created
service "nginx-test" created
root@kubectld-1180979449-j2qtb:/#
```

通过以下命令获取外网访问地址：

```
kubectl get services
```

```
root@kubectld-1180979449-b8kkr:/# kubectl get service
NAME         CLUSTER-IP      EXTERNAL-IP       PORT(S)        AGE
nginx        172.16.255.15   111.230.          80:32218/TCP   3m
root@kubectld-1180979449-b8kkr:/#
```

在浏览器上输入该 IP， 便可直接访问 nginx 服务。



kubectl logs

在容器中打印容器的日志。

```
// Return a snapshot of the logs from pod <pod-name>.
$ kubectl logs <pod-name>
```

```
root@kubectld-1180979449-j2qtb:/# kubectl get pod
NAME                          READY     STATUS    RESTARTS   AGE
kubectld-1180979449-j2qtb     1/1       Running   0          1h
nginx-test-4016666277-q44kg   1/1       Running   0          1m
root@kubectld-1180979449-j2qtb:/# kubectl logs nginx-test-4016666277-q44kg
10.0.0.104 - - [05/Feb/2018:09:54:36 +0000] "GET / HTTP/1.1" 200 612 "-" "Mozilla/5
10.0.0.104 - - [05/Feb/2018:09:54:36 +0000] "GET /favicon.ico HTTP/1.1" 404 571 "ht
537.36" "-"
2018/02/05 09:54:36 [error] 6#6: *1 open() "/usr/share/nginx/html/favicon.ico" fail
146.34", referrer: "http://203.195.146.34/"
10.0.0.112 - - [05/Feb/2018:09:54:43 +0000] "GET / HTTP/1.1" 200 612 "-" "Mozilla/5
10.0.0.105 - - [05/Feb/2018:09:54:43 +0000] "GET / HTTP/1.1" 200 612 "-" "Mozilla/5
10.0.0.104 - - [05/Feb/2018:09:54:43 +0000] "GET / HTTP/1.1" 200 612 "-" "Mozilla/5
root@kubectld-1180979449-j2qtb:/#
```

版权所有：腾讯云计算（北京）有限责任公司

通过 Kubectl 连接集群

最近更新时间：2018-07-25 17:18:19

您可以通过 Kubernetes 命令行工具 kubectl 从本地客户端机器连接到 TKE 集群。

操作步骤

一. 安装 kubectl 工具

如果您已经安装 kubectl 工具，请忽略本步骤。详细安装过程参考 Installing and Setting up kubectl。

下载 kubectl 工具

- Mac OS X 系统

```
curl -LO https://storage.googleapis.com/kubernetes-release/release/v1.8.13/bin/darwin/amd64/kubectl
```

- Linux 系统

```
curl -LO https://storage.googleapis.com/kubernetes-release/release/v1.8.13/bin/linux/amd64/kubectl
```

- Windows 系统

```
curl -LO https://storage.googleapis.com/kubernetes-release/release/v1.8.13/bin/windows/amd64/kubectl.exe
```

> 注：v1.8.13可根据需求替换成业务所需的kubectl版本。

添加执行权限

使用以下命令添加执行权限：

```
chmod +x ./kubectl
sudo mv ./kubectl /usr/local/bin/kubectl
```

测试安装结果

测试安装结果，输入以下命令，输出版本信息即安装成功。

```
kubectl version
Client Version: version.Info{Major:"1", Minor:"5", GitVersion:"v1.5.2", GitCommit:"08e099554f3c31f6e6f07b448ab3ed78d0520507", GitTreeState:"clean", BuildDate:"2017-01-12T04:57:25Z", GoVersion:"go1.7.4", Compiler:"gc", Platform:"linux/amd64"}
```

二. 获取集群账号密码以及证书信息

1. 登录 容器服务控制台 > 集群，单击需要连接的**集群 ID/名称**，查看集群详情。

2. 在集群信息页，单击【显示凭证】，查看用户名、密码和证书信息。

3. 复制或下载证书文件到本地。

4. 获取访问入口。

   - 获取公网访问入口：开启公网访问地址后可参考 步骤三 直接使用。
   - 获取 VPC 内网访问入口：目前暂不支持内网 **DNS** 解析，因此需要指定客户端主机的 **hosts**，来支持域名解析。即：在 /etc/hosts 文件后追加内网返回的 **IP** 和域名。用户可以参考下面的代码，也可以手动设置。

     ```
     sudo sed -i '$a **IP地址** **域名**' /etc/hosts
     ```

     配置完成后，可参考 步骤三 使用内网访问地址域名来访问。
     注：若集群无可用节点（包括节点异常,已封锁等状态），内网访问将在集群内有可用节点时生效。
   - 集群内直接访问：无须任何配置，可直接在集群内的主机上执行 kubectl 命令。

三. 通过证书信息使用 kubectl 操作集群

单次 kubectl 操作请求，附带证书信息

该方法适用于单次操作集群，不将容器集群的证书信息保存到机器上。
请求方法，kubectl 命令格式：

```
-s "域名信息" --username=用户名 --password=密码 --certificate-authority=证书路径
```

示例：

```
kubectl get node -s "https://cls-66668888.ccs.tencent-cloud.com" --username=admin --password=6666o9oIB2gHD88882quIfLMy6666 --certificate-authority=/etc/kubernetes/cluster-ca.crt
```

修改 kubectl 配置文件，长期有效

该方法适用于长期通过 kubectl 操作集群， 一次配置，只要文件不修改就长期有效。
设置 kubectl 配置，修改以下命令中的密码、证书信息。

```
kubectl config set-credentials default-admin --username=admin --password=6666o9oIB2gHD88882quIfLMy6666
kubectl config set-cluster default-cluster --server=https://cls-66668888.ccs.tencent-cloud.com --certificate-authority=/etc/kubernetes/cluster-ca.crt
kubectl config set-context default-system --cluster=default-cluster --user=default-admin
kubectl config use-context default-system
```

配置完成，直接使用 kubectl 命令：

```
kubectl get nodes
NAME STATUS AGE
10.0.0.61 Ready 10h
```

设置 kubectl 命令自动补全

您可以通过配置 kubectl 自动补全，提高可使用性。

```
source <(kubectl completion bash)
```

创建Service

最近更新时间：2018-10-18 11:21:10

创建公网服务

使用以下命令创建 mysvc.yaml 文件：

```
$ kubectl create -f mysvc.yaml
```

mysvc.yaml 文件内容如下：

```
kind: Service
apiVersion: v1
metadata:
name: my-service
spec:
selector:
app: MyApp
ports:
- protocol: TCP
port: 80
targetPort: 9376
type: LoadBalancer
```

创建内网服务

创建通过 VPC 内网访问的服务，只需在 metadata 下的 annotations 中增加 **service.kubernetes.io/qcloud-loadbalancer-internal-subnetid** 即可，即创建的内网 LB 所在子网的唯一 ID，此子网必须在集群 VPC 下。
命令如下：

```
metadata:
annotations:
service.kubernetes.io/qcloud-loadbalancer-internal-subnetid: subnet-xxxxxxxx
```

版权所有：腾讯云计算（北京）有限责任公司

指定 Loadbalance 只绑定指定节点

当您的集群规模较大时， 入口类型的应用设置亲和性调度到部分节点，你可以配置 Servce 的 Loadbalance 只绑定指定节点，annotations 如下：

```
metadata:
annotations:
service.kubernetes.io/qcloud-loadbalancer-backends-label: key in (value1, value2) ## LabelSelector 格式
```

1. 建议配合工作负载的亲和性调度使用。
2. 前提条件是 Node 已根据业务需求设置 Label。
3. 采用原生LabelSelector格式如：
   - service.kubernetes.io/qcloud-loadbalancer-backends-label: key1=values1, key2=values2
   - service.kubernetes.io/qcloud-loadbalancer-backends-label: key1 in (value1),key2 in (value2)
   - service.kubernetes.io/qcloud-loadbalancer-backends-label: key in (value1, value2)
   - service.kubernetes.io/qcloud-loadbalancer-backends-label: key1, key2 notin (value2)
4. 增量的节点若匹配，将自动绑定到该 Loadbalance。

带宽上移用户

带宽上移用户在创建公网访问方式的服务时需要指定如下两个 annotations 项：

- **service.kubernetes.io/qcloud-loadbalancer-internet-charge-type**
  公网带宽计费方式， 可选值有：TRAFFIC_POSTPAID_BY_HOUR（按使用流量计费），BANDWIDTH_POSTPAID_BY_HOUR（按带宽计费）。
- **service.kubernetes.io/qcloud-loadbalancer-internet-max-bandwidth-out**
  带宽上限，范围：[1,2000] Mbps。

创建命令：

```
metadata:
annotations:
service.kubernetes.io/qcloud-loadbalancer-internet-charge-type: TRAFFIC_POSTPAID_BY_HOUR
service.kubernetes.io/qcloud-loadbalancer-internet-max-bandwidth-out: "10"
```

版权所有：腾讯云计算（北京）有限责任公司

创建 Ingress

最近更新时间：2018-07-06 16:42:41

创建公网 Ingress

使用以下命令创建 myingress.yaml 文件：

```
$ kubectl create -f myingress.yaml
```

myingress.yaml 文件内容如下：

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
annotations:
kubernetes.io/ingress.class: qcloud
name: my-ingress
namespace: default
spec:
rules:
- host: localhost
http:
paths:
- backend:
serviceName: non-service
servicePort: 65535
path: /
```

创建内网 Ingress

内网 Ingress 只需在 metadata 下的 annotations 中增加 **kubernetes.io/ingress.subnetId** 即可，即创建的内网 LB 所在子网的唯一 ID，此子网必须在集群 VPC下。

```
metadata:
annotations:
kubernetes.io/ingress.subnetId: subnet-xxxxxxxx
```

版权所有：腾讯云计算（北京）有限责任公司

带宽上移用户

带宽上移用户在创建公网 Ingress 时需要指定如下两个 annotations 项：

- **kubernetes.io/ingress.internetChargeType**
  公网带宽计费方式， 可选值有：TRAFFIC_POSTPAID_BY_HOUR（按使用流量计费），BANDWIDTH_POSTPAID_BY_HOUR（按带宽计费）。
- **kubernetes.io/ingress.internetMaxBandwidthOut**
  带宽上限，范围：[1,2000] Mbps。

创建命令如下：

```
metadata:
annotations:
kubernetes.io/ingress.internetChargeType: TRAFFIC_POSTPAID_BY_HOUR
kubernetes.io/ingress.internetMaxBandwidthOut: "10"
```

创建 PV&PVC

最近更新时间：2018-09-17 17:29:12

当前仅支持 CBS 类型的 PV&PVC， 敬请期待 CFS 和 COS 存储类型支持。

静态创建 CBS 类型 PV&PVC

1. 创建 PV（可选）

通过已有 CBS 创建 PV。若未通过本步骤创建 PV，直接执行步骤 2 时，创建 PVC 将自动创建对应的 PV。

```
apiVersion: v1
kind: PersistentVolume
metadata:
name: nginx-pv
spec:
capacity:
storage: 10Gi
accessModes:
- ReadWriteOnce
qcloudCbs:
cbsDiskId: disk-xxxxxxx ## 指定已有的CBS id
fsType: ext4
storageClassName: cbs
```

2. 创建 PVC

创建命令如下：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
name: nginx-pv-claim
spec:
storageClassName: cbs
accessModes:
- ReadWriteOnce
resources:
requests:
storage: 10Gi
```

- 普通云盘大小必须是 10 的倍数，最小为 10，最大为 4000。

版权所有：腾讯云计算（北京）有限责任公司

- 高效云盘最小为 50 GB。
- SSD 云硬盘最小为 200 GB，具体策略见 云硬盘文档。

3. 使用 PVC

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
name: nginx-deployment
spec:
replicas: 1
selector:
matchLabels:
qcloud-app: nginx-deployment
template:
metadata:
labels:
qcloud-app: nginx-deployment
spec:
containers:
- image: nginx
imagePullPolicy: Always
name: nginx
volumeMounts:
- mountPath: "/opt/"
name: pvc-test
volumes:
- name: pvc-test
persistentVolumeClaim:
claimName: nginx-pv-claim # 已经创建好的pvc
```

动态创建 CBS 类型 PV&PVC

1. 创建 StorageClass

如果不创建 StorageClass， 集群内将默认存在 name 为 cbs 的 StorageClass。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
# annotations:
# storageclass.beta.kubernetes.io/is-default-class: "true"
```

版权所有：腾讯云计算（北京）有限责任公司

```
# 如果有这一条，则会成为default-class，创建pvc时不指定类型则自动使用此类型
name: cloud-premium
provisioner: cloud.tencent.com/qcloud-cbs ##tke集群自带的provisioner
parameters:
type: cloudPremium
# 支持 cloudBasic,cloudPremium,cloudSSD 如果不识别则当做cloudBasic
# zone:ap-shanghai-1
# zone 支持指定zone，如果指定，则讲云盘创建到此zone，如果不指定，则拉取所有node的zone信息，随机挑选一个
# paymode: PREPAID
# paymode为云盘的计费模式，PREPAID模式（包年包月：仅支持Retain保留的回收策略），默认是POSTPAID（按量计费：支持Retain保留和Delete删除策略，Retain仅在高于1.8的集群版本生效）
# aspid:asp-123
# 支持指定快照策略，创建云盘后自动绑定此快照策略,绑定失败不影响创建
```

2. 创建多实例 StatefulSet

使用云硬盘 CBS 创建多实例 StatefulSet：

```
apiVersion: apps/v1beta1
kind: StatefulSet
metadata:
name: web
spec:
serviceName: "nginx"
replicas: 3
template:
metadata:
labels:
app: nginx
spec:
terminationGracePeriodSeconds: 10
containers:
- name: nginx
image: nginx
ports:
- containerPort: 80
name: web
volumeMounts:
- name: www
mountPath: /usr/share/nginx/html
volumeClaimTemplates: # 自动创建pvc，进而自动创建pv
- metadata:
name: www
spec:
```

```
accessModes: [ "ReadWriteOnce" ]
storageClassName: cloud-premium
resources:
requests:
storage: 10Gi
```

版权所有：腾讯云计算（北京）有限责任公司

开源组件

最近更新时间：2018-08-09 16:12:19

tencentcloud-cloud-controller-manager

tencentcloud-cloud-controller-manager 是腾讯云容器服务的 Cloud Controller Manager 的实现。使用该组件，可以在通过腾讯云云服务器自建的 Kubenrentes 集群上实现以下功能：

- nodecontroller：更新 Kubernetes node 相关的 addresses 信息。
- routecontroller：负责创建 VPC 内 pod 网段内的路由。
- servicecontroller：当集群中创建了类型为 LoadBalancer 的 service 的时候，创建相应的 LoadBalancers。

更多安装使用说明，可查看 GitHub 详细介绍。

kubernetes-csi-tencentcloud

kubernetes-csi-tencentloud 是腾讯云 Cloud Block Storage 服务的一个满足 CSI 标准实现的插件。使用该组件，可以在通过腾讯云云服务器自建的 Kubenrentes 集群使用 Cloud Block Storage。
该插件适用与自建 kubernetes 集群的时候使用cbs的插件，与TKE集群自带的 provisioner: cloud.tencent.com/qcloud-cbs 不相同。

更多安装使用说明，可查看 GitHub 详细介绍。

ingress-tke-bm

ingress-tke-bm 是腾讯云 TKE 黑石集群的 ingress controller。该 controller 会监听 ingress 资源，创建黑石 lb 并绑定到对应 service。
更多安装使用说明，可查看 Github 详细介绍

版权所有：腾讯云计算（北京）有限责任公司