2021.3.30 kubernetes 命令工具

一．Kubectl 命令工具

Kubectl 是一个用于操作 Kubernetes 集群的命令行工具，利用 Kubectl 的各种子命令可以实现各种功能，在管理 Kubernetes 集群过程中 kubectl,是非常实用的工具。

Kube-apiserver 是整个 Kubernetes 集群管理的入口。API Server 运行在 Kubernetes 集群的主管理节点(Master）上，用户需要通过 APl Server 配置和组织集群，同时集群中各节点与 Etcd,存储的交互也是通过 API Server 来进行的。

APl Server 实现了一套 RESTfull 的接口，用户可以直接使用 API 与 APl Server 进行交互。另外官方还提供了一个客户端 Kubectl 工具集，可直接通过 Kubectl 命令行的方式与集群进行交互。

1.1 kubectl 命令行的语法

Kubectl[command][TYPE][NAME]{flags}

·Command：子命令。用于操作 kubernetes 集群资源对象的命令。例如：create，delete，describe，get，apply

·TYPE：资源对象的类型，区分大小写，能以单数，复数或者简写形式表示。例如： kubectl get pod pod1，kubectl get pods pod1，kubectl get po pod1 三种 TYPE 是等价的

·NAME：资源对象名称，区分大小写，如果不能指定名称，系统则返回属于 TYPE 的全部对象，例如，执行 kubectl get pods 命令即可返回所有 pod 的列表

·flags：kubectl 的子命令的可选参数，例如使用“-s”指定 PAI server 的 URL 地址而不用默认值

·kubectl 的子命令非常丰富，涵盖了对 kubernetes 集群的主要操作，包括资源对象的创建，删除，查看，修改，配置，运行等操作

1.2 kubectl 命令列表

Kubectl --help 命令帮助

类型	命令	描述
基础命令	create	通过文件名或标准输入创建资源
	expose	将一个资源公开为一个新的 service
	run	在集群中运行一个特定的镜像（docker）
	set	在对象上设置新的功能
	get	显示一个或多个资源
	explain	文档参考资料
	edit	使用默认的编辑器编辑一个资源
	delete	通过文件名，标准输入，资源名称或者标签选择器来删除资源
部署命令	Rollout	管理资源发布
	Rolling-update	滚动升级
	Scale	扩容或缩容 pod 数量，Deployment，Replicaset
	autoscale	创建一个自动选择扩容或者缩容并设置 pod 数量
	certificate	修改证书资源

集群管理命令	cluster-info	显示集群信息
	top	显示资源（CPU/Memory/Storage），使用需要 heapster
	cordon	标记节点不可调度
	uncordon	标记节点可调度
	drain	驱逐节点上的应用，准备下线维护
	taint	修改节点 taint 标记

故障诊断

故障诊断和调试命令	命令	描述
	describe	显示特定资源或资源组的详细信息
	logs	在一个 pod 中打印一个容器，如果 pod 容器只有一个，容器名称是可选的
	attach	附加到一个运行的容器
	prot-forward	转发一个或多个本地端口到一个 pod
	proxy	运行一个 porxy kubernetes APIserver
	cp	拷贝文件或目录到容器中
	auth	检查授权
	exec	执行命令到容器
高级命令	apply	通过文件名或标准输入对资源的应用配置
	patch	使用补丁修改，更新资源的字段
	replace	通过文件的名或者标准输入替换一个资源

	convert	不同的 API 版本之间转换配置文件
设置命令	label	更新资源上的标签
	annotate	更新资源上的注释
	completion	用于实现 kubectl 工具补全
其他命令	api-versions	打印受支持的 API 版本
	config	修改 kubeconfig 文件（用于访问 API，比如配置认证信息）
	help	所有命令帮助
	plugin	运行一个命令插件
	version	打印客户端和服务版本信息

1.3 使用 kubectl 工具容器资源

Kubectl 是管理 Kubernetes 集群的命令行工具,通过生成的 json 格式传递给 APIl Server 进行创建、查看、管理的操作。

使用 kubectl --help 命令可以查看 Kubectl 帮助命令，其中包括基本命令、部署命令、群集管理命令、调试命令以及高级命令等。

```
[root@k8s-master ~]# kubectl  --help
```

在一个完整的项目周期中，包含创建-->发布-->更新-->回滚-->删除等过程，下面针对该过程依次进行操作命令的演示

1.3.1，创建容器

Kubectl run 命令可以创建并运行一个或多个容器镜像，也可以创建一个 deployment 或 job 来管理容器。此命令和 docker run 相类似，也是实现容器镜像的创建，先从仓库中拉取基础镜像，然后对容器进行操作: kubectl run 的命令语法如下所示。

```
kubectl run NAME --image=image [--env="key=value"] [--port=port] [--replicas=replicas] [--dry-run=bool] [--overrides=inline-json] [--command] -- [COMMAND] [args...] [options]
```

各选项的作用分别如下所示

·NAME:指定容器运行的名称;

·Image:指定运行的基础镜像;

·env:指定在容器中设置的环境参数;

·port:指定容器暴露的端口;

·replicas:指定启动容器设置的副本数;

·dry-run: dry-run 值如果为 true，则只打印要发送的对象，而不发送它;

·overrides:生成对象的内联 JSON 重写。如果非空，则用于覆盖生成的对象。要求对象提供有效的 apiVersion 字段。

通过 kubectl run 命令创建 Nginx 容器，指定名称为 nginx-deployment,指定基础镜像为 Nginx 目前最新版本，指定对外暴露的端口为 80 以及副本数为 3。Nginx 容器创建完成后使用 get pod 命令查看 Pod 的信息，可以发现确实有 3 众容器资源，并且处于 Running 状态。还可以查看 deployment，也显示的是 3

```
[root@k8s-master ~]# kubectl run nginx-deployment --image=nginx:1.14 --port=80 --replicas=3
kubectl run --generator=deployment/apps.v1 is DEPRECATED and will be removed in a future version. Use kubectl run --generator=run-pod/v1 or kubectl create instead.
deployment.apps/nginx-deployment created

[root@k8s-master ~]# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-5567b746cd-7vq8r   1/1     Running   0          6m19s
nginx-deployment-5567b746cd-9h7qf   1/1     Running   0          9s
nginx-deployment-5567b746cd-nx65b   1/1     Running   0          6m19s
```

node 主机上传 nginx 软件包，作导成镜像（两台主机）

注意：两台主机需要有 nginx-1.14 和 nginx-1.19 两个镜像

这里有导入镜像和改名两个操作

```
[root@k8s-node01 ~]# rz    // nginx-1.19.tar  // nginx-1.14.tar
[root@k8s-node01~]# docker load <  nginx-1.14.tar  // nginx-1.19.tar
[root@k8s-node02 ~]# rz    // nginx-1.19.tar  // nginx-1.14.tar
[root@k8s-node02~]# docker load <  nginx-1.14.tar  // nginx-1.19.tar
[root@k8s-node01 ~]# docker tag  nginx:latest  nginx:1.19
[root@k8s-node02 ~]# docker tag  nginx:latest  nginx:1.19
```

运行容器实例：

```
[root@k8s-master ~]# kubectl  run nginx-deployment --image=nginx:1.14 --port=80 --replicas=3
kubectl run --generator=deployment/apps.v1 is DEPRECATED and will be removed in a future version. Use kubectl run --generator=run-pod/v1 or kubectl create instead.
deployment.apps/nginx-deployment created
[root@k8s-master ~]# kubectl  get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-6f8754f958-c222c   1/1     Running   0          81s
nginx-deployment-6f8754f958-pqc9c   1/1     Running   0          81s
nginx-deployment-6f8754f958-s5w8t   1/1     Running   0          81s
```

1.3.2 发布服务

容器资源创建完成，就需要完成发布工作，确保 Pod 能够对外提供服务，保证客户端能够正常访问，使用 kubectl expose 命令可以实现该目的。kubectl expose 的命令语法如下

```
kubectl expose (-f FILENAME | TYPE NAME) [--port=port] [--protocol=TCP|UDP] [--target-port=number-or-name] [--name=name] [--external-ip=external-ip-of-service] [--type=type]
```

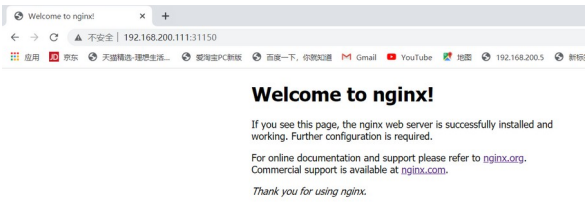
发布示例：

```
[root@k8s-master~]#kubectl expose deployment nginx-deployment --port=80 --target-port=80 --name=nginx-service --type=NodePort
service/nginx-service exposed
```

查看发布结果

```
[root@k8s-master ~]# kubectl  get svc
NAME         TYPE       CLUSTER-IP     EXTERNAL-IP   PORT(S)       AGE
kubernetes   ClusterIP  10.96.0.1      <none>        443/TCP       3h37m
nginx-service  NodePort   10.96.80.101  <none>        80:31150/TCP  68s
```

访问 192.168.200.111:31150



查看 server 的信息

```
[root@k8s-master ~]# kubectl get endpoints
NAME         ENDPOINTS                     AGE
kubernetes   192.168.200.111:6443          3h41m
nginx-service  10.244.1.2:80,10.244.1.3:80,10.244.2.3:80  4m37s

[root@k8s-master ~]# kubectl  get pods -o wide   #查看 pods 的详细信息

NAME                                READY   STATUS    RESTARTS   AGE   IP           NODE
NOMINATED NODE   READINESS GATES
nginx-deployment-6f8754f958-c222c   1/1     Running   0          11m   10.244.2.3   k8s-
node02   <none>           <none>
nginx-deployment-6f8754f958-pqc9c   1/1     Running   0          11m   10.244.1.2   k8s-
node01   <none>           <none>
nginx-deployment-6f8754f958-s5w8t   1/1     Running   0          11m   10.244.1.3   k8s-
```

```
node01   <none>
```

1.3.3 版本更新

一般来说,生产环境中的线上项目会随着时间的推进不断地进行更新、维护、版本升级、兼容老版本。而此时如果需要对 Nginx 更换版本，就需要滚动更新。

查看 nginx 当前版本

```
[root@k8s-master ~]# kubectl  exec nginx-deployment-6f8754f958-s5w8t  -- nginx -v
nginx version: nginx/1.14.2
```

通过 set 选项将 Nginx 版本换成 1.19，再监使用-w 先处于监听状态进行听，更新速度快。如果不使用 Ctrl+c 中断监听,会一直持续。

```
[root@k8s-master ~]# kubectl  get deploy,pods   #查看一个 server
NAME                               READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/nginx-deployment   3/3     3            3           16m

NAME                                   READY   STATUS    RESTARTS   AGE
pod/nginx-deployment-6f8754f958-c222c   1/1     Running   0          16m
pod/nginx-deployment-6f8754f958-pqc9c   1/1     Running   0          16m
pod/nginx-deployment-6f8754f958-s5w8t   1/1     Running   0          16m
[root@k8s-master ~]# kubectl  set image deployment.apps/nginx-deployment nginx-deployment=nginx:1.19   #滚动更新
deployment.apps/nginx-deployment image updated
[root@k8s-master ~]# kubectl  get pods -w
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-76775dd78f-hv555   1/1     Running   0          37s
nginx-deployment-76775dd78f-rdgcv   1/1     Running   0          33s
nginx-deployment-76775dd78f-tt8hj   1/1     Running   0          34s
[root@k8s-master ~]# kubectl  exec nginx-deployment-76775dd78f-hv555  -- nginx -v  #再次查看当前版本
nginx version: nginx/1.19.6
```

1.3.4 版本回滚

如果想回滚到上一个版本 可以使用 rollout

查看历史发布的版本信息

```
[root@k8s-master ~]# kubectl  rollout history deployment.apps/nginx-deployment
deployment.apps/nginx-deployment
REVISION  CHANGE-CAUSE
1         <none>
2         <none>
```

使用 undo 回滚

```
[root@k8s-master ~]# kubectl  rollout undo deployment.apps/nginx-deployment
[root@k8s-master ~]# kubectl  exec nginx-deployment-6f8754f958-ckbjx  -- nginx -v
nginx version: nginx/1.14.2
```

1.3.5 删除资源

使用 delete

```
[root@k8s-master ~]# kubectl  get deploy
NAME               READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deployment   3/3     3            3           29m
[root@k8s-master ~]# kubectl  delete deployment/nginx-deployment
deployment.apps "nginx-deployment" deleted
[root@k8s-master ~]# kubectl  get deploy
No resources found in default namespace.

[root@k8s-master ~]# kubectl  delete svc/nginx-service  #删除 service

service "nginx-service" deleted
```

1.3.7 执行容器命令

```
[root@k8s-master ~]# kubectl  get pods
NAME                     READY   STATUS              RESTARTS   AGE
nginx-dd6b5d745-2rfr5    0/1     ContainerCreating   0          13s
nginx-dd6b5d745-55g6r    0/1     ContainerCreating   0          13s
nginx-dd6b5d745-ldqqk    0/1     ContainerCreating   0          13s
```

```
[root@k8s-master ~]# kubectl  exec -it nginx-dd6b5d745-ldqqk bash
root@nginx-dd6b5d745-ldqqk:/#
```

二 YAML 资源清单

在 Kubernetes 中可以使用 YAML 格式的文件来创建符合预期期望的 Pod，这样的 YAML 文件一般称之为资源清单。

2.1 YAML 语言

YAML 语言是一个可读性高，用来表达数据序列的语言格式。YAML 是"XMlAin't aMarkupLanguage”(YAML 不是一种标记语言)的递归缩写。在开发这个语言时，YAML 的意思其实是:"YetAnother Markup Language"(仍是一种标记语言)，但为了强调这种语言以数据做为中心，而不是以标记语言为重点，而用反向缩略语重命名。

（1）基本语法

缩进时不允许使用 Tab 键，只允许使用空格

缩进的空格数目不重要，只要相同层级的元素左侧对齐即可;

#用于标识注释，从这个字符一直到行尾，都会被解释器忽略。

（2）支持的数据结构

对象:键值对的集合，又称为映射（mapping) /哈希(hashes) /字典(dictionary);

数组:一组按次序排列的值，又称为序列(sequence)/列表（list);

纯量(scalars):单个的、不可再分的值字符串。包括布尔值、整数、浮点数 Null、时间、日期。

2.2 通过资源清单管理容器资源

前面介绍了使用 kubectl 命令创建容器资源方法。基于这种命令方式创建容器资源，优点在于简单直观快捷、上手比较快，适合临时测试或实验。除了 kubectl 命令方式创建资源之外，还可以通过 YAML 配置文件来创建容器资源。

基于 YAML 配置文件创建容器资源的方式，优点在于配置文件提供了创建资源的模板，能够重复部署，可以像管理代码一样管理部署，适合正式的、跨环境的、规模化部署。

YAML 语法格式:

缩进标识层级关系;

不支持制表符（Tab 键）缩进，使用空格缩进;

通常开头缩进两个空格;

字符后缩进一个空格，如冒号、逗号等;

“---”表示 YAML 格式，一个文件的开始;

“#”表示注释。

在创建 Deployment 资源清单之前先创建 demo 目录，用于存放资源清单的文件。在创建的 nginx.deploxment.xaml 资源清单中，定义以下信息。

```
apiVersion: apps/v1      #指定 API 版本
kind: Deployment         #指定资源类型
metadata:                #指定属性
  name: nginx-deployment  #指定名称
  labels:                #定义标签
    app: nginx
spec:                    #定义详细信息
  replicas: 3            #指定副本数量
  selector:              #定义选择器信息
    matchLabels:
      app: nginx
  template:              #指定模板
    metadata:
      labels:
        app: nginx
    spec:                #定义容器信息
      containers:
      - name: nginx
        image: nginx:1.19.6
        ports:
        - containerPort: 80
```

2.2 使用 kubectl 命令创建 YAML 模板

[root@k8s-master ~]# kubectl run nginx-deployment --image=nginx:1.14 --port=80 -o yaml --dry-run > nginx.yaml

kubectl run --generator=deployment/apps.v1 is DEPRECATED and will be removed in a future version. Use kubectl run --generator=run-pod/v1 or kubectl create instead.

[root@k8s-master ~]# ls

```
[root@k8s-master ~]# ls
anaconda-ks.cfg             images              k8s-images-1.17.0.tar  公共  图片  音
docker                      init-config.yaml    kube-flannel.yml       模板  文档  桌
flannel_v0.12.0-amd64.tar   initial-setup-ks.cfg nginx.yaml             视频  下载
```

创建 nginx 容器资源

```
[root@k8s-master ~]# kubectl  create -f nginx.yaml
deployment.apps/nginx-deployment created
[root@k8s-master ~]# kubectl  get pod
NAME                     READY   STATUS    RESTARTS   AGE
nginx-dd6b5d745-2rfr5          1/1     Running   0          22m
nginx-dd6b5d745-55g6r          1/1     Running   0          22m
nginx-dd6b5d745-ldqqk          1/1     Running   0          22m
nginx-deployment-6f8754f958-f2hjz   1/1     Running   0          9s
```

创建 service 模板

```
[root@k8s-master ~]#
kubectl expose deployment nginx-deployment --port=80 --target-port=80 --name=nginx-service --type=NodePort -o yaml --dry-run > nginx-service.yaml
[root@k8s-master ~]# kubectl get svc
NAME          TYPE        CLUSTER-IP    EXTERNAL-IP   PORT(S)        AGE
kubernetes    ClusterIP   10.96.0.1     <none>        443/TCP        4h33m
nginx-service NodePort    10.96.0.22    <none>        80:32127/TCP   15s
```

访问 192.168.200.111:32127