**Kubernets常用命令**

# 查看所有namespace的pods运行情况

kubectl get pods --all-namespaces

kubectl get service -ALL

kubectl get deployment -ALL

kubectl get service web01 -o json ##输出json

kubectl get pods web01-bbc5667-xb7cd -o json

kubectl get service web01 -o wide ##输出网络信息

kubectl get endpoints web01 ##如果有值，说明endpoints 控制器已经为您的 Service 找到了对应的Pods，否则可能创建Service时出错，如：暴露端口expose时，名称写错了，导出创建的Service找不到对应的pod

# 查看具体pods，记得后边跟namespace名字哦

kubectl get pods kubernetes-dashboard-76479d66bb-nj8wr --namespace=kube-system

# 查看pods具体信息

kubectl get pods -o wide kubernetes-dashboard-76479d66bb-nj8wr --namespace=kube-system

# 查看集群健康状态

kubectl get cs

查看节点加入集群信息

kubectl get csr

# 获取所有deployment

kubectl get deployment --all-namespaces

# 列出该 namespace 中的所有 pod 包括未初始化的

kubectl get pods --include-uninitialized

# 查看deployment()

kubectl get deployment nginx-app

# 查看rc和servers

kubectl get rc,services

# 查看pods结构信息（重点，通过这个看日志分析错误）对控制器和服务，node同样有效

kubectl describe pods xxxxpodsname --namespace=xxxnamespace

其他控制器类似吧，就是kubectl get 控制器 控制器具体名称

# 查看pod日志

kubectl logs $POD\_NAME

# 查看pod变量

kubectl exec my-nginx-5j8ok -- printenv | grep SERVICE

kubectl get cs #### 集群健康情况  
kubectl cluster-info #### 集群核心组件运行情况  
kubectl get namespaces #### 表空间名  
kubectl version #### 版本  
kubectl api-versions #### API  
kubectl get events #### 查看事件  
kubectl get nodes //获取全部节点  
kubectl delete node k8s2 //删除节点  
kubectl rollout status deploy nginx-test

# 创建

kubectl create -f ./nginx.yaml #### 创建资源  
kubectl create -f . #### 创建当前目录下的所有yaml资源  
kubectl create -f ./nginx1.yaml -f ./mysql2.yaml #### 使用多个文件创建资源  
kubectl create -f ./dir #### 使用目录下的所有清单文件来创建资源  
kubectl create -f https://git.io/vPieo #### 使用 url 来创建资源  
kubectl run -i --tty busybox --image=busybox ----创建带有终端的pod  
kubectl run nginx --image=nginx #### 启动一个 nginx 实例  
kubectl run mybusybox --image=busybox --replicas=5 ----启动多个pod  
kubectl explain pods,svc #### 获取 pod 和 svc 的文档

# 更新

kubectl rolling-update python-v1 -f python-v2.json #### 滚动更新 pod frontend-v1  
kubectl rolling-update python-v1 python-v2 --image=image:v2 #### 更新资源名称并更新镜像  
kubectl rolling-update python --image=image:v2 #### 更新 frontend pod 中的镜像  
kubectl rolling-update python-v1 python-v2 --rollback #### 退出已存在的进行中的滚动更新  
cat pod.json | kubectl replace -f - #### 基于 stdin 输入的 JSON 替换 pod  
强制替换，删除后重新创建资源。会导致服务中断。  
kubectl replace --force -f ./pod.json  
为 nginx RC 创建服务，启用本地 80 端口连接到容器上的 8000 端口  
kubectl expose rc nginx --port=80 --target-port=8000

更新单容器 pod 的镜像版本（tag）到 v4  
kubectl get pod nginx-pod -o yaml | sed 's/(image: myimage):.\*$/\1:v4/' | kubectl replace -f -  
kubectl label pods nginx-pod new-label=awesome #### 添加标签  
kubectl annotate pods nginx-pod icon-url=http://goo.gl/XXBTWq #### 添加注解  
kubectl autoscale deployment foo --min=2 --max=10 #### 自动扩展 deployment “foo”

# 编辑资源

kubectl edit svc/[**docker**](https://debian.cn/tag/docker/)-registry #### 编辑名为 docker-registry 的 service  
KUBE\_EDITOR="nano" kubectl edit svc/docker-registry #### 使用其它编辑器

# 动态伸缩pod

kubectl scale --replicas=3 rs/foo #### 将foo副本集变成3个  
kubectl scale --replicas=3 -f foo.yaml #### 缩放“foo”中指定的资源。  
kubectl scale --current-replicas=2 --replicas=3 deployment/mysql #### 将deployment/mysql从2个变成3个  
kubectl scale --replicas=5 rc/foo rc/bar rc/baz #### 变更多个控制器的数量  
kubectl rollout status deploy deployment/mysql #### 查看变更进度

# 删除

kubectl delete -f ./pod.json #### 删除 pod.json 文件中定义的类型和名称的 pod  
kubectl delete pod,service baz foo #### 删除名为“baz”的 pod 和名为“foo”的 service  
kubectl delete pods,services -l name=myLabel #### 删除具有 name=myLabel 标签的 pod 和 serivce  
kubectl delete pods,services -l name=myLabel --include-uninitialized #### 删除具有 name=myLabel 标签的 pod 和 service，包括尚未初始化的  
kubectl -n my-ns delete po,svc --all #### 删除 my-ns namespace下的所有 pod 和 serivce，包括尚未初始化的  
kubectl delete pods prometheus-7fcfcb9f89-qkkf7 --grace-period=0 --force 强制删除

# 交互

kubectl logs nginx-pod #### dump 输出 pod 的日志（stdout）  
kubectl logs nginx-pod -c my-container #### dump 输出 pod 中容器的日志（stdout，pod 中有多个容器的情况下使用）  
kubectl logs -f nginx-pod #### 流式输出 pod 的日志（stdout）  
kubectl logs -f nginx-pod -c my-container #### 流式输出 pod 中容器的日志（stdout，pod 中有多个容器的情况下使用）  
kubectl run -i --tty busybox --image=busybox -- sh #### 交互式 shell 的方式运行 pod  
kubectl attach nginx-pod -i #### 连接到运行中的容器  
kubectl port-forward nginx-pod 5000:6000 #### 转发 pod 中的 6000 端口到本地的 5000 端口  
kubectl exec nginx-pod -- ls / #### 在已存在的容器中执行命令（只有一个容器的情况下）  
kubectl exec nginx-pod -c my-container -- ls / #### 在已存在的容器中执行命令（pod 中有多个容器的情况下）

kubectl exec -it -n default nginx01-67fdf8d7c7-stvwt /bin/bash ###进入pod容器中，可以修改服务的配置，如nginx、tomcate的配置

kubectl top pod POD\_NAME --containers #### 显示指定 pod和容器的指标度量

# 调度配置

kubectl cordon k8s-node #### 标记 my-node 不可调度  
kubectl drain k8s-node #### 清空 my-node 以待维护  
kubectl uncordon k8s-node #### 标记 my-node 可调度  
kubectl top node k8s-node #### 显示 my-node 的指标度量  
kubectl cluster-info dump #### 将当前集群状态输出到 stdout  
kubectl cluster-info dump --output-directory=/path/to/cluster-state #### 将当前集群状态输出到 /path/to/cluster-state

# 如果该键和影响的污点（taint）已存在，则使用指定的值替换

kubectl taint nodes foo dedicated=special-user:NoSchedule

# 在被删除的node节点清空集群信息

kubeadm reset**（谨慎使用）**

# 初始化master

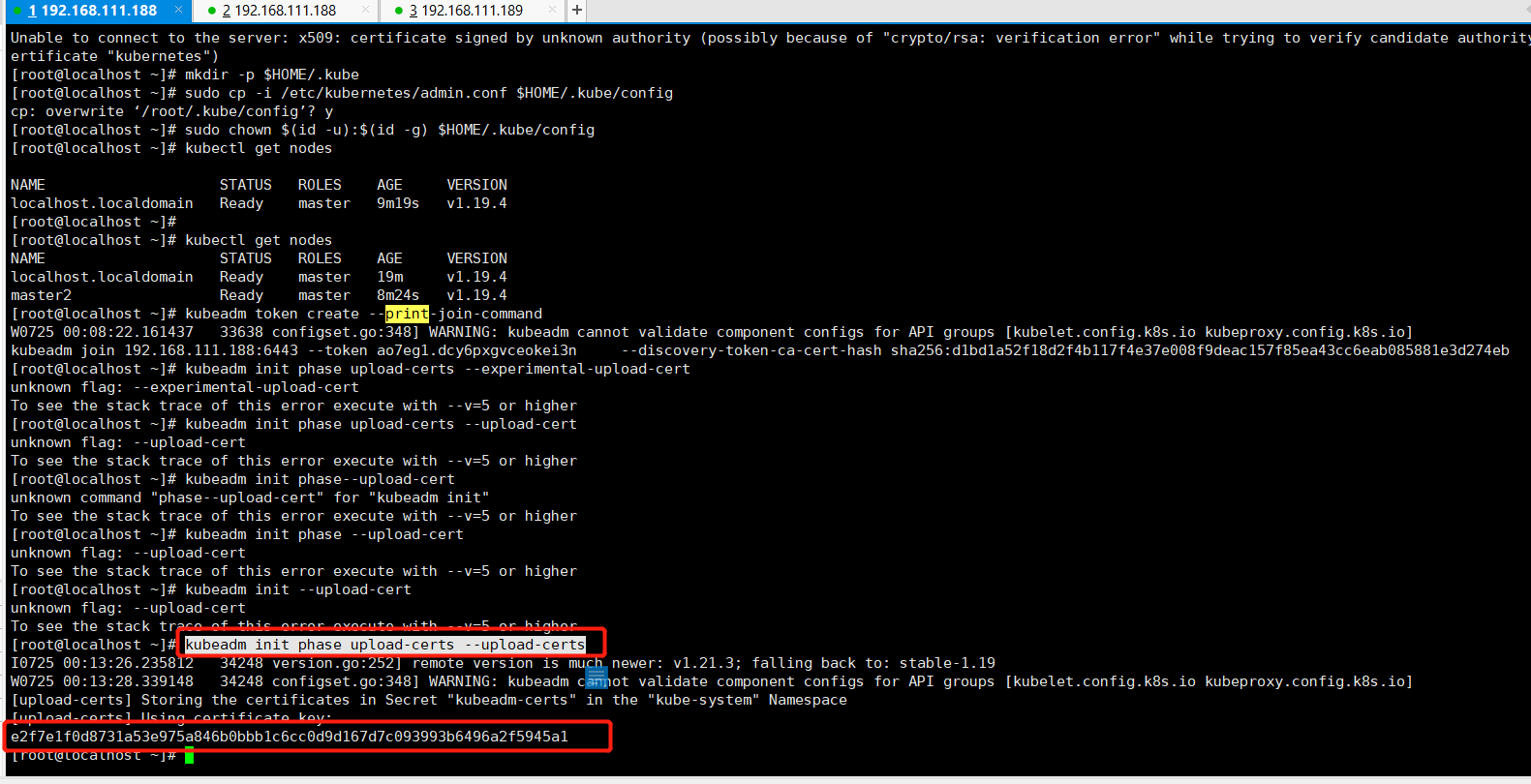
kubeadm init --apiserver-advertise-address=192.168.126.188 --image-repository registry.aliyuncs.com/google\_containers --kubernetes-version v1.19.4 --service-cidr=10.96.0.0/12 --pod-network-cidr=10.244.0.0/16

# 查看相关证书是否过期

kubeadm alpha certs check-expiration

# 在master上生成用于新master加入的证书（添加新Master）

kubeadm init phase upload-certs --upload-certs



e2f7e1f0d8731a53e975a846b0bbb1c6cc0d9d167d7c093993b6496a2f5945a1

先后执行以下2条命令来组合成生成新Master的命令：

kubeadm token create --print-join-command

kubeadm init phase upload-certs --upload-certs

示例如下：

kubeadm join 20.78.200.190:16443 --token abcdef.0123456789abcdef \

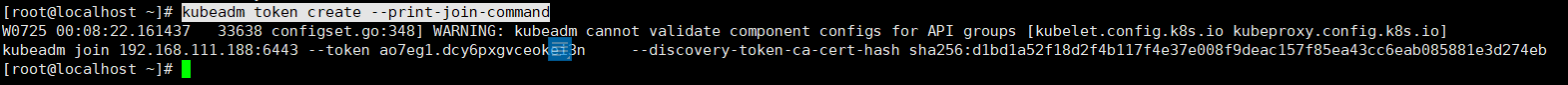
--discovery-token-ca-cert-hash sha256:a5bffb8aacbd60a03925e4a4c1ff65bb07e25618edfc356842e2b458f446d980 \

--control-plane --certificate-key dee8bfe34fffcaabdf9c9602ba25100c6f77eae4294203672d2b3c77f1ea15e0

# 在master节点查看集群的token值（添加Node）

kubeadm token create --print-join-command ####重新生成token

kubeadmin token list ####查看token



kubeadm join 192.168.111.188:6443 --token ao7eg1.dcy6pxgvceokei3n --discovery-token-ca-cert-hash sha256:d1bd1a52f18d2f4b117f4e37e008f9deac157f85ea43cc6eab085881e3d274eb

# 创建初始化配置文件

kubeadm config print init-defaults > kubeadm-config.yaml

# 在集群搭建完成后可以使用如下命令查看生效的配置文件：

kubectl -n kube-system get cm kubeadm-config -oyaml