Kubernetes集群安装文档

注：本次安装共三个节点:

master（node04.ift.com|192.168.254.104）

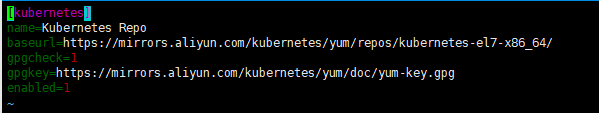
node01（node05.ift.com|192.168.254.105）

node02（node06.ift.com|192.168.254.106）

每台服务器都需要安装Docker和K8S

1. 由于Kubernetes安装过于复杂所以使用官方工具kubeadm进行安装。建议使用yum安装。注：安装之前请确定iptables和firewalld没有开启，最好禁止开机自动启动，因为k8s安装时会大量操作iptables规则。
2. 进入master配置yum仓库：
   1. cd /etc/yum.repos.d/
   2. vim kubernetes.repo，将以下内容复制到该文件中：

如图：



[kubernetes]

name=Kubernetes Repo

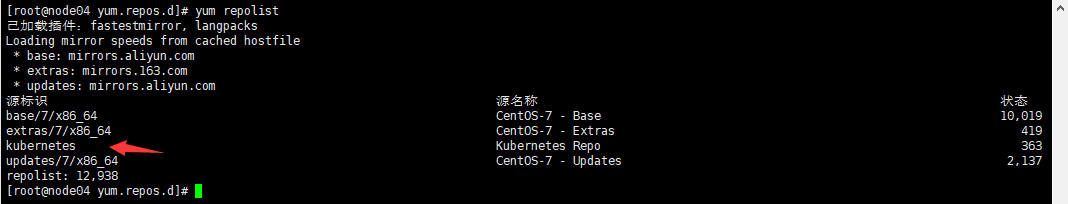
baseurl=https://mirrors.aliyun.com/kubernetes/yum/repos/kubernetes-el7-x86\_64/

gpgcheck=1

gpgkey=https://mirrors.aliyun.com/kubernetes/yum/doc/yum-key.gpg

enabled=1

* 1. 使用命令：yum repolist，检查仓库是否可用：



* 1. 拷贝kubernetes.repo和docker-ce.repo到node05.ift.com和node06.ift.com的/etc/yum.repos.d/目录下：

scp docker-ce.repo kubernetes.repo node05.ift.com:/etc/yum.repos.d/

* 1. 执行安装操作：

yum install docker-ce kubelet kubeadm kubectl

注：如果安装过程中提示gpg key校验错误则执行以下命令：

1. 将kubernetes.repo中的gpgcheck=1改为0
2. cd
3. wget <https://mirrors.aliyun.com/kubernetes/yum/doc/yum-key.gpg>
4. rpm --import yum-key.gpg
5. 重新执行安装命令
   1. 确认iptabls内部的nf-call的桥接的相关功能是否打开：

cat /proc/sys/net/bridge/bridge-nf-call-ip6tables

cat /proc/sys/net/bridge/bridge-nf-call-iptables

都为1则打开了，否则改为1，如果不为1则docker在创建ipatbles规则时则会失败。

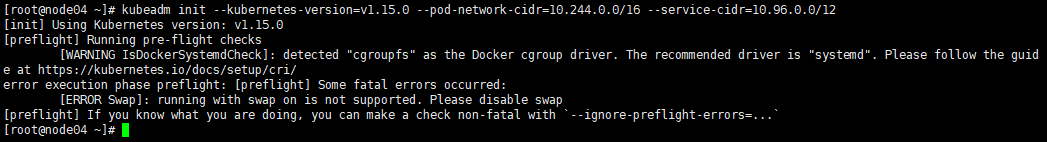
* 1. 将kubelet和docker设置为开机自动启动：

systemctl enable kubelet docker

1. 进行初始化操作：

注：初始化之前一定要保证防火墙处于关闭状态，并且确认iptabls内部的nf-call的桥接的相关功能已经打开

kubeadm init --kubernetes-version=v1.15.0 --pod-network-cidr=10.244.0.0/16 --service-cidr=10.96.0.0/12



如果主机开启了Swap会出现如上提示，如果不使用Swap则禁用后重试，否则：

vim /etc/sysconfig/kubelet，添加如下参数，--fail-swap-on=false表示启用swap时不让它报错

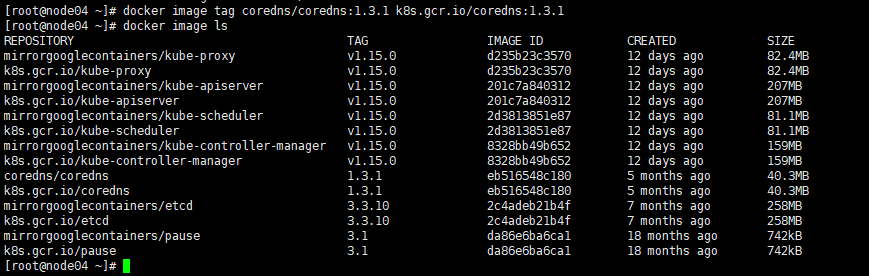


执行命令重新安装：

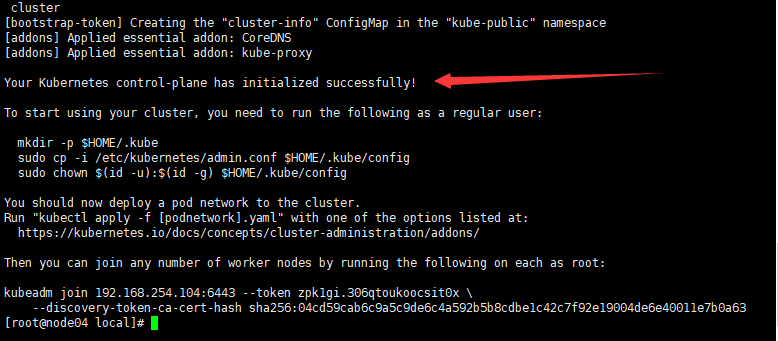
kubeadm init --kubernetes-version=v1.15.0 --pod-network-cidr=10.244.0.0/16 --service-cidr=10.96.0.0/12 --ignore-preflight-errors=Swap

此时安装将会失败，代理服务不通导致的，解决办法：

1. 科学上网
2. DockerHub上有一些人做的k8s相关镜像可以pull到Docker中然后修改标签然后重新安装即可。



1. 出现下图结果即安装成功：



如果其他节点要加入该集群则在其服务器上执行，前提是已经完成初始化操作：

kubeadm join 192.168.254.104:6443 --token zpk1gi.306qtoukoocsit0x \

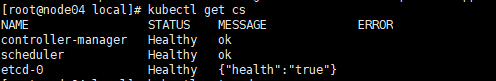
--discovery-token-ca-cert-hash sha256:04cd59cab6c9a5c9de6c4a592b5b8cdbe1c42c7f92e19004de6e40011e7b0a63

1. 成功后执行以下命令：

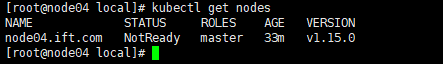
mkdir -p $HOME/.kube

cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

1. 使用kubectl get cs；可查看组件状态：



1. 使用kubectl get nodes；可查看当前集群的节点：

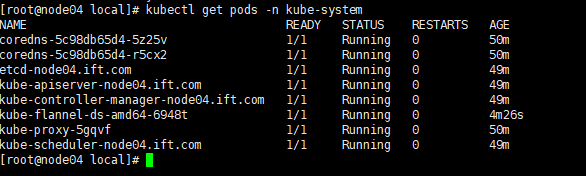


Status为NotReady是因为还缺少一个组件：flannel，flannel负责各Pod之间的通信的网络配置，如果缺少该组件Pod之间是无法进行通信的。

1. 安装flannel：
   1. 执行命令：

kubectl apply -f <https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml>

* 1. docker image ls；如果出现quay.io/coreos/flannel这个image表示已经拖取image完成；
  2. kubectl get nodes；查看集群状态可发现Status已经变为Ready
  3. kubectl get pods –n kube-system；该命令用于查看当前节点上属于名称空间kube-system的所有Pod：



* 1. kubectl get ns：可查看当前节点上的所有名称空间

1. 子节点
   1. 先安装docker-ce、kubelet、kubectl、kubeadm
   2. 将docker、kubelet修改为开机自启
   3. 执行脚本：pull\_k8s\_images.sh拖取相关镜像并安装flannel或者使用docker save和docker load将master相关镜像导入到子节点的Docker中并重新打标docker tag：
   4. 将master的/etc/sysconfig/kubelet拷贝到子节点的相同目录或编辑子节点的改文件修改为：



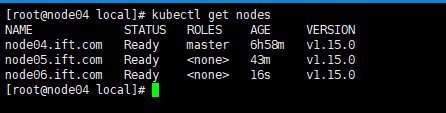
* 1. 执行如下加入命令：

kubeadm join 192.168.254.104:6443 --token zpk1gi.306qtoukoocsit0x --discovery-token-ca-cert-hash sha256:04cd59cab6c9a5c9de6c4a592b5b8cdbe1c42c7f92e19004de6e40011e7b0a63 --ignore-preflight-errors=Swap

注：如果不记得以上命令参数请使用以下命令查看：

kubeadm token create --print-join-command --ttl 0

1. 在master执行kubectl get nodes出现如下状态则安装成功：



1. 常用命令，所有命令详情使用--help查看：
2. 查看kubectl版本：kubectl version
3. 查看集群信息：kubectl cluster-info
4. 创建并运行Pod：kubectl run nginx-deploy --image=nginx:1.14-alpine --port=80 --replicas=1

以上命令表示使用镜像nginx:1.14-alpine创建一个名叫nginx-deploy的控制器并暴露80端口，pod的数量为1

1. 查看创建的控制器：kubectl get deployment
2. 查看当前节点上的Pod：kubectl get pods
3. 查看详细信息：kubectl get pods -o wide

注：该命令结果中有一个IP，该IP即为当前Pod的IP地址，该IP地址只能在集群内部访问，如果需要在集群外部访问需要使用Service；

1. 删除Pod：kubeclt delete pods PodName

PodName：Pod的名字，可以使用命令kubectl get pods查看

1. 删除后控制器会自动再次创建一个新的Pod因为创建控制器时指定的—replicas的值为1
2. 暴露服务：kubectl expose deployment nginx-deploy --name=nginx --port=80 --target-port=80 --protocol=TCP

deployment：被暴露的服务类型为deployment

nginx-deploy：deployment的名称

--name=nginx：指定暴露的服务名字

--port：指定暴露的服务端口

--target-port：指定容器的端口

--protocol：指定暴露的服务类型TCP|UDP，默认为TCP

--type：指定暴露的类型默认为ClusterIP，也就是只能在集群内部访问，这种地址更多的时候是被Pod客户端所访问的

1. 查看服务：kubectl get services；可查看到刚才暴露的服务信息。
2. service的IP地址也是动态分配的，可以通过服务名访问，但是由于节点的DNS不能解析所以还是不能够通过服务名去访问，因为必须要使用CoreDNS才能够解析。
3. 运行以交互式接口运行一个busybox：kubectl run client --image=busybox --replicas=1 --restart=Never –it

--restart：退出后不再启动一个新的Pod

-it：与busybox交互

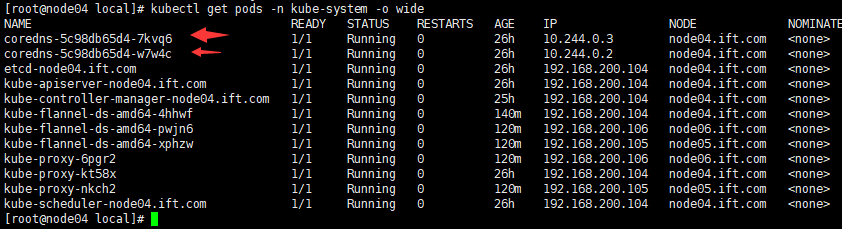
在运行的busybox中可以直接使用刚才暴露的服务名称来访问nginx如：

wget –O - -q nginx

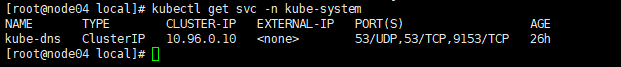
1. 查看名称空间kube-system的详细信息：kubectl get pods -n kube-system -o wide

kube-system：为k8s的系统名称空间。

-o wide：查看详细信息

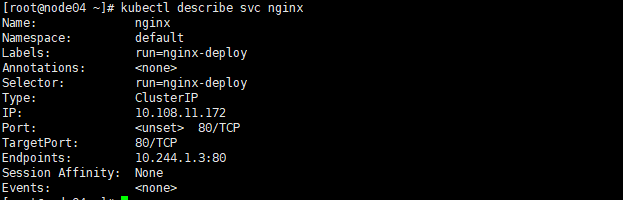


如图coredns的IP地址为10.244.0.3和10.244.0.2使用这两个IP地址可以解析但是coredns也是有服务名的，使用命令：kubectl get svc -n kube-system；可以查看到kube-dns的IP地址：



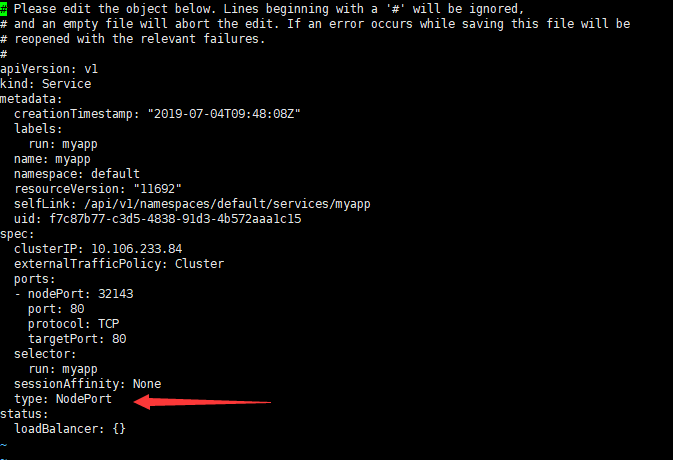
svc=services=service

1. 查看service详细信息：kubectl describe svc nginx

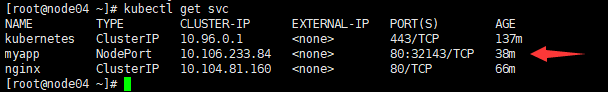


1. 查看Pod的标签：kubectl get pods --show-labels
2. 改变Pod数量：kubectl scale --replicas=5 deployment podName
3. 更新Pod中的容器版本：kubectl set image deployment myapp myapp=ikubernetes/myapp:v2；该方式为灰度更新，就是一个一个地替换
4. 查看替换过程：kubectl rollout status deployment myapp
5. 将service暴露到集群外部可访问：kubectl edit svc myapp

将type：的值改为NodePort即可



随后查看该服务的信息：kubectl get svc：



端口已经映射到主机的32134端口；