一：k8s集群环境搭建：

k8s集群环境主要是kubernetes管理端服务(kube-apiserver、kube-controller-manager、kube-scheduler)的高可用实现，以及node节点上的(kubelet、kube-proxy)客户端服务的部署。

Kubernetes设计架构：

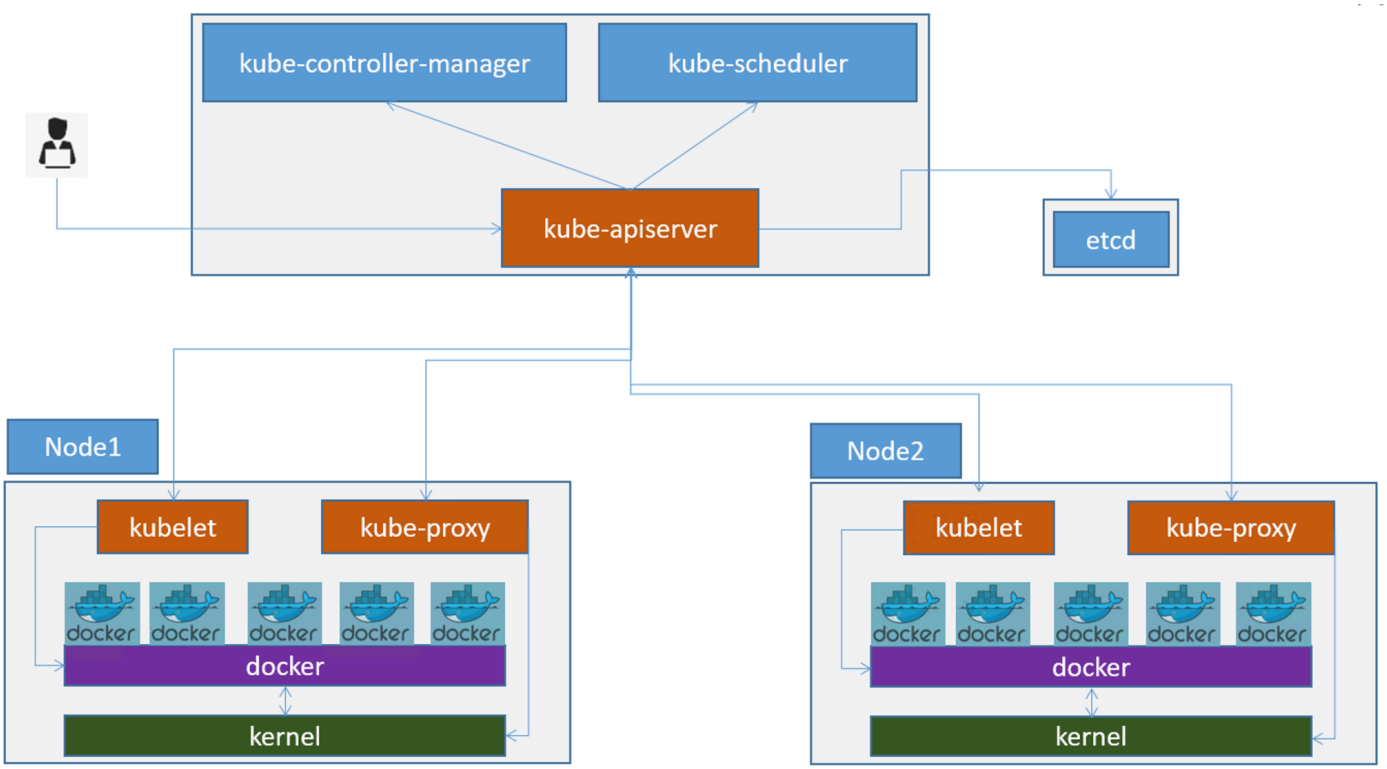
```
https://www.kubernetes.org.cn/kubernetes%E8%AE%BE%E8%AE%A1%E6%9E%B6%E6%9E%84
```

1.1：k8s高可用集群环境规划信息：

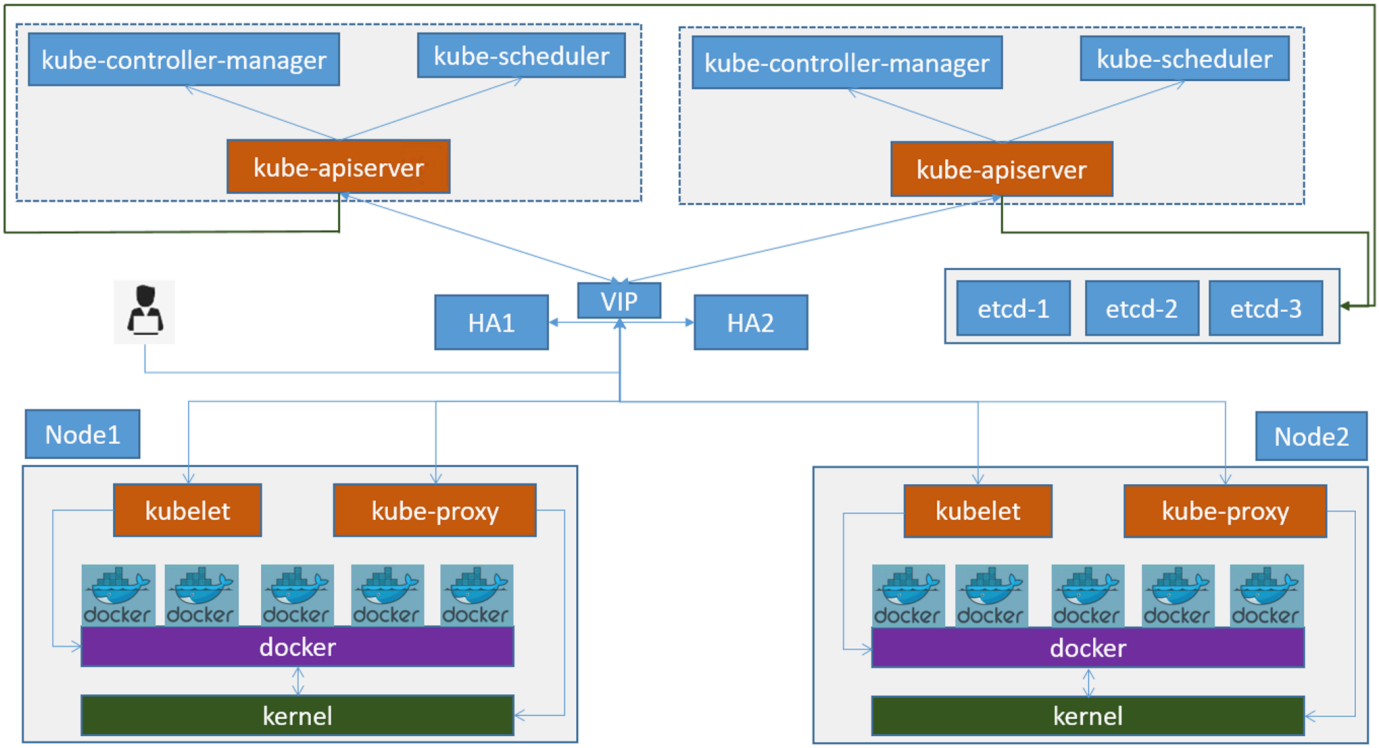
按照实际环境需求，进行规划与部署相应的单master或者多master的高可用k8s运行环境。

1.1.1：单master环境：

见 **kubeadm 安装k8s**



1.1.2：多master环境：



1.1.3：服务器统计：

类型	服务器IP地址	备注
Ansible(2台)	172.31.7.101/102	K8S集群部署服务器，可以和其他服务器混用
K8S Master(3台)	172.31.7.101/102/103	K8s控制端，通过一个VIP做主备高可用
Harbor(2台)	172.31.7.104/105	高可用镜像服务器
Etcd(最少3台)	172.31.7.106/107/108	保存k8s集群数据的服务器
Hproxy(2台)	172.31.7.109/110	高可用etcd代理服务器
Node节点(2-N台)	172.31.7.111/112/113/xxx	真正运行容器的服务器，高可用环境至少两台

1.2：服务器准备：

服务器可以是私有云的虚拟机或物理机，也可以是公有云环境的虚拟机环境，如果是公司托管的IDC环境，可以直接将harbor和node节点部署在物理机环境，master节点、etcd、负载均衡等可以是虚拟机。

类型	服务器IP	主机名	VIP
K8S Master1	172.31.7.101	k8s-master1.magedu.net	172.31.7.188
K8S Master2	172.31.7.102	k8s-master2.magedu.net	172.31.7.188
K8S Master3	172.31.7.103	k8s-master3.magedu.net	172.31.7.188
Harbor1	172.31.7.104	k8s-harbor1.magedu.net	
Harbor2	172.31.7.105	k8s-harbor2.magedu.net	
etcd节点1	172.31.7.106	k8s-etcd1.magedu.net	
etcd节点2	172.31.7.107	k8s-etcd2.magedu.net	
etcd节点3	172.31.7.108	k8s-etcd3.magedu.net	
Haproxy1	172.31.7.109	k8s-ha1.magedu.net	
Haproxy2	172.31.7.110	k8s-ha2.magedu.net	
Node节点1	172.31.7.111	k8s-node1.magedu.net	
Node节点2	172.31.7.112	k8s-node2.magedu.net	
Node节点3	172.31.7.113	k8s-node3.magedu.net	

1.3：k8s集群软件清单：

见当前目录下Excel文件 **kubernetes软件清单**

API端口：

```
端口：172.31.7.188:6443  #需要配置在负载均衡上实现反向代理
操作系统：ubuntu server 20.04.x
k8s版本：  1.21.x
```

1.4：基础环境准备：

系统主机名配置、IP配置、系统参数优化，以及依赖的负载均衡和Harbor部署

1.4.1：系统配置：

主机名、iptables、防火墙、内核参数与资源限制等系统配置略

1.4.2：高可用负载均衡：

k8s高可用反向代理

参见博客http://blogs.studylinux.net/?p=4579

1.4.2.1：keepalived：

```
root@k8s-ha1:~# cat /etc/keepalived/keepalived.conf
vrrp_instance VI_1 {
    state MASTER
    interface eth0
    virtual_router_id 1
    priority 100
    advert_int 3
    unicast_src_ip 172.31.7.101
    unicast_peer {
        172.31.7.110
    }

    authentication {
        auth_type PASS
        auth_pass 123abc
    }
    virtual_ipaddress {
        172.31.7.188 dev eth0 label eth0:1
    }
}
```

1.4.2.2：haproxy：

```
listen k8s_api_nodes_6443
    bind 172.31.7.188:6443
    mode tcp
    #balance leastconn
    server 172.31.7.101  172.31.7.101:6443  check inter 2000 fall 3 rise 5
    server 172.31.7.102  172.31.7.102:6443  check inter 2000 fall 3 rise 5
    server 172.31.7.103  172.31.7.103:6443  check inter 2000 fall 3 rise 5
```

1.4.3：安装docker：

各master及node节点安装docker。

```
root@k8s-node3:/usr/local/src# pwd
/usr/local/src

root@k8s-node3:/usr/local/src# tar xvf docker-19.03.15-binary-install.tar.gz
root@k8s-node3:/usr/local/src# bash docker-install.sh
root@k8s-node3:/usr/local/src# reboot
```

1.4.4：Harbor之https：

内部镜像将统一保存在内部Harbor服务器，不再通过互联网在线下载，为实现镜像的安全传输，harbor服务使用https上传与下载镜像。

```
root@k8s-harbor1:/apps# pwd
/apps

#解压
root@k8s-harbor1:/apps# tar xvf harbor-offline-installer-v2.3.2.tgz
root@k8s-harbor1:/apps/harbor# mkdir  certs
root@k8s-harbor1:/apps/harbor# cd certs/

#生成私有key
root@k8s-harbor1:/apps/harbor/certs# openssl genrsa -out harbor-ca.key
Generating RSA private key, 2048 bit long modulus (2 primes)
......+++++
.....................................................................+++++
e is 65537 (0x010001)

#签发证书
root@k8s-harbor1:/apps/harbor/certs# openssl req -x509 -new -nodes -key harbor-ca.key
-subj "/CN=harbor.magedu.net" -days 7120 -out harbor-ca.crt

#修改harbor配置文件
root@k8s-harbor1:/apps/harbor/certs# cd ..
root@k8s-harbor1:/apps/harbor# vim harbor.yml
root@k8s-harbor1:/apps/harbor# grep -v "#" harbor.yml | grep -v "^$"
hostname: harbor.magedu.net
http:
  port: 80
https:
  port: 443
  certificate: /apps/harbor/certs/harbor-ca.crt
  private_key: /apps/harbor/certs/harbor-ca.key
```

```
harbor_admin_password: 123456
database:
  password: root123
  max_idle_conns: 100
  max_open_conns: 900
data_volume: /data

root@k8s-harbor1:/apps/harbor# ./install.sh  --with-trivy
```

1.4.4.1:客户端节点同步证书：

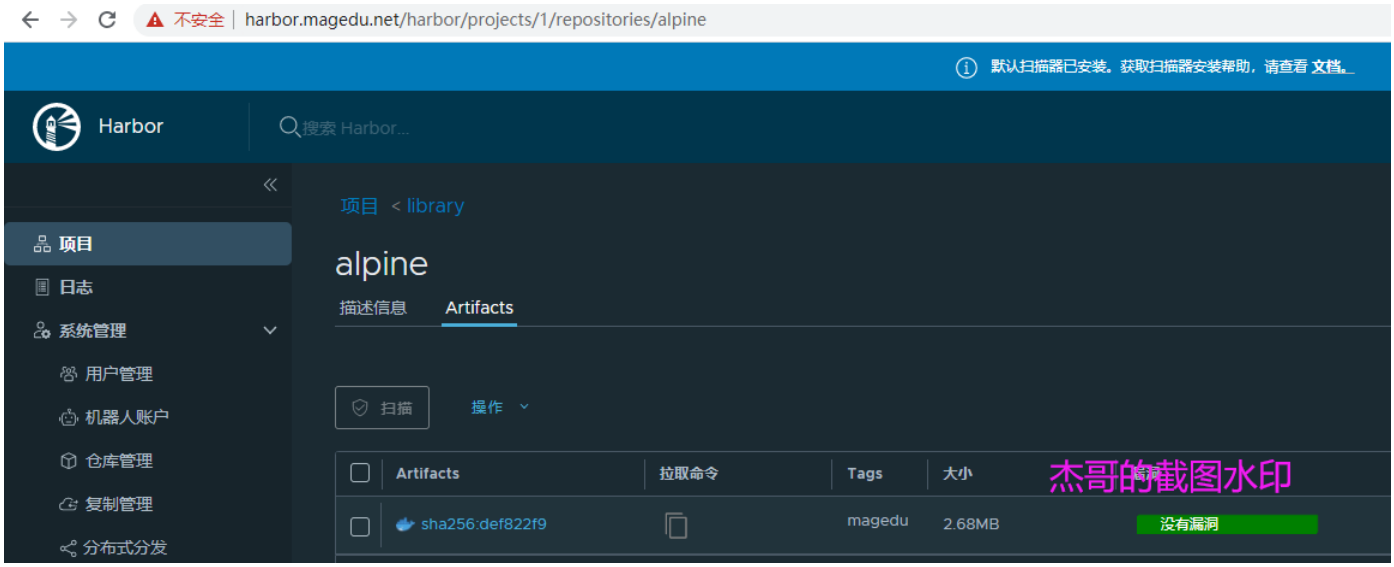
```
#同步证书
root@k8s-harbor2:~# mkdir   /etc/docker/certs.d/harbor.magedu.net -p
root@k8s-harbor1:/usr/local/src# scp /apps/harbor/certs/harbor-ca.crt
172.31.7.105:/etc/docker/certs.d/harbor.magedu.net

#添加host文件解析
root@k8s-harbor2:~# vim /etc/hosts
172.31.7.104 harbor.magedu.net

#重启docker
root@k8s-harbor2:~# systemctl  restart docker
```

1.4.4.2:测试登录harbor：

```
root@k8s-harbor2:~# docker login harbor.magedu.net
Username: admin
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

1.4.4.3:测试push镜像到harbor：

```
root@k8s-harbor2:~# docker pull alpine

root@k8s-harbor2:~# docker tag alpine harbor.magedu.net/library/alpine
root@k8s-harbor2:~# docker push harbor.magedu.net/library/alpine
The push refers to repository [harbor.magedu.net/library/alpine]
e2eb06d8af82: Pushed
latest: digest: sha256:69704ef328d05a9f806b6b8502915e6a0a4faa4d72018dc42343f511490daf8a
size: 528
```



1.5：手动二进制部署：

见 k8s 1.11 ubuntu1804部署文档

1.6：ansible部署：

基于ansible进安装部署kubernetes集群。

1.6.1：安装ansible：

```
部署节点安装ansible
root@k8s-master1:~# apt  install python3-pip git
root@k8s-master1:~# pip3 install ansible -i https://mirrors.aliyun.com/pypi/simple/
root@k8s-master1:~# ansible --version
ansible [core 2.11.4]
  config file = None
  configured module search path = ['/root/.ansible/plugins/modules',
'/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/local/lib/python3.8/dist-packages/ansible
  ansible collection location =
/root/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/local/bin/ansible
  python version = 3.8.10 (default, Jun  2 2021, 10:49:15) [GCC 9.4.0]
  jinja version = 2.10.1
  libyaml = True
```

1.6.2：配置免秘钥登录：

```
 #生成密钥对
root@k8s-master1:~# ssh-keygen

#安装sshpass命令用于同步公钥到各k8s服务器
# apt-get install sshpass


#分发公钥脚本：
root@k8s-master1:~# cat scp-key.sh
#!/bin/bash
#目标主机列表
IP="
172.31.7.101
172.31.7.102
172.31.7.103
172.31.7.106
172.31.7.107
172.31.7.108
172.31.7.111
172.31.7.112
172.31.7.113
"
```

```
for node in ${IP};do
  sshpass -p 123456 ssh-copy-id  ${node}  -o StrictHostKeyChecking=no
  if [ $? -eq 0 ];then
    echo "${node} 秘钥copy完成"
  else
    echo "${node} 秘钥copy失败"
  fi
done

root@k8s-master1:~# bash scp-key.sh
```

1.6.2：部署节点下载部署项目及组件：

使用harbor2作为部署节点

```
root@k8s-master1:~# export release=3.1.0
root@k8s-master1:~# curl -C- -fLO --retry 3
https://github.com/easzlab/kubeasz/releases/download/${release}/ezdown

https://hub.docker.com/r/easzlab/kubeasz-k8s-bin/tags?page=1&ordering=last_updated
root@k8s-master1:~# chmod  a+x ezdown
root@k8s-master1:~# vim ezdown
  # default settings, can be overridden by cmd line options, see usage
  DOCKER_VER=19.03.15
  KUBEASZ_VER=3.1.0
  K8S_BIN_VER=v1.21.0
root@k8s-master1:~# ./ezdown -D

root@k8s-master1:~# ll /etc/kubeasz/down/
total 1239424
drwxr-xr-x  2 root root      4096 Jun 10 11:20 ./
drwxrwxr-x 11 root root       209 Jun 10 11:11 ../
-rw-------  1 root root 451969024 Jun 10 11:14 calico_v3.15.3.tar
-rw-------  1 root root  42592768 Jun 10 11:15 coredns_1.8.0.tar
-rw-------  1 root root 227933696 Jun 10 11:17 dashboard_v2.2.0.tar
-rw-r--r--  1 root root  62436240 Jun 10 11:09 docker-19.03.15.tgz
-rw-------  1 root root  58150912 Jun 10 11:18 flannel_v0.13.0-amd64.tar
-rw-------  1 root root 124833792 Jun 10 11:16 k8s-dns-node-cache_1.17.0.tar
-rw-------  1 root root 179014144 Jun 10 11:20 kubeasz_3.1.0.tar
-rw-------  1 root root  34566656 Jun 10 11:19 metrics-scraper_v1.0.6.tar
-rw-------  1 root root  41199616 Jun 10 11:19 metrics-server_v0.3.6.tar
-rw-------  1 root root  45063680 Jun 10 11:20 nfs-provisioner_v4.0.1.tar
-rw-------  1 root root    692736 Jun 10 11:20 pause_3.4.1.tar
-rw-------  1 root root    692736 Jun 10 11:20 pause.tar
```

1.6.3：生成ansible hosts文件：

```
root@k8s-master1:~# cd /etc/kubeasz/
root@k8s-master1:/etc/kubeasz# pwd
/etc/kubeasz

root@k8s-master1:/etc/kubeasz# ./ezctl new k8s-01
2021-06-10 11:22:54 DEBUG generate custom cluster files in /etc/kubeasz/clusters/k8s-01
2021-06-10 11:22:54 DEBUG set version of common plugins
2021-06-10 11:22:54 DEBUG cluster k8s-01: files successfully created.
2021-06-10 11:22:54 INFO next steps 1: to config '/etc/kubeasz/clusters/k8s-01/hosts'
2021-06-10 11:22:54 INFO next steps 2: to config '/etc/kubeasz/clusters/k8s-
01/config.yml'

root@k8s-master1:/etc/kubeasz# vim  clusters/k8s-01/hosts
root@k8s-master1:/etc/kubeasz# vim clusters/k8s-01/config.yml
```

1.6.3.1：编辑ansible hosts文件：

指定etcd节点、master节点、node节点、VIP、运行时、网络组建类型、service IP与pod IP范围等配置信息。

```
root@k8s-master1:/etc/kubeasz# cat  /etc/kubeasz/clusters/k8s-01/hosts
# 'etcd' cluster should have odd member(s) (1,3,5,...)
[etcd]
172.31.7.106
172.31.7.107
172.31.7.108

# master node(s)
[kube_master]
172.31.7.101
172.31.7.102

# work node(s)
[kube_node]
172.31.7.111
172.31.7.112

# [optional] harbor server, a private docker registry
# 'NEW_INSTALL': 'true' to install a harbor server; 'false' to integrate with existed
one
[harbor]
#172.31.7.8 NEW_INSTALL=false

# [optional] loadbalance for accessing k8s from outside
[ex_lb]
172.31.7.109 LB_ROLE=backup EX_APISERVER_VIP=172.31.7.188 EX_APISERVER_PORT=6443
172.31.7.110 LB_ROLE=master EX_APISERVER_VIP=172.31.7.188 EX_APISERVER_PORT=6443
```

```
# [optional] ntp server for the cluster
[chrony]
#172.31.7.1

[all:vars]
# --------- Main Variables ---------------
# Secure port for apiservers
SECURE_PORT="6443"

# Cluster container-runtime supported: docker, containerd
CONTAINER_RUNTIME="docker"

# Network plugins supported: calico, flannel, kube-router, cilium, kube-ovn
CLUSTER_NETWORK="flannel"

# Service proxy mode of kube-proxy: 'iptables' or 'ipvs'
PROXY_MODE="ipvs"

# K8S Service CIDR, not overlap with node(host) networking
SERVICE_CIDR="10.00.0.0/16"

# Cluster CIDR (Pod CIDR), not overlap with node(host) networking
CLUSTER_CIDR="10.200.0.0/16"

# NodePort Range
NODE_PORT_RANGE="30000-65535"

# Cluster DNS Domain
CLUSTER_DNS_DOMAIN="magedu.local"

# -------- Additional Variables (don't change the default value right now) ---
# Binaries Directory
bin_dir="/usr/local/bin"

# Deploy Directory (kubeasz workspace)
base_dir="/etc/kubeasz"

# Directory for a specific cluster
cluster_dir="{{ base_dir }}/clusters/k8s-01"

# CA and other components cert/key Directory
ca_dir="/etc/kubernetes/ssl"
```

1.6.3.2：编辑config.yml 文件：

```
root@k8s-master1:/etc/kubeasz# cat  clusters/k8s-01/config.yml
############################
# prepare
############################
# 可选离线安装系统软件包 (offline|online)
INSTALL_SOURCE: "online"

# 可选进行系统安全加固 github.com/dev-sec/ansible-collection-hardening
OS_HARDEN: false

# 设置时间源服务器【重要：集群内机器时间必须同步】
ntp_servers:
  - "ntp1.aliyun.com"
  - "time1.cloud.tencent.com"
  - "0.cn.pool.ntp.org"

# 设置允许内部时间同步的网络段，比如"10.0.0.0/8"，默认全部允许
local_network: "0.0.0.0/0"


############################
# role:deploy
############################
# default: ca will expire in 100 years
# default: certs issued by the ca will expire in 50 years
CA_EXPIRY: "876000h"
CERT_EXPIRY: "438000h"

# kubeconfig 配置参数
CLUSTER_NAME: "cluster1"
CONTEXT_NAME: "context-{{ CLUSTER_NAME }}"


############################
# role:etcd
############################
# 设置不同的wal目录，可以避免磁盘io竞争，提高性能
ETCD_DATA_DIR: "/var/lib/etcd"
ETCD_WAL_DIR: ""


############################
# role:runtime [containerd,docker]
############################
# ------------------------------------------- containerd
# [.]启用容器仓库镜像
ENABLE_MIRROR_REGISTRY: true
```

```
# [containerd]基础容器镜像
SANDBOX_IMAGE: "easzlab/pause-amd64:3.4.1"

# [containerd]容器持久化存储目录
CONTAINERD_STORAGE_DIR: "/var/lib/containerd"

# ------------------------------------------- docker
# [docker]容器存储目录
DOCKER_STORAGE_DIR: "/var/lib/docker"

# [docker]开启Restful API
ENABLE_REMOTE_API: false

# [docker]信任的HTTP仓库
INSECURE_REG: '["127.0.0.1/8"]'


############################
# role:kube-master
############################
# k8s 集群 master 节点证书配置，可以添加多个ip和域名（比如增加公网ip和域名）
MASTER_CERT_HOSTS:
  - "10.1.1.1"
  - "k8s.test.io"
  #- "www.test.com"

# node 节点上 pod 网段掩码长度（决定每个节点最多能分配的pod ip地址）
# 如果flannel 使用 --kube-subnet-mgr 参数，那么它将读取该设置为每个节点分配pod网段
# https://github.com/coreos/flannel/issues/847
NODE_CIDR_LEN: 24


############################
# role:kube-node
############################
# Kubelet 根目录
KUBELET_ROOT_DIR: "/var/lib/kubelet"

# node节点最大pod 数
MAX_PODS: 300

# 配置为kube组件（kubelet,kube-proxy,dockerd等）预留的资源量
# 数值设置详见templates/kubelet-config.yaml.j2
KUBE_RESERVED_ENABLED: "yes"

# k8s 官方不建议草率开启 system-reserved, 除非你基于长期监控，了解系统的资源占用状况;
# 并且随着系统运行时间，需要适当增加资源预留，数值设置详见templates/kubelet-config.yaml.j2
# 系统预留设置基于 4c/8g 虚机，最小化安装系统服务，如果使用高性能物理机可以适当增加预留
```

```
# 另外，集群安装时候apiserver等资源占用会短时较大，建议至少预留1g内存
SYS_RESERVED_ENABLED: "no"

# haproxy balance mode
BALANCE_ALG: "roundrobin"


##############################
# role:network [flannel,calico,cilium,kube-ovn,kube-router]
##############################
# ------------------------------------------- flannel
# [flannel]设置flannel 后端"host-gw","vxlan"等
FLANNEL_BACKEND: "vxlan"
DIRECT_ROUTING: false

# [flannel] flanneld_image: "quay.io/coreos/flannel:v0.10.0-amd64"
flannelVer: "v0.13.0-amd64"
flanneld_image: "easzlab/flannel:{{ flannelVer }}"

# [flannel]离线镜像tar包
flannel_offline: "flannel_{{ flannelVer }}.tar"

# ------------------------------------------- calico
# [calico]设置 CALICO_IPV4POOL_IPIP="off",可以提高网络性能，条件限制详见 docs/setup/calico.md
CALICO_IPV4POOL_IPIP: "Always"

# [calico]设置 calico-node使用的host IP，bgp邻居通过该地址建立，可手工指定也可以自动发现
IP_AUTODETECTION_METHOD: "can-reach={{ groups['kube_master'][0] }}"

# [calico]设置calico 网络 backend: brid, vxlan, none
CALICO_NETWORKING_BACKEND: "brid"

# [calico]更新支持calico 版本: [v3.3.x] [v3.4.x] [v3.8.x] [v3.15.x]
calico_ver: "v3.15.3"

# [calico]calico 主版本
calico_ver_main: "{{ calico_ver.split('.')[0] }}.{{ calico_ver.split('.')[1] }}"

# [calico]离线镜像tar包
calico_offline: "calico_{{ calico_ver }}.tar"

# ------------------------------------------- cilium
# [cilium]CILIUM_ETCD_OPERATOR 创建的 etcd 集群节点数 1,3,5,7...
ETCD_CLUSTER_SIZE: 1

# [cilium]镜像版本
cilium_ver: "v1.4.1"

# [cilium]离线镜像tar包
```

```
cilium_offline: "cilium_{{ cilium_ver }}.tar"

# ------------------------------------------- kube-ovn
# [kube-ovn]选择 OVN DB and OVN Control Plane 节点，默认为第一个master节点
OVN_DB_NODE: "{{ groups['kube_master'][0] }}"

# [kube-ovn]离线镜像tar包
kube_ovn_ver: "v1.5.3"
kube_ovn_offline: "kube_ovn_{{ kube_ovn_ver }}.tar"

# ------------------------------------------- kube-router
# [kube-router]公有云上存在限制，一般需要始终开启 ipinip；自有环境可以设置为 "subnet"
OVERLAY_TYPE: "full"

# [kube-router]NetworkPolicy 支持开关
FIREWALL_ENABLE: "true"

# [kube-router]kube-router 镜像版本
kube_router_ver: "v0.3.1"
busybox_ver: "1.28.4"

# [kube-router]kube-router 离线镜像tar包
kuberouter_offline: "kube-router_{{ kube_router_ver }}.tar"
busybox_offline: "busybox_{{ busybox_ver }}.tar"


############################
# role:cluster-addon
############################
# coredns 自动安装
dns_install: "no" #是否自动配置dns缓存
corednsVer: "1.8.0"
ENABLE_LOCAL_DNS_CACHE: false
dnsNodeCacheVer: "1.17.0"
# 设置 local dns cache 地址
LOCAL_DNS_CACHE: "169.254.20.10"

# metric server 自动安装
metricsserver_install: "no"
metricsVer: "v0.3.6"

# dashboard 自动安装
dashboard_install: "no" #是否自动安装dashboard
dashboardVer: "v2.2.0"
dashboardMetricsScraperVer: "v1.0.6"

# ingress 自动安装
ingress_install: "no" #是否自动安装ingress
ingress_backend: "traefik"
```

```
traefik_chart_ver: "9.12.3"

# prometheus 自动安装
prom_install: "no" #是否自动安装prometheus
prom_namespace: "monitor"
prom_chart_ver: "12.10.6"

# nfs-provisioner 自动安装
nfs_provisioner_install: "no"
nfs_provisioner_namespace: "kube-system"
nfs_provisioner_ver: "v4.0.1"
nfs_storage_class: "managed-nfs-storage"
nfs_server: "192.168.1.10"
nfs_path: "/data/nfs"

############################
# role:harbor
############################
# harbor version, 完整版本号
HARBOR_VER: "v2.1.3"
HARBOR_DOMAIN: "harbor.yourdomain.com"
HARBOR_TLS_PORT: 8443

# if set 'false', you need to put certs named harbor.pem and harbor-key.pem in
directory 'down'
HARBOR_SELF_SIGNED_CERT: true

# install extra component
HARBOR_WITH_NOTARY: false
HARBOR_WITH_TRIVY: false
HARBOR_WITH_CLAIR: false
HARBOR_WITH_CHARTMUSEUM: true
```

1.6.4：部署k8s集群：

通过ansible脚本初始化环境及部署k8s 高可用集群

1.6.4.1：步骤1-基础环境初始化

```
root@k8s-master1:/etc/kubeasz# ./ezctl help setup
Usage: ezctl setup <cluster> <step>
available steps:
    01  prepare            to prepare CA/certs & kubeconfig & other system settings
    02  etcd               to setup the etcd cluster
    03  container-runtime  to setup the container runtime(docker or containerd)
    04  kube-master        to setup the master nodes
    05  kube-node          to setup the worker nodes
```

```
    06  network             to setup the network plugin
    07  cluster-addon       to setup other useful plugins
    90  all                 to run 01~07 all at once
    10  ex-lb               to install external loadbalance for accessing k8s from
outside
    11  harbor              to install a new harbor server or to integrate with an
existed one

examples: ./ezctl setup test-k8s 01  (or ./ezctl setup test-k8s prepare)
    ./ezctl setup test-k8s 02  (or ./ezctl setup test-k8s etcd)
          ./ezctl setup test-k8s all
          ./ezctl setup test-k8s 04 -t restart_master

root@k8s-master1:/etc/kubeasz# vim playbooks/01.prepare.yml #系统基础初始化主机配置
root@k8s-master1:/etc/kubeasz# ./ezctl setup k8s-01 01 #准备CA和基础系统设置
```

1.6.4.2：步骤2-部署etcd集群：

可更改启动脚本路径及版本等自定义配置

```
root@k8s-master1:/etc/kubeasz# ./ezctl help setup
root@k8s-master1:/etc/kubeasz# ./ezctl setup k8s-01 02 #部署etcd集群
```

验证各etcd节点服务状态：

```
root@etcd1:~# export NODE_IPS="172.31.7.106 172.31.7.107 172.31.7.108"

root@etcd1:~# for ip in ${NODE_IPS}; do   ETCDCTL_API=3 /usr/local/bin/etcdctl   --
endpoints=https://${ip}:2379    --cacert=/etc/kubernetes/ssl/ca.pem   --
cert=/etc/kubernetes/ssl/etcd.pem   --key=/etc/kubernetes/ssl/etcd-key.pem   endpoint
health; done
https://172.31.7.106:2379 is healthy: successfully committed proposal: took =
11.648312ms
https://172.31.7.107:2379 is healthy: successfully committed proposal: took =
11.742806ms
https://172.31.7.108:2379 is healthy: successfully committed proposal: took =
13.327166ms

注：以上返回信息表示etcd集群运行正常，否则异常！
```

1.6.4.3：部署docker:

node节点必须安装docker，docker可以自行使用yum或者安装安装也可以使用二进制安装，因此此步骤为可选步骤！

1.6.4.3.1：配置harbor客户端证书：

```
root@k8s-master1:/etc/kubeasz# mkdir /etc/docker/certs.d/harbor.magedu.net -p #创建证书保存目录
root@k8s-harbor1:~# scp /apps/harbor/certs/harbor-ca.crt 172.31.7.101:/etc/docker/certs.d/harbor.magedu.net/ #从harbor服务器拷贝证书

root@k8s-master1:/etc/kubeasz# cat  /etc/hosts #添加harbor域名解析
172.31.7.104 harbor.magedu.net

root@k8s-master1:/etc/kubeasz# systemctl  restart  docker #重启docker
root@k8s-master1:/etc/kubeasz# docker  login  harbor.magedu.net
Username: admin
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

1.6.4.3.2：同步harbor客户端证书：

```
#同步docker证书脚本：
root@k8s-master1:/etc/kubeasz# cat /root/scp-key.sh
#!/bin/bash
#目标主机列表
IP="
172.31.7.101
172.31.7.102
172.31.7.103
172.31.7.106
172.31.7.107
172.31.7.108
172.31.7.111
172.31.7.112
172.31.7.113
"

for node in ${IP};do
  sshpass -p 123456 ssh-copy-id  ${node}  -o StrictHostKeyChecking=no
  if [ $? -eq 0 ];then
    echo "${node} 秘钥copy完成"
    echo "${node} 秘钥copy完成,准备环境初始化....."
      ssh  ${node}  "mkdir /etc/docker/certs.d/harbor.magedu.net -p"
      echo "Harbor 证书目录创建成功!"
      scp /etc/docker/certs.d/harbor.magedu.net/harbor-ca.crt ${node}:/etc/docker/certs.d/harbor.magedu.net/harbor-ca.crt
      echo "Harbor 证书拷贝成功!"
      ssh ${node}   "echo "172.31.7.104 harbor.magedu.net" >> /etc/hosts"
```

```
      echo "host 文件拷贝完成"
      #scp -r  /root/.docker  ${node}:/root/
      #echo "Harbor 认证文件拷贝完成!"
  else
    echo "${node} 秘钥copy失败"
  fi
done

#执行脚本进行证书分发
root@k8s-master1:/etc/kubeasz# bash /root/scp-key.sh
```

1.6.4.3.3：测试上传pause镜像：

```
# 基础容器镜像
root@k8s-master1:/etc/kubeasz# grep SANDBOX_IMAGE ./clusters/* -R
./clusters/k8s-01/config.yml:SANDBOX_IMAGE: "easzlab/pause-amd64:3.4.1"


root@k8s-master1:/etc/kubeasz# docker pull easzlab/pause-amd64:3.4.1

root@k8s-master1:/etc/kubeasz# docker tag easzlab/pause-amd64:3.4.1
harbor.magedu.net/baseimages/pause-amd64:3.4.1
root@k8s-master1:/etc/kubeasz# docker push harbor.magedu.net/baseimages/pause-
amd64:3.4.1

root@k8s-master1:/etc/kubeasz# vim ./clusters/k8s-01/config.yml
 48 # [containerd]基础容器镜像
 49 SANDBOX_IMAGE: "harbor.magedu.net/baseimages/pause-amd64:3.4.1"

root@k8s-master1:/etc/kubeasz# ./ezctl setup k8s-01 03

#node节点验证docker：
root@k8s-node2:~# docker version
Client: Docker Engine - Community
 Version:           19.03.15
 API version:       1.40
 Go version:        go1.13.15
 Git commit:        99e3ed8
 Built:             Sat Jan 30 03:11:43 2021
 OS/Arch:           linux/amd64
 Experimental:      false

Server: Docker Engine - Community
 Engine:
  Version:          19.03.15
  API version:      1.40 (minimum version 1.12)
  Go version:       go1.13.15
  Git commit:       99e3ed8
  Built:            Sat Jan 30 03:18:13 2021
```

```
 OS/Arch:          linux/amd64
 Experimental:     false
containerd:
 Version:          v1.3.9
 GitCommit:        ea765aba0d05254012b0b9e595e995c09186427f
runc:
 Version:            1.0.0-rc10
 GitCommit:        dc9208a3303feef5b3839f4323d9beb36df0a9dd
docker-init:
 Version:          0.18.0
 GitCommit:        fec3683
```

1.6.4.4：部署master节点：

可选更改启动脚本参数及路径等自定义功能

```
root@k8s-master1:/etc/kubeasz# ./ezctl  help setup
root@k8s-master1:/etc/kubeasz# ./ezctl setup k8s-01 04

#验证服务器

root@k8s-master1:/etc/kubeasz# kubectl get node
NAME           STATUS                     ROLES    AGE   VERSION
172.31.7.101   Ready,SchedulingDisabled   master   25s   v1.21.0
172.31.7.102   Ready,SchedulingDisabled   master   25s   v1.21.0
```

1.6.4.5：部署node节点：

node节点必须安装docker

```
root@k8s-master1:/etc/kubeasz# ./ezctl setup k8s-01 05

#验证服务器
root@k8s-master1:/etc/kubeasz# kubectl  get node
NAME           STATUS                     ROLES    AGE    VERSION
172.31.7.101   Ready,SchedulingDisabled   master   3m4s   v1.21.0
172.31.7.102   Ready,SchedulingDisabled   master   3m4s   v1.21.0
172.31.7.111   Ready                      node     23s    v1.21.0
172.31.7.112   Ready                      node     23s    v1.21.0
```

```
TASK [kube-node : 轮询等待node达到Ready状态] ********************************************************************************
changed: [172.31.7.111]
changed: [172.31.7.112]

TASK [kube-node : 设置node节点role] ****************************************************************************************
changed: [172.31.7.111]
changed: [172.31.7.112]

PLAY RECAP ******************************************************************************************************************
172.31.7.111               : ok=37   changed=34   unreachable=0    failed=0    skipped=5    rescued=0    ignored=0
172.31.7.112               : ok=35   changed=33   unreachable=0    failed=0    skipped=5    rescued=0    ignored=0

root@k8s-master1:/etc/kubeasz# kubectl  get node
root@k8s-master1:/etc/kubeasz# kubectl  get node
NAME           STATUS                     ROLES    AGE    VERSION
172.31.7.101   Ready,SchedulingDisabled   master   3m4s   v1.21.0
172.31.7.102   Ready,SchedulingDisabled   master   3m4s   v1.21.0
172.31.7.111   Ready                      node     23s    v1.21.0
172.31.7.112   Ready                      node     23s    v1.21.0
root@k8s-master1:/etc/kubeasz#
```

马哥教育，杰哥的截图水印

1.6.4.5：部署网络服务：

网络组件可以使用calico或者flannel。

可选更改calico服务启动脚本路径，csr证书信息

1.6.4.5.1：使用calico网络组件：

```
root@k8s-master1:/etc/kubeasz# vim ./clusters/k8s-01/config.yml

# ---------------------------------------- calico
# [calico]设置 CALICO_IPV4POOL_IPIP="off",可以提高网络性能，条件限制详见 docs/setup/calico.md
CALICO_IPV4POOL_IPIP: "Always"

# [calico]设置 calico-node使用的host IP，bgp邻居通过该地址建立，可手工指定也可以自动发现
IP_AUTODETECTION_METHOD: "can-reach={{ groups['kube_master'][0] }}"

# [calico]设置calico 网络 backend: brid, vxlan, none
CALICO_NETWORKING_BACKEND: "brid"

# [calico]更新支持calico 版本: [v3.3.x] [v3.4.x] [v3.8.x] [v3.15.x]
calico_ver: "v3.15.3"


root@k8s-master1:/etc/kubeasz# grep image  roles/calico/templates/calico-v3.15.yaml.j2
          image: calico/cni:v3.15.3
          image: calico/pod2daemon-flexvol:v3.15.3
          image: calico/node:v3.15.3
          image: calico/kube-controllers:v3.15.3


root@k8s-master1:/etc/kubeasz# docker pull calico/cni:v3.15.3
root@k8s-master1:/etc/kubeasz# docker tag calico/cni:v3.15.3
harbor.magedu.net/baseimages/calico-cni:v3.15.3
root@k8s-master1:/etc/kubeasz# docker push harbor.magedu.net/baseimages/calico-
cni:v3.15.3
```

```
root@k8s-master1:/etc/kubeasz# docker pull calico/pod2daemon-flexvol:v3.15.3
root@k8s-master1:/etc/kubeasz# docker tag docker.io/calico/pod2daemon-flexvol:v3.15.3
harbor.magedu.net/baseimages/calico-pod2daemon-flexvol:v3.15.3
root@k8s-master1:/etc/kubeasz# docker push harbor.magedu.net/baseimages/calico-
pod2daemon-flexvol:v3.15.3


root@k8s-master1:/etc/kubeasz# docker pull calico/node:v3.15.3
root@k8s-master1:/etc/kubeasz# docker tag  calico/node:v3.15.3
harbor.magedu.net/baseimages/calico-node:v3.15.3
root@k8s-master1:/etc/kubeasz# docker push harbor.magedu.net/baseimages/calico-
node:v3.15.3

root@k8s-master1:/etc/kubeasz# docker pull calico/kube-controllers:v3.15.3
root@k8s-master1:/etc/kubeasz# docker tag calico/kube-controllers:v3.15.3
harbor.magedu.net/baseimages/calico-kube-controllers:v3.15.3
root@k8s-master1:/etc/kubeasz# docker push harbor.magedu.net/baseimages/calico-kube-
controllers:v3.15.3

修改镜像地址：
root@k8s-master1:/etc/kubeasz# vim roles/calico/templates/calico-v3.15.yaml.j2
root@k8s-master1:/etc/kubeasz# grep image  roles/calico/templates/calico-v3.15.yaml.j2
          image: harbor.magedu.net/baseimages/calico-cni:v3.15.3
          image: harbor.magedu.net/baseimages/calico-pod2daemon-flexvol:v3.15.3
          image: harbor.magedu.net/baseimages/calico-node:v3.15.3
          image: harbor.magedu.net/baseimages/calico-kube-controllers:v3.15.3


root@k8s-master1:/etc/kubeasz# ./ezctl  help setup
root@k8s-master1:/etc/kubeasz# ./ezctl setup k8s-01 06
```

```
TASK [calico : 下载calicoctl 客户端] ********************************************************
*****
ok: [172.31.7.101] => (item=calicoctl)
ok: [172.31.7.102] => (item=calicoctl)
ok: [172.31.7.111] => (item=calicoctl)
ok: [172.31.7.112] => (item=calicoctl)

TASK [calico : 准备 calicoctl配置文件] ******************************************************
******
ok: [172.31.7.101]
ok: [172.31.7.102]
ok: [172.31.7.111]
ok: [172.31.7.112]

TASK [calico : 轮询等待calico-node 运行，视下载镜像速度而定] ********************************
*****************
changed: [172.31.7.101]
changed: [172.31.7.102]
changed: [172.31.7.111]
changed: [172.31.7.112]

PLAY RECAP ***********************************************************************************
172.31.7.101               : ok=16   changed=8    unreachable=0    failed=0    skipped=52
172.31.7.102               : ok=12   changed=5    unreachable=0    failed=0    skipped=40
172.31.7.111               : ok=12   changed=5    unreachable=0    failed=0    skipped=40
172.31.7.112               : ok=12   changed=5    unreachable=0    failed=0    skipped=40

root@k8s-master1:/etc/kubeasz#
```


验证calico:

```
root@k8s-master1:/etc/kubeasz# calicoctl  node status
Calico process is running.

IPv4 BGP status
+--------------+-------------------+-------+----------+-------------+
| PEER ADDRESS |     PEER TYPE     | STATE |  SINCE   |    INFO     |
+--------------+-------------------+-------+----------+-------------+
| 172.31.7.111 | node-to-node mesh | up    | 07:19:07 | Established |
| 172.31.7.102 | node-to-node mesh | up    | 07:19:10 | Established |
| 172.31.7.112 | node-to-node mesh | up    | 07:19:13 | Established |
+--------------+-------------------+-------+----------+-------------+

IPv6 BGP status
No IPv6 peers found.
```

1.6.4.5.2：使用flannel网络组件：

```
############################
# role:network [flannel,calico,cilium,kube-ovn,kube-router]
############################
# ------------------------------------------- flannel
# [flannel]设置flannel 后端"host-gw","vxlan"等
```

```
FLANNEL_BACKEND: "vxlan"
DIRECT_ROUTING: false

# [flannel] flanneld_image: "quay.io/coreos/flannel:v0.10.0-amd64"
flannelVer: "v0.13.0-amd64"
flanneld_image: "easzlab/flannel:{{ flannelVer }}"

# [flannel]离线镜像tar包
flannel_offline: "flannel_{{ flannelVer }}.tar"
```

1.6.4.6：创建容器测试网络通信：

```
root@k8s-master1:/etc/kubeasz# docker pull alpine
root@k8s-master1:/etc/kubeasz# docker tag alpine harbor.magedu.net/baseimages/alpine
root@k8s-master1:/etc/kubeasz# docker push  harbor.magedu.net/baseimages/alpine

#创建pod测试夸主机网络通信是否正常
root@k8s-master1:/etc/kubeasz# kubectl run net-test1  --
image=harbor.magedu.net/baseimages/alpine   sleep 360000
pod/net-test1 created
root@k8s-master1:/etc/kubeasz# kubectl run net-test2  --
image=harbor.magedu.net/baseimages/alpine   sleep 360000
pod/net-test2 created
root@k8s-master1:/etc/kubeasz# kubectl run net-test3  --
image=harbor.magedu.net/baseimages/alpine   sleep 360000
pod/net-test3 created
```

```
root@k8s-master1:/etc/kubeasz# kubectl  get pod -o wide
NAME        READY   STATUS    RESTARTS   AGE   IP           NODE           NOMINATED NODE   READINESS GATES
net-test1   1/1     Running   0          17m   10.200.3.2   172.31.7.112   <none>           <none>
net-test2   1/1     Running   0          17m   10.200.2.2   172.31.7.111   <none>           <none>
net-test3   1/1     Running   0          17m   10.200.3.3   172.31.7.112   <none>           <none>
root@k8s-master1:/etc/kubeasz# kubectl  exec -it net-test1 sh
kubectl exec [POD] [COMMAND] is DEPRECATED and will be removed in a future version. Use kubectl exec [POD] -- [COMMAND] instead.
/ # ping 10.200.3.3
PING 10.200.3.3 (10.200.3.3): 56 data bytes
64 bytes from 10.200.3.3: seq=0 ttl=64 time=0.184 ms
64 bytes from 10.200.3.3: seq=1 ttl=64 time=0.115 ms
^C
--- 10.200.3.3 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.115/0.149/0.184 ms
/ # ping 223.6.6.6
PING 223.6.6.6 (223.6.6.6): 56 data bytes
64 bytes from 223.6.6.6: seq=0 ttl=127 time=6.074 ms
64 bytes from 223.6.6.6: seq=1 ttl=127 time=6.470 ms
^C
--- 223.6.6.6 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 6.074/6.272/6.470 ms
/ #
```

- 马哥教育，杰哥的截图水印

1.6.5：集群管理：

集群管理主要是添加master、添加node、删除master与删除node等节点管理及监控

当前集群状态：

```
root@k8s-master1:/etc/kubeasz# kubectl  get node
NAME            STATUS                     ROLES    AGE   VERSION
172.31.7.101    Ready,SchedulingDisabled   master   31m   v1.21.0
172.31.7.102    Ready,SchedulingDisabled   master   31m   v1.21.0
172.31.7.111    Ready                      node     28m   v1.21.0
172.31.7.112    Ready                      node     28m   v1.21.0
root@k8s-master1:/etc/kubeasz#
root@k8s-master1:/etc/kubeasz#
root@k8s-master1:/etc/kubeasz#
```


1.6.5.1：添加node节点：

```
root@k8s-master1:/etc/kubeasz# ./ezctl  help
root@k8s-master1:/etc/kubeasz# ./ezctl  add-node k8s-01 172.31.7.113
```

```
TASK [flannel : 配置 flannel DaemonSet yaml文件] ********************************************************************
ok: [172.31.7.113]

TASK [flannel : 运行 flannel网络] ***********************************************************************************
changed: [172.31.7.113]

TASK [flannel : 删除默认cni配置] ************************************************************************************
changed: [172.31.7.113]
FAILED - RETRYING: 轮询等待flannel 运行，视下载镜像速度而定 (15 retries left).

TASK [flannel : 轮询等待flannel 运行，视下载镜像速度而定] ***********************************************************
changed: [172.31.7.113]

TASK [推送cluster-addon的离线镜像包] ********************************************************************************
changed: [172.31.7.113] => (item=/etc/kubeasz/down/coredns_1.8.0.tar)
changed: [172.31.7.113] => (item=/etc/kubeasz/down/dashboard_v2.2.0.tar)
changed: [172.31.7.113] => (item=/etc/kubeasz/down/metrics-scraper_v1.0.6.tar)
changed: [172.31.7.113] => (item=/etc/kubeasz/down/metrics-server_v0.3.6.tar)

TASK [导入离线镜像（若执行失败，可忽略）] ***************************************************************************
changed: [172.31.7.113]

PLAY RECAP ***********************************************************************************************************
172.31.7.113               : ok=89   changed=76   unreachable=0    failed=0    skipped=192  rescued=0    ignored=0

root@k8s-master1:/etc/kubeasz#
```


验证node节点信息：

```
root@k8s-master1:/etc/kubeasz# kubectl  get node
NAME            STATUS                     ROLES    AGE     VERSION
172.31.7.101    Ready,SchedulingDisabled   master   37m     v1.21.0
172.31.7.102    Ready,SchedulingDisabled   master   37m     v1.21.0
172.31.7.111    Ready                      node     34m     v1.21.0
172.31.7.112    Ready                      node     34m     v1.21.0
172.31.7.113    Ready                      node     2m30s   v1.21.0
root@k8s-master1:/etc/kubeasz#
```

1.6.5.2：添加master节点：

```
root@k8s-master1:/etc/kubeasz# ./ezctl  help
root@k8s-master1:/etc/kubeasz# ./ezctl  add-master k8s-01 172.31.7.103
```

```
PLAY [kube_node] ***************************************************************************************************

TASK [kube-lb : 创建kube-lb的配置文件] ******************************************************************************
changed: [172.31.7.113]
changed: [172.31.7.111]
changed: [172.31.7.112]

TASK [kube-lb : 创建kube-lb的systemd unit文件] **********************************************************************
ok: [172.31.7.113]
ok: [172.31.7.111]
ok: [172.31.7.112]

TASK [kube-lb : 开启kube-lb服务] ************************************************************************************
changed: [172.31.7.113]
changed: [172.31.7.111]
changed: [172.31.7.112]

TASK [kube-lb : 以轮询的方式等待kube-lb服务启动] ********************************************************************
changed: [172.31.7.113]
changed: [172.31.7.111]
changed: [172.31.7.112]
```


1.6.5.3：验证当前节点：

```
root@k8s-master1:/etc/kubeasz# kubectl  get node
NAME           STATUS                     ROLES    AGE     VERSION
172.31.7.101   Ready,SchedulingDisabled   master   44m     v1.21.0
172.31.7.102   Ready,SchedulingDisabled   master   44m     v1.21.0
172.31.7.103   Ready,SchedulingDisabled   master   2m19s   v1.21.0
172.31.7.111   Ready                      node     42m     v1.21.0
172.31.7.112   Ready                      node     42m     v1.21.0
172.31.7.113   Ready                      node     9m56s   v1.21.0
root@k8s-master1:/etc/kubeasz#
```


1.6.5.4：验证网络组件calico状态：

```
root@k8s-master1:/etc/kubeasz# calicoctl  node status
Calico process is running.

IPv4 BGP status
+--------------+-------------------+-------+----------+-------------+
| PEER ADDRESS |     PEER TYPE     | STATE |  SINCE   |    INFO     |
+--------------+-------------------+-------+----------+-------------+
| 172.31.7.102 | node-to-node mesh | up    | 06:07:17 | Established |
| 172.31.7.103 | node-to-node mesh | up    | 06:10:09 | Established |
| 172.31.7.111 | node-to-node mesh | up    | 06:08:44 | Established |
| 172.31.7.112 | node-to-node mesh | up    | 06:09:29 | Established |
| 172.31.7.113 | node-to-node mesh | up    | 06:08:00 | Established |
+--------------+-------------------+-------+----------+-------------+

IPv6 BGP status
No IPv6 peers found.

root@k8s-master1:/etc/kubeasz#
```

1.6.5.5：验证node节点路由：

```
root@k8s-node1:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         172.31.7.254    0.0.0.0         UG    0      0        0 eth0
10.100.39.64    0.0.0.0         255.255.255.192 U     0      0        0 *
10.100.39.65    0.0.0.0         255.255.255.255 UH    0      0        0 cali2b2e7c9e43e
10.100.142.128  172.31.7.112    255.255.255.192 UG    0      0        0 eth0
172.17.0.0      0.0.0.0         255.255.0.0     U     0      0        0 docker0
172.31.0.0      0.0.0.0         255.255.248.0   U     0      0        0 eth0
root@k8s-node1:~#
```


1.7：DNS服务:

目前在kubernetes中常用的dns组件有kube-dns和coredns两个，kube-dns和coredns用于解析k8s集群中service name所对应得到IP地址，在k8s版本 1.17.x和之前的版本都可以使用kube-dns，但是k8s 在1.18之后不再支持kube-dns。

https://console.cloud.google.com/gcr/images/google-containers/GLOBAL #google的镜像仓库地址

1.7.1：部署kube-dns：

k8s 1.18版本以后将不再支持kube-dns。

https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.18.md#downloads-for-v1180

kubeadm: kube-dns is deprecated and will not be supported in a future version

```
1.skyDNS/kube-dns/coreDNS
kube-dns：提供service name域名的解析
dns-dnsmasq：提供DNS缓存，降低kubedns负载，提高性能
dns-sidecar：定期检查kubedns和dnsmasq的健康状态

2.导入镜像并上传至本地harbor
# docker load -i k8s-dns-kube-dns-amd64_1.14.13.tar.gz
# docker images
# docker tag gcr.io/google-containers/k8s-dns-kube-dns-amd64:1.14.13
harbor.magedu.net/baseimages/k8s-dns-kube-dns-amd64:1.14.13
# docker push harbor.magedu.net/baseimages/k8s-dns-kube-dns-amd64:1.14.13


# docker load -i  k8s-dns-sidecar-amd64_1.14.13.tar.gz
# docker images
# docker tag gcr.io/google-containers/k8s-dns-sidecar-amd64:1.14.13
harbor.magedu.net/baseimages/k8s-dns-sidecar-amd64:1.14.13
# docker push harbor.magedu.net/baseimages/k8s-dns-sidecar-amd64:1.14.13

# docker load -i k8s-dns-dnsmasq-nanny-amd64_1.14.13.tar.gz
```

```
# docker imgaes
# docker  tag gcr.io/google-containers/k8s-dns-dnsmasq-nanny-amd64:1.14.13
harbor.magedu.net/baseimages/k8s-dns-dnsmasq-nanny-amd64:1.14.13
# docker push harbor.magedu.net/baseimages/k8s-dns-dnsmasq-nanny-amd64:1.14.13

3.修改yaml文件中的镜像地址为本地harbor地址
# vim kube-dns-magedu.yaml
      - name: kubedns
        image: harbor.magedu.net/baseimages/k8s-dns-kube-dns-amd64:1.14.13

      - name: dnsmasq
        image: harbor.magedu.net/baseimages/k8s-dns-dnsmasq-nanny-amd64:1.14.13

      - name: sidecar
        image: harbor.magedu.net/baseimages/k8s-dns-sidecar-amd64:1.14.13

apiVersion: v1
kind: Service
metadata:
  name: kube-dns
  namespace: kube-system
  labels:
    k8s-app: kube-dns
    kubernetes.io/cluster-service: "true"
    addonmanager.kubernetes.io/mode: Reconcile
    kubernetes.io/name: "KubeDNS"
spec:
  selector:
    k8s-app: kube-dns
  clusterIP: 10.200.0.2 #自定义kube-dns的service IP


4.创建服务
# kubectl apply -f  kube-dns-magedu.yaml
service/kube-dns created
serviceaccount/kube-dns created
configmap/kube-dns created
deployment.apps/kube-dns created
```

1.7.2：测试域名解析测试：

```
# docker load -i busybox-online.tar.gz
# docker tag quay.io/prometheus/busybox:latest
harbor.magedu.net/baseimages/busybox:latest
# docker push harbor.magedu.net/baseimages/busybox:latest
# kubectl apply -f  busybox.yaml

如果启用了缓存，需要启用缓存容器，如果没有缓存次步骤可忽略。
```

```
# grep 169.254.20.10 /etc/kubeasz/ -R  #DNS缓存
/etc/kubeasz/example/config.yml:LOCAL_DNS_CACHE: "169.254.20.10"
Binary file /etc/kubeasz/down/kubeasz_3.1.0.tar matches
/etc/kubeasz/clusters/k8s-01/yml/nodelocaldns.yaml:        bind 169.254.20.10
/etc/kubeasz/clusters/k8s-01/yml/nodelocaldns.yaml:        health 169.254.20.10:8080
/etc/kubeasz/clusters/k8s-01/yml/nodelocaldns.yaml:        bind 169.254.20.10
/etc/kubeasz/clusters/k8s-01/yml/nodelocaldns.yaml:        bind 169.254.20.10
/etc/kubeasz/clusters/k8s-01/yml/nodelocaldns.yaml:        bind 169.254.20.10
/etc/kubeasz/clusters/k8s-01/yml/nodelocaldns.yaml:        args: [ "-localip",
"169.254.20.10", "-conf", "/etc/Corefile", "-upstreamsvc", "kube-dns-upstream" ]
/etc/kubeasz/clusters/k8s-01/yml/nodelocaldns.yaml:            host: 169.254.20.10
/etc/kubeasz/clusters/k8s-01/config.yml:LOCAL_DNS_CACHE: "169.254.20.10"


# docker pull  easzlab/k8s-dns-node-cache:1.16.0
# docker tag  easzlab/k8s-dns-node-cache:1.16.0  harbor.magedu.net/baseimages/k8s-dns-
node-cache:1.16.0
# docker push  harbor.magedu.net/baseimages/k8s-dns-node-cache:1.16.0
# pwd
/etc/kubeasz/clusters/k8s-01/yml
# vim  nodelocaldns.yaml
image: harbor.magedu.net/baseimages/k8s-dns-node-cache:1.16.0

# kubectl apply -f nodelocaldns.yaml
root@k8s-master1:~/coredns/kube-dns# pwd
/root/coredns/kube-dns
root@k8s-master1:~/coredns/kube-dns# kubectl apply -f busybox.yaml


测速域名解析
# kubectl exec busybox nslookup kubernetes
kubectl exec [POD] [COMMAND] is DEPRECATED and will be removed in a future version. Use
kubectl exec [POD] -- [COMMAND] instead.
Server:    169.254.20.10
Address 1: 169.254.20.10

Name:      kubernetes
Address 1: 10.200.0.1 kubernetes.default.svc.jie.local

root@k8s-master1:~/coredns/kube-dns#  kubectl exec busybox nslookup
kubernetes.default.svc.jie.local
kubectl exec [POD] [COMMAND] is DEPRECATED and will be removed in a future version. Use
kubectl exec [POD] -- [COMMAND] instead.
Server:    169.254.20.10
Address 1: 169.254.20.10

Name:      kubernetes.default.svc.jie.local
Address 1: 10.200.0.1 kubernetes.default.svc.jie.local
```

```
#root@k8s-master1:~/coredns/kube-dns# kubectl exec busybox nslookup kube-dns.kube-system.svc.jie.local
kubectl exec [POD] [COMMAND] is DEPRECATED and will be removed in a future version. Use kubectl exec [POD] -- [COMMAND] instead.
Server:    169.254.20.10
Address 1: 169.254.20.10

Name:      kube-dns.kube-system.svc.jie.local
Address 1: 10.200.0.2 kube-dns.kube-system.svc.jie.local
```

1.7.3：部署coredns：

将kube-dns更换为coredns

https://github.com/coredns/coredns

coredns 1.2/1.3/1.4/1.5版本：

```
# docker tag gcr.io/google-containers/coredns:1.2.6 harbor.magedu.net/baseimages/coredns:1.2.6
# docker push harbor.magedu.net/baseimages/coredns:1.2.6
```

1.6及以上新版本：

https://github.com/coredns/deployment/tree/master/kubernetes #1.6部署方式

1.7.3.1：主要配置参数：

```
error:  #错误日志输出到stdout。
health:  #CoreDNS的运行状况报告为http://localhost:8080/health.
cache:  #启用coredns缓存。
reload:#配置自动重新加载配置文件，如果修改了ConfigMap的配置，会在两分钟后生效.
loadbalance:#一个域名有多个记录会被轮询解析。
cache 30 #缓存时间
kubernetes:#CoreDNS将根据指定的service domain名称在Kubernetes SVC中进行域名解析。
forward:  #不是Kubernetes集群域内的域名查询都进行转发指定的服务器（/etc/resolv.conf）
prometheus:#CoreDNS的指标数据可以配置Prometheus 访问http://coredns svc:9153/metrics 进行收集。
ready:#当coredns 服务启动完成后会进行在状态监测，会有个URL 路径为/ready返回200状态码，否则返回报错。
```

1.7.3.2：部署coredns：

```
# unzip deployment-master.zip
# cd deployment-master/kubernetes/
# cp coredns.yaml.sed  coredns-1.8.3.yaml #手动编辑yaml文件
# ./deploy.sh > coredns-jie.yaml  #此步骤依赖于提前部署好DNS服务，才能自动生成yaml文件，如果没有提前部署过dns服务可以手动配置coredns的yaml文件。

# cat  coredns-jie.yaml
root@k8s-master1:/etc/kubeasz# cat coredns.yaml
apiVersion: v1
kind: ServiceAccount
metadata:
  name: coredns
  namespace: kube-system
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  labels:
    kubernetes.io/bootstrapping: rbac-defaults
  name: system:coredns
rules:
  - apiGroups:
    - ""
    resources:
    - endpoints
    - services
    - pods
    - namespaces
    verbs:
    - list
    - watch
  - apiGroups:
    - discovery.k8s.io
    resources:
    - endpointslices
    verbs:
    - list
    - watch
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  annotations:
    rbac.authorization.kubernetes.io/autoupdate: "true"
  labels:
    kubernetes.io/bootstrapping: rbac-defaults
  name: system:coredns
```

```
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: system:coredns
subjects:
- kind: ServiceAccount
  name: coredns
  namespace: kube-system
---
apiVersion: v1
kind: ConfigMap
metadata:
  name: coredns
  namespace: kube-system
data:
  Corefile: |
    .:53 {
        errors
        health {
          lameduck 5s
        }
        bind 0.0.0.0
        ready
        kubernetes magedu.local in-addr.arpa ip6.arpa {
          fallthrough in-addr.arpa ip6.arpa
        }
        prometheus :9153
        forward . /etc/resolv.conf {
          max_concurrent 1000
        }
        cache 30
        loop
        reload
        loadbalance
    }
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: coredns
  namespace: kube-system
  labels:
    k8s-app: kube-dns
    kubernetes.io/name: "CoreDNS"
spec:
  # replicas: not specified here:
  # 1. Default is 1.
  # 2. Will be tuned in real time if DNS horizontal auto-scaling is turned on.
  strategy:
```

```
    type: RollingUpdate
    rollingUpdate:
      maxUnavailable: 1
  selector:
    matchLabels:
      k8s-app: kube-dns
  template:
    metadata:
      labels:
        k8s-app: kube-dns
    spec:
      priorityClassName: system-cluster-critical
      serviceAccountName: coredns
      tolerations:
        - key: "CriticalAddonsOnly"
          operator: "Exists"
      nodeSelector:
        kubernetes.io/os: linux
      affinity:
         podAntiAffinity:
           preferredDuringSchedulingIgnoredDuringExecution:
           - weight: 100
             podAffinityTerm:
               labelSelector:
                 matchExpressions:
                   - key: k8s-app
                     operator: In
                     values: ["kube-dns"]
               topologyKey: kubernetes.io/hostname
      containers:
      - name: coredns
        image: coredns/coredns:1.8.3
        imagePullPolicy: IfNotPresent
        resources:
          limits:
            memory: 170Mi
          requests:
            cpu: 100m
            memory: 70Mi
        args: [ "-conf", "/etc/coredns/Corefile" ]
        volumeMounts:
        - name: config-volume
          mountPath: /etc/coredns
          readOnly: true
        ports:
        - containerPort: 53
          name: dns
          protocol: UDP
        - containerPort: 53
```

```
          name: dns-tcp
          protocol: TCP
        - containerPort: 9153
          name: metrics
          protocol: TCP
        securityContext:
          allowPrivilegeEscalation: false
          capabilities:
            add:
            - NET_BIND_SERVICE
            drop:
            - all
          readOnlyRootFilesystem: true
        livenessProbe:
          httpGet:
            path: /health
            port: 8080
            scheme: HTTP
          initialDelaySeconds: 60
          timeoutSeconds: 5
          successThreshold: 1
          failureThreshold: 5
        readinessProbe:
          httpGet:
            path: /ready
            port: 8181
            scheme: HTTP
      dnsPolicy: Default
      volumes:
        - name: config-volume
          configMap:
            name: coredns
            items:
            - key: Corefile
              path: Corefile
---
apiVersion: v1
kind: Service
metadata:
  name: kube-dns
  namespace: kube-system
  annotations:
    prometheus.io/port: "9153"
    prometheus.io/scrape: "true"
  labels:
    k8s-app: kube-dns
    kubernetes.io/cluster-service: "true"
    kubernetes.io/name: "CoreDNS"
spec:
```

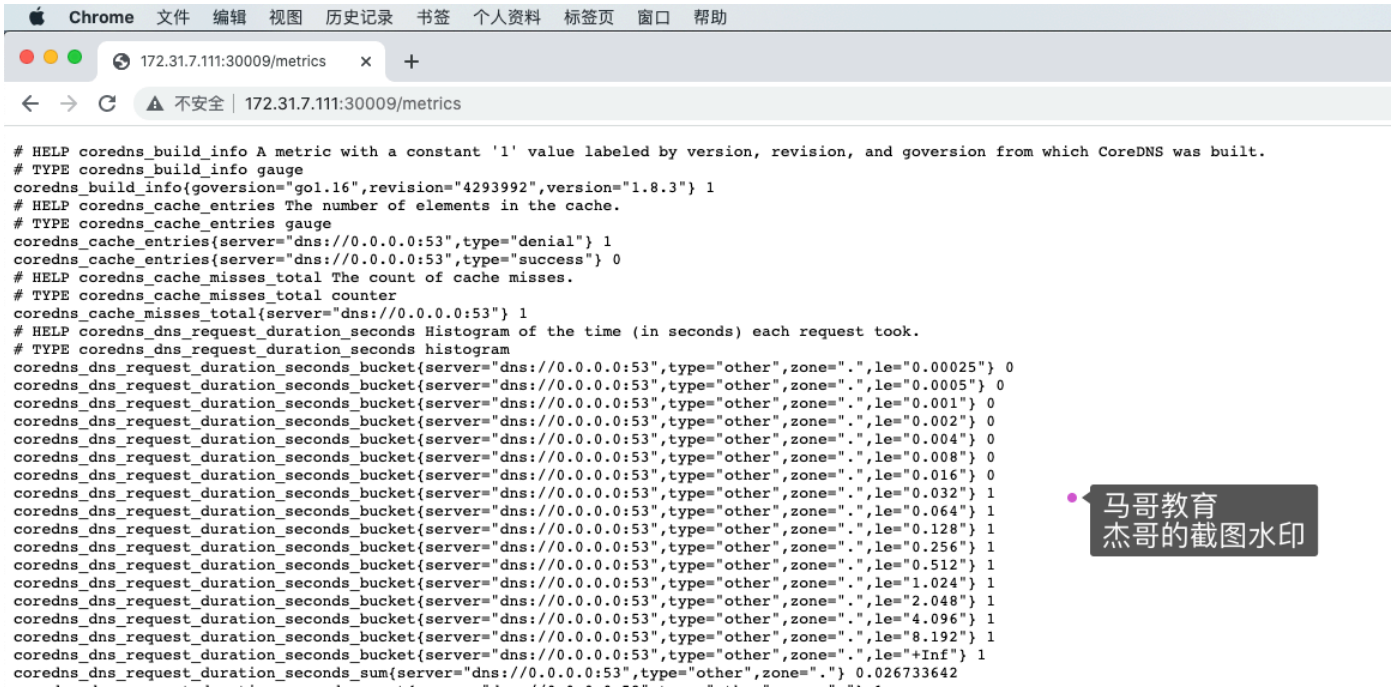
```
  type: NodePort
  selector:
    k8s-app: kube-dns
  clusterIP: 10.0.0.2
  ports:
  - name: dns
    port: 53
    protocol: UDP
  - name: dns-tcp
    port: 53
    protocol: TCP
  - name: metrics
    port: 9153
    protocol: TCP
    targetPort: 9153
    nodePort: 30009

# kubectl  delete -f /etc/ansible/manifests/dns/kube-dns/kube-dns.yaml #删除kube-dns
# kubectl  apply -f  coredns.yaml  #部署coredns

# kubectl apply -f coredns-jie.yaml
serviceaccount/coredns created
clusterrole.rbac.authorization.k8s.io/system:coredns created
clusterrolebinding.rbac.authorization.k8s.io/system:coredns created
configmap/coredns created
deployment.apps/coredns created
service/kube-dns created
```

1.7.3.3：测试域名解析：

```
root@k8s-master1:/etc/kubeasz# kubectl  exec -it net-test1  sh
kubectl exec [POD] [COMMAND] is DEPRECATED and will be removed in a future version. Use
kubectl exec [POD] -- [COMMAND] instead.
/ # ping kubernetes.default.svc.magedu.local
PING kubernetes.default.svc.magedu.local (10.0.0.1): 56 data bytes
64 bytes from 10.0.0.1: seq=0 ttl=64 time=0.062 ms
64 bytes from 10.0.0.1: seq=1 ttl=64 time=0.117 ms
^C
--- kubernetes.default.svc.magedu.local ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.062/0.089/0.117 ms
```

1.7.3.4：验证coredns指标数据：

http://172.31.7.111:30009/metrics



1.8：dashboard：

部署kubernetes的web管理界面dashboard

https://github.com/kubernetes/dashboard

1.8.1：部署dashboard：

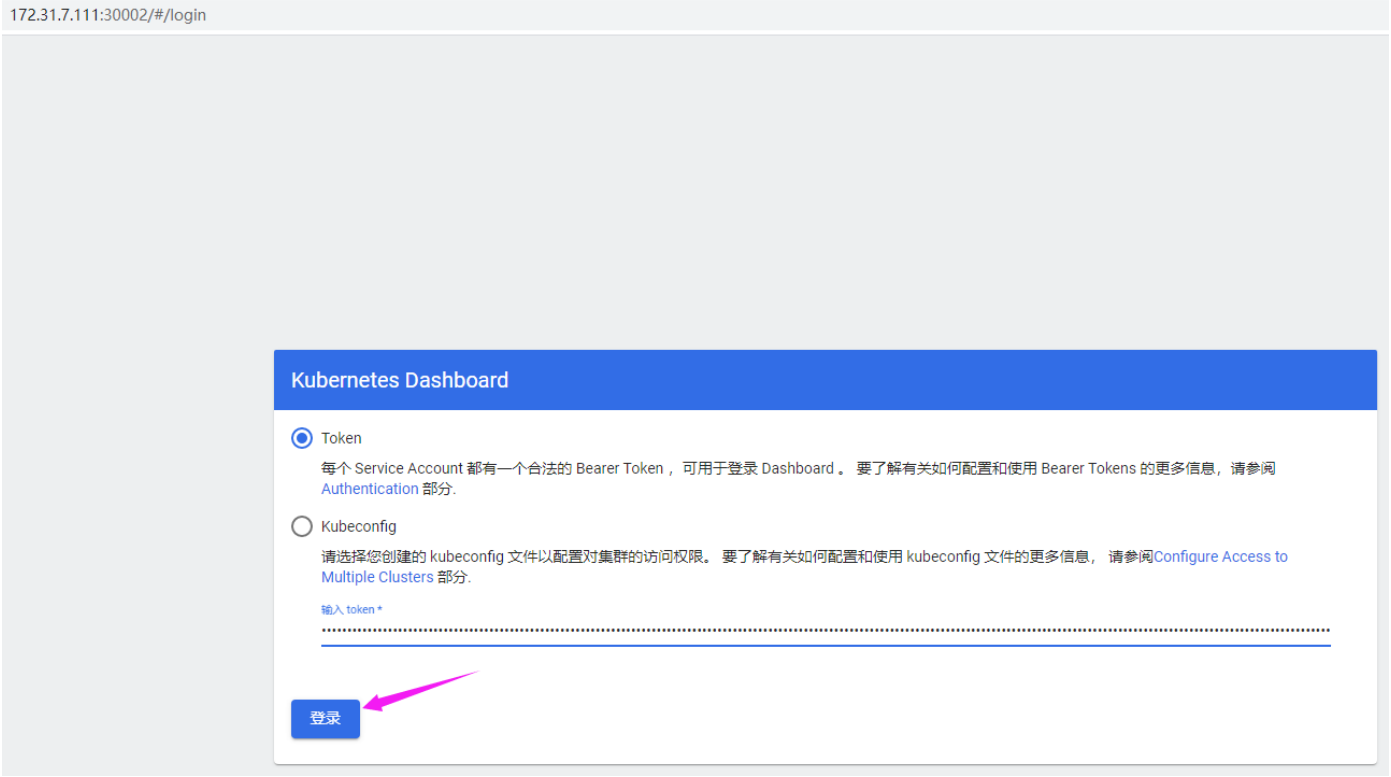
```
root@k8s-master1:~# docker pull kubernetesui/dashboard:v2.3.1
root@k8s-master1:~# docker tag kubernetesui/dashboard:v2.3.1
harbor.magedu.net/baseimages/dashboard:v2.3.1
root@k8s-master1:~# docker push harbor.magedu.net/baseimages/dashboard:v2.3.1

root@k8s-master1:~# docker pull kubernetesui/metrics-scraper:v1.0.6
root@k8s-master1:~# docker tag kubernetesui/metrics-scraper:v1.0.6
harbor.magedu.net/baseimages/metrics-scraper:v1.0.6
root@k8s-master1:~# docker push harbor.magedu.net/baseimages/metrics-scraper:v1.0.6

root@k8s-master1:~# wget
https://raw.githubusercontent.com/kubernetes/dashboard/v2.3.1/aio/deploy/recommended.ya
ml
root@k8s-master1:~# mv recommended.yaml  dashboard-v2.3.1.yaml
root@k8s-master1:~# vim dashboard-v2.3.1.yaml
root@k8s-master1:~# kubectl apply  -f dashboard-v2.3.1.yaml
root@k8s-master1:~# kubectl apply -f admin-user.yml
```

1.8.1.2：token登录dashboard：

```
root@k8s-master1:~# kubectl  get secret -A | grep admin
root@k8s-master1:~# kubectl  -n kubernetes-dashboard describe secret admin-user-token-cgg2f
```

172.31.7.111:30002

172.31.7.111:30002/#/login



不安全 | 172.31.7.111:30002/#/pod?namespace=_all

kubernetes 全部命名空间 搜索

Workloads > Pods

工作负载 N
- Cron Jobs
- Daemon Sets
- Deployments
- Jobs
- Pods
- Replica Sets
- Replication Controllers
- Stateful Sets

服务 N
- Ingresses
- Services

配置和存储
- Config Maps N
- Persistent Volume Claims N

Pods

杰哥的截图水印

名称	命名空间	标签	节点	状态
dashboard-metrics-scraper-696b4ffdb7-6ntmf	kubernetes-dashboard	k8s-app: dashboard-metrics-scraper pod-template-hash: 696b4ffdb7	172.31.7.112	Running
kubernetes-dashboard-65d7f76c69-kh7gz	kubernetes-dashboard	k8s-app: kubernetes-dashboard pod-template-hash: 65d7f76c69	172.31.7.113	Running
calico-node-f2zr2	kube-system	controller-revision-hash: 75579b74d9 k8s-app: calico-node pod-template-generation: 2	172.31.7.103	Running
calico-node-lq9md	kube-system	controller-revision-hash: 75579b74d9 k8s-app: calico-node pod-template-generation: 2	172.31.7.112	Running
calico-node-l6phl	kube-system	controller-revision-hash: 75579b74d9 k8s-app: calico-node pod-template-generation: 2	172.31.7.111	Running

1.8.1.3：Kubeconfig登录：

制作kubeconfig文件

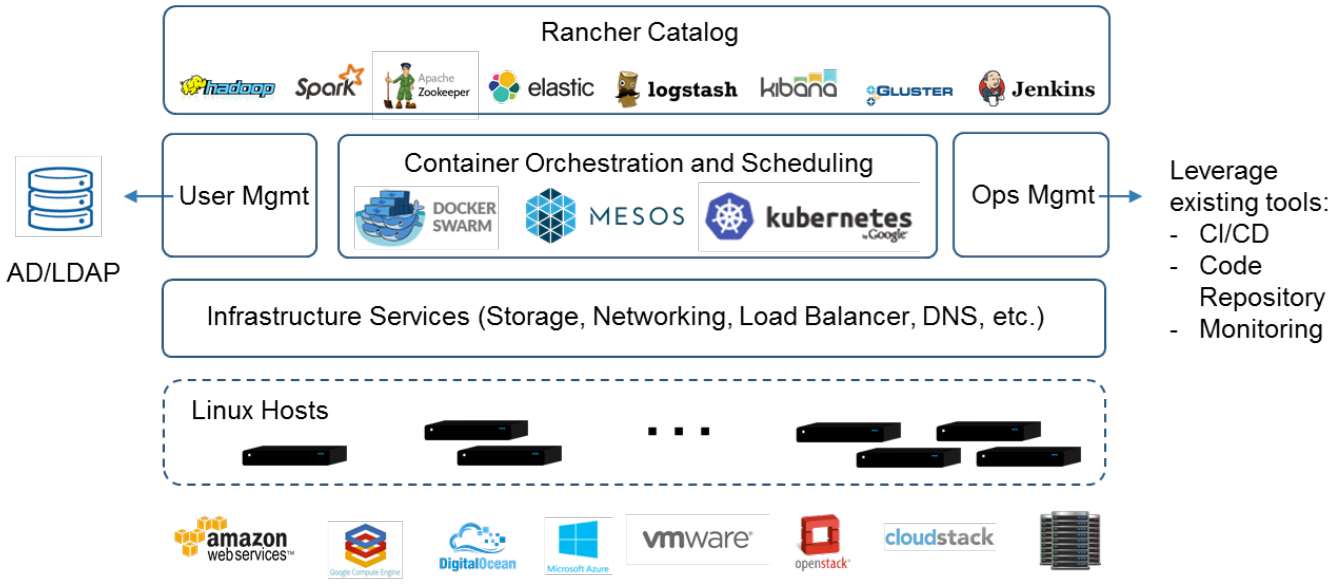
1.8.1.5：设置token登录会话保持时间

```
root@k8s-master1:~# vim dashboard-v2.3.1.yaml
    spec:
      containers:
        - name: kubernetes-dashboard
          image: kubernetesui/dashboard:v2.3.1
          imagePullPolicy: Always
          ports:
            - containerPort: 8443
              protocol: TCP
          args:
            - --auto-generate-certificates
            - --namespace=kubernetes-dashboard
            - --token-ttl=43200


root@k8s-master1:~# kubectl apply -f dashboard-v2.3.1.yaml
```

1.8.2：rancher：

https://rancher.com/docs/rancher/v1.6/zh/

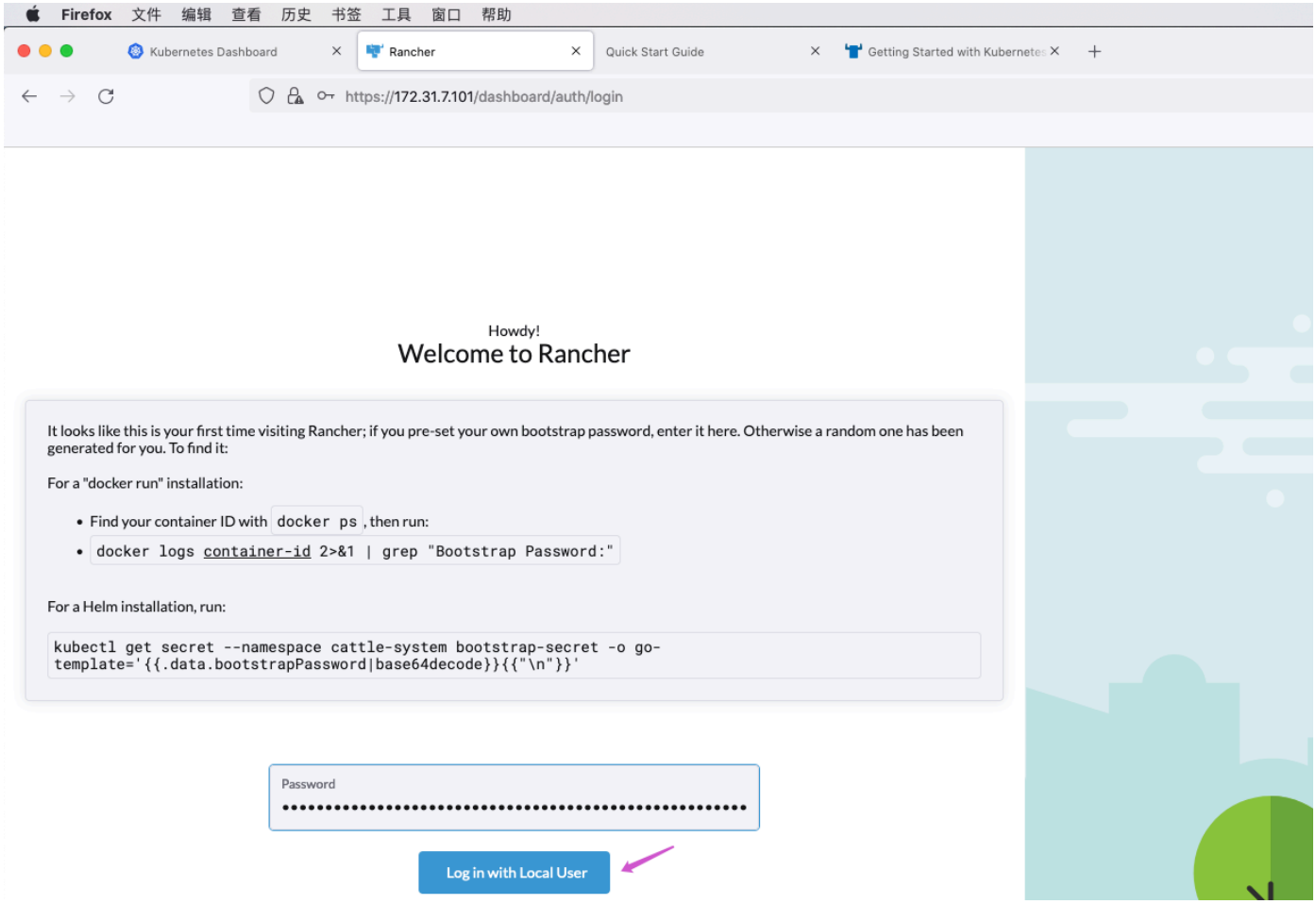
https://rancher.com/quick-start/



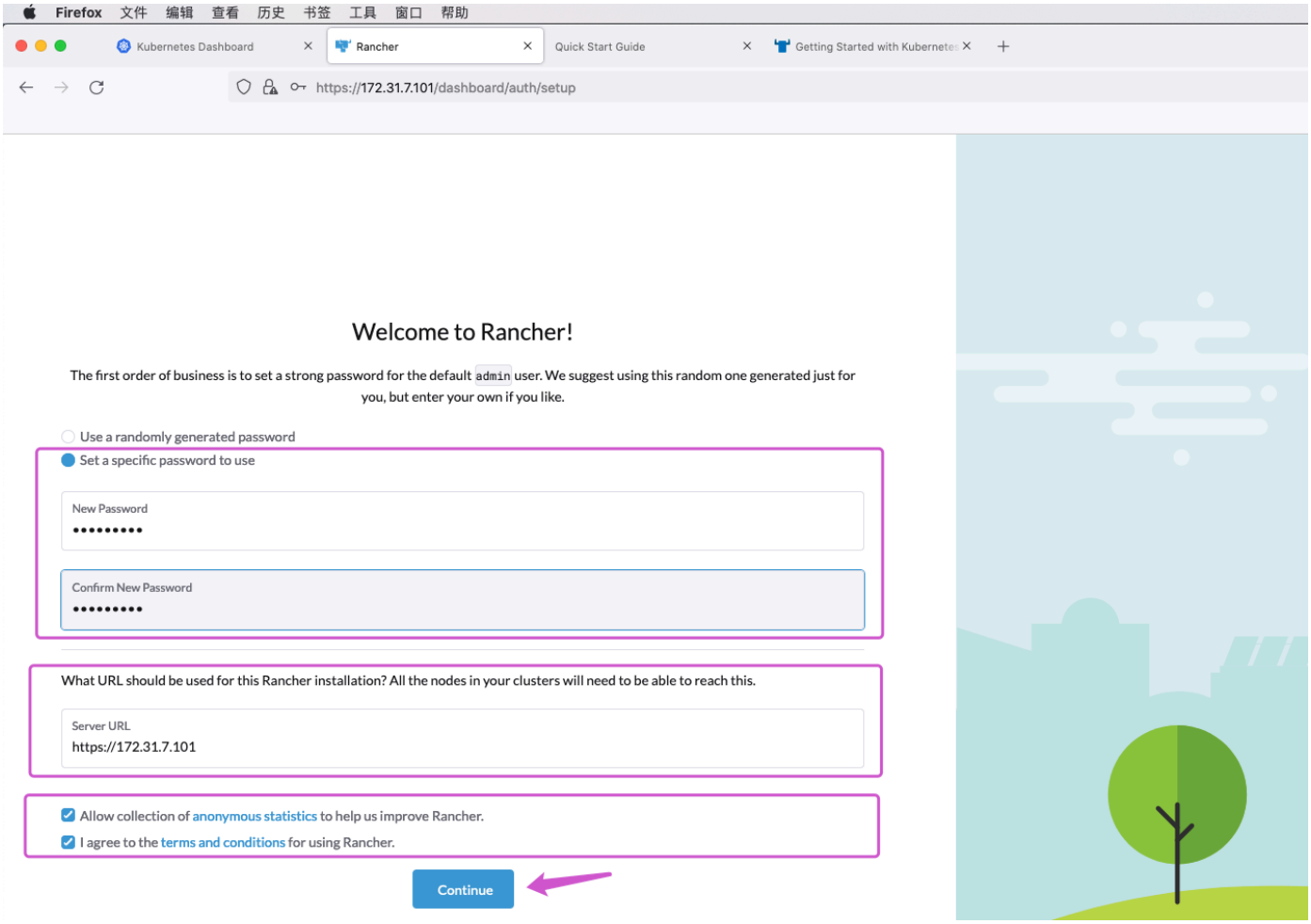
1.8.2.1:安装rancher：

```
root@k8s-master1:~# sudo docker run --privileged -d --restart=unless-stopped -p 80:80 -p 443:443 rancher/rancher
root@k8s-master1:~# docker logs b8463c48a2b8  2>&1 | grep "Bootstrap Password:"
2021/09/10 11:45:46 [INFO] Bootstrap Password: jrxfp749g55h2gnw8rdd7sqrzxphtmvwv422xdbc22d699lqsb5m2r
```

1.8.2.2：访问dashboard：

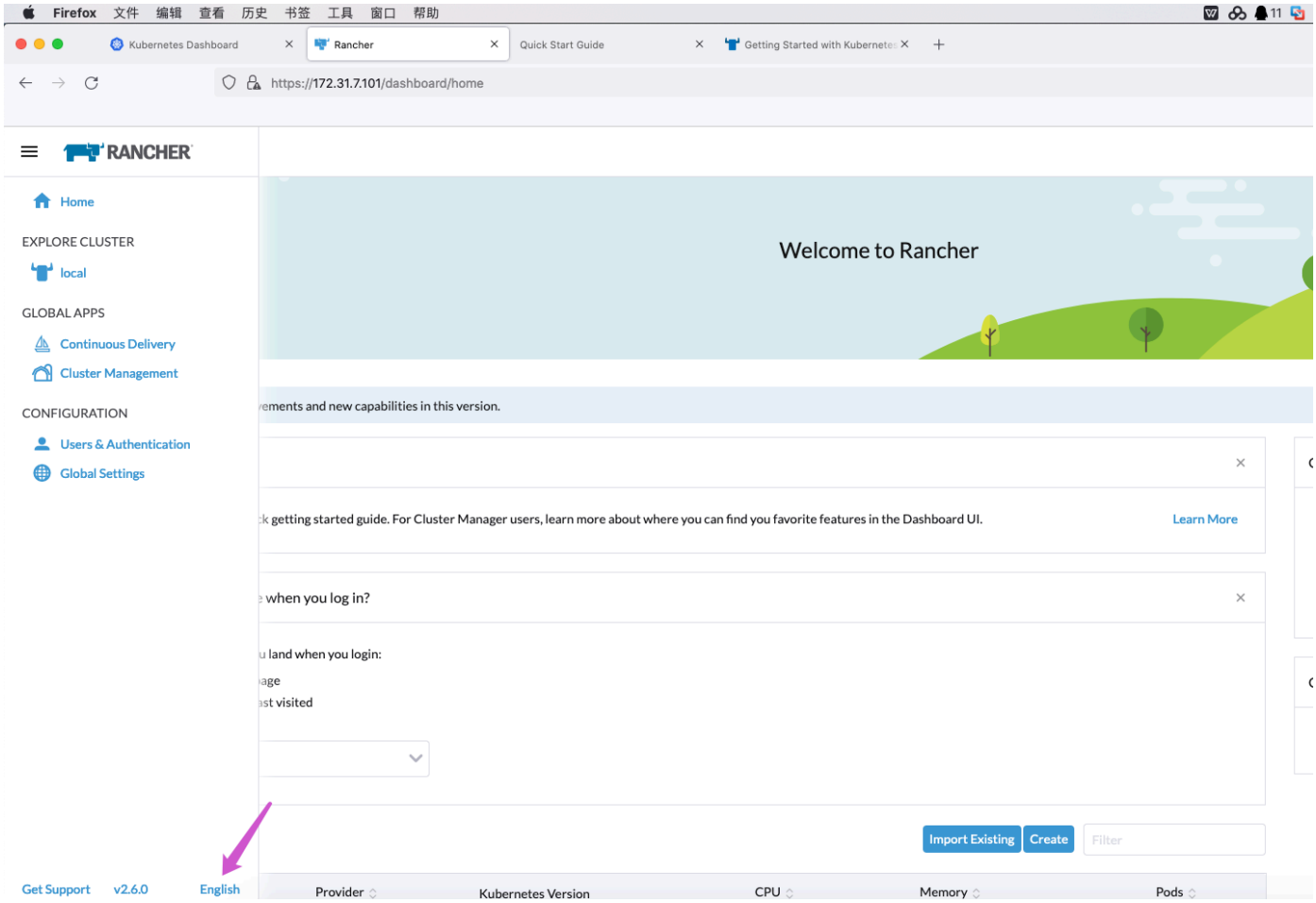


1.8.2.3：创建密码：

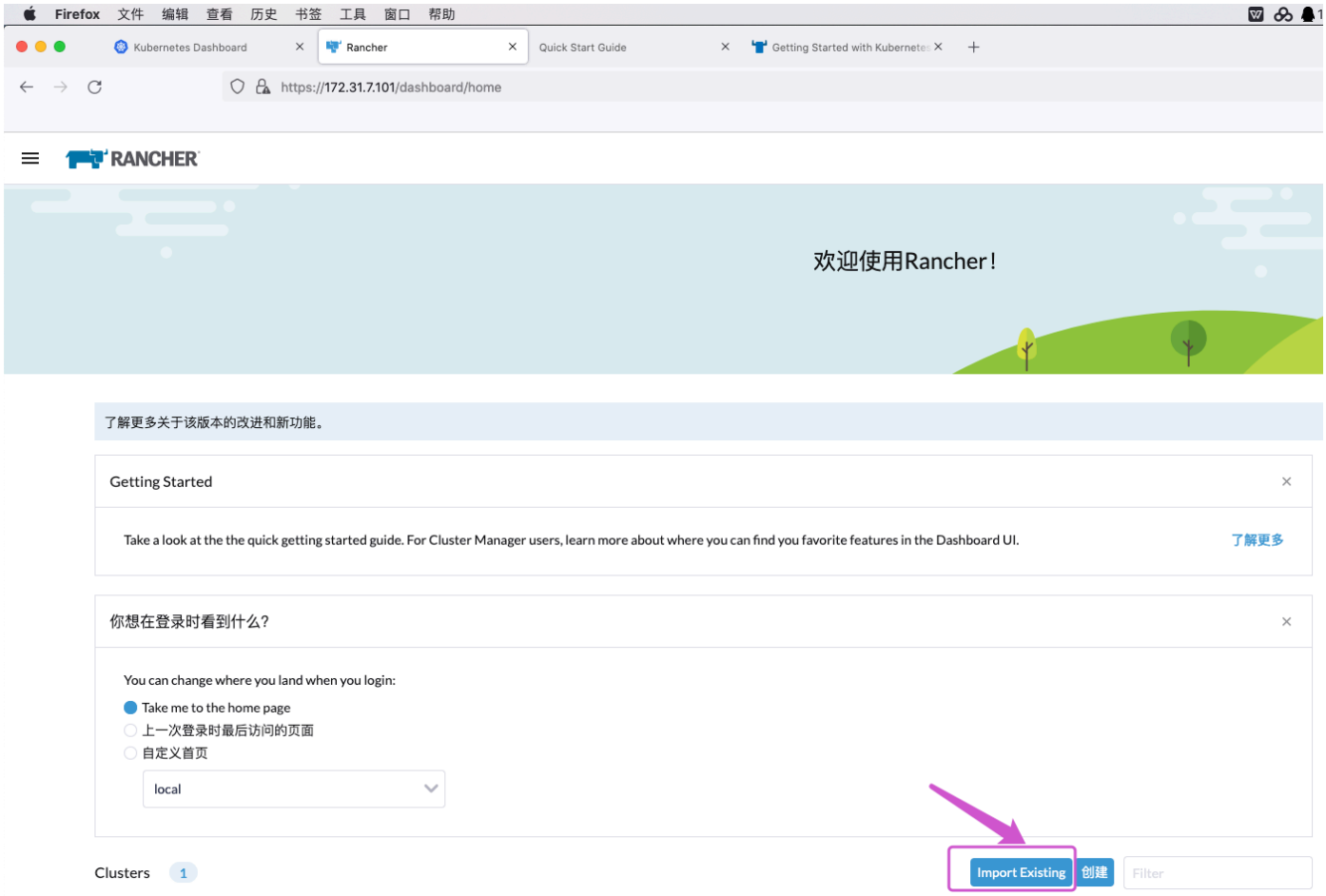


1.8.2.3:导入集群：

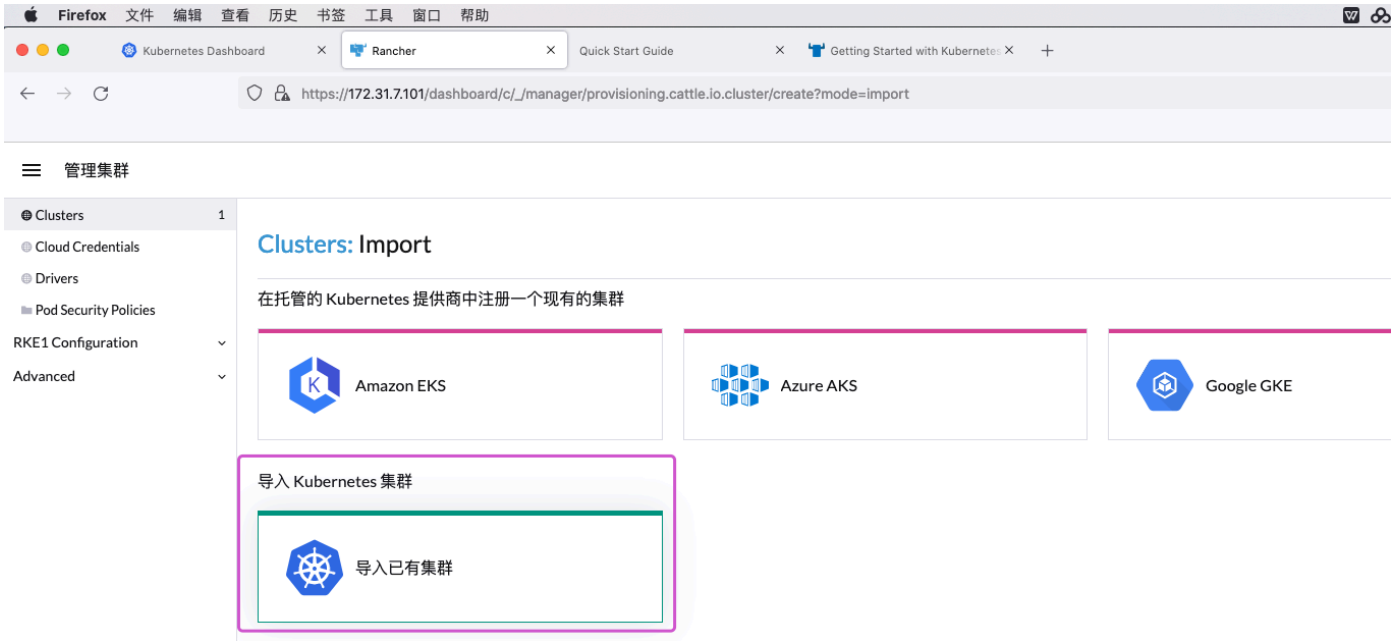
语言设置：

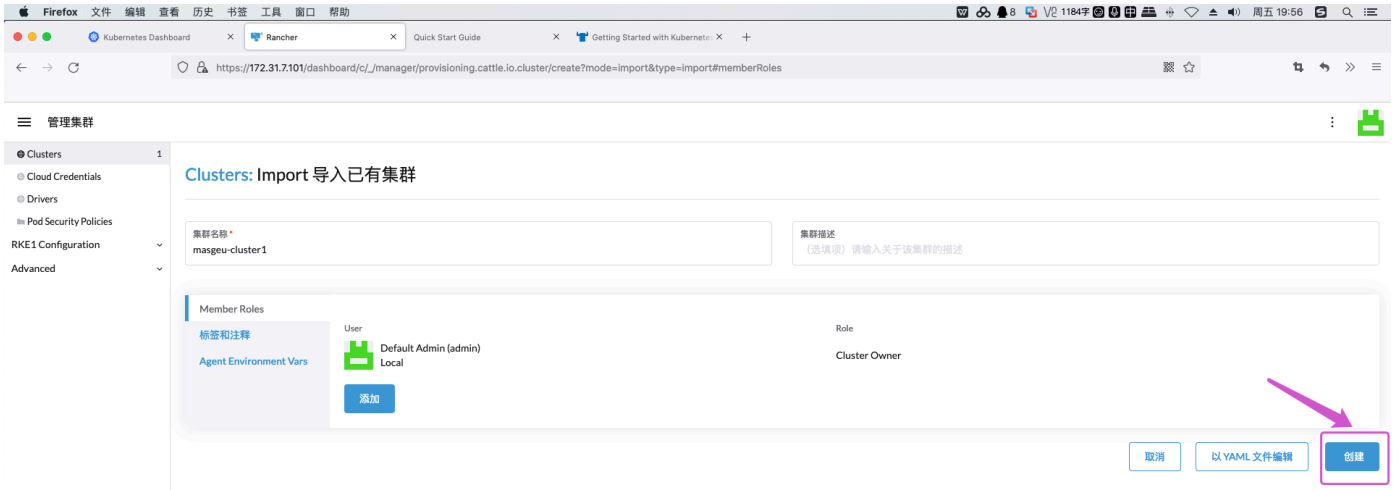

Firefox 文件 编辑 查看 历史 书签 工具 窗口 帮助
Kubernetes Dashboard
Rancher
Quick Start Guide
Getting Started with Kubernetes
https://172.31.7.101/dashboard/home
RANCHER
Home
EXPLORE CLUSTER
local
GLOBAL APPS
Continuous Delivery
Cluster Management
CONFIGURATION
Users & Authentication
Global Settings
Welcome to Rancher
ements and new capabilities in this version.
k getting started guide. For Cluster Manager users, learn more about where you can find you favorite features in the Dashboard UI.
Learn More
when you log in?
u land when you login:
age
st visited
Import Existing
Create
Filter
Get Support
v2.6.0
English
Provider
Kubernetes Version
CPU
Memory
Pods

导入集群：

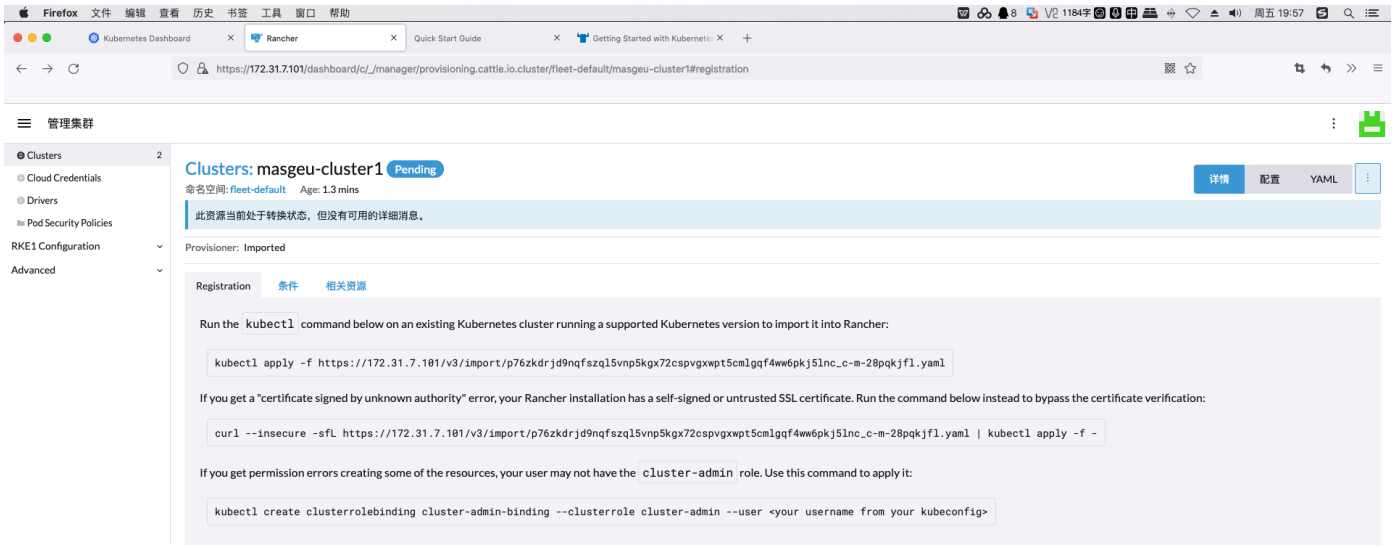

Firefox 文件 编辑 查看 历史 书签 工具 窗口 帮助
Kubernetes Dashboard
Rancher
Quick Start Guide
Getting Started with Kubernetes
https://172.31.7.101/dashboard/home
RANCHER
欢迎使用Rancher!
了解更多关于该版本的改进和新功能。
Getting Started
Take a look at the the quick getting started guide. For Cluster Manager users, learn more about where you can find you favorite features in the Dashboard UI.
了解更多
你想在登录时看到什么?
You can change where you land when you login:
Take me to the home page
上一次登录时最后访问的页面
自定义首页
local
Clusters 1
Import Existing
创建
Filter

导入已有集群:


Firefox 文件 编辑 查看 历史 书签 工具 窗口 帮助
Kubernetes Dashboard
Rancher
Quick Start Guide
Getting Started with Kubernetes
https://172.31.7.101/dashboard/c/_/manager/provisioning.cattle.io.cluster/create?mode=import
管理集群
Clusters 1
Cloud Credentials
Drivers
Pod Security Policies
RKE1 Configuration
Advanced
Clusters: Import
在托管的 Kubernetes 提供商中注册一个现有的集群
Amazon EKS
Azure AKS
Google GKE
导入 Kubernetes 集群
导入已有集群



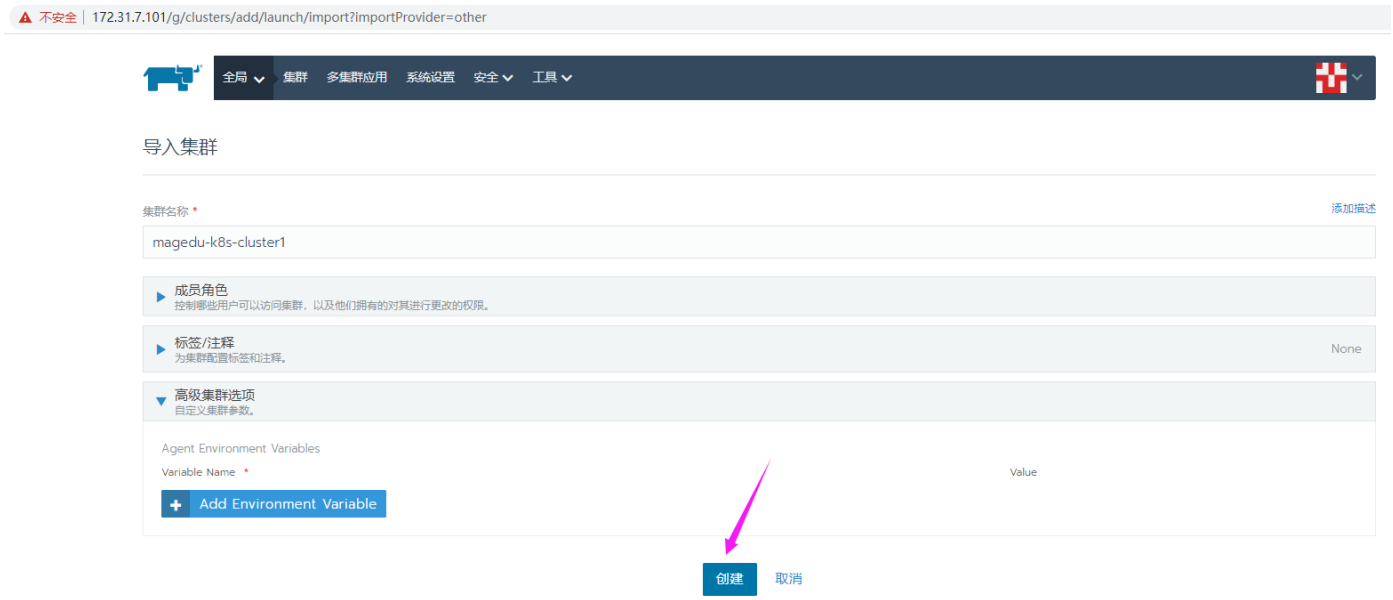
添加集群后要在k8s客户端：



k8s创建客户端：

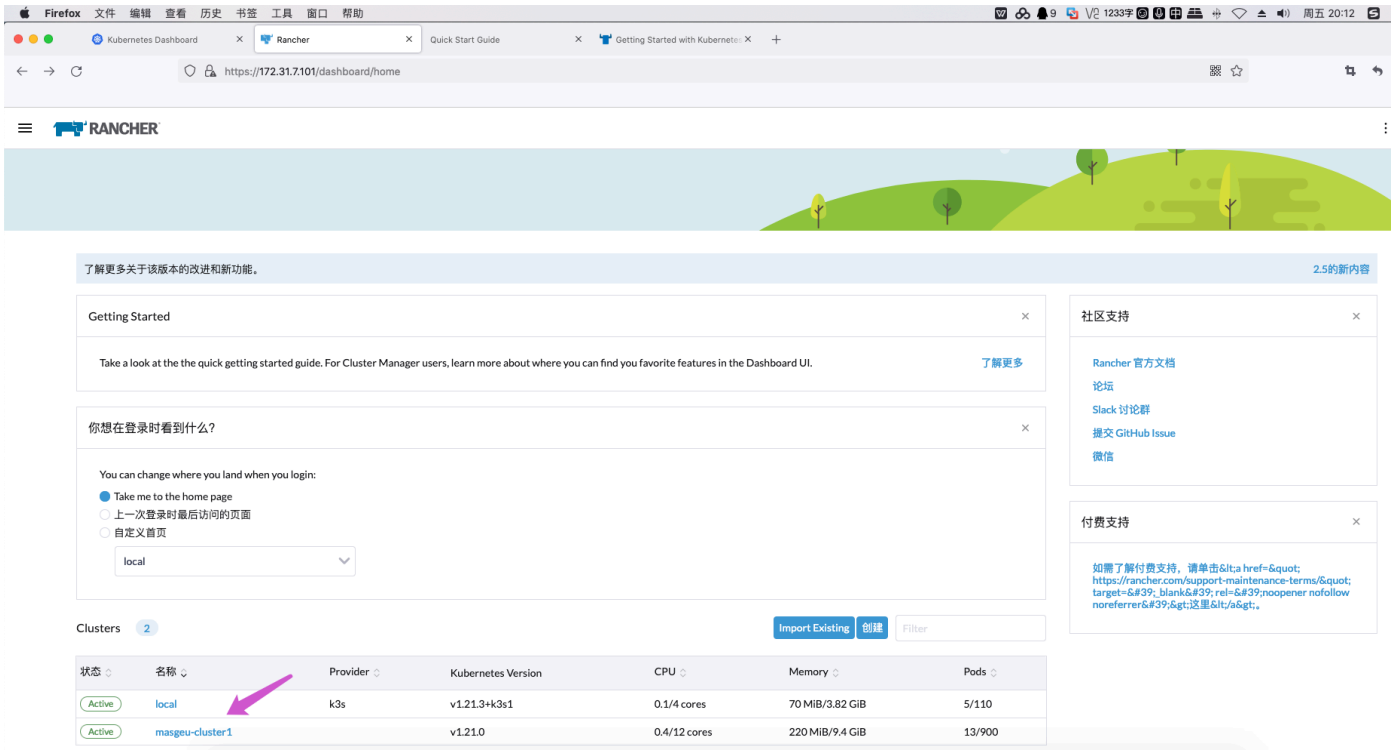
```
root@k8s-master1:~# curl --insecure -sfL
https://172.31.7.101/v3/import/p76zkdrjd9nqfszql5vnp5kgx72cspvgxwpt5cmlgqf4ww6pkj5lnc_c
-m-28pqkjfl.yaml | kubectl apply -f -
clusterrole.rbac.authorization.k8s.io/proxy-clusterrole-kubeapiserver created
clusterrolebinding.rbac.authorization.k8s.io/proxy-role-binding-kubernetes-master
created
namespace/cattle-system created
serviceaccount/cattle created
clusterrolebinding.rbac.authorization.k8s.io/cattle-admin-binding created
secret/cattle-credentials-552ffc2 created
clusterrole.rbac.authorization.k8s.io/cattle-admin created
deployment.apps/cattle-cluster-agent created
service/cattle-cluster-agent created
```

集群配置：

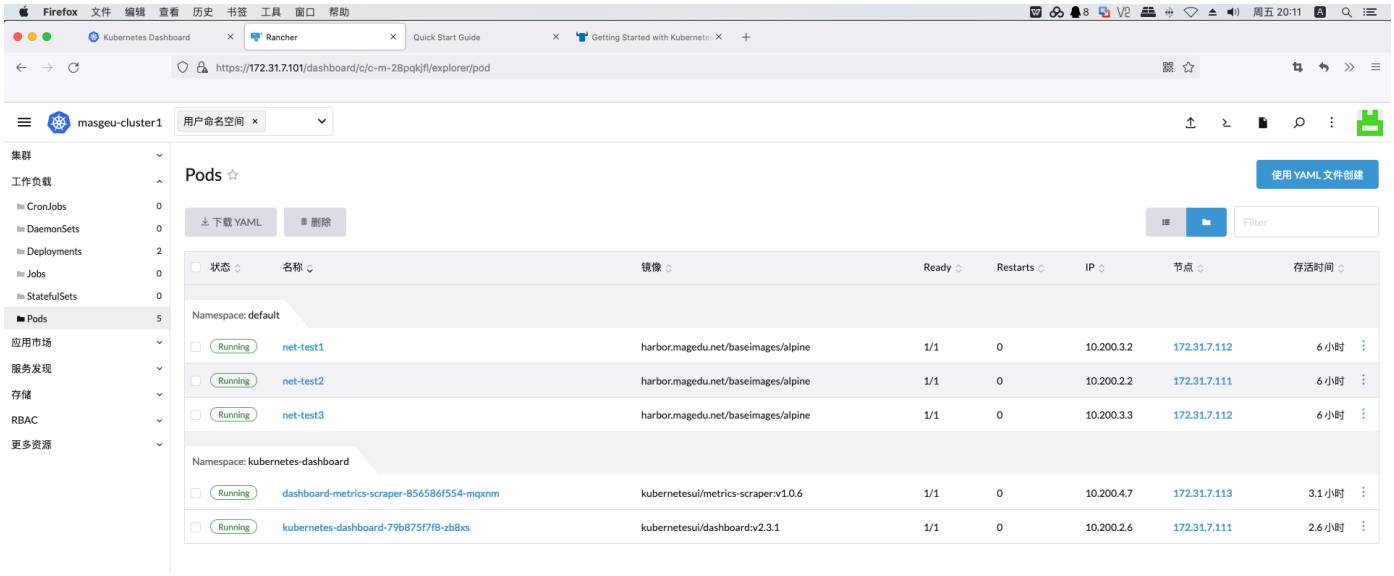



```
# curl --insecure -sfL
https://172.31.7.101/v3/import/c8khcm67zhvxrhbx4ddpmmc88htxmkwb5b4f4fpttn2jlkfnwbscsx_c
-wq7t7.yaml | kubectl apply -f -
```

web页面管理集群：



管理页面：



1.8.3：kuboard:

https://kuboard.cn/support/#kuboard-%E4%BB%8B%E7%BB%8D #kuboard-介绍

https://kuboard.cn/install/v3/install.html#%E5%85%BC%E5%AE%B9%E6%80%A7 #kuboard与kubernetes兼容性

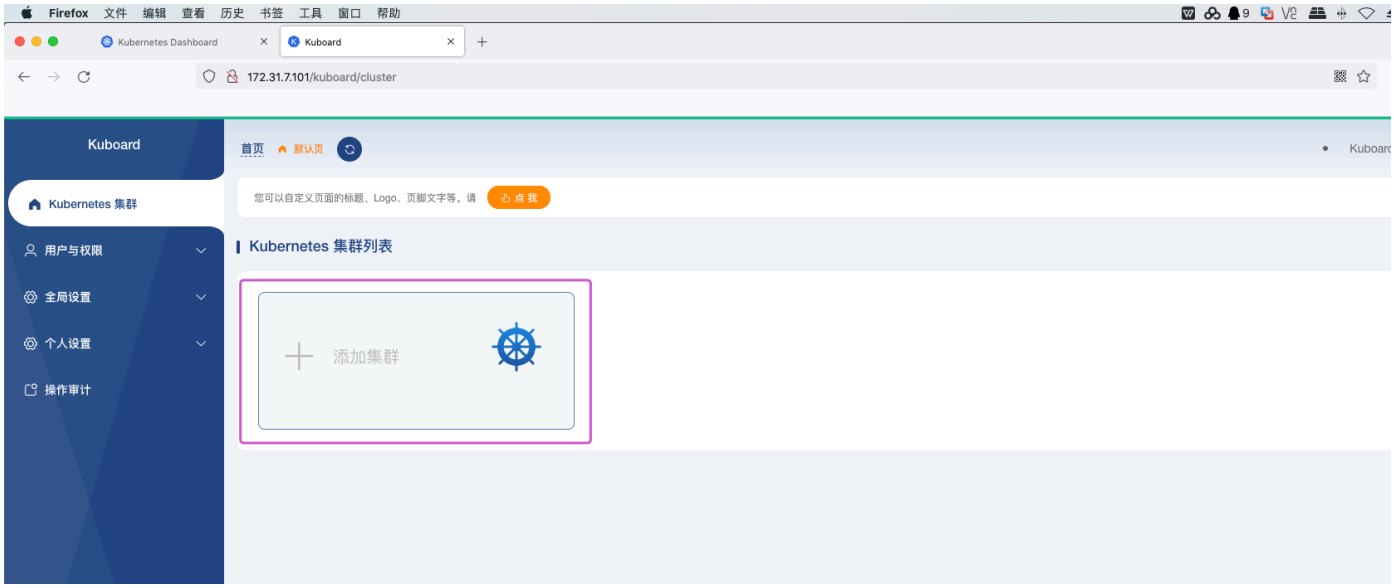
https://kuboard.cn/install/v3/install-built-in.html#%E9%83%A8%E7%BD%B2%E8%AE%A1%E5%88%92 #安装教程

1.8.3.1：部署kuboard：

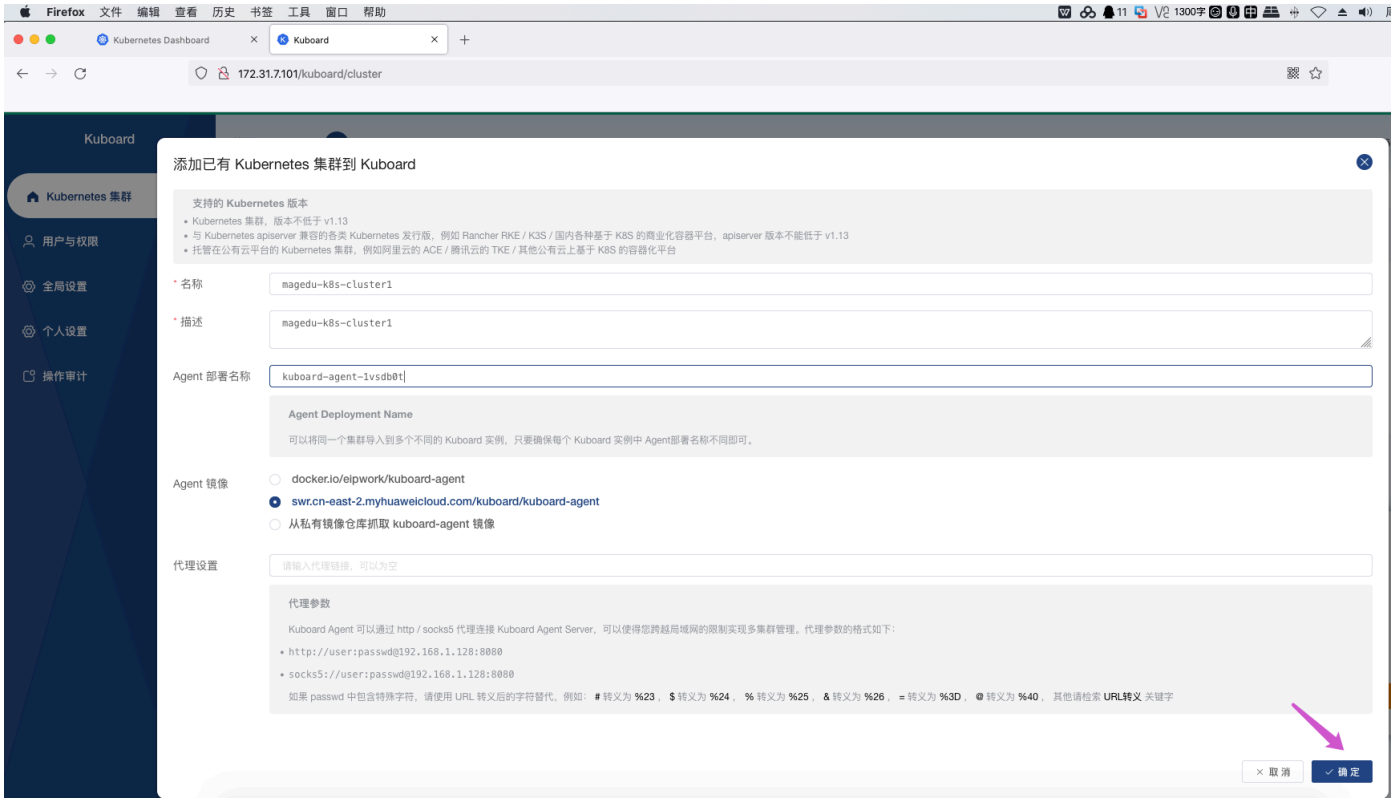
```
root@k8s-master1:~# sudo docker run -d \
  --restart=unless-stopped \
  --name=kuboard \
  -p 80:80/tcp \
  -p 10081:10081/tcp \
  -e KUBOARD_ENDPOINT="http://172.31.7.101:80" \
  -e KUBOARD_AGENT_SERVER_TCP_PORT="10081" \
  -v /root/kuboard-data:/data \
  swr.cn-east-2.myhuaweicloud.com/kuboard/kuboard:v3

在浏览器输入 http://your-host-ip:80 即可访问 Kuboard v3.x 的界面，登录方式：

用户名： admin
密 码： Kuboard123
```

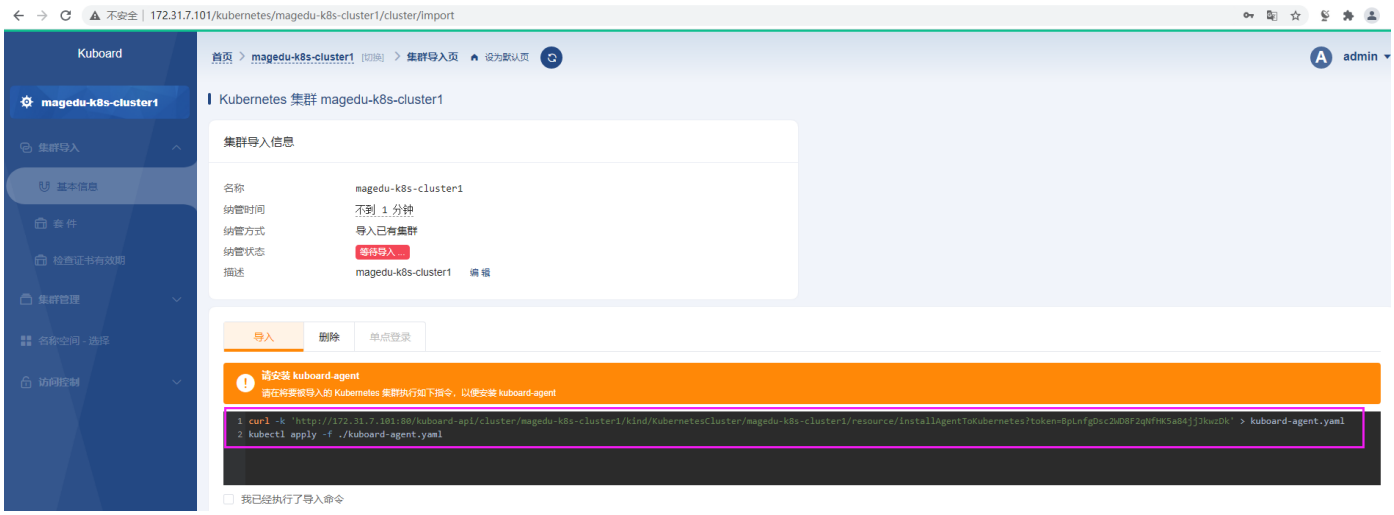
登录界面：

添加集群：

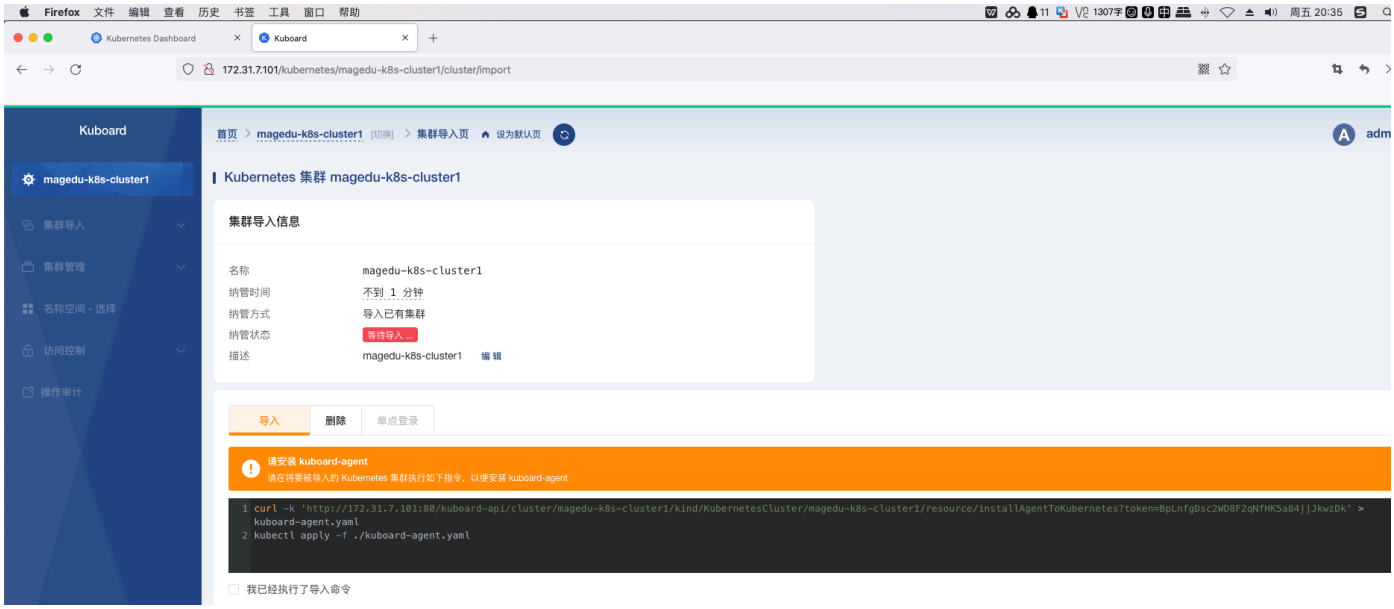


添加集群-生成agent：



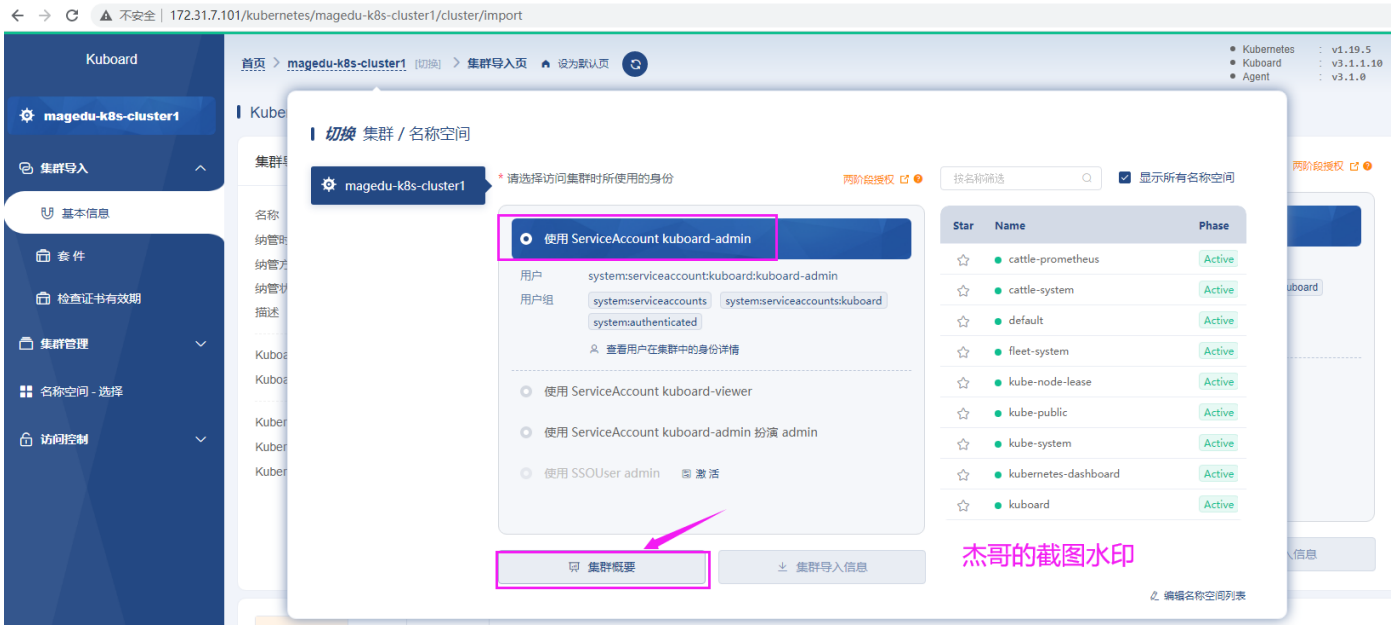


k8s管理节点部署agent：

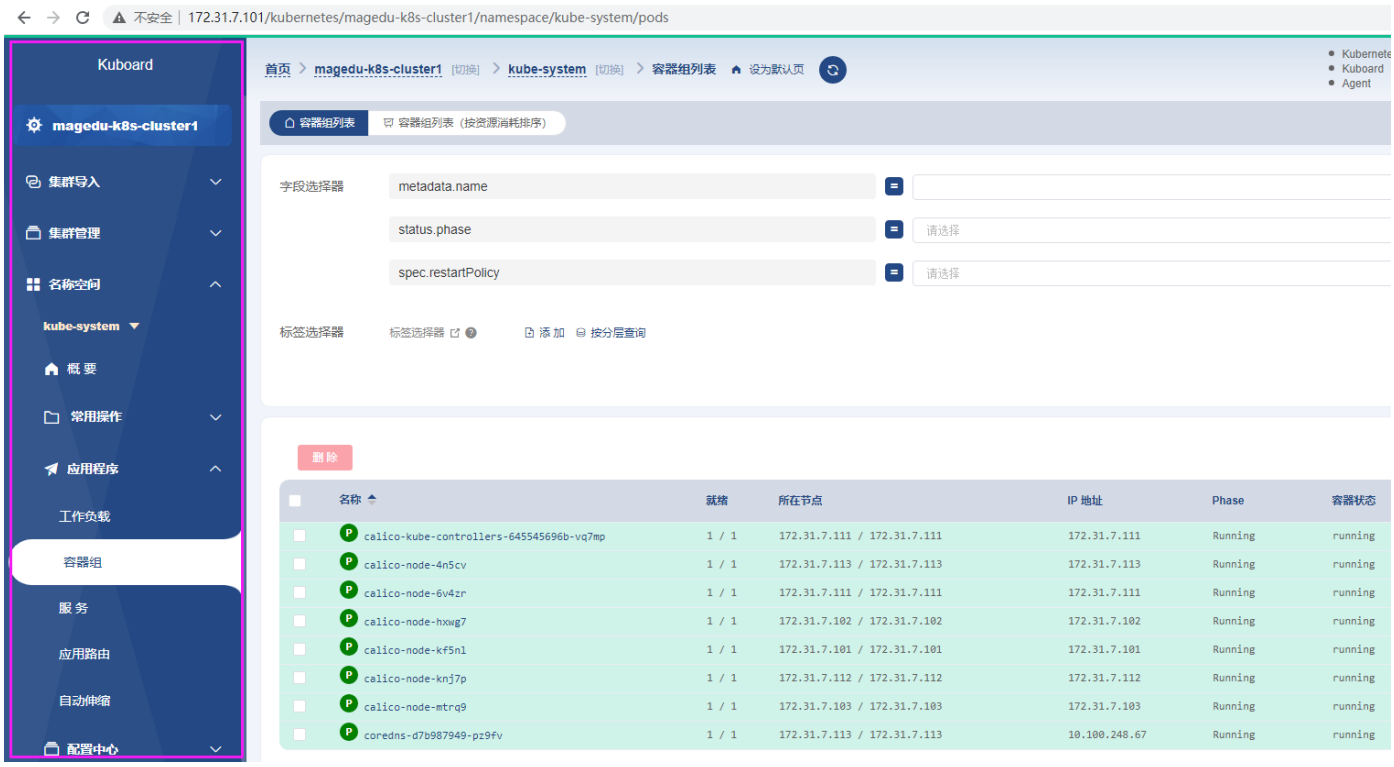



```
root@k8s-master1:~# curl -k 'http://172.31.7.101:80/kuboard-api/cluster/magedu-k8s-cluster1/kind/KubernetesCluster/magedu-k8s-cluster1/resource/installAgentToKubernetes?token=BpLnfgDsc2WD8F2qNfHK5a84jjJkwzDk' > kuboard-agent.yaml
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  5881    0  5881    0     0  1148k      0 --:--:-- --:--:-- --:--:-- 1148k
root@k8s-master1:~# kubectl apply -f ./kuboard-agent.yaml
namespace/kuboard created
serviceaccount/kuboard-admin created
clusterrolebinding.rbac.authorization.k8s.io/kuboard-admin-crb created
serviceaccount/kuboard-viewer created
clusterrolebinding.rbac.authorization.k8s.io/kuboard-viewer-crb created
deployment.apps/kuboard-agent-1vsdb0t created
deployment.apps/kuboard-agent-1vsdb0t-2 created
```

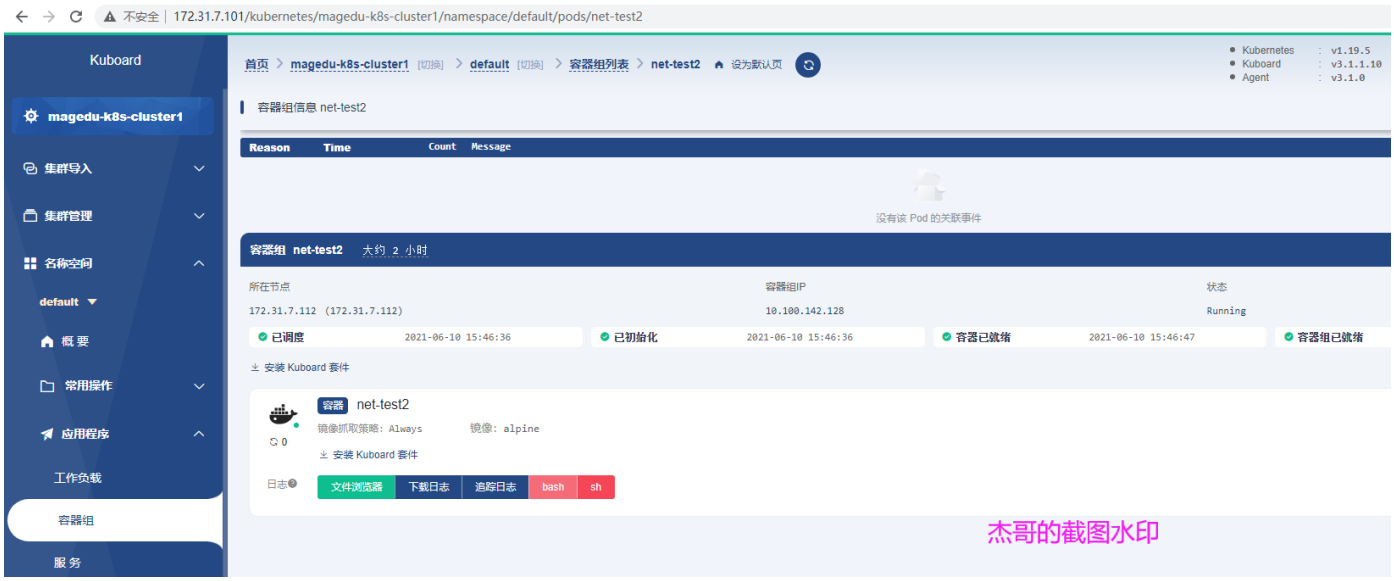
验证是否可管理k8s集群：



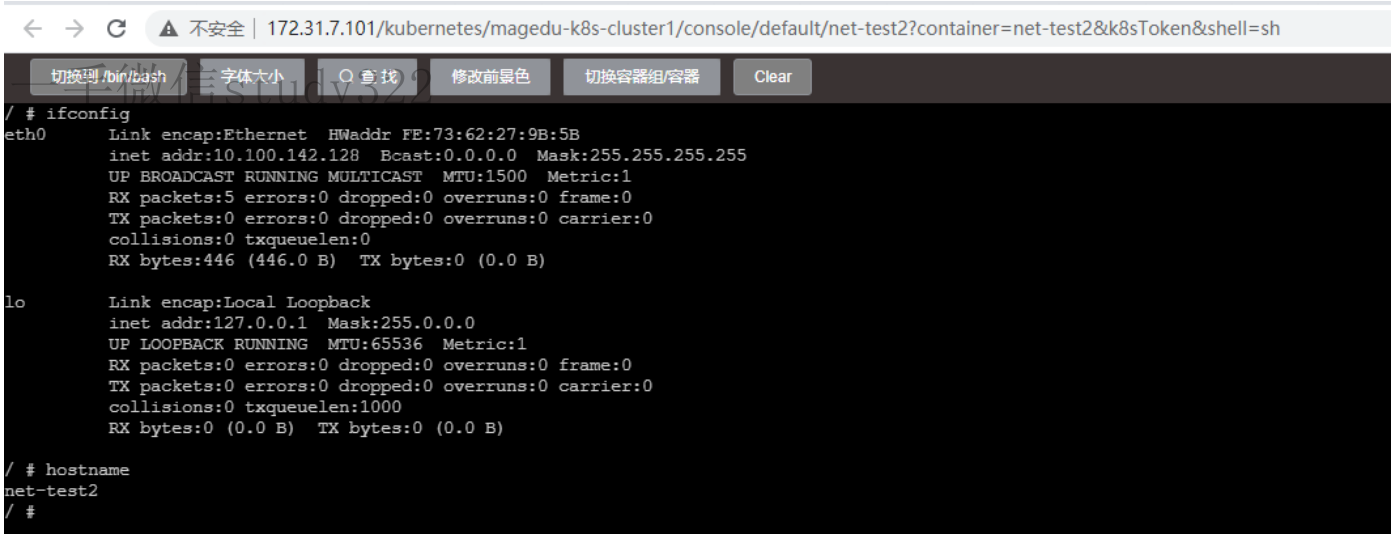
对k8s中的容器和其他资源进行管理：



#进入容器进行单独日志查等操作：



进入容器进行操作：


不安全 | 172.31.7.101/kubernetes/magedu-k8s-cluster1/console/default/net-test2?container=net-test2&k8sToken&shell=sh
切换到 /bin/bash
字体大小
查找
修改前景色
切换容器组/容器
Clear
/ # ifconfig
eth0 Link encap:Ethernet HWaddr FE:73:62:27:9B:5B
inet addr:10.100.142.128 Bcast:0.0.0.0 Mask:255.255.255.255
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:5 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:446 (446.0 B) TX bytes:0 (0.0 B)
lo Link encap:Local Loopback
inet addr:127.0.0.1 Mask:255.0.0.0
UP LOOPBACK RUNNING MTU:65536 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)
/ # hostname
net-test2
/ #