k8s完整搭建教程

一、服务器准备
二、CentOS环境配置
- 2.1 网络配置
- 2.2 设置主机名
- 2.3 关闭交换分区
- 2.4 检查MAC是否唯一
- 2.5 检查product_uuid是否唯一
- 2.6 关闭Linux安全配置
- 2.7 允许iptables桥接流量
- 2.8 安装工具
- 2.9 关闭防火墙
- 2.10 设置yum源
- 2.11 安装文件上传工具

三、Docker安装配置
- 3.1 docker-ce安装
- 3.2 配置镜像加速
- 3.3 其他命令

四、k8s的安装
- 4.1 设置仓库
- 4.2 安装k8s
- 4.2 初始化集群
- 4.3 安装可视化面板

一、服务器准备

采用虚拟机vmware软件创建三台虚拟CentOS7服务器，本文以NAT模式上网为例。

192.168.153.5	2G	2核	20G
192.168.153.6	2G	2核	20G
192.168.153.7	2G	2核	20G

二、CentOS环境配置

2.1 网络配置

（1）编辑网卡配置文件

```
cat /etc/sysconfig/network-scripts/ifcfg-ens33
vim /etc/sysconfig/network-scripts/ifcfg-ens33
```

（2）修改三台服务器配置

```
ONBOOT=yes
BOOTPROTO=static
DNS1=114.114.114.114
IPADDR=192.168.153.5
NETMASK=255.255.255.0
GATEWAY=192.168.153.2
```

DNS的配置还可以这样配置（用于访问互联网）：

```
vim /etc/resolv.conf
nameserver 114.114.114.114
```

（3）重启网络服务

```
service network restart
```

2.2 设置主机名

（1）查看主机名

```
hostname
```

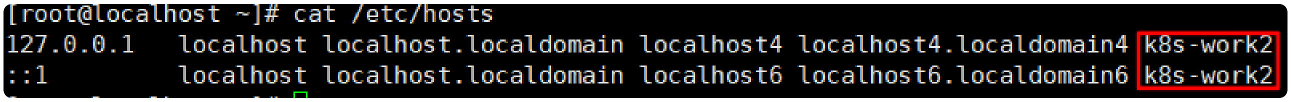
（2）分别给3台服务器设置主机名

```

hostnamectl set-hostname k8s-master
hostnamectl set-hostname k8s-work1
hostnamectl set-hostname k8s-work2
```

（3）编辑host添加域名映射

```
vim /etc/hosts
```



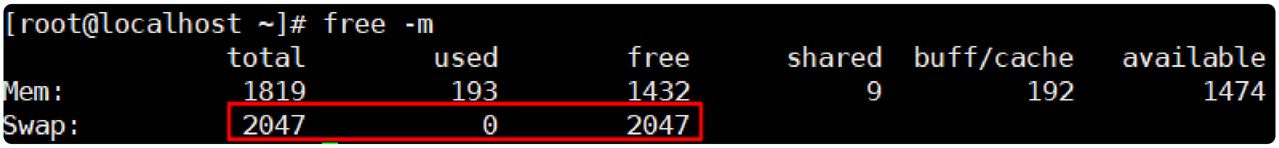
```
[root@localhost ~]# cat /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4 k8s-work2
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6 k8s-work2
```

2.3 关闭交换分区

（1）查看是否开启了交换分区

```
free -m
```

出现了红框中的内容表示开启了交换分区：

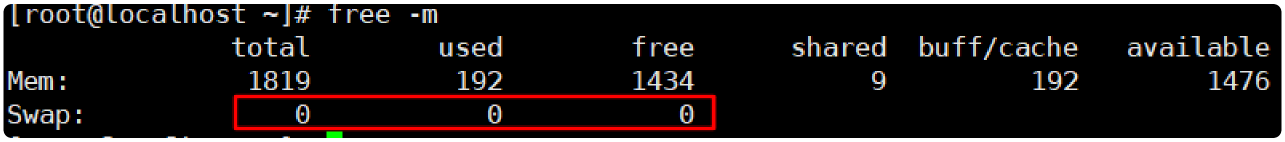


```
[root@localhost ~]# free -m
              total        used        free      shared  buff/cache   available
Mem:           1819         193        1432           9         192        1474
Swap:          2047           0        2047
```

（2）分别3台服务器执行如下命令关闭交换分区

```
swapoff -a
sed -ri 's/.*swap.*/#&/' /etc/fstab
```

出现下面表示关闭了交换分区：

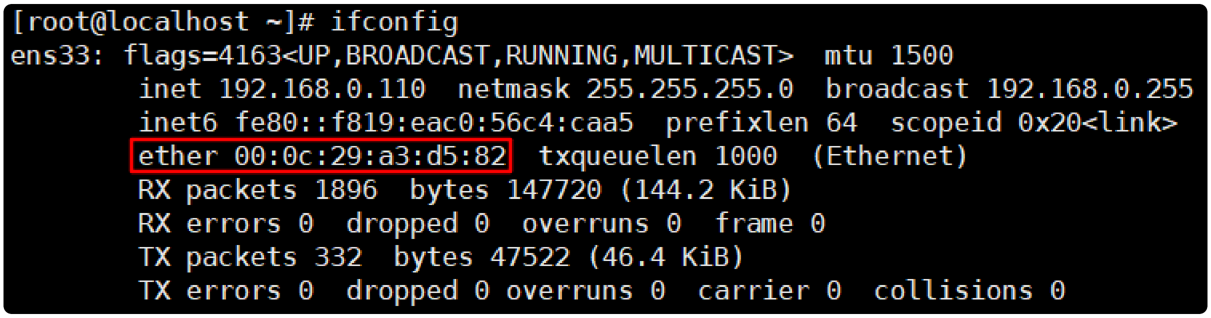


```
[root@localhost ~]# free -m
              total        used        free      shared  buff/cache   available
Mem:           1819         192        1434           9         192        1476
Swap:             0           0           0
```

2.4 检查MAC是否唯一

分别在3台服务器执行如下命令，检查是否都是由唯一的mac地址。

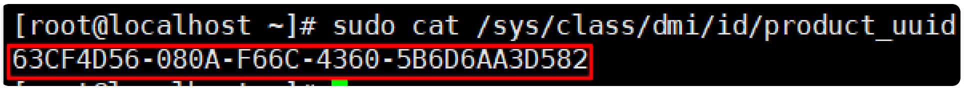
```
ifconfig
```



```
[root@localhost ~]# ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.0.110  netmask 255.255.255.0  broadcast 192.168.0.255
        inet6 fe80::f819:eac0:56c4:caa5  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:a3:d5:82  txqueuelen 1000  (Ethernet)
        RX packets 1896  bytes 147720 (144.2 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 332  bytes 47522 (46.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

2.5 检查product_uuid是否唯一

```
sudo cat /sys/class/dmi/id/product_uuid
```



```
[root@localhost ~]# sudo cat /sys/class/dmi/id/product_uuid
63CF4D56-080A-F66C-4360-5B6D6AA3D582
```

2.6 关闭Linux安全配置

分别在3台服务器执行如下命令：

```
sudo setenforce 0
sudo sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config
```

2.7 允许iptables桥接流量

分别在3台服务器执行如下命令：

```
cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf
br_netfilter
EOF
```

```
cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-ip6tables=1
net.bridge.bridge-nf-call-iptables=1
EOF
```

```
sudo sysctl --system
```

2.8 安装工具

```
yum install wget
```

2.9 关闭防火墙

```
systemctl stop firewalld.service
```

2.10 设置yum源

（1）下载yum到/etc/yum.repos.d/CentOS-Base.repo目录

```
wget -O /etc/yum.repos.d/CentOS-Base.repo https://mirrors.aliyun.com/repo/Centos-7.repo
或
curl -o /etc/yum.repos.d/CentOS-Base.repo https://mirrors.aliyun.com/repo/Centos-7.repo
```

（2）清空缓存

```
yum clean all && yum makecache
```

2.11 安装文件上传工具

（1）安装

```
yum install -y lrzsz
```

（2）上传文件

```
rz -y
```

（3）下载文件

```
sz filename
```

三、Docker安装配置

3.1 docker-ce安装

分别在3台服务器上面安装docker-ce容器。

（1）更新yum

```
yum update
```

（1）先卸载老版本

```
sudo yum remove docker \
                  docker-client \
                  docker-client-latest \
                  docker-common \
                  docker-latest \
                  docker-latest-logrotate \
                  docker-logrotate \
                  docker-engine
```

（2）设置仓库

```
sudo yum install -y yum-utils
sudo yum-config-manager --add-repo https://download.docker.com/linux/centos/docker-ce.repo

sudo yum makecache fast #更新索引包
```

（3）查找可以安装的版本

```
yum list docker-ce --showduplicates | sort -r
```

（4）安装指定版本的Docker

```
yum install -y docker-ce-20.10.7 docker-ce-cli-20.10.7  containerd.io-1.4.6
```

（5）启动Docker

```
sudo systemctl start docker
```

（6）设置开机自启

```
systemctl enable docker.service
```

3.2 配置镜像加速

（1）阿里云加镜像加速器官网

阿里云镜像加速器官网

（2）在3台服务器配置镜像加速器

```
sudo mkdir -p /etc/docker
```

```
sudo tee /etc/docker/daemon.json <<-'EOF'
{
  "registry-mirrors": ["https://1t76n4pl.mirror.aliyuncs.com"],
  "exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
```

```
sudo systemctl daemon-reload
sudo systemctl restart docker
```

3.3 其他命令

（1）重启docker服务

```
systemctl restart  docker
```

（2）关闭开机自启

```
systemctl disable docker.service
```

（3）查看是否设置了开机自启

```
systemctl list-unit-files | grep enable
```

四、k8s的安装

```
name  architectures
registry.k8s.io/conformance:v1.22.17  amd64, arm, arm64, ppc64le, s390x
registry.k8s.io/kube-apiserver:v1.22.17 amd64, arm, arm64, ppc64le, s390x
registry.k8s.io/kube-controller-manager:v1.22.17  amd64, arm, arm64, ppc64le, s390x
registry.k8s.io/kube-proxy:v1.22.17 amd64, arm, arm64, ppc64le, s390x
registry.k8s.io/kube-scheduler:v1.22.17 amd64, arm, arm64, ppc64le, s390x
```

4.1 设置仓库

```
cat <<EOF | sudo  tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=http://mirrors.aliyun.com/kubernetes/yum/repos/kubernetes-el7-x86_64
enabled=1
gpgcheck=0
repo_gpgcheck=0
gpgkey=http://mirrors.aliyun.com/kubernetes/yum/doc/yum-key.gpg
  http://mirrors.aliyun.com/kubernetes/yum/doc/rpm-package-key.gpg
exclude=kubelet kubeadm kubectl
EOF
```

4.2 安装k8s

（1）在每台服务器如下命令安装

```
sudo yum install -y kubelet-1.22.17 kubeadm-1.22.17 kubectl-1.22.17 --disableexcludes=kubernetes
```

（2）启动k8s

```
sudo systemctl enable --now kubelet
```

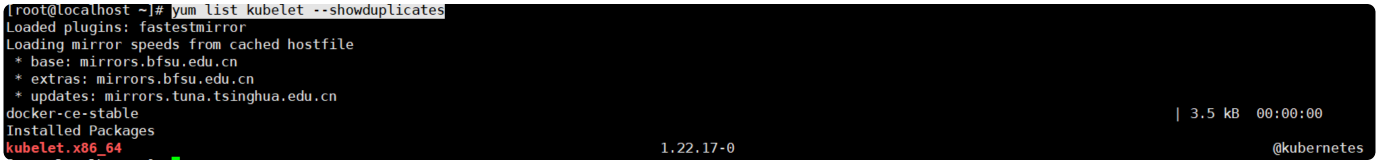
4.2 初始化集群

（1）在3个服务器添加域名映射

```
echo "192.168.153.5 cluster-endpoint" >> /etc/hosts
```

（2）查看可以安装的kubernete版本

```
yum list kubelet --showduplicates
```



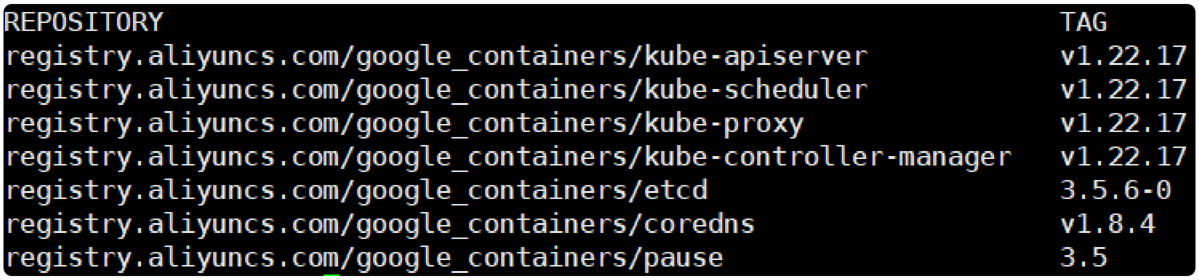
```
[root@localhost ~]# yum list kubelet --showduplicates
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: mirrors.bfsu.edu.cn
 * extras: mirrors.bfsu.edu.cn
 * updates: mirrors.tuna.tsinghua.edu.cn
docker-ce-stable                                                                                                    | 3.5 kB  00:00:00
Installed Packages
kubelet.x86_64                                           1.22.17-0                                                               @kubernetes
```

（3）查看所需要下载的镜像

```
kubeadm --kubernetes-version 1.22.17 config images list
```

```
registry.k8s.io/kube-apiserver:v1.22.17
registry.k8s.io/kube-controller-manager:v1.22.17
registry.k8s.io/kube-scheduler:v1.22.17
registry.k8s.io/kube-proxy:v1.22.17
registry.k8s.io/pause:3.5
registry.k8s.io/etcd:3.5.6-0
registry.k8s.io/coredns/coredns:v1.8.4
```

（4）下载镜像



```
REPOSITORY                                                   TAG
registry.aliyuncs.com/google_containers/kube-apiserver           v1.22.17
registry.aliyuncs.com/google_containers/kube-scheduler           v1.22.17
registry.aliyuncs.com/google_containers/kube-proxy               v1.22.17
registry.aliyuncs.com/google_containers/kube-controller-manager  v1.22.17
registry.aliyuncs.com/google_containers/etcd                     3.5.6-0
registry.aliyuncs.com/google_containers/coredns                  v1.8.4
registry.aliyuncs.com/google_containers/pause                    3.5
```

```
sudo tee ./k8simages.sh <<-'EOF'
#!/bin/bash
images=(
kube-apiserver:v1.22.17
kube-proxy:v1.22.17
kube-controller-manager:v1.22.17
kube-scheduler:v1.22.17
coredns:v1.8.4
etcd:3.5.6-0
pause:3.5
)
for imageName in ${images[@]};do
docker pull registry.aliyuncs.com/google_containers/$imageName
done
EOF
```

```
chmod +x ./k8simages.sh &&./k8simages.sh
```

（4）初始化集群

```
kubeadm init \
  --kubernetes-version 1.22.17 \
  --apiserver-advertise-address=192.168.153.5 \
  --control-plane-endpoint=cluster-endpoint \
  --service-cidr=10.96.0.0/16 \
  --pod-network-cidr=192.168.0.0/16 \
  --image-repository registry.aliyuncs.com/google_containers
```

（5）按照提示在创建相应的文件

```
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

```
export KUBECONFIG=/etc/kubernetes/admin.conf
```

（6）安装网络插件

网络插件下载地址官网

这里以calico为例，如果下载慢，可以下载后上传到服务器。

```
curl https://docs.projectcalico.org/manifests/calico.yaml -O
```

```
kubectl apply -f calico.yaml
```

（7）将master加入集群（由多个master的情况下）

```
kubeadm join cluster-endpoint:6443 --token 8p2b25.d98qeepyaonvz5xs \
    --discovery-token-ca-cert-hash sha256:66e4611a76767628bfbf6a9cbb791f79f5a83559bae35c189616e388df20c1f4 \
    --control-plane
```

（8）将worker加入集群

```
kubeadm join cluster-endpoint:6443 --token 8p2b25.d98qeepyaonvz5xs \
    --discovery-token-ca-cert-hash sha256:66e4611a76767628bfbf6a9cbb791f79f5a83559bae35c189616e388df20c1f4
```

4.3 安装可视化面板

（1）下载dashboard并命名为k8s-dashboard.yaml

```
https://raw.githubusercontent.com/kubernetes/dashboard/v2.7.0/aio/deploy/recommended.yaml
```

（2）安装dashboard

```
kubectl apply -f k8s-dashboard.yaml
```

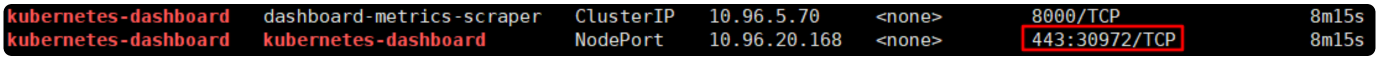
（3）修改配置

```
kubectl edit svc kubernetes-dashboard -n kubernetes-dashboard
```

搜索type=ClusterIP改成NodePort

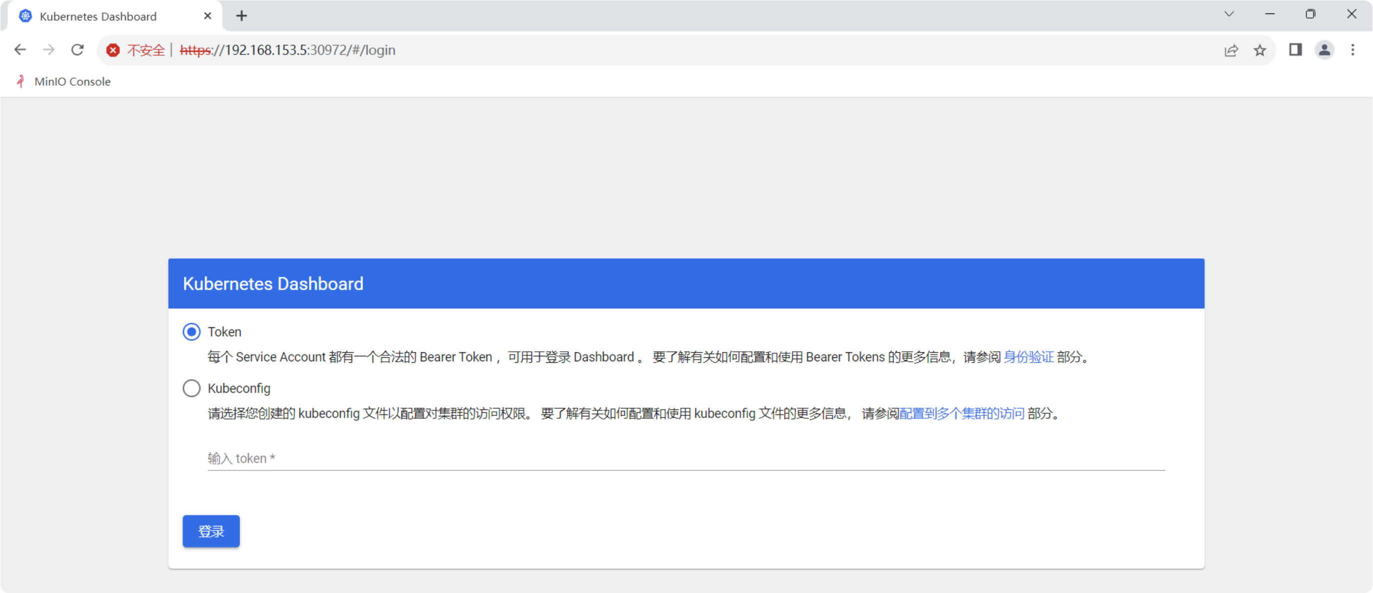
（4）查看dashboard信息

```
kubectl get svc -A|grep kubernetes-dashboard
```



```
kubernetes-dashboard   dashboard-metrics-scraper   ClusterIP   10.96.5.70     <none>        8000/TCP        8m15s
kubernetes-dashboard   kubernetes-dashboard        NodePort    10.96.20.168   <none>        443:30972/TCP   8m15s
```

访问https://192.168.153.5:30972即可进入界面：



Kubernetes Dashboard

● Token

每个 Service Account 都有一个合法的 Bearer Token，可用于登录 Dashboard。 要了解有关如何配置和使用 Bearer Tokens 的更多信息，请参阅 身份验证 部分。

○ Kubeconfig

请选择您创建的 kubeconfig 文件以配置对集群的访问权限。 要了解有关如何配置和使用 kubeconfig 文件的更多信息， 请参阅配置到多个集群的访问 部分。

输入 token *

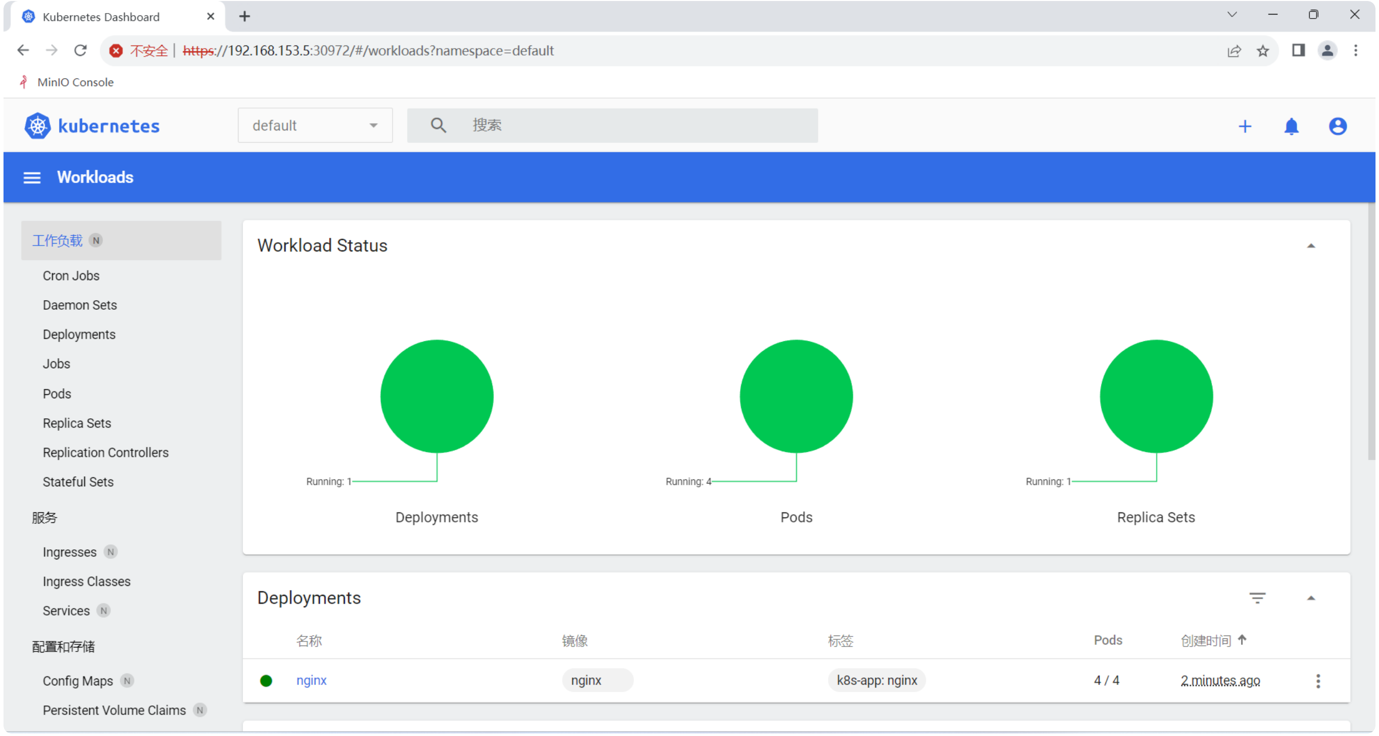
（5）创建访问账号account.yaml

```yaml
apiVersion: v1
kind: ServiceAccount
metadata:
  name: admin-user
  namespace: kubernetes-dashboard
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: admin-user
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: cluster-admin
subjects:
  - kind: ServiceAccount
    name: admin-user
    namespace: kubernetes-dashboard
```

（6）获取访问密钥Token

```
kubectl   -n kubernetes-dashboard get secret $(kubectl -n kubernetes-dashboard get sa/admin-user -o  jsonpath="{.secrets[0].name}") -o go-template="{{.data.token |base64decode}}"
```

```
eyJhbGciOiJSUzI1NiIsImtpZCI6InVkdG0xc3h5OHVKNE5BSE85OVpLQk1mcTVwbkNwemQ0WjFDS2U2eWhnWUkifQ.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZpY2VhY2NvdW50Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9uYW1lc3BhY2UiOiJrdWJlcm5ldGVzLWRhc2hib2FyZCIsImt1YmVybmV0ZXMuaW8vc2VydmljZWFjY291bnQvc2VjcmV0Lm5hbWUiOiJhZG1pbi11c2VyLXRva2VuLWxtMnY0Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZXJ2aWNlLWFjY291bnQubmFtZSI6ImFkbWluLXVzZXIiLCJrdWJlcm5ldGVzLmlvL3NlcnZpY2VhY2NvdW50L3NlcnZpY2UtYWNjb3VudC51aWQiOiJkOTI0YzcwNS0wM2VkLTRhYzctODRmOS04MTlmZWE2ZjgyMjUiLCJzdWIiOiJzeXN0ZW06c2VydmljZWFjY291bnQ6a3ViZXJuZXRlcy1kYXNoYm9hcmQ6YWRtaW4tdXNlciJ9.IKg3ti_OtCSv1jbZjjETv1TeFvAQRlgfvjZbR-0bMvX_i1MeZn_DHKCNJ4l62PoBqV3IBocL5gAVmrqJV_mbaWPGGKOlQwVWeLjD4yeqq4j6RDovq6C5-AjnIAAO1DrBfAG1SpLvD1p-Rh9idO4VJid4lRaWyJ-hOxV2vn0pDJGuND3cMQ9M5uEPXYczzFDVy2qu8E4ufJSBwz6_qC-THYh7YTWrvzM-agxlXWhaAzXReFHdGtJzaOWHDxVeJysb4TCCO9aEZwcgJWmQK03ZVAtLBcnYmUIfZuAC4ljH0Etlw4aBfmjtTETFRFdvpGF4NydzKg_E5ZDEFMmI88UX5w
```



Workload Status

Deployments

	名称	镜像	标签	Pods	创建时间 ↑
●	nginx	nginx	k8s-app: nginx	4 / 4	2 minutes ago